Олійник А. О.,
Субботін С. О.,
Олійник О. О.

ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ

Запорізький національний
технічний університет
2011

МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ
ТЕХНІЧНИЙ УНІВЕРСИТЕТ

А. О. Олійник, С. О. Субботін, О. О. Олійник

ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ

Навчальний посібник

Рекомендовано Міністерством освіти і науки, молоді та спорту України як навчальний посібник для студентів вищих навчальних закладів, які навчаються за напрямом підготовки «Програмна інженерія»

Видання здійснено за підтримки Міжнародного проекту «Професійно-орієнтовані магістерські програми в галузі інжинірингу в Росії, Україні, Узбекистані» (*510920-Tempus-1-2010-1-De-Tempus-JPCR*) *за програмою Tempus Європейської Комісії*

Запоріжжя • ЗНТУ • 2012

УДК 004.4:004.05
ББК 32.973
О-53

Гриф надано Міністерством освіти і науки, молоді та спорту України
(лист № 1/11-6226 від 15.07.11 р.)

Рецензенти: доктор технічних наук, професор, професор кафедри штучного інтелекту Харківського національного університету радіоелектроніки *Бодянський Є. В.*;
доктор технічних наук, професор, декан математичного факультету Запорізького національного університету *Гоменюк С. І.*;
доктор технічних наук, професор, професор кафедри обчислювальної техніки та програмування Національного технічного університету «Харківський політехнічний інститут» *Дмитрієнко В. Д.*

Олійник А. О.

О-53 Інтелектуальний аналіз даних : навчальний посібник / А. О. Олійник, С. О. Субботін, О. О. Олійник. – Запоріжжя : ЗНТУ, 2012. – 278 с.

ISBN 978-617-529-052-1

Книга містить систематизований виклад основних понять, моделей і методів інтелектуального аналізу даних, які можуть використовуватися при вирішенні практичних завдань розпізнавання образів, прийняття рішень, класифікації та прогнозування. Розглянуто методи відбору інформативних ознак та критерії оцінювання індивідуальної і групової інформативності, описано моделі інтелектуального аналізу даних: асоціативні правила, дерева рішень, нейронні та нейронечіткі мережі. Наведено аналіз градієнтних, еволюційних та мультиагентних методів синтезу інтелектуальних моделей.

Видання призначено для студентів комп'ютерних спеціальностей вищих навчальних закладів, а також може використовуватися педагогічними працівниками та практичними фахівцями.

УДК 004.4:004.05
ББК 32.973

ISBN 978-617-529-052-1

ЗМІСТ

ВСТУП

Розв'язання завдань технічного та біомедичного діагностування, розпізнавання образів, класифікації та прогнозування пов'язано з необхідністю виконання аналізу даних.

Актуальність створення, вивчення та використання методів, моделей та комп'ютерних систем, що в своїй роботі застосовують методи інтелектуального аналізу даних, підтверджується Державною програмою розвитку промисловості на 2003–2011 роки, схваленою Постановою Кабінету Міністрів України № 1174 від 28.07.2003 та Указом Президента України №102/2003 від 12.02.2003 «Про Концепцію державної промислової політики», що визначають як пріоритет інформатизації створення систем підтримки прийняття рішень та штучного інтелекту; Постановою Кабінету Міністрів України №1896 від 10.12.2003, яка передбачає «… розроблення методології інтелектуального аналізу даних … на основі застосування сучасних методів нечіткої логіки, штучного інтелекту та добування знань із баз даних»; Постановою Кабінету Міністрів України №789 від 15 липня 1997 р. «Про першочергові заходи інформатизації», Законом України №75/98–ВР від 04.02.1998 «Про Концепцію Національної програми інформатизації», Законом України №76/98–ВР від 04.02.1998 «Про затвердження Завдань Національної програми інформатизації на 1998–2000 роки», Постановою Верховної Ради України № 914–XIV від 13.07.1999 «Про затвердження Завдань Національної програми інформатизації на 1999–2001 роки», які передбачають, зокрема, «створити діючі зразки та прототипи конкурентоспроможних засобів та систем: методичне та програмне забезпечення проектування і розроблення комп'ютеризованих систем для застосування в управлінні, програмно-технічні засоби підтримки експертного прийняття рішень, високопродуктивні оброблювачі інформації з нейромережевою архітектурою, проблемно-орієнтовані робочі станції, нейрокомп'ютери та нейромережеві технології, програмне забезпечення інформаційно-аналітичної обробки текстових, фактографічних та статистичних даних, конкурентоспроможні інформа-

ційні технології формування аналітичних електронних оглядів і довідок, засоби інтелектуалізації широкого застосування».

Метою даної книги є систематизований виклад основних понять, моделей і методів інтелектуального аналізу даних, які можуть використовуватися при вирішенні практичних завдань розпізнавання образів, прийняття рішень, класифікації та прогнозування.

У **першому розділі** розглянуто методи пошуку найбільш інформативної комбінації ознак у навчальній вибірці. Наведено постанову завдання відбору ознак, проаналізовано основні методи вибору інформативної комбінації (метод повного перебору, пошук у глибину, пошук вглибину, метод гілок і границь, метод групового врахування аргументів, ранжирування ознак, випадковий пошук з адаптацією та ін.), розглянуто критерії оцінювання індивідуальної (коефіцієнт парної кореляції, кореляції знаків, кореляції Фехнера, дисперсійне відношення, коефіцієнт зв'язку, інформаційний критерій, ентропія ознаки та ін.) та групової (множинний коефіцієнт кореляції, коефіцієнт кореляції Пірсона, множинне дисперсійне відношення, множинний коефіцієнт зв'язку, інформаційний критерій, ентропія набору ознак, різні види похибок та відхилень) значущості ознак.

Другий розділ присвячено моделям інтелектуального аналізу даних. Описано нейромережеві моделі, асоціативні правила, дерева рішень, нейро-нечіткі моделі. Надано загальну характеристику описаних моделей, виконано їх класифікацію, розглянуто програмні засоби для подання та обробки інтелектуальних моделей.

У **третьому розділі** наведено опис методів синтезу інтелектуальних моделей. Проаналізовано градієнтні методи (метод Коші, метод Ньютона, алгоритми спряжених градієнтів, метод Левенберга-Марквардта та ін.). Досліджено еволюційні методи, зокрема надано узагальнену схему роботи генетичних методів, описано оператори ініціалізації, відбору, схрещування та мутації, наведено принципи формування нового покоління та критерії зупинення еволюційного пошуку. Надано загальну характеристику інтелектуальних методів мультиагентної оптимізації та програмних засоби розв'язання оптимізаційних завдань для побудови інтелектуальних моделей.

Для спрощення **самостійного опрацювання** та кращого засвоєння матеріалу книги наприкінці кожного розділу наведено контрольні питання, а також практичні та тестові завдання.

Видання орієнтоване на студентів комп'ютерних спеціальностей вищих навчальних закладів, а також може використовуватися педагогічними працівниками та практичними фахівцями.

Матеріал, наведений у книзі, призначений для вивчення та апробований авторами при читанні курсу «Інтелектуальний аналіз даних» у Запорізькому національному технічному університеті. Книга також може використовуватися при вивченні окремих розділів дисциплін «Дискретні структури», «Інтелектуальні системи», «Технологія та використання штучних нейронних мереж».

Більш детальна інформація про використання матеріалу книги у навчальному процесі, а також посилання корисні літературні джерела та рекомендовані програмні засоби доступні на веб-сайті авторів за адресою: `http://csit.narod.ru`

Терміни та назви методів, визначення яких наводиться у наступному тексті виділено **напівжирним шрифтом**.

Для спрощення пошуку навчальних завдань, прикладів, контрольних питань і тестів для самоперевірки використовується така **система умовних позначень**:

 – контрольні питання,

 – практичні завдання, що мають бути виконані з використанням комп'ютера,

 – завдання, що виконуються вручну,

 – тестові завдання.

1 ВІДБІР ІНФОРМАТИВНИХ ОЗНАК

У задачах розпізнавання образів для визначення значення вихідного параметра застосовуються математичні моделі. Для синтезу таких моделей використовується вхідна навчальна вибірка, що складається з великого набору ознак, які характеризують досліджуваний об'єкт або процес. Масиви даних великого розміру, як правило, містять надлишкові й неінформативні ознаки, які ускладнюють не тільки процес синтезу моделі, але й приводять до її надлишковості, що збільшує час класифікації по такій моделі. Таким чином, при вирішенні задач розпізнавання образів важливим етапом є процес редукції вхідного набору ознак.

Складність вирішення задач вибору максимально значимої комбінації ознак полягає в її комбінаторному характері. Використання повного перебору всіх можливих комбінацій при великій кількості ознак приводить до комбінаторного вибуху. Тому такий підхід на практиці виявляється неприйнятним, у результаті чого були розроблені методи скороченого перебору комбінацій ознак.

У даному розділі приводиться загальна постановка задачі відбору інформативних ознак, опис відомих підходів до її вирішення. Розглядається структура методів відбору ознак, а також критерії оцінювання індивідуальної та спільної значущості ознак.

1.1 Загальна постановка задачі відбору інформативних ознак для синтезу розпізнаючих моделей

Нехай задана вибірка вихідних даних у вигляді:

$$< X = \{X_1, X_2, ..., X_L\} = \{X_i\},\ \ Y = \{y_1, y_2, \ldots, y_m\} = \{y_j\} >,$$

де X – вихідний набір значень ознак, що характеризують досліджуваний об'єкт або процес; Y – масив значень вихідного параметра в заданій вибірці; $X_i = \{x_{ij}\}$ – i-та ознака у вихідній вибірці, $i = 1, 2, ..., L$; x_{ij} – значення i-ої ознаки для j-го екземпляра вибірки, $j = 1, 2, ..., m$; y_j – значення прогнозованого параметра для j-го екземпляра; L – загальна кількість ознак у вихідному наборі; m – кількість екземплярів вибірки.

Тоді постановка задачі відбору інформативних ознак може бути представлена одним з таких способів.

1. Ідеалізована постановка: виділити комбінацію ознак X^* з вихідного масиву даних, при якій досягається мінімум заданого критерію оцінювання набору ознак:

$$J(X^*) = \min_{Xe \in XS} J(Xe),$$

де Xe – елемент множини XS; $J(Xe)$ – критерій оцінювання значимості набору ознак Xe; XS – множина всіх можливих комбінацій ознак, отримана з вихідного набору ознак X.

2. Класична постановка: відібрати з множини вхідних L ознак комбінацію, що складається не більш, ніж з L_0 ознак ($L_0 < L$), при якій досягається оптимум заданого критерію:

$$J(X^*) = \min_{Xe \in XS, |Xe| \le L_0} J(Xe),$$

де $|Xe|$ – кількість елементів у множині Xe.

3. Знайти набір ознак мінімального розміру, що забезпечує досягнення заданого значення критерію оцінювання значимості набору ознак:

$$\left|X^*\right| = \min_{Xe \in XS, J(Xe) < \varepsilon} \left|Xe\right|,$$

де ε – задане значення критерію оцінювання набору ознак J.

Результатом виконання процедури відбору ознак є оптимальний набір ознак X^*, що має достатню інформативність. Інформативність ознаки (набору ознак) – це величина, що відображає ступінь взаємозв'язку ознаки (набору ознак) із прогнозованим параметром. Інформативність комбінації ознак дорівнює сумі інформативності окремих ознак тільки при їхній незалежності. Якщо ознаки є залежними одна від одної, то інформативність набору не виражається через інформативність окремих ознак.

Таким чином, отриманий у результаті відбору ознак оптимальний набір X^*, маючи достатню інформативність, найбільш повно відображає досліджуваний об'єкт або процес. При цьому з вхідного набору X виключаються:

– незначущі ознаки – ознаки, що не впливають на вихідний параметр;

– надлишкові ознаки – ознаки, значення яких залежать від інших ознак. Такі ознаки не приводять до поліпшення якості прогнозування по синтезованій моделі.

1.2 Аналіз методів відбору інформативних ознак

Виділення максимально значимої комбінації інформативних ознак є досить важким і ресурсномістким завданням, оскільки вона пов'язана з необхідністю комбінаторного перебору. У цей час запропоновані різні методи виділення набору ознак, серед яких найбільше поширення отримали:

– метод повного перебору (exhaustive search);

– пошук у глибину (depth-first search);

– пошук завширшки (breath-first search);

– метод гілок і границь (branch and bound method) або скорочений пошук у глибину;

– метод групового врахування аргументів або скорочений пошук завширшки;

– метод послідовного додавання ознак (forward selection);

– метод послідовного видалення ознак (backward selection);

– метод почергового додавання й видалення ознак (combined selection);

– ранжирування ознак;

– кластеризація ознак (unsupervised learning for feature selection);

– випадковий пошук з адаптацією (adaptive stochastic search);

– еволюційний пошук (evolutionary search).

Класифікація методів відбору ознак наведена на рис. 1.1.

У **методі класичного повного перебору** аналізуються всі можливі комбінації ознак, серед яких вибирається найкраща.

Крок 1. Згенерувати всі можливі набори ознак.

Крок 2. Оцінити всі отримані на попередньому кроці комбінації Xe, обчисливши для кожної з них значення критерію оцінювання набору ознак: $J(Xe)$.

Крок 3. Визначити оптимальне значення критерію оцінювання набору ознак: $J_{\text{opt}} = \min(J(Xe))$.

Крок 4. Визначити оптимальний набір ознак: $X^* = \arg\min J(Xe)$.

Крок 5. Зупинення.

Класичний повний перебір є найбільш простим для реалізації й гарантує одержання оптимального рішення. Недоліком такого методу є його обмеженість при рішенні практичних завдань, зумовлена великими обчислювальними витратами при його використанні.

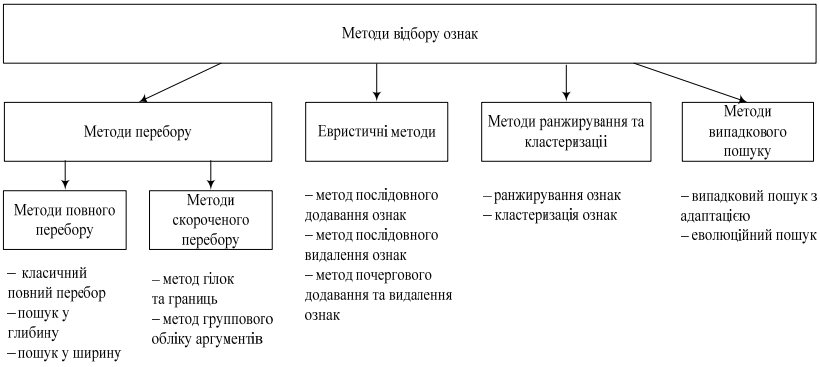


Рисунок 1.1 – Класифікація методів відбору ознак

Виконати повний перебір всіх можливих комбінацій ознак можливо за допомогою *обходу дерева можливих наборів ознак*, у якому вузли відповідають наборам ознак. Кореневий вузол відповідає порожньому набору. Кожний наступний набір утвориться шляхом приєднання деякої ознаки до попереднього батьківського вузла. З метою уникання появи в дереві вузлів, що відповідають однаковим наборам, що відрізняються тільки порядком ознак, до дочірніх вузлів додають тільки ті ознаки, номери яких перевищують максимальний номер ознаки в батьківському наборі.

При використанні **методу пошуку в глибину** обхід дерева можливих наборів ознак відбувається по напрямку від кореневого (батьківського) вузла до вузла-нащадка, що характеризується найбільшою кількістю ознак.

Для обходу дерева використовується рекурсивний виклик процедури нарощування поточного вузла Наростити(Xe), у такий спосіб дерево в явному виді не будується. Процедура нарощування по черзі приєднує до поточного набору по одній ознаці, і для кожного з отриманих наборів спочатку обчислює значення критерію оцінювання, а потім викликає себе рекурсивно.

Крок 1. Ініціалізувати початковий набір ознак (кореневий вузол): $Xe = \varnothing$. Виконати: $X^* = Xe$, $J_{опт} = J(X^*)$, де X^* – оптимальний набір ознак; $J_{опт}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 2. Виконати нарощування поточного вузла дерева, викликавши процедуру Наростити(Xe).

Крок 3. Зупинення. Результатом виконання методу є оптимальний набір ознак X^* і оптимальне значення критерію оцінювання набору ознак $J_{\text{опт}}$, отримані в ході рекурсивного виконання процедури Наростити(*Xe*).

Процедура Наростити(*Xe*) виконується в наступній послідовності.

Крок 1. З метою усунення можливості генерації наборів ознак, що відрізняються тільки порядком ознак, визначити максимальний номер ознаки max у вихідному наборі *Xe*: $\max = \max\limits_{X_i \in Xe} i$. У випадку, якщо max = *L*, тоді виконати перехід до кроку 9.

Крок 2. Встановити лічильник включених у переглянуті набори ознак: $i = \max + 1$.

Крок 3. Згенерувати новий набір ознак шляхом додавання *i*-ї ознаки до поточного набору: $Xt = \{Xe \cup X_i\}$.

Крок 4. Оцінити комбінацію ознак *Xt*, обчисливши значення критерію оцінювання набору ознак: *J*(*Xt*).

Крок 5. У випадку, якщо $J(Xt) < J_{\text{опт}}$, тоді виконати: $J_{\text{опт}} = J(Xt)$ і $X^* = Xt$.

Крок 6. Наростити вузол дерева *Xt*, для чого рекурсивно перейти до виконання процедури нарощування поточного вузла дерева: Наростити(*Xt*).

Крок 7. Виконати: $i = i + 1$.

Крок 8. У випадку, якщо $i \le L$, тоді виконати перехід до кроку 3.

Крок 9. Зупинення.

При обході дерева можливих наборів ознак за допомогою **пошуку в ширину** відбувається послідовний перегляд вузлів дерева по рівнях, тобто комбінації ознак аналізуються по збільшенню їхнього розміру. Таким чином, на початку проглядаються всі можливі одноознакові комбінації, потім комбінації, що складаються із двох ознак і т. д.

Крок 1. Ініціалізувати початковий набір ознак: $Xe = \varnothing$. Виконати: $X^* = Xe$, $J_{\text{опт}} = J(X^*)$, де X^* – оптимальний набір ознак; $J_{\text{опт}}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 2. Встановити лічильник аналізуємого рівня дерева (кількості ознак в аналізуємому наборі): $k = 1$.

Крок 3. Згенерувати всі можливі набори ознак розміром k.

Крок 4. Оцінити всі отримані на попередньому кроці комбінації Xe_k, обчисливши для кожної з них значення критерію оцінювання набору ознак: $J(Xe_k)$.

Крок 5. У випадку, якщо знайдено комбінацію ознак зі значенням критерію оцінювання, що краще поточного оптимального значення ($J(Xe_k) < J_{\text{опт}}$), тоді виконати: $J_{\text{опт}} = J(Xe_k)$ і $X^* = Xe_k$.

Крок 6. Збільшити лічильник кількості ознак в аналізованому наборі: $k = k + 1$.

Крок 7. У випадку, якщо $k \le L$, тоді виконати перехід до кроку 3.

Крок 8. Зупинення.

Аналогічно методу повного перебору недоліком методів пошуку в глибину й пошуку в ширину є їхня обмеженість практичного застосування.

Пошук оптимальних комбінацій за допомогою дерева наборів ознак може бути легко перетворений від повного перебору рішень до скороченого, заснованому на використанні деякої додаткової інформації, отриманої з вихідного масиву даних.

До методів скороченого перебору відносяться метод гілок і границь (скорочений пошук у глибину) і метод групового обліку аргументів (скорочений пошук в ширину).

Скорочення кількості комбінацій, що перебираються, у **методі гілок і границь** досягається за рахунок відмови від нарощування гілки дерева у випадку, якщо вже є краща гілка. Тобто вузол, що відповідає набору ознак Xe, не наращується, якщо значення критерію оцінювання набору ознак $J(Xe)$ виявиться гірше, ніж на найкращому із уже оцінених наборів меншої розмірності.

З метою пошуку більш оптимального набору спочатку ознаки ранжируються в порядку убування їхньої індивідуальної значимості.

Крок 1. Ініціалізувати початковий набір ознак (кореневий вузол): $Xe = \varnothing$. Виконати: $X^* = Xe$, $J_{\text{опт}} = J(X^*)$, де X^* – оптимальний набір ознак; $J_{\text{опт}}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 2. Упорядкувати ознаки по убуванню інформативності.

Крок 3. Виконати нарощування поточного вузла дерева, викликавши процедуру Наростити(Xe).

Крок 4. Зупинення. Результатом виконання методу є оптимальний набір ознак X^* і оптимальне значення критерію оцінювання набору ознак $J_{\text{опт}}$, отримані в ході рекурсивного виконання процедури Наростити(*Xe*).

Процедура Наростити(*Xe*) виконується в наступній послідовності.

Крок 1. У випадку, якщо із числа вже оцінених наборів ознак найдеться така комбінація *Xk*, що $J(Xe) > J(Xk)$ і $|Xe| \geq |Xk|$, тоді виконати перехід до кроку 10.

Крок 2. З метою уникнення можливості генерації наборів ознак, що відрізняються тільки порядком ознак, визначити максимальний номер ознаки max у вихідному наборі *Xe*: $\max = \max_{X_i \in Xe} i$. У випадку, якщо max = *L*, тоді виконати перехід до кроку 10.

Крок 3. Встановити лічильник включених у переглянуті набори ознак: $i = \max + 1$.

Крок 4. Згенерувати новий набір ознак шляхом додавання *i*-ї ознаки до поточного набору: $Xt = \{Xe \cup X_i\}$.

Крок 5. Оцінити комбінацію ознак *Xt*, обчисливши значення критерію оцінювання набору ознак: $J(Xt)$.

Крок 6. У випадку, якщо $J(Xt) < J_{\text{опт}}$, тоді виконати: $J_{\text{опт}} = J(Xt)$ і $X^* = Xt$.

Крок 7. Наростити вузол дерева *Xt*, для чого рекурсивно перейти до виконання процедури нарощування поточного вузла дерева: Наростити(*Xt*).

Крок 8. Виконати: $i = i + 1$.

Крок 9. У випадку, якщо $i \leq L$, тоді виконати перехід до кроку 4.

Крок 10. Зупинення.

Застосування методу гілок і границь дозволяє скоротити час, необхідний для пошуку. Недоліком такого методу є послідовне додавання ознак до оцінюваного набору ознак, що часто приводить до виключення з розгляду комбінацій ознак, що володіють максимальною інформативністю.

У **методі групового врахування аргументів** (МГВА) на кожній *t*-ій ітерації оцінюється не один набір ознак, а множина P_t ($|P_t| = N$) наборів, що називається *t*-им рядом. Для переходу від по-

точного P_t до наступного P_{t+1} ряду від кожного набору $Xe \in P_t$ породжується $L - t$ нових наборів шляхом приєднання одного з ознак, що не належать набору Xe. Зі згенерованих $N(L - t)$ наборів ознак у наступний ряд відбирається не більше N наборів, кращих за значенням критерію оцінювання набору ознак. Таким чином, на кожній ітерації розмір наборів ознак збільшується на одиницю.

Кількість наборів ознак на кожній ітерації N називається шириною пошуку. Зокрема при $N = 1$ метод групового обліку аргументів являє собою евристичний метод послідовного додавання ознак.

Крок 1. Встановити лічильник ітерацій (рядів): $t = 1$. Ініціалізувати початкову множину наборів ознак одноознаковими комбінаціями: $P_t = \{Xe \mid Xe = \{X_i\}, i = 1, 2, \ldots, L\}$... Виконати: $X^* = \varnothing$, $J_{\text{опт}} = \infty$, де X^* – оптимальний набір ознак; $J_{\text{опт}}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 2. Обчислити значення критерію оцінювання кожного набору ознак з t-го ряду: $J(Xe)$, $Xe \in P_t$.

Крок 3. У випадку, якщо найдеться такий набір $Xe \in P_t$, що $J(Xe) < J_{\text{опт}}$, тоді виконати: $J_{\text{опт}} = J(Xe)$ і $X^* = Xe$.

Крок 4. Відсортувати ряд P_t по зростанню значення критерію оцінювання набору ознак.

Крок 5. Залишити в ряді P_t N кращих наборів ознак. Всі інші набори видалити.

Крок 6. Збільшити лічильник ітерацій (рядів): $t = t + 1$.

Крок 7. Перевірити критерії закінчення пошуку (досягнення максимально можливого розміру набору ознак L_0, перевищення припустимої кількості ітерацій T і т. п.). У випадку, якщо такі критерії досягнуті, тоді виконати перехід до кроку 10.

Крок 8. Згенерувати наступний ряд. Для цього сформувати для кожного набору $Xe \in P_{t-1}$ $L - t$ нових наборів шляхом приєднання одного з ознак, що не належать набору Xe.

Крок 9. Виконати перехід до кроку 2.

Крок 10. Зупинення.

Такий метод дозволяє позбутися необхідності оцінювання кожної з $2^L - 1$ можливих комбінацій ознак, але є більше складним у порівнянні з методом гілок і границь.

Евристичні методи відбору ознак використовують жадібні стратегії (greedy strategy) для додавання або видалення ознак на

кожній ітерації. До евристичних методів відносяться метод послідовного додавання ознак, метод послідовного видалення ознак, а також метод почергового додавання й видалення ознак.

У **методі послідовного додавання ознак** на основі оптимального набору ознак, знайденого на попередній ітерації X^*_{t-1}, на поточній ітерації формуються всі можливі комбінації ознак $Xe_{t,k}$ шляхом додавання однієї ознаки, ще не включеного в набір X^*_{t-1}. У наступній ітерації формування нових рішень відбувається на основі оптимального набору $X^*_t = \text{argmin}\, J(Xe_{t,k})$, знайденого на поточній ітерації.

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$. Ініціалізувати початковий набір ознак: $Xe_t = \varnothing$. Виконати: $X^* = Xe_t$, $J_{\text{опт}} = J(X^*)$, де X^* – оптимальний набір ознак; $J_{\text{опт}}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 2. Збільшити лічильник ітерацій: $t = t + 1$. Згенерувати всі можливі нові набори ознак шляхом додавання однієї ознаки до набору X^*, одержавши в такий спосіб $M = L - |X^*|$ пробних комбінацій ознак $Xe_{t,k}$, $k = 1, 2, \ldots, M$, де L – кількість ознак у вихідному масиві даних; $|X^*|$ – кількість відібраних ознак в оптимальному наборі, отриманому на попередньому кроці.

Крок 3. Оцінити кожний з $Xe_{t,k}$ набір ознак, розрахувавши значення критерію оцінювання набору ознак $J(Xe_{t,k})$.

Крок 4. Перевірити критерії закінчення пошуку.

Як такі критерії можуть бути використані:

– виникнення ситуації, при якій всі згенеровані на поточній ітерації набори ознак гірше оптимального набору, отриманого раніше: $J_{\text{опт}} < \min(J(Xe_{t,k}))$;

– досягнення максимально можливого розміру набору ознак L_0;

– перевищення припустимої кількості ітерацій T.

У випадку, якщо такі критерії досягнуті, тоді виконати перехід до кроку 7.

Крок 5. Виконати: $J_{\text{опт}} = \min(J(Xe_{t,k}))$ і $X^* = \text{argmin}\, J(Xe_{t,k})$.

Крок 6. Перейти до виконання кроку 2.

Крок 7. Зупинення.

Метод послідовного додавання ознак є простим у реалізації, а також не вимагає значних тимчасових витрат при його використанні: обчислювальна складність методу $O(L^2)$, що значно нижче, ніж у методів повного перебору.

Недолік такого методу викликаний неоптимальністю жадібної стратегії пошуку: при використанні методу послідовного додавання ознак часто в оптимальний набір включаються надлишкові ознаки.

У **методі послідовного видалення ознак** на основі оптимального набору ознак, знайденого на попередній ітерації X^*_{t-1}, на поточній ітерації формуються всі можливі комбінації ознак $Xe_{t,k}$ шляхом видалення однієї ознаки, ще з набору X^*_{t-1}. У наступній ітерації формування нових рішень відбувається на основі оптимального набору $X^*_t = \text{argmin}\, J(Xe_{t,k})$, знайденого на поточній ітерації.

Таким чином, у методі послідовного видалення ознак на кожній ітерації виключаються ознака, що мінімально погіршує критерій оцінювання набору ознак.

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$. Ініціалізувати початковий набір ознак комбінацією із всіх можливих ознак, що містяться у вихідному наборі даних: $Xe_t = \{X_1, X_2, \ldots, X_L\}$... Виконати: $X^* = Xe_t$, $J_{\text{опт}} = J(X^*)$, де X^* – оптимальний набір ознак; $J_{\text{опт}}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 2. Збільшити лічильник ітерацій: $t = t + 1$. Згенерувати всі можливі нові набори ознак шляхом видалення однієї ознаки з набору X^*, одержавши в такий спосіб $|X^*|$ пробних комбінацій ознак $Xe_{t,k}$, $k = 1, 2, \ldots, |X^*|$, де $|X^*|$ – кількість відібраних ознак в оптимальному наборі, отриманому на попередньому кроці.

Крок 3. Оцінити кожний з $Xe_{t,k}$ набір ознак, розрахувавши значення критерію оцінювання набору ознак $J(Xe_{t,k})$.

Крок 4. Перевірити критерії закінчення пошуку.

Як такі критерії можуть бути використані:

– виникнення ситуації, при якій значення критеріїв оцінювання всіх згенерованих на поточній ітерації наборів ознак гірше максимально припустимої величини ε, заданої користувачем: $\min(J(Xe_{t,k})) > \varepsilon$. Величина (визначає мінімально прийнятне значення критерію оцінювання набору ознак;

– виникнення ситуації, при якій всі згенеровані на поточній ітерації набори ознак є значно гірше оптимального набору, отриманого раніше: $|\min(J(Xe_{t,k})) - J_{\text{опт}}| > \xi$, де ξ – величина, що визначає максимально можливе погіршення значення критерію оцінювання набору ознак на одній ітерації;

– досягнення максимально можливого розміру набору ознак L_0;

– перевищення припустимої кількості ітерацій T.

У випадку, якщо такі критерії досягнуті, тоді виконати перехід до кроку 7.

Крок 5. Виконати: $J_{\text{опт}} = \min(J(Xe_{t,k}))$ і $X^* = \text{argmin}\, J(Xe_{t,k})$.

Крок 6. Перейти до виконання кроку 2.

Крок 7. Зупинення.

Метод послідовного видалення ознак також, як і попередній метод, простий у реалізації. До основного недоліку такого методу варто віднести неоптимальність жадібної стратегії. Важливо відзначити, що метод послідовного видалення ознак працює повільніше в порівнянні з методом послідовного додавання ознак, оскільки на початкових ітераціях необхідно оцінювати набори ознак, що складаються із всіх або майже всіх ознак. Такий метод застосовують у випадках, коли відомо, що інформативних ознак значно більше, ніж малоінформативних або надлишкових.

Метод послідовного додавання та видалення ознак сполучає в собі ідеї двох розглянутих раніше методів, що діють протилежно, у результаті чого виходить нежадібна стратегія пошуку.

Ідея такого методу полягає в тому, щоб використовувати стратегію додавання ознак доти, поки не виникне ситуація, при якій збільшення кількості ознак в оптимальному наборі не приводить до поліпшення значення критерію оцінювання набору ознак $J(X^*)$. Після цього запускається процедура видалення ознак, що припиняє своє функціонування за тих самих умов, що й процедура додавання ознак. Процедури додавання й видалення ознак чергуються доти, поки значення критерію $J(X^*)$ в оптимальних точках X^* не перестане поліпшуватися.

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$. Ініціалізувати початковий набір ознак: $Xe_t = \varnothing$. Виконати: $X^* = Xe_t$, $J_{\text{опт}} = J(X^*)$, де X^* – оптимальний набір ознак; $J_{\text{опт}}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 2. Виконати процедуру додавання ознак.

Крок 2.1. Збільшити лічильник ітерацій: $t = t + 1$. Згенерувати всі можливі нові набори ознак шляхом додавання однієї ознаки до набору X^*, одержавши в такий спосіб $M = L - |X^*|$ пробних комбінацій ознак $Xe_{t,k}$, $k = 1, 2, \ldots, M$, де L – кількість ознак у ви-

хідному масиві даних; $|X^*|$ – кількість відібраних ознак в оптимальному наборі, отриманому на попередньому кроці.

Крок 2.2. Оцінити кожний з $Xe_{t,k}$ набір ознак, розрахувавши значення критерію оцінювання набору ознак $J(Xe_{t,k})$.

Крок 2.3. Перевірити критерії закінчення виконання процедури додавання ознак. У випадку, якщо виникла ситуація, при якій всі згеровані на поточній ітерації набори ознак гірше оптимального набору, отриманого раніше: $J_{\text{опт}} < \min(J(Xe_{t,k}))$, тоді завершити процедуру додавання ознак і перейти до кроку 3.

Крок 2.4. Виконати: $J_{\text{опт}} = \min(J(Xe_{t,k}))$ і $X^* = \text{argmin}\, J(Xe_{t,k})$.

Крок 2.5. Виконати перехід до кроку 2.1.

Крок 3. Виконати процедуру видалення ознак.

Крок 3.1. Збільшити лічильник ітерацій: $t = t + 1$. Згенерувати всі можливі нові набори ознак шляхом видалення однієї ознаки з набору X^*, одержавши в такий спосіб $|X^*|$ пробних комбінацій ознак $Xe_{t,k}$, $k = 1, 2, \ldots, |X^*|$.

Крок 3.2. Оцінити кожний з $Xe_{t,k}$ набір ознак, розрахувавши значення критерію оцінювання набору ознак $J(Xe_{t,k})$.

Крок 3.3. Перевірити критерії закінчення виконання процедури видалення ознак.

Як такі критерії можуть бути використані:

– виникнення ситуації, при якій значення критеріїв оцінювання всіх згенерованих на поточній ітерації наборів ознак гірше максимально припустимої величини ε, заданої користувачем: $\min(J(Xe_{t,k})) > \varepsilon$. Величина ε визначає мінімально прийнятне значення критерію оцінювання набору ознак;

– виникнення ситуації, при якій всі згенеровані на поточній ітерації набори ознак є значно гірше оптимального набору, отриманого раніше: $|\min(J(Xe_{t,k})) - J_{\text{опт}}| > \xi$, де ξ – величина, що визначає максимально можливе погіршення значення критерію оцінювання набору ознак на одній ітерації;

У випадку, якщо критерії закінчення виконання процедури видалення ознак виконуються, тоді завершити процедуру видалення ознак і перейти до кроку 4.

Крок 3.4. Виконати: $J_{\text{опт}} = \min(J(Xe_{t,k}))$ і $X^* = \text{argmin}\, J(Xe_{t,k})$.

Крок 3.5. Виконати перехід до кроку 3.1.

Крок 4. Перевірити критерії закінчення пошуку (досягнення максимально можливого розміру набору ознак L_0, перевищення припустимої кількості ітерацій T і т. п.). У випадку, якщо такі критерії досягнуті, тоді виконати перехід до кроку 6.

Крок 5. Перейти до виконання кроку 2.

Крок 6. Зупинення.

Метод почергового додавання й видалення ознак є більше складним у реалізації й працює довше в порівнянні з методами послідовного додавання й послідовного видалення ознак окремо й також не гарантує оптимальності знайденого рішення. Однак рішення, отримані за допомогою такого методу, як правило, виявляються більш оптимальними в порівнянні з рішеннями, отриманими шляхом застосування методів послідовного додавання й послідовного видалення ознак.

Методи ранжирування й кластеризації не використовують критерії оцінювання спільного впливу набору ознак на вихідний параметр. Так при ранжируванні використовуються критерії оцінювання індивідуальної значимості ознак. У методах кластеризації застосовуються метрики відстані на ознаках.

Для відбору ознак також використовується **ранжирування ознак**. При такому підході виконується сортування ознак по оптимальності обраного критерію індивідуальної значимості. Після індивідуального оцінювання кожної ознаки і їхнього сортування відбувається вибір певної кількості ознак, індивідуальна значимість яких задовольняє заданим умовам.

Для оцінювання індивідуальної значимості ознак можуть використовуватися критерії кореляції (парний, Фехнера, знаків), інформаційний критерій, ентропія ознак.

Крок 1. Обчислити значення критерію оцінювання індивідуальної значимості кожної ознаки у вихідному наборі ознак.

Крок 2. Упорядкувати ознаки по убуванню інформативності.

Крок 3. Сформувати комбінацію з перших L_0 ознак або з ознак, значення індивідуальної значимості яких вище граничного. Отримана комбінація вважається оптимальним набором.

Крок 4. Зупинення.

Перевагою такого підходу є простота реалізації, а недоліком – можливість застосування тільки у випадку, якщо ознаки у вихідному наборі є статистично незалежними. Як правило, при рішенні

практичних завдань ознаки статистично залежать друг від друга, у результаті чого при використанні такого підходу для відбору ознак виходять комбінації, що містять надлишкові ознаки, отже, такі комбінації виявляються далеко не оптимальними.

Виділити максимально значиму комбінацію можна за допомогою **кластеризації ознак** (unsupervised learning for feature selection).

Методи **кластеризації** застосовують для розбивки вибірки на кластери, що складаються зі схожих екземплярів, і виділення в кожній групі одного найбільш типового екземпляра. Аналогічні дії можна виконати не над екземплярами, а над ознаками, якщо ввести функцію відстані на ознаках.

Крок 1. Для кожної ознаки у вихідному наборі обчислити відстань d_{ab} від нього до інших ознак.

Крок 2. На основі розрахованих на попередньому кроці відстаней d_{ab} між ознаками виконати кластеризацію ознак, згрупувавши їх по кластерах.

Крок 3. Виділити типових представників у кожному кластері.

Крок 4. Сформувати інформативну комбінацію з ознак, виділених на попередньому етапі.

Крок 5. Зупинення.

Недоліком кластеризації ознак є можливість існування кластерів, що цілком складаються з неінформативних ознак, у результаті чого типові представники таких кластерів можуть увійти в комбінацію ознак, що вважається найбільш інформативною.

Кластеризацію ознак доцільно застосовувати на початкових етапах інших методів відбору ознак для формування груп (кластерів) схожих ознак. Після цього в процесі пошуку оптимальної комбінації занадто схожим ознакам, що належать одному класу, забороняється входити в той самий набір.

У **методі випадкового пошуку з адаптацією** на кожної t-ій ітерації використовується популяція P_t, що складається з N рішень. Формування нових рішень відбувається шляхом випадкової генерації N наборів ознак залежно від розподілу ймовірностей включення ознак у генеруємі набори.

Ідея адаптації полягає у тому, щоб при генерації нових наборів ознак імовірність включення в них ознак, які частіше входять

у кращі набори, була більшою в порівнянні з імовірністю мало використовуваних у кращих наборах ознак.

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$. Встановити рівні ймовірності включення ознак у генеруємі набори: $p_1 = p_2 = \ldots = p_L = 1/L$, де p_i – імовірність включення i-ї ознаки у формований набір.

Крок 2. Ініціалізувати початкову множину наборів ознак P_t, згенерував N наборів ознак відповідно до розподілу $\{p_1, p_2, \ldots, p_L\}$... Обчислити значення критерію оцінювання кожного набору ознак з P_t: $J(Xe)$, $Xe \in P_t$. Виконати: $J_{\text{опт}} = \min(J(Xe))$ і $X^* = \text{argmin}\, J(Xe)$, де X^* – оптимальний набір ознак; $J_{\text{опт}}$ – оптимальне значення критерію оцінювання набору ознак.

Крок 3. Визначити найкращу $X_{\min} = \text{argmin}\, J(Xe)$ і найгіршу $X_{\max} = \text{argmax}\, J(Xe)$ комбінації ознак у поточній популяції. У випадку, якщо $J(X_{\min}) < J_{\text{опт}}$, тоді виконати: $J_{\text{опт}} = J(X_{\min})$ і $X^* = X_{\min}$.

Крок 4. Встановити $s = 0$, де s – сумарне значення зміни ймовірностей включення ознак у генеруємі набори.

Крок 5. Зменшити ймовірності включення в нові набори ознак, що входять у найгірший набір $X_{\max}$.

Крок 5.1. Якщо $p_i > h$ $(X_i \in X_{\max})$, тоді виконати: $p_i = p_i - h$ і $s = s + h$, де $h << 1/L$ – крок зміни ймовірності включення ознак у генеруємі набори.

Крок 5.2. Якщо $p_i \le h$ $(X_i \in X_{\max})$, тоді виконати: $p_i = 0$ і $s = s + p_i$.

Крок 6. Збільшити ймовірності включення в нові набори ознак, що входять у найкращий набір $X_{\min}$: $p_i = p_i + h/|X_{\min}|$ $(X_i \in X_{\min})$.

Крок 7. Перевірити критерії закінчення пошуку (досягнення прийнятного значення критерію оцінювання набору ознак $J(Xe)$, перевищення припустимої кількості ітерацій T і т. п.). У випадку, якщо такі критерії досягнуті, тоді виконати перехід до кроку 10.

Крок 8. Збільшити лічильник ітерацій: $t = t + 1$.

Крок 9. Одержати нову множину наборів ознак P_t, згенерував N наборів ознак відповідно до розподілу $\{p_1, p_2, \ldots, p_L\}$. Обчислити значення критерію оцінювання кожного набору ознак з P_t: $J(Xe)$, $Xe \in P_t$. Виконати перехід до кроку 3.

Крок 10. Зупинення.

Кількість наборів ознак N у кожній ітерації повинне бути по можливості мінімальним, але одночасно достатнім для того, щоб кращий набір $X_{\min}$ був близький до оптимального, а гірший $X_{\max}$ – до самого неінформативного набору ознак.

Параметр h, що задає ступінь адаптації, вибирається таким чином, щоб імовірність включення ознаки не могла стати нульовою. При $h = 0$ метод випадкового пошуку з адаптацією перетвориться до неадаптивного випадкового пошуку.

Перевага методу випадкового пошуку з адаптацією полягає в тому, що в більшості випадків він знаходить більш оптимальні набори ознак у порівнянні з методом почергового додавання й видалення ознак. Недоліком такого методу є досить повільна збіжність.

Для відбору інформативних ознак з вихідного масиву, що містить L ознак, за допомогою **методів еволюційного пошуку** рішення (хромосома) представляється бітовим рядком розміру L. Якщо біт хромосоми приймає одиничне значення, то відповідний йому ознака вважається інформативним і враховується при оцінюванні набору ознак, що відповідає хромосомі. У противному випадку, коли біт приймає нульове значення, ознака вважається неінформативним і не використовується при оцінюванні комбінації ознак.

Перевага такого подання полягає в тому, що класичні еволюційні оператори схрещування й мутації можуть бути застосовані для відбору ознак без внесення в них яких-небудь змін.

Для рішення задачі відбору інформативних ознак еволюційний пошук здійснюється шляхом виконання наступних кроків.

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$.

Крок 2. Згенерувати початкові комбінації ознак у вигляді хромосом – бітових рядків H_j розмірності L, де $j = 1, 2, \ldots, N$ – номер хромосоми в популяції; N – кількість згенерованих хромосом; L – кількість ознак.

Крок 3. Обчислити значення фітнес-функції хромосом $H_j \in P_t$. У якості фітнес-функції використовується значення критерію оцінювання набору ознак Xe_j, що відповідає хромосомі H_j.

Крок 4. Перевірити умови закінчення пошуку, у якості яких можуть бути використані: досягнення обмеження часу, кількості ітерацій, прийнятного значення критерію оцінювання набору ознак. Якщо критерії закінчення вдоволені, тоді перейти до кроку 9.

Крок 5. Збільшити лічильник ітерацій (часу): $t = t + 1$.

Крок 6. Згенерувати нові рішення шляхом застосування еволюційних операторів схрещування й мутації до особин поточного покоління.

Крок 7. Обчислити значення фітнес-функції хромосом $H_j \in P_t$.

Крок 8. Відібрати N кращих хромосом для переходу в наступне покоління. Виконати перехід до кроку 4.

Крок 9. Зупинення.

Перевагою еволюційного пошуку є те, що він має можливості для виходу з локальних оптимумів і пристосований для знаходження нових рішень за рахунок об'єднання кращих рішень, отриманих на різних ітераціях. Крім того, еволюційний пошук адаптується до особливостей цільової функції. Створені в процесі схрещування, нові рішення тестують усе більше широкі області простору ознак і переважно розташовуються в області оптимуму. Відносно рідкі мутації перешкоджають виродженню популяції, що рівносильне рідкому, але не припиняємому пошуку оптимуму у всіх інших областях простору ознак.

Недоліками еволюційного пошуку є відносно повільна збіжність і залежність від початкових умов пошуку.

У табл. 1.1 наведено порівняльну характеристика проаналізованих методів відбору ознак.

Таким чином, виділення максимально значимого набору ознак, як правило, виконується в наступній послідовності кроків.

Крок 1. Згенерувати початкове рішення.

Крок 2. Оцінити поточну комбінацію ознак.

Крок 3. Перевірити критерії закінчення пошуку. У випадку, якщо такі критерії задоволені, тоді виконати перехід до кроку 5.

Крок 4. Згенерувати нове рішення. Виконати перехід до кроку 2.

Крок 5. Зупинення.

Таблиця 1.1 – Порівняльна характеристика методів відбору ознак

	Критерій порівняння	Методи відбору ознак											
		Методи перебору					Евристичні методи			Методи ранжирування й кластеризації		Методи випадкового пошуку	
		Методи повного перебору			Методи скороченого перебору								
		Класичний повний перебір	Пошук у глибину	Пошук завширшки	Метод гілок і границь	МГВА	Метод послідовного додавання ознак	Метод послідовного видалення ознак	Метод почергового додавання та видалення ознак	Ранжирування ознак	Кластеризація ознак	Випадковий пошук з адаптацією	Еволюційний пошук
1	Оптимальність пошуку (оптимальність знайдених рішень)	Оптимальний	Оптимальний	Оптимальний	Субоптимальний	Субоптимальний	Субоптимальний	Субоптимальний	Субоптимальний	Субоптимальний	Субоптимальний	Субоптимальний	Субоптимальний
2	Кількість рішень на кожній ітерації	Одне	Одне	Одне	Одне	Декілька N	Одне	Одне	Одне	Одне	Одне	Декілька N	Декілька N
3	Детермінованість пошуку	Детермінований	Детермінований	Детермінований	Детермінований	Детермінований	Детермінований	Детермінований	Детермінований	Детермінований	Детермінований	Стохастичний	Стохастичний
4	Теоретичне обгрунтування	Високе	Високе	Високе	Середнє	Середнє	Середнє	Середнє	Середнє	Середнє	Середнє	Низьке	Середнє
5	Відносна складність реалізації	Низька	Низька	Низька	Середня	Середня	Середня	Середня	Середня	Низька	Середня	Середня	Висока
6	Принцип формування початкового рішення (початкова точка пошуку)	Порожня множина	Порожня множина	Порожня множина	Порожня множина	Декілька (N) одноознакових наборів	Порожня множина	Множина, що містить всі ознаки вихідного набору	Порожня множина	Порожня множина	Порожня множина	Декілька (N) випадково згенерованих або отриманих за певними правилами наборів	Декілька (N) випадково згенерованих або отриманих за певними правилами наборів
7	Критерії оцінювання інформативності ознак	Групові	Групові	Групові	Групові	Групові	Групові	Групові	Групові	Індивідуальні	Метрики відстані	Групові	Групові
8	Обчислювальна складність	$O(2^L)$	$O(2^L)$	$O(2^L)$	$O(L^2)$	$O(NL^2)$	$O(L^2)$	$O(L^2)$	$O(L^3)$	$O(L)$	$O(L^2)$	$O(NLT)$	$O(NLT)$

1.3 Структура методів виділення інформативної комбінації ознак

Виходячи з наведеної вище послідовності пошуку максимально значимого набору ознак, можуть бути виділені наступні параметри (процедури, компоненти, блоки) методів відбору ознак:

– початкова точка пошуку;

– процедура пошуку оптимального набору ознак;

– стратегія оцінювання набору ознак;

– критерії зупинення.

1.3.1 Початкова точка пошуку

Будь-який метод відбору ознак на етапі ініціалізації генерує початкову комбінацію ознак, з якої починається пошук набору ознак, що володіє максимальною інформативністю.

Як початкова точка пошуку можуть бути використані наступні множини:

– порожня множина. При цьому метод на кожній ітерації додає ознака (один або трохи), максимально поліпшуючий критерій оцінювання набору ознак;

– вихідний набір ознак, з якого ітеративно видаляються ознаки, виключення яких призводить до мінімального погіршення критерію оцінювання комбінації ознак;

– випадково згенерована або отримана за певними правилами множина.

1.3.2 Процедура пошуку оптимального набору ознак

Процедура пошуку оптимального набору ознак генерує новий набір ознак за певними правилами з метою оптимізації заданого критерію оцінювання комбінації ознак.

Найбільш відомими є наступні:

– методи перебору – послідовно генерують всі або більшість із $2^L - 1$ можливих комбінацій ознак. У випадку застосування ненадлишкових методів, які перебирають не всі можливі комбінації, генерація нових рішень відбувається за певними правилами, що відкидає малоефективні комбінації;

– евристичні методи – група методів, що використовують евристичні процедури для визначення напрямку пошуку;

– ранжирування ознак – підхід, при якому генерується одна комбінація ознак, що складає з ознак, що володіють максимальною індивідуальною значимістю;

– методи випадкового пошуку – генерують набори ознак випадковим або випадково спрямованим чином. Процедура випадкової генерації нових рішень триває доти, поки не буде знайдена комбінація ознак, що задовольняє заданим критеріям, або поки не буде досягнута максимально припустима кількість ітерацій.

1.3.3 Стратегії оцінювання набору ознак

Оцінювання комбінації ознак залежно від типу використовуваного критерію може бути здійснене за допомогою одного з наступних методів:

– фільтруючі методи (filters) – методи, при яких відбір ознак відбувається незалежно від побудови моделі;

– вбудовуючі методи (wrappers) – методи, які в процесі пошуку оптимальної комбінації ознак синтезують моделі на основі оцінюваємого ознакового набору;

– вбудовуємі методи (embedded methods) – методи, при яких процедура відбору ознак вбудовується в процедуру побудови оптимальної моделі.

Фільтруючі методи передбачають виключення неінформативних ознак з вихідного набору до побудови математичної моделі, що описує досліджуваний об’єкт або процес. При цьому використовується критерій оцінювання набору ознак, що не залежить від точності моделі, синтезованої на його основі.

Одним з переваг таких методів є те, що вони не мають потреби в повторному запуску якщо буде потреба синтезу нової моделі по вже відібраних ознаках.

Фільтри є обчислювально більш простими в порівнянні з іншими методами й ефективно можуть застосовуватися для відбору інформативних ознак з масивів даних дуже великого розміру.

Однак у результаті використання фільтруючих методів можуть бути отримані такі комбінації ознак, на основі яких не вдасться побудувати модель, що забезпечує необхідну точність. Це

викликано тим, що такі методи безпосередньо не пов'язані з математичною моделлю, що буде використовуватися для опису досліджуваного об'єкта, процесу або системи.

Критерії, використовувані для оцінювання ознак у фільтруючих методах, можуть бути класифіковані на:

– критерії оцінювання індивідуальної інформативності, застосовуємі при відборі ознак за допомогою ранжирування;

– критерії оцінювання спільного впливу набору ознак, використовуємі в більше складних пошукових процедурах.

Вбудовуючі методи оцінюють набір ознак за допомогою помилки прогнозування або класифікації по моделі, побудованої на основі ознак з аналізуємого набору. Як синтезовані моделі можуть використовуватися регресійні, нечіткологічні, нейромережеві, нейронечіткологічні та інші.

Використання помилок синтезованих моделей для оцінювання інформативності набору ознак є більше ресурсовмісткою процедурою, оскільки синтез математичних моделей на основі оцінюваної комбінації ознак займає значно більший час у порівнянні з оцінюванням ознак шляхом застосування критеріїв оцінювання спільного впливу ознак, використовуваних у фільтрах.

Як правило, такі методи приводять до кращих результатів у порівнянні з фільтруючими методами, оскільки вони орієнтовані на пошук інформативної комбінації ознак для конкретної моделі, що надалі буде застосовуватися на практиці.

Однак це приводить до зменшення гнучкості результатів у вигляді набору інформативних ознак. І у випадку ухвалення рішення про зміну типу моделі, використовуємої для опису досліджуваного об'єкта або процесу, необхідно буде запускати метод для повторного пошуку комбінації інформативних ознак, що відповідає новій моделі.

Вбудовуємі методи відбор, ознак впроваджуються в процес побудови оптимальної моделі. Прикладами таких методів можуть служити методи ID3, C4.5 і CART, що використовуються для побудови дерев вирішуючих правил. Вбудовуємі методи, при побудові дерев вирішуючих правил на кожному кроці використовують функцію оцінювання інформативності для вибору ознаки, по якій піде розбивка вихідної множини на підмножини.

1.3.4 Критерії зупинення

Критерії зупинення визначають умови закінчення пошуку. В якості таких критеріїв можуть бути використані наступні:

– досягнення заданого прийнятного значення критерію оцінювання набору ознак;

– неможливість генерації нового набору ознак, що поліпшує досягнуте на поточній ітерації значення оцінки комбінації ознак;

– перевищення максимально можливої кількості ітерацій або часу функціонування методу.

1.4 Критерії оцінювання інформативності ознак

Критерії, що використовуються для оцінювання інформативності ознак, можуть бути класифіковані на критерії, що дозволяють оцінити індивідуальну інформативність ознак, і критерії, що оцінюють спільний вплив деякого набору ознак на вихідний параметр.

1.4.1 Оцінювання індивідуальної інформативності

Критерії оцінювання індивідуальної інформативності застосовуються, як правило, при відборі ознак за допомогою ранжирування або на початкових етапах більше складних методів відбору ознак.

Індивідуальна інформативність ознак може бути оцінена за допомогою наступних критеріїв:

– коефіцієнт парної кореляції;

– коефіцієнт кореляції знаків;

– коефіцієнт кореляції Фехнера;

– дисперсійне відношення;

– коефіцієнт зв’язку;

– інформаційний критерій;

– теоретико-інформаційний критерій;

– ентропія ознаки;

– критерій, заснований на імовірнісному підході;

– критерій, заснований на статистичному підході.

Коефіцієнт парної кореляції застосовується у випадках, коли значення досліджуваної ознаки й вихідного параметра є безперервними.

Якщо ознака й вихідний параметр приймають дискретні значення, тоді для аналізу індивідуальної інформативності застосовують інші критерії. Такі критерії можуть бути використані також для аналізу інформативності безперервних ознак, однак значення таких ознак необхідно попередньо дискретизувати. Для дискретизації весь діапазон зміни ознаки X_i розбивається на N_i однакових інтервалів, де N_i залежить від кількості екземплярів у вибірці даних і, як правило, визначається за формулою:

$$N_i = \text{Ціле}(\log_2(m)) + 1,$$

де m – кількість екземплярів у вибірці.

Для відбору інформативних ознак при використанні критерію, заснованого на імовірнісному підході, або ентропії ознаки виконують пошук мінімуму, у випадку використання інших критеріїв виконують їхню максимізацію.

1. **Коефіцієнт парної кореляції** $r_{\text{п}}(X_i)$ ознаки X_i і вихідного параметра Y дозволяє оцінити наявність лінійного зв'язку між ними й розраховується за формулою:

$$r_{\text{п}}(X_i) = \sum_{p=1}^{m} \left(x_{ip} - \overline{x_i}\right)\left(y_p - \overline{y}\right) \Bigg/ \sqrt{\sum_{p=1}^{m} \left(x_{ip} - \overline{x_i}\right)^2 \sum_{p=1}^{m} \left(y_p - \overline{y}\right)^2},$$

де $\overline{x_i} = \frac{1}{m}\sum_{p=1}^{m} x_{ip}$ й $\overline{y} = \frac{1}{m}\sum_{p=1}^{m} y_p$ – середні значення ознаки X_i і відгуку Y.

Для оцінювання інформативності ознак використовується модуль значення коефіцієнта парної кореляції.

2. **Коефіцієнт кореляції знаків** $r_{\text{з}}(X_i)$ визначається за формулою:

$$r_{\text{з}}(X_i) = \frac{C^{+}_{x_i y} - C^{+}_{x_i} C^{+}_{y}}{\sqrt{C^{+}_{x_i} C^{+}_{y} \left(1 - C^{+}_{x_i}\right)\left(1 - C^{+}_{y}\right)}},$$

де $C^{+}_{x_i y}$ – кількість збігів позитивних знаків різниць $(x_{ip} - \overline{x_i})$ і $(y_p - \overline{y})$, поділене на m – кількість екземплярів у вибірці; $C^{+}_{x_i}$, C^{+}_{y} – частки від розподілу кількості позитивних знаків різниць $(x_{ip} - \overline{x_i})$ і $(y_p - \overline{y})$ на m для кожної змінної X_i і Y окремо.

3. **Коефіцієнт кореляції Фехнера** $r_{\Phi}(X_i)$ може бути розрахований таким чином:

$$r_{\Phi}(X_i) = \frac{C_i - D_i}{C_i + D_i} = \frac{2C_i}{m} - 1,$$

де C_i – кількість збігів однакових, як позитивних, так і негативних знаків різниць $(x_{i\,p} - \overline{x_i})$ і $(y_p - \overline{y})$ для i-ї ознаки відповідно; D_i – кількість розбіжностей знаків різниць $(x_{i\,p} - \overline{x_i})$ і $(y_p - \overline{y})$ для i-ї ознаки.

4. **Дисперсійне (кореляційне) відношення** $\eta(X_i)$ є мірою зв'язку вихідного параметра Y і ознаки X_i, що враховує зміну величини умовного математичного очікування $M(Y/X_i)$:

$$\eta(X_i) = \sqrt{\frac{DM(Y/X_i)}{DY}} = \sqrt{\frac{\frac{1}{m}\sum_{k=1}^{N_i} n_k (\overline{y}_k - \overline{y})^2}{\frac{1}{m}\sum_{p=1}^{m} (y_p - \overline{y})^2}} = \sqrt{\frac{\sum_{k=1}^{N_i} n_k (\overline{y}_k - \overline{y})^2}{\sum_{p=1}^{m} (y_p - \overline{y})^2}},$$

де $DM(Y/X_i)$ – дисперсія умовного математичного очікування $M(Y/X_i)$ – характеризує ту частину коливань значень вихідної змінної Y, що викликана впливом вхідної змінної X_i; n_k – кількість значень ознаки X_i, що потрапили в k-ий інтервал групування; N_i – кількість інтервалів з діапазону зміни i-ї ознаки, в які він може потрапити; $\overline{y}_k = \frac{1}{n_k}\sum_{j=1}^{n_k} y_{k,j}$ – середнє значення вихідного параметра для k-го інтервалу діапазону зміни i-ї ознаки; $y_{k,j}$ – j-е значення вихідного параметра Y за умови, якщо вхідна змінна X_i потрапить в k-ий інтервал; $\overline{y} = \frac{1}{m}\sum_{p=1}^{m} y_p$ – середнє значення вихідної змінної Y; DY – дисперсія вихідного параметра Y.

5. **Коефіцієнт зв'язку** $\xi(X_i)$ у порівнянні з дисперсійним відношенням дозволяє одержати більш об'єктивну оцінку інформативності ознаки X_i, оскільки характеризує залежність вихідного параметра Y від ознаки X_i не тільки в результаті зміни величини

умовного математичного очікування $M(Y/X_i)$, але й у результаті зміни величини умовної дисперсії $D(Y/X_i)$:

$$\xi(X_i) = \sqrt[4]{\frac{D\left(M\left(Y/X_i\right)\right)^2}{DY^2 - 2B}},$$

де $D\left(M\left(Y/X_i\right)\right)^2 = \frac{1}{m}\sum_{k=1}^{N_i} n_k\left(\left(\overline{y}_k\right)^2 - \overline{y^2}\right)^2$ – дисперсія квадрата умовного математичного очікування $M(Y/X_i)$; $\overline{y^2} = \frac{1}{m}\sum_{p=1}^{m} y_p^2$ – середнє значення квадрата вихідної змінної Y; $DY^2 = \frac{1}{m}\sum_{p=1}^{m}\left(y_p^2 - \overline{y^2}\right)^2$ – дисперсія квадрата вихідного параметра Y; $B = \operatorname{cov}\left(\left(M(Y/X_i)\right)^2, \left(D(Y/X_i)\right)\right) = \frac{1}{m}\sum_{k=1}^{N_i} n_k\left(\left(\overline{y}_k\right)^2 - \overline{y^2}\right)\left(\overline{y^2}_k - \left(\overline{y}_k\right)^2 - \frac{1}{m}\sum_{a=1}^{N_i} n_a\left(\overline{y^2}_a - \left(\overline{y}_a\right)^2\right)\right)$;

$\overline{y^2}_k = \frac{1}{n_k}\sum_{j=1}^{n_k} y_{k,j}^2$ – середнє значення квадрата вихідного параметра для k-го інтервалу діапазону зміни i-ої ознаки.

6. **Інформаційний критерій** $I(X_i)$ – передбачає використання кількості інформації, що одержує система в процесі розпізнавання об'єктів у результаті використання оцінюваної ознаки:

$$I\left(X_i\right) = -\sum_{l=1}^{N_y} p_l \log_2 p_l + \sum_{l=1}^{N_y} p_l \sum_{q=1}^{N_y}\sum_{k=1}^{N_i} p_{lk}\, p_{kq} \log_2 p_{kq},$$

де N_y – кількість інтервалів з діапазону зміни вихідного параметра y; $p_l = \frac{n_l}{m}$ – імовірність попадання значення вихідного параметра в l-ий інтервал діапазону його зміни; n_l – кількість значень вихідного параметра, що належать l-му інтервалу діапазону його зміни; $p_{lk} = p\left(X_i \in \left[x_{i,k}; x_{i,k+1}\right) \middle/ y \in \left[y_l; y_{l+1}\right)\right) = \frac{n_{lk}}{n_l}$ – умовна ймовірність попадання значення i-ої ознаки в k-ий інтервал діапазону його зміни за умови, що вихідний параметр y потрапить в l-ий інтервал; n_{lk} – кількість значень i-ої ознаки приналежних k-му ін-

тервалу діапазону його зміни за умови, що значення вихідного параметра y належить l-му інтервалу; $p_{kq} = \frac{n_{kq}}{n_k}$ – умовна ймовірність влучення значення вихідного параметра y в q-ий інтервал, за умови, що i-а ознака потрапить в k-ий інтервал; n_{kq} – кількість значень вихідного параметра y, що належать q-му інтервалу діапазону його зміни за умови, що значення i-ої ознаки належить k-му інтервалу діапазону його зміни; n_k – кількість значень i-ої ознаки, що належать k-му інтервалу діапазону її зміни.

Перевагою інформаційного підходу для оцінювання значимості ознак є наявність строгого математичного обґрунтування. Недолік обчислення інформаційного критерію складається в необхідності дискретизації ознак, що приймають безперервні значення.

7. **Теоретико-інформаційний критерій** $TI(X_i)$ – передбачає використання кількості інформації, що одержує система в процесі розпізнавання об'єктів у результаті використання оцінюваємої ознаки:

$$TI(X_i) = \sum_{a=1}^{N_y} \sum_{b=1}^{N_i} p_{ab} \log_2 \frac{p_{ab}}{p_a p_b},$$

де $p_{ab} = p\left(X_i \in [x_{i,b}; x_{i,b+1}), y \in [y_a; y_{a+1})\right) = \frac{n_{ab}}{m}$ – імовірність одночасного влучення значення i-ої ознаки в b-ий інтервал діапазону його зміни й вихідного параметра y потрапить в a-ий інтервал; n_{ab} – кількість екземплярів вибірки, для яких виконуються умови: $X_i \in [x_{i,b}; x_{i,b+1})$ і $y \in [y_a; y_{a+1})$; p_a – імовірність попадання значення вихідного параметра в a-ий інтервал діапазону її зміни; p_b – імовірність попадання значення i-ої ознаки в b-ий інтервал діапазону її зміни.

8. **Ентропія ознаки** $e(X_i)$ також використовує інформаційний підхід до визначення його значимості й розраховується за формулою:

$$e(X_i) = -\sum_{k=1}^{N_i} \left(p_k \sum_{l=1}^{N_y} p_{kl} \log_2 p_{kl} \right),$$

де $p_k = \frac{n_k}{m}$ – імовірність влучення значення i-ої ознаки в k-ий інтервал діапазону її зміни; n_k – кількість значень i-ої ознаки, що належать k-му інтервалу діапазону її зміни; N_i – кількість інтерва-

лів з діапазону зміни i-ої ознаки, у які вона може потрапити; N_y – кількість інтервалів з діапазону зміни вихідного параметра y; $p_{kl} = \frac{n_{kl}}{n_k}$ – умовна ймовірність влучення значення вихідного параметра y в l-ий інтервал, за умови, що i-а ознака потрапить в k-ий інтервал; n_{kl} – кількість значень вихідного параметра y, що належать l-му інтервалу діапазону його зміни за умови, що значення i-ї ознаки належить k-му інтервалу діапазону її зміни.

У теорії інформації передбачається, що значення ймовірностей станів систем точно відомі. Однак, як правило, ці ймовірності визначаються на основі статистичних даних і являють собою випадкові величини. Тому тільки при нескінченно великому обсязі вибірок їхнього значення можна вважати точними.

9. **Критерій, заснований на імовірнісному підході** $Z(X_i)$ заснований на тім, що ознаки можуть бути умовно підрозділені на дві групи.

До першої групи відносяться ознаки, значення яких незначно змінюються при переході від одного екземпляра даного класу до іншого об'єкта й досить помітно змінюються при переході від екземпляра одного класу до екземплярів інших класів. До другої групи відносяться ознаки, значення яких чутливі до переходів від одного екземпляра даного класу до іншого екземпляра й лише незначно змінюються при переходах від екземплярів одного класу до екземплярів інших класів.

Ознаки, що відносяться до першої групи, є більше інформативними в порівнянні з ознаками, що відносяться до другої групи.

Для оцінювання інформативності ознак використовують величину $Z(X_i)$:

$$Z(X_i) = \frac{M(d_{il})}{D(m_{il})} = \frac{\sum\limits_{l=1}^{N_y} d_{il} p_l}{\frac{1}{N_y}\sum\limits_{l=1}^{N_y}(m_{il} - M_i)^2} = \frac{\sum\limits_{l=1}^{N_y}\left(\frac{\sum\limits_{j=1}^{n_l}(x_{il,j} - m_{il})^2}{n_l} \cdot \frac{n_l}{m}\right)}{\frac{1}{N_y}\sum\limits_{l=1}^{N_y}(m_{il} - M_i)^2} = \frac{\frac{1}{m}\sum\limits_{l=1}^{N_y}\sum\limits_{j=1}^{n_l}(x_{il,j} - m_{il})^2}{\frac{1}{N_y}\sum\limits_{l=1}^{N_y}(m_{il} - M_i)^2},$$

де $M(d_{il})$ – математичне очікування дисперсії i-ої ознаки по класах. Очевидно, що чим менше величина $M(d_{il})$, тим більше значи-

мою є ознака X_i, оскільки невисокі значення $M(d_{il})$ характеризують більш компактне розташування екземплярів уздовж осі i-ої ознаки; $D(m_{il})$ – дисперсія математичного очікування розподілу i-ої ознаки при переході від класу до класу. Чим більше дисперсія $D(m_{il})$, тим далі уздовж осі i-ої ознаки розташовуються екземпляри, що відносяться до різних класів, що характеризує високу значимість i-ої ознаки; d_{il} – дисперсія i-ої ознаки за умови, що вихідний параметр y потрапить в l-ий інтервал діапазону своєї зміни;

$m_{il} = m\left(X_i / y \in [y_l; y_{l+1})\right) = \frac{1}{n_l} \sum_{j=1}^{n_l} x_{il,j}$ – математичне очікування i-ої ознаки за умови, що вихідний параметр y потрапить в l-ий інтервал діапазону своєї зміни; $x_{il,j}$ – j-е значення i-ої ознаки за умови, що вихідний параметр y потрапить в l-ий інтервал; n_l – кількість значень i-ої ознаки за умови, що вихідний параметр y потрапить в l-ий інтервал, $l = 1, 2, \ldots, N_y$; N_y – кількість інтервалів з діапазону зміни вихідного параметра y; $M_i = \frac{1}{m} \sum_{j=1}^{m} x_{ij}$ – математичне очікування i-ої ознаки; m – кількість екземплярів у вибірці.

10. **Критерій, заснований на статистичному підході** $S(X_i)$ використовується при наявності досить великого обсягу статистичних даних. У випадку нормального розподілу значень ознаки X_i у кожному класі критерій інформативності $S(X_i)$ може бути обчислений по формулі:

$$S(X_i) = \frac{1}{2}\left(\sigma_{j,1}^2 - \sigma_{j,2}^2\right)\left(\frac{1}{\sigma_{j,1}^2} - \frac{1}{\sigma_{j,2}^2}\right) + \frac{1}{2}\left(\frac{1}{\sigma_{j,1}^2} + \frac{1}{\sigma_{j,2}^2}\right)\left(\overline{x_{j,1}} - \overline{x_{j,2}}\right)^2,$$

де $\sigma_{j,1}$ і $\sigma_{j,2}$ – середньоквадратичні відхилення j-ої ознаки, за умови, що екземпляр ставиться до першого й другого класів, відповідно; $\overline{x_{j,1}}$ і $\overline{x_{j,2}}$ – середні значення j-ої ознаки, за умови, що екземпляр відноситься до першого й другого класів, відповідно.

Недоліками статистичного критерію є необхідність нормального розподілу в кожному класі, а також неможливість оцінювання інформативності при рішенні завдання класифікації для випадку декількох класів.

1.4.2 Критерії оцінювання спільного впливу набору ознак

Для оцінювання спільного впливу комбінації ознак залежно від використоємої стратегії оцінювання набору ознак застосовують:

– критерії, використовувані у фільтруючих методах;

– помилки синтезованих за допомогою оцінюваної комбінації ознак моделей.

1.4.2.1 Критерії, що використовуються у фільтруючих методах

У фільтруючих методах для оцінювання інформативності наборів ознак використовують наступні критерії:

– множинний коефіцієнт кореляції;

– коефіцієнт кореляції Пірсона;

– множинне дисперсійне відношення;

– множинний коефіцієнт зв'язку;

– інформаційний критерій;

– ентропія набору ознак;

– критерій, заснований на статистичному підході;

– критерій оцінювання набору ознак на основі теорії чітких множин;

– критерій, отриманий на основі теорії нечітких множини.

1. **Множинний коефіцієнт кореляції** $R(Xe)$ узагальнює поняття парної кореляції на багатомірний випадок і служить для виміру ступеня залежності між однією випадковою величиною Y і множиною випадкових величин Xe. Цей показник множинного зв'язку є аналогом абсолютної величини парного коефіцієнта кореляції й вимірює якість найкращого лінійного наближення:

$$r(Xe) = \frac{\sum_{p=1}^{m} \left(y_{\text{оц},p} - \bar{y}\right)^2}{\sum_{p=1}^{m} \left(y_p - \bar{y}\right)^2},$$

де $y_{\text{оц},p}$ – p-ий елемент вектора $Y_{\text{оц}} = A(A^T A)^{-1} A^{\mathrm{T}} Y$ оцінок значень вихідного параметра Y, обчисленого при припущенні, що залежність між Y і Xe є лінійною; A – матриця, перший стовпець якої містить одиничні значення, а інші значення визначаються в такий спосіб: $A(b,c) = Xe(b,c-1)$.

2. **Коефіцієнт кореляції Пірсона** $r_{\text{Пир}}(Xe)$ може бути розрахований за формулою:

$$r_{\text{Пир}}(Xe) = \frac{k\overline{r_y}}{\sqrt{k + k(k-1)\overline{r_x}}},$$

де $k = |Xe|$ – кількість ознак у розглянутому наборі Xe; $\overline{r_y}$ – середнє значення коефіцієнта кореляції між кожною ознакою й вихідним параметром; $\overline{r_x}$ – середнє значення коефіцієнта кореляції між ознаками.

3. **Множинне дисперсійне відношення** $\eta(Xe)$ ураховує зміну величини умовного математичного очікування $M(Y/Xe)$:

$$\eta(Xe) = \sqrt{\frac{DM(Y/Xe)}{DY}} = \sqrt{\frac{\frac{1}{m}\sum_{k=1}^{N_{Xe}} n_k\left(\overline{y}_k - \overline{y}\right)^2}{\frac{1}{m}\sum_{p=1}^{m}\left(y_p - \overline{y}\right)^2}} = \sqrt{\frac{\sum_{k=1}^{N_{Xe}} n_k\left(\overline{y}_k - \overline{y}\right)^2}{\sum_{p=1}^{m}\left(y_p - \overline{y}\right)^2}},$$

де $DM(Y/Xe)$ – дисперсія умовного математичного очікування $M(Y/Xe)$ – характеризує ту частину коливань значень вихідної змінної Y, що викликана впливом набору вхідних змінних Xe; $N_{Xe} = \sum_{l=1}^{|Xe|} N_l$ – кількість комбінацій інтервалів діапазону зміни значень набору ознак Xe. Ця величина визначається як добуток кількостей інтервалів розбивки для кожної ознаки, що входить у набір Xe; n_k – кількість значень набору ознак Xe, що потрапили в k-ий інтервал групування; $\overline{y}_k = \frac{1}{n_k}\sum_{j=1}^{n_k} y_{k,j}$ – середнє значення вихідного параметра для k-го інтервалу діапазону значень набору ознак Xe; $y_{k,j}$ – j-е значення вихідного параметра Y за умови, що значення набору ознак Xe потраплять в k-ий інтервал; $\overline{y} = \frac{1}{m}\sum_{p=1}^{m} y_p$ – середнє значення вихідної змінної Y; DY – дисперсія вихідного параметра Y.

4. **Множинний коефіцієнт зв'язку** $\xi(Xe)$ для оцінювання інформативності набору ознак *Xe* стосовно вихідного параметра *Y* за допомогою оцінки зміни умовного математичного очікування $M(Y/Xe)$ і умовної дисперсії $D(Y/Xe)$:

$$\xi(Xe) = \sqrt[4]{\frac{D(M(Y/Xe))^2}{DY^2 - 2B}},$$

де $D(M(Y/Xe))^2 = \frac{1}{m}\sum_{k=1}^{N_{Xe}} n_k\left(\left(\overline{y}_k\right)^2 - \overline{y^2}\right)^2$ – дисперсія квадрата умовного математичного очікування $M(Y/Xe)$; $\overline{y^2} = \frac{1}{m}\sum_{p=1}^{m} y_p^2$ – середнє значення квадрата вихідної змінної *Y*; $DY^2 = \frac{1}{m}\sum_{p=1}^{m}\left(y_p^2 - \overline{y^2}\right)^2$ – дисперсія квадрата вихідного параметра *Y*;

$$B = \operatorname{cov}\left(\left(M(Y/Xe)\right)^2, \left(D(Y/Xe)\right)\right) =$$

$$= \frac{1}{m}\sum_{k=1}^{Ne} n_k\left(\left(\overline{y}_k\right)^2 - \overline{y^2}\right)\left(\overline{y^2}_k - \left(\overline{y}_k\right)^2 - \frac{1}{m}\sum_{a=1}^{Ne} n_a\left(\overline{y^2}_a - \left(\overline{y}_a\right)^2\right)\right);$$

$\overline{y^2}_k = \frac{1}{n_k}\sum_{j=1}^{n_k} y_{k,j}^2$ – квадрат середнього значення вихідного параметра для *k*-го інтервалу діапазону зміни *i*-ої ознаки.

5. **Інформаційний критерій** $I(Xe)$ розраховується в такий спосіб:

$$I(Xe) = -\sum_{l=1}^{N_y} p_l \log_2 p_l + \sum_{l=1}^{N_y} p_l \sum_{q=1}^{N_y}\sum_{k=1}^{N_{Xe}} p_{lk} p_{kq} \log_2 p_{kq},$$

де N_y – кількість інтервалів з діапазону зміни вихідного параметра *y*; $p_l = \frac{n_l}{m}$ – імовірність попадання значення вихідного параметра в *l*-ий інтервал діапазону його зміни; n_l – кількість значень вихідного параметра, що належать *l*-му інтервалу діапазону його зміни; N_{Xe} – кількість комбінацій інтервалів діапазону зміни набору ознак *Xe*, у які можуть потрапити значення ознак набору *Xe*;

p_{lk} – умовна ймовірність влучення комбінації значень набору ознак Xe в k-ту комбінацію інтервалів діапазону зміни набору ознак за умови, що вихідний параметр y потрапить в l-ий інтервал; p_{kq} – умовна ймовірність влучення значення вихідного параметра y в q-ий інтервал, за умови, що комбінації значень набору ознак Xe потрапить в k-ту комбінацію інтервалів діапазону зміни набору ознак.

6. **Ентропія $e(Xe)$ набору ознак** Xe визначається за формулою

$$e(Xe) = -\sum_{k=1}^{N_{Xe}} p_k \left(\sum_{l=1}^{N_y} p_{kl} \log_2 p_{kl} \right),$$

де p_k – імовірність влучення комбінації значень набору ознак Xe в k-ту комбінацію інтервалів діапазону зміни набору ознак; N_{Xe} – кількість комбінацій інтервалів діапазону зміни набору ознак Xe, у які можуть потрапити значення ознак набору Xe; N_y – кількість інтервалів з діапазону зміни вихідного параметра y; p_{kl} – умовна ймовірність влучення значення вихідного параметра y в l-ий інтервал, за умови, що набір ознак Xe потрапить в k-ту комбінацію інтервалів діапазону зміни набору ознак.

7. **Критерій, заснований на статистичному підході** $S(Xe)$ може бути застосований за тих самих умов, що й аналогічний критерій, що використовується для оцінювання індивідуальної інформативності ознак. Інформативність сукупності ознак за допомогою статистичного критерію може бути визначена за формулою:

$$S(Xe) = \frac{1}{2}\mathrm{tr}\left(\left(V_1 - V_2\right)\left(V_1^{-1} - V_2^{-1}\right)\right) + \frac{1}{2}\mathrm{tr}\left(\left(V_1^{-1} + V_2^{-1}\right)\left(\overline{Xe}_1 - \overline{Xe}_2\right)\left(\overline{Xe}_1 - \overline{Xe}_2\right)^{\mathrm{T}}\right),$$

де $V_1 = \{v_{ij}\}_1$ і $V_2 = \{v_{ij}\}_2$ – коваріаційні матриці для випадків, коли екземпляр відноситься до першого й другого класів, відповідно; $v_{ij} = M\left(\left(x_i - \overline{x_i}\right)\left(x_j - \overline{x_j}\right)\right)$ – загальний елемент коваріаційної матриці, що прораховується окремо для кожного класу; M – оператор математичного очікування; V_1^{-1} і V_2^{-1} – зворотні матриці до V_1 та V_2, відповідно; $\mathrm{tr}(V)$ – слід матриці V, обчислений як сума діагональних елементів; $\overline{Xe} = \left(\overline{x_1}, \overline{x_2}, ..., \overline{x_{|Xe|}}\right)$ – вектор середніх значень ознак; A^{T} – матриця, транспонована до A.

8. **Критерій оцінювання набору ознак на основі теорії чітких множин** використовується для відбору ознак при рішенні завдання класифікації. Якщо буде потреба застосування такого критерію для виділення інформативної комбінації ознак при прогнозуванні значення ознак і вихідного параметра необхідно квантувати.

Критерій оцінювання набору ознак *Xe* на основі теорії чітких множин розраховується за формулою:

$$\gamma_Y(Xe) = \frac{|POS_{Xe}(Y)|}{|U|},$$

де U – кінцева множин екземплярів; $|U| = m$ – потужність множини U, рівна кількості екземплярів у вихідній вибірці; $|POS_{Xe}(Y)|$ – величина, що визначає кількість екземплярів, які можуть бути вірно класифіковані за допомогою набору ознак *Xe*. При цьому в множину $POS_{Xe}(Y)$ включаються номери вірно класифікуємих екземплярів за допомогою набору ознак *Xe*. Таким чином, у множину $POS_{Xe}(Y)$ не включаються номери тих екземплярів, які мають однакові значення ознак з набору *Xe*, але різні значення вихідного параметра.

Математично множина $POS_{Xe}(Y)$ може бути визначене в такий спосіб:

$$POS_{Xe}(Y) = \bigcup_{A \in U/Y} \underline{Xe}(A),$$

де U/Y = IND(Y) – множина, кожний l-ий елемент якої містить номера екземплярів, для яких значення класу вихідного параметра Y дорівнює l. $\underline{Xe}(A) = \{x | [a]_{Xe} \subseteq A\}$ – нижнє наближення множини A до множини Xe; $[a]_{Xe}$ – еквівалентні класи *Xe*-нерозрізненого зв'язку, тобто такі, які визначаються операцією IND(*Xe*):

$$\text{IND}(Xe) = U/Xe = \{(x, y) \in U^2 \mid \forall \text{a} \in Xe\ \text{a}(x) = \text{a}(y)\}.$$

Якщо $\gamma_Y(Xe) = 1$, тоді вихідний параметр Y повністю прогнозується (класифікується) за допомогою набору ознак *Xe*.

9. **Оцінка набору ознак *Xe* на основі теорії нечітких** множин може бути розрахована в такий спосіб:

$$\beta_Y(Xe) = \frac{\sum_{l=1}^{m} \mu_{POS_{Xe}(Y)}(l)}{|U|},$$

де U – кінцева множина екземплярів; $|U| = m$ – потужність множини U, рівна кількості екземплярів у вихідній вибірці; l – номер екземпляра; $\mu_{POS_{Xe}(Y)}(l) = \max\limits_{A=y_1,y_2,\ldots,y_m} \mu_{\underline{Xe}A}(l)$ – функція приналежності нечіткої множини $POS_{Xe}(Y)$;

$$\mu_{\underline{Xe}A}(l) = \max_{b=1,2,\ldots,B_{Xe}} \left[\min\left(\mu_{Xe_b}(l); \min_{a=1,2,\ldots,m} \max\left(1 - \mu_{Xe_b}(a); \mu_A(a)\right)\right)\right];$$

$B_{Xe} = \prod\limits_{X_i \in Xe} |X_i|$ – кількість різних комбінацій значень, які можуть приймати ознаки з множини Xe; Xe_b – b-та комбінація значень нечітких термів множини Xe.

1.4.2.2 Використання помилок синтезованих моделей

Залежно від типу розв'язуваної задачі в методах, які синтезують моделі для оцінювання спільного впливу набору ознак, використовуються наступні критерії:

– критерії оцінювання спільного впливу набору ознак у задачах прогнозування;

– критерії оцінювання спільного впливу комбінації ознак при класифікації.

Критерії оцінювання спільного впливу набору ознак у завданнях прогнозування

У випадку відбору ознак при рішенні задачі прогнозування для розрахунку помилки синтезованої моделі можуть бути використані:

– середньоквадратична помилка;

– сума квадратів відхилень;

– середня абсолютна помилка;

– сума значень абсолютних відхилень;

– максимальне абсолютне відхилення;

– середня відносна помилка;

– сума відносних відхилень;

– максимальне відносне відхилення.

1. **Середньоквадратична помилка** *MSE* (mean squared error performance function), що розраховується за формулою:

$$MSE(Xe) = \frac{1}{m}\sum_{p=1}^{m}\left(y^p - y_{Xe}^p\right)^2,$$

де y^p_{Xe} – значення вихідного параметра p-го екземпляра, розраховане по синтезованій моделі; y^p – реальне значення вихідного параметра p-го екземпляра; m – кількість екземплярів у вихідній вибірці даних.

2. **Сума квадратів відхилень** *SSE* (sum squared error performance function):

$$SSE(Xe) = m \cdot MSE(Xe) = \sum_{p=1}^{m}\left(y^p - y_{Xe}^p\right)^2.$$

3. **Середня абсолютна помилка** *MAE* (mean absolute error performance function) – характеризує абсолютні відхилення реальних значень вихідного параметра від значень, отриманих за допомогою синтезованої моделі:

$$MAE(Xe) = \frac{1}{m}\sum_{p=1}^{m}\left|y^p - y_{Xe}^p\right|.$$

4. **Сума значень абсолютних відхилень** *SAE* (sum absolute error performance function):

$$SAE(Xe) = m \cdot MAE(Xe) = \sum_{p=1}^{m}\left|y^p - y_{Xe}^p\right|.$$

5. **Максимальне абсолютне відхилення** *MaxAE* (maximum absolute error performance function):

$$MaxAE(Xe) = \max_{p=1,2,\ldots,m}\left|y^p - y_{Xe}^p\right|.$$

6. **Середня відносна помилка** *MARE* (mean absolute relative error performance function) або *MAPE* (mean absolute percentage error performance function) – оцінює відносні відхилення реальних значень вихідного параметра від значень, отриманих за допомогою синтезованої моделі:

$$MARE(Xe) = \frac{1}{m}\sum_{p=1}^{m}\left|\frac{y^p - y_{Xe}^p}{y^p}\right|, \quad y^p \neq 0.$$

7. **Сума відносних відхилень** *SRE* (sum relative error performance function):

$$SRE(Xe) = m \cdot MARE(Xe) = \sum_{p=1}^{m} \left| \frac{y^p - y^p_{Xe}}{y^p} \right|, \; y^p \neq 0.$$

8. **Максимальне відносне відхилення** *MaxRE* (maximum relative error performance function):

$$MaxRE(Xe) = \max_{p=1,2,\ldots,m} \left| \frac{y^p - y^p_{Xe}}{y^p} \right|, \; y^p \neq 0.$$

Оцінювання спільного впливу комбінації ознак при класифікації

При виділенні максимально інформативного набору ознак у випадку рішення завдання класифікації як критерій оцінювання інформативності набору ознак може бути використана **ймовірність прийняття помилкових рішень** *E* по моделі, що відповідає оцінюваної комбінації ознак:

$$E(Xe) = \frac{n}{m},$$

де *n* – кількість екземплярів вибірки, невірно класифікованих за допомогою побудованої моделі.

Крім імовірності прийняття помилкових рішень у завданнях класифікації для оцінювання інформативності набору ознак також може бути використаний **критерій Фишера**, що розраховується по формулах:

$$F(Xe) = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \text{ або } F(Xe) = \frac{|\mu_1 - \mu_2|}{\sigma_1^2 + \sigma_2^2},$$

де μ_1, μ_2 і σ^2_1, σ^2_2 – математичні очікування й дисперсії приналежності вихідного параметра *y* до першого й другого класу, відповідно, обчислені за допомогою моделі, синтезованої на основі набору ознак *Xe*.

У табл. 1.2 наведені результати порівняльного аналізу критеріїв оцінювання інформативності ознак, де д – дискретні значення, б – безперервні, бін – бінарні, неч – нечіткі.

Таблиця 1.2 – Порівняльна характеристика критеріїв оцінювання інформативності ознак

№	Критерій інформативності	Тип значень ознак	Тип значень вихідного параметра	Необхідність побудови моделі для оцінювання інформатив-ності	Можливість оцінювання наборів, що складаються з декількох ознак	Теоретичне обгрунтування	Відносна складність реалізації	Час обчислення
1	Коефіцієнт парної кореляції	б, д	б, д	немає	немає	високе	низька	низький
2	Коефіцієнт кореляції знаків	б, д	б, д	немає	немає	середнє	низька	низький
3	Коефіцієнт кореляції Фехнера	б, д	б, д	немає	немає	середнє	низька	низький
4	Дисперсійне відношення	д	б, д	немає	так	високе	середня	низький
5	Коефіцієнт зв'язку	д	б, д	немає	так	високе	середня	низький
6	Інформаційний критерій	д	д	немає	так	високе	висока	середній
7	Теоретико-інформаційний критерій	д	д	немає	так	високе	висока	середній
8	Ентропія	д	д	немає	так	високе	висока	середній
9	Критерій, заснований на імовірнісному підході	б, д	д	немає	немає	середнє	середня	низький
10	Критерій, заснований на статистичному підході	б, д	бін	немає	так	середнє	низька	низький
11	Множинний коефіцієнт кореляції	б, д	б, д	немає	так	високе	низька	низький
12	Коефіцієнт кореляції Пірсона	б, д	б, д	немає	так	середнє	низька	низький
13	Критерій, заснований на теорії чітких множин	д	д	немає	так	середнє	середня	низький
14	Критерій, заснований на теорії нечітких множин	неч	неч	немає	так	середнє	висока	середній
15	Середньоквадратична помилка	б, д	б, д	так	так	високе	висока	високий
16	Сума квадратів відхилень	б, д	б, д	так	так	високе	висока	високий
17	Середня абсолютна помилка	б, д	б, д	так	так	високе	висока	високий
18	Сума значень абсолютних відхилень	б, д	б, д	так	так	високе	висока	високий
19	Максимальне абсолютне відхилення	б, д	б, д	так	так	середнє	висока	високий
20	Середня відносна помилка	б, д	б, д	так	так	високе	висока	високий
21	Сума відносних відхилень	б, д	б, д	так	так	високе	висока	високий
22	Максимальне відносне відхилення	б, д	б, д	так	так	середнє	висока	високий
23	Імовірність прийняття помилкових рішень	б, д	д	так	так	високе	висока	високий
24	Критерій Фішера	б, д	бін	так	так	високе	висока	високий

1.4.3 Метрики, використовувані при кластеризації ознак

При кластеризації ознак застосовуються метрики відстані, що дозволяють оцінити не інформативність ознаки або набору ознак, а визначити міру близькості їхнього розташування по відношенню друг до друга в просторі екземплярів.

До метрик, використовуваним при кластеризації й дозволяючим оцінити відстань між ознаками x_a і x_b, що характеризуються m екземплярами, відносять наступні:

1) Евклідову відстань: $d = \left(\sum_{j=1}^{m} \left(x_{aj} - x_{bj} \right)^2 \right)^{\frac{1}{2}}$;

2) відстань Хеммінга (відстань по Манхеттену, метрика міських кварталів): $d = \sum_{j=1}^{m} \left| x_{aj} - x_{bj} \right|$;

3) відстань Мінковского (узагальнена відстань): $d = \left(\sum_{j=1}^{m} \left| x_{aj} - x_{bj} \right|^{v} \right)^{\frac{1}{v}}$, де v – деяке ціле число, що задається дослідником;

4) відстань у неізотропному просторі ознак: $d = \sum_{j=1}^{m} \left(\alpha_j \right)^2 \left(x_{aj} - x_{bj} \right)^2$, де α_j – коефіцієнт значимості j-го екземпляра;

5) діагностична міра відстані: $d = \left(\sum_{j=1}^{m} \left| x_{aj} - x_{bj} \right|^{v} \right)^{\frac{\mu}{v}}$, де v, μ – деякі цілі числа, що задаються дослідником;

6) узагальнена відстань у просторі ознак: $d_k = \left(\sum_{j=1}^{m} \alpha_{jk}^{v} \left| x_{aj} - x_{bj} \right|^{v} \right)^{\frac{\mu}{v}}$, де α_{jk} – коефіцієнт значимості j-го екземпляра для k-го класу;

7) відстань у нелінійному просторі: $d = \sum_{j=1}^{m} (\alpha_j)^p \left| (x_{aj})^p - (x_{bj})^p \right|$,

де p – ступінь нелінійності, призначена зменшити помилку класифікації (як правило, $p = 2, 3$);

8) узагальнена (зважене) відстань Махалонобіаса: $d = \sqrt{(\bar{x}_a - \bar{x}_b)^T \Lambda^T \Sigma^{-1} \Lambda (\bar{x}_a - \bar{x}_b)}$, де Σ – коваріаційна матриця генеральної сукупності ознак; Λ – деяка симетрична ненегативно визначена матриця вагових коефіцієнтів, що найчастіше вибирається діагональною;

9) відстань Камберра: $d = \sum_{j=1}^{m} \frac{\left| x_{aj} - x_{bj} \right|}{\left| x_{aj} + x_{bj} \right|}$;

10) відстань Чебишева: $d = \max_j \left| x_{aj} - x_{bj} \right|$;

11) кореляційна відстань: $d = \left(\sum_{j=1}^{m} x_{aj} x_{bj} \right) - \frac{1}{m} \left(\sum_{j=1}^{m} x_{aj} \right) \left(\sum_{j=1}^{m} x_{bj} \right)$;

12) кутова відстань:

$$d = \cos\gamma = \frac{x_a x_b}{\left| x_a \right\| x_b \right|} = \frac{\sum_{j=1}^{m} x_{aj} x_{bj}}{\sqrt{\left(\sum_{j=1}^{m} x_{aj}^2 \right) \left(\sum_{j=1}^{m} x_{bj}^2 \right)}}.$$

Вибір тої або іншої метрики при кластеризації ознак спирається на апріорні відомості про можливі особливості розподілу значень ознак в обраній сукупності екземплярів, а також залежить від результатів, отриманих раніше, і обмежень на обчислювальні витрати при проведенні експериментів.

1.5 Висновки за розділом

Виділення максимально значимої комбінації ознак є одним з найбільш важливих етапів при синтезі моделей розпізнаючих пристроїв, оскільки включення в модель малоінформативних ознак ускладнює процес побудови моделі, а також її подальше використання.

Проведене дослідження показало, що в цей час не існує єдиної постановки задачі відбору ознак. Визначено, що задача відбору ознак може бути задана в одній з наступних постановок:

– виділити комбінацію ознак, при якій досягається мінімум заданого критерію оцінювання набору ознак;

– знайти максимально інформативний набір ознак заданого розміру;

– визначити комбінацію ознак мінімального розміру, що забезпечує досягнення заданого значення критерію оцінювання значимості набору ознак.

Виконано порівняльний аналіз різних методів скорочення розмірності ознакового простору. Проаналізовані: метод повного перебору, пошук у глибину, пошук завширшки, метод галузей і границь, метод групового обліку аргументів, метод послідовного додавання ознак, метод послідовного видалення ознак, метод почергового додавання й видалення ознак, ранжирування ознак, кластеризація ознак, випадковий пошук з адаптацією й еволюційний пошук для відбору ознак.

Показано, що при рішенні задач прогнозування й класифікації, як правило, доводиться мати справи із системою статистично залежних ознак, інформативність набору яких не виражається через інформативність окремих ознак. Тому при рішенні практичних завдань необхідно оцінювати набір ознак, а не кожну ознаку окремо. Таким чином, використання ранжирування ознак по обчисленій індивідуальній оцінці є неприйнятним.

Застосування методу повного перебору вимагає оцінки всіх можливих комбінацій ознак, складених з вихідної сукупності, що унеможливлює використання такого підходу при великій кількості ознак у вихідному наборі, оскільки вимагає величезних обчислювальних витрат.

Методи евристичного пошуку недостатньо ефективні через неоптимальність жадібної стратегії пошуку, що послідовно додає або видаляє по одній ознаці, у результаті чого одержуваний набір ознак містить надлишкові ознаки, що корелюють із іншими ознаками в наборі. Крім того, при відборі ознак з набору, що має велику розмірність, евристичний пошук також вимагає значних витрат на оцінку наборів ознак.

Для пошуку комбінації інформативних ознак в умовах їхньої взаємної залежності друг від друга представляється доцільним вибрати еволюційний пошук, оскільки він більш пристосований для знаходження нових рішень за рахунок об'єднання кращих рішень, отриманих на різних ітераціях, має можливості для виходу з локальних оптимумів.

Досліджено структуру методів виділення комбінації ознак, що має найбільшу інформативність. Показано, що при побудові процедури відбору ознак необхідно визначити: початкову точку пошуку, процедуру пошуку оптимального набору ознак, стратегію оцінювання набору ознак і критерії зупинки.

Наведено критерії оцінювання інформативності ознак: критерії, що дозволяють оцінити індивідуальну інформативність ознак, і критерії, що оцінюють спільний вплив деякого набору ознак на вихідний параметр.

1.6 Приклади розрахунку критеріїв оцінювання інформативності ознак

***Приклад* 1**. Для заданої вибірки (табл. 1.3), ознаки й вихідний параметр якої приймають безперервні значення, обчислити значення:

а) коефіцієнта парної кореляції для ознак X_1, X_2 і X_4: $r_{\text{п}}(X_1)$, $r_{\text{п}}(X_2)$ і $r_{\text{п}}(X_4)$;

б) множинного коефіцієнта кореляції для набору ознак $Xe = \{X_1, X_2, X_4\}$: $r\,(Xe)$.

в) коефіцієнта кореляції Пірсона для набору ознак $Xe = \{X_1, X_2, X_4\}$: $r_{\text{Пир}}(Xe)$.

г) метрики, використовувані при кластеризації ознак: Евклідова відстань, відстань Хеммінга, відстань Мінковского (при $v = 3$), відстань у неізотропному просторі ознак (при $\alpha_1 = 0{,}3$, $\alpha_2 = 0{,}1$, $\alpha_3 = 0{,}4$, $\alpha_4 = 0{,}05$, $\alpha_5 = 0{,}15$), діагностичну міру відстані (при $v = 3$, $\mu = 2$), відстань у нелінійному просторі (при $p = 2$ і $\alpha_1 = 0{,}3$, $\alpha_2 = 0{,}1$, $\alpha_3 = 0{,}4$, $\alpha_4 = 0{,}05$, $\alpha_5 = 0{,}15$), відстань Камберра, відстань Чебишева, кореляційна відстань, кутова відстань між ознаками X_3 і X_5.

Таблиця 1.3 – Вибірка з безперервними значеннями ознак і вихідного параметра

№ екземпляра	X_1	X_2	X_3	X_4	X_5	X_6	Y
1	1,2	7,2	8,6	5,6	2,3	6,4	2,5
2	2,3	3,4	5,4	8,9	4,5	4,2	4,7
3	3,7	5,6	3,2	11,5	7,9	3,8	8,1
4	4,5	6,9	1,1	12,6	9,3	3,5	9,1
5	5,9	8,1	0,8	20,4	10,8	2,7	11,6

а) для розрахунку значень коефіцієнтів парної кореляції ознак X_1, X_2 і X_4 і вихідний параметри обчислимо попередньо середні значення $\overline{x_1}$, $\overline{x_2}$, $\overline{x_4}$, $\overline{y}$:

$$\overline{x_1} = \frac{1}{5}(1{,}2 + 2{,}3 + 3{,}7 + 4{,}5 + 5{,}9) = 3{,}52,$$

$$\overline{x_2} = \frac{1}{5}(7{,}2 + 3{,}4 + 5{,}6 + 6{,}9 + 8{,}1) = 6{,}24,$$

$$\overline{x_4} = \frac{1}{5}(5{,}6 + 8{,}9 + 11{,}5 + 12{,}6 + 20{,}4) = 11{,}8,$$

$$\overline{y} = \frac{1}{5}(2{,}5 + 4{,}7 + 8{,}1 + 9{,}1 + 11{,}6) = 7{,}2.$$

Розрахуємо значення коефіцієнтів парної кореляції ознак:

$$r_{\text{п}}(X_1) = \frac{(1{,}2-3{,}52)(2{,}5-7{,}2)+(2{,}3-3{,}52)(4{,}7-7{,}2)+\ldots+(5{,}9-3{,}52)(11{,}6-7{,}2)}{\sqrt{\left((1{,}2-3{,}52)^2+(2{,}3-3{,}52)^2+\ldots+(5{,}9-3{,}52)^2\right)\left((2{,}5-7{,}2)^2+(4{,}7-7{,}2)^2+\ldots+(11{,}6-7{,}2)^2\right)}} =$$

$$= \frac{26{,}45}{\sqrt{13{,}528 \cdot 52{,}12}} = 0{,}9961.$$

Аналогічно $r_{\text{п}}(X_2) = 0{,}435$ і $r_{\text{п}}(X_4) = 0{,}9484$. Оскільки $|\, r_{\text{п}}(X_1)\,| > |\, r_{\text{п}}(X_4)\,| > |\, r_{\text{п}}(X_2)\,|$, то можна стверджувати, що X_1 впливає на Y. При цьому перевіряється наявність лінійної залежності між X_i і Y;

б) розрахуємо значення множинного коефіцієнта кореляції для набору ознак $Xe = \{X_1, X_2, X_4\}$: $r\,(Xe)$. Для цього визначимо вектор $Y_{\text{оц}} = A(A^{\text{T}}A)^{-1}A^{\text{T}}Y$:

$$Y_{\text{оц}} = \begin{pmatrix} 1 & 1{,}2 & 7{,}2 & 5{,}6 \\ 1 & 2{,}3 & 3{,}4 & 8{,}9 \\ 1 & 3{,}7 & 5{,}6 & 11{,}5 \\ 1 & 4{,}5 & 6{,}9 & 12{,}6 \\ 1 & 5{,}9 & 8{,}1 & 20{,}4 \end{pmatrix} \cdot \left(\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1{,}2 & 2{,}3 & 3{,}7 & 4{,}5 & 5{,}9 \\ 7{,}2 & 3{,}4 & 5{,}6 & 6{,}9 & 8{,}1 \\ 5{,}6 & 8{,}9 & 11{,}5 & 12{,}6 & 20{,}4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1{,}2 & 7{,}2 & 5{,}6 \\ 1 & 2{,}3 & 3{,}4 & 8{,}9 \\ 1 & 3{,}7 & 5{,}6 & 11{,}5 \\ 1 & 4{,}5 & 6{,}9 & 12{,}6 \\ 1 & 5{,}9 & 8{,}1 & 20{,}4 \end{pmatrix} \right)^{-1} \times$$

$$\times \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1{,}2 & 2{,}3 & 3{,}7 & 4{,}5 & 5{,}9 \\ 7{,}2 & 3{,}4 & 5{,}6 & 6{,}9 & 8{,}1 \\ 5{,}6 & 8{,}9 & 11{,}5 & 12{,}6 & 20{,}4 \end{pmatrix} \cdot \begin{pmatrix} 2{,}5 \\ 4{,}7 \\ 8{,}1 \\ 9{,}1 \\ 11{,}6 \end{pmatrix} = \begin{pmatrix} 2{,}5015 \\ 4{,}8619 \\ 7{,}6822 \\ 9{,}3352 \\ 11{,}6192 \end{pmatrix}.$$

Обчислимо значення множинного коефіцієнта кореляції:

$$r(Xe) = \frac{(2{,}5015-7{,}2)^2 + (4{,}8619-7{,}2)^2 + (7{,}6822-7{,}2)^2 + (9{,}3352-7{,}2)^2 + (11{,}6192-7{,}2)^2}{(2{,}5-7{,}2)^2 + (4{,}7-7{,}2)^2 + (8{,}1-7{,}2)^2 + (9{,}1-7{,}2)^2 + (11{,}6-7{,}2)^2} =$$

$$= \frac{51{,}86}{52{,}12} = 0{,}99\ ;$$

в) для розрахунку значення коефіцієнта кореляції Пірсона для набору ознак $Xe = \{X_1, X_2, X_4\}$ спочатку визначимо значення k, $\overline{r_x}$ і $\overline{r_y}$.

Оскільки набір Xe складається із трьох ознак, то $k = 3$.

$\overline{r_y}$ – середнє значення коефіцієнта кореляції між кожною ознакою оцінюваного набору й вихідним параметром. Використовуючи отримані раніше дані, одержуємо:

$$\overline{r_y} = \frac{r_{\text{п}}(X_1) + r_{\text{п}}(X_2) + r_{\text{п}}(X_4)}{3} = \frac{0{,}9961 + 0{,}435 + 0{,}9484}{3} = 0{,}7932\ .$$

З метою розрахунку середнього значення коефіцієнта кореляції між ознаками $\overline{r_x}$ розрахуємо коефіцієнти парної кореляції між всіма ознаками в оцінюваному наборі Xe: $r_{\text{п}}(X_1; X_2)$, $r_{\text{п}}(X_1; X_4)$ і $r_{\text{п}}(X_2; X_4)$, де $r_{\text{п}}(X_i; X_j)$ – коефіцієнт кореляції між i-ою і j-ою ознаками в наборі Xe: $r_{\text{п}}(X_1; X_2) = 0{,}4621$, $r_{\text{п}}(X_1; X_4) = 0{,}9648$, $r_{\text{п}}(X_2; X_4) = 0{,}4727$. Таким чином:

$$\overline{r_x} = \frac{r_{\text{п}}(X_1; X_2) + r_{\text{п}}(X_1; X_4) + r_{\text{п}}(X_2; X_4)}{3} = \frac{0{,}4621 + 0{,}9648 + 0{,}4727}{3} = 0{,}6332\ .$$

Підставляючи знайдені значення k, $\overline{r_x}$ і $\overline{r_y}$ у формулу для розрахунку коефіцієнта кореляції Пірсона, одержуємо:

$$r_{\text{Пир}}(Xe) = \frac{k\overline{r_y}}{\sqrt{k + k(k-1)\overline{r_x}}} = \frac{3 \cdot 0{,}7932}{\sqrt{3 + 3(3-1) \cdot 0{,}6332}} = 0{,}9126;$$

г) розрахуємо різні відстані між ознаками X_3 і X_5:

– Евклідову відстань:

$$d = \sqrt{(8{,}6-2{,}3)^2 + (5{,}4-4{,}5)^2 + (3{,}2-7{,}9)^2 + (1{,}1-9{,}3)^2 + (0{,}8-10{,}8)^2} = 15{,}16;$$

– відстань Хеммінга:

$$d = |8{,}6-2{,}3| + |5{,}4-4{,}5| + |3{,}2-7{,}9| + |1{,}1-9{,}3| + |0{,}8-10{,}8| = 30{,}1;$$

– відстань Мінковського (при $v = 3$):

$$d = \left(|8{,}6-2{,}3|^3 + |5{,}4-4{,}5|^3 + |3{,}2-7{,}9|^3 + |1{,}1-9{,}3|^3 + |0{,}8-10{,}8|^3\right)^{\frac{1}{3}} = 12{,}4;$$

– відстань у неізотропному просторі ознак (при $\alpha_1 = 0{,}3$, $\alpha_2 = 0{,}1$, $\alpha_3 = 0{,}4$, $\alpha_4 = 0{,}05$, $\alpha_5 = 0{,}15$):

$$d = 0{,}3^2(8{,}6-2{,}3)^2 + 0{,}1^2(5{,}4-4{,}5)^2 + 0{,}4^2(3{,}2-7{,}9)^2 + 0{,}05^2(1{,}1-9{,}3)^2 +$$
$$+ 0{,}15^2(0{,}8-10{,}8)^2 = 39{,}19;$$

– діагностична міра відстані (при $v = 3$, $\mu = 2$):

$$d = \left(|8{,}6-2{,}3|^3 + |5{,}4-4{,}5|^3 + |3{,}2-7{,}9|^3 + |1{,}1-9{,}3|^3 + |0{,}8-10{,}8|^3\right)^{\frac{2}{3}} = 153{,}72;$$

– відстань у нелінійному просторі (при $p = 2$ і $\alpha_1 = 0{,}3$, $\alpha_2 = 0{,}1$, $\alpha_3 = 0{,}4$, $\alpha_4 = 0{,}05$, $\alpha_5 = 0{,}15$):

$$d = 0{,}3^2\left|8{,}6^2 - 2{,}3^2\right| + 0{,}1^2\left|5{,}4^2 - 4{,}5^2\right| + 0{,}4^2\left|3{,}2^2 - 7{,}9^2\right| + 0{,}05^2\left|1{,}1^2 - 9{,}3^2\right| +$$
$$+ 0{,}15^2\left|0{,}8^2 - 10{,}8^2\right| = 17{,}44;$$

– відстань Камберра:

$$d = \frac{|8{,}6-2{,}3| + |5{,}4-4{,}5| + |3{,}2-7{,}9| + |1{,}1-9{,}3| + |0{,}8-10{,}8|}{|8{,}6+2{,}3| + |5{,}4+4{,}5| + |3{,}2+7{,}9| + |1{,}1+9{,}3| + |0{,}8+10{,}8|} = 0{,}558;$$

– відстань Чебишева:

$$d = \max\left(|8{,}6-2{,}3|;\ |5{,}4-4{,}5|;\ |3{,}2-7{,}9|;\ |1{,}1-9{,}3|;\ |0{,}8-10{,}8|\right) = 10;$$

– кореляційна відстань:

$$d = 8{,}6 \cdot 2{,}3 + 5{,}4 \cdot 4{,}5 + 3{,}2 \cdot 7{,}9 + 1{,}1 \cdot 9{,}3 + 0{,}8 \cdot 10{,}8 -$$

$$-\frac{1}{5}(8{,}6 + 5{,}4 + 3{,}2 + 1{,}1 + 0{,}8)(2{,}3 + 4{,}5 + 7{,}9 + 9{,}3 + 10{,}8) = -44{,}71;$$

– кутова відстань:

$$d = \frac{8{,}6 \cdot 2{,}3 + 5{,}4 \cdot 4{,}5 + 3{,}2 \cdot 7{,}9 + 1{,}1 \cdot 9{,}3 + 0{,}8 \cdot 10{,}8}{\left(8{,}6^2 + 5{,}4^2 + 3{,}2^2 + 1{,}1^2 + 0{,}8^2\right)\left(2{,}3^2 + 4{,}5^2 + 7{,}9^2 + 9{,}3^2 + 10{,}8^2\right)} = 0{,}0026 \cdot$$

***Приклад* 2**. Для заданої вибірки (табл. 1.4), ознаки й вихідний параметр якої приймають дискретні значення, обчислити значення:

а) коефіцієнта кореляції знаків для ознак X_2 і X_4: $r_з(X_2)$ і $r_з(X_4)$;

б) коефіцієнта кореляції Фехнера для ознак X_2 і X_4: $r_Ф(X_2)$ і $r_Ф(X_4)$;

в) дисперсійного відношення для ознак X_1 і X_3: $\eta(X_1)$ і $\eta(X_3)$;

г) коефіцієнта зв'язку для ознаки X_3: $\xi(X_3)$;

д) інформаційного критерію для ознаки X_4: $I(X_4)$;

е) ентропії ознак X_2 і X_4: $e(X_2)$ і $e(X_4)$;

ж) ентропії набору ознак $Xe = \{X_2, X_4\}$: $e(Xe)$;

з) критерію, заснованого на імовірнісному підході, для ознак X_5 і X_6: $Z(X_5)$ і $Z(X_6)$;

и) критерію, заснованого на статистичному підході, для ознаки X_5: $S(X_5)$, вважаючи, що значення вихідного параметра для першого екземпляра дорівнює одиниці, а також ознаки по класах (1, 2, 1, 2, 0) і (0, 1, 2) розподілені нормально;

Таблиця 1.4 – Вибірка з дискретними значеннями ознак і вихідного параметра

№ екземпляра	X_1	X_2	X_3	X_4	X_5	X_6	Y
1	2	0	2,2	2	1	3	0
2	1	1	4,7	1	0	5	2
3	1	0	4	0	2	5	1
4	2	1	5,6	0	1	3	2
5	0	0	3,3	2	1	2	1
6	1	2	4,1	0	2	5	1
7	1	1	6,2	1	2	5	2
8	0	1	4,3	1	0	2	1

к) критерію оцінювання набору ознак $Xe = \{X_2, X_4\}$ на основі теорії чітких множин: $\gamma_Y(Xe)$;

Розв'язання:

а) для розрахунку значень коефіцієнтів кореляції знаків для ознак X_2 і X_4 визначимо попередньо значення $\overline{x_2}$, $\overline{x_4}$, $\overline{y}$, $C^+_{x_i y}$, $C^+_{x_i}$ і C^+_y: $\overline{x_2} = 0{,}75$, $\overline{x_4} = 0{,}875$, $\overline{y} = 1{,}25$, $C^+_{x_2 y} = 3/8 = 0{,}375$, $C^+_{x_2} = 5/8 = 0{,}625$, $C^+_y = 3/8 = 0{,}375$, $C^+_{x_4 y} = 2/8 = 0{,}25$, $C^+_{x_4} = 5/8 = 0{,}625$.

Таким чином:

$$r_з(X_2) = \frac{0{,}375 - 0{,}625 \cdot 0{,}375}{\sqrt{0{,}625 \cdot 0{,}375(1 - 0{,}625)(1 - 0{,}375)}} = 0{,}6,$$

$$r_з(X_4) = \frac{0{,}25 - 0{,}625 \cdot 0{,}375}{\sqrt{0{,}625 \cdot 0{,}375(1 - 0{,}625)(1 - 0{,}375)}} = 0{,}0667;$$

б) з метою розрахунку коефіцієнта кореляції Фехнера необхідно визначити C_i – кількість збігів однакових, як позитивних, так і негативних знаків різниць $(x_{ip} - \overline{x_i})$ і $(y_p - \overline{y})$ для i-ої ознаки: $C_2 = 6$, $C_4 = 4$. Тому:

$$r_Ф(X_2) = \frac{2 \cdot 6}{8} - 1 = 0{,}5,$$

$$r_Ф(X_2) = \frac{2 \cdot 4}{8} - 1 = 0;$$

в) для обчислення значення дисперсійного відношення $\eta(X_1)$ ознаки X_1 урахуємо, що він 2 рази приймає значення, рівне нулю, 4 рази дорівнює одиниці й 2 рази дорівнює двом.

При $X_1 = 0$ вихідний параметр два рази приймає значення, рівне одиниці. При $X_1 = 1$ вихідний параметр два рази приймає значення, рівне одиниці, і два рази дорівнює двом. При $X_1 = 2$ вихідний параметр один раз дорівнює нулю й один раз дорівнює двом.

Тому: $\overline{y}_1 = \overline{y}(X_1 = 0) = \frac{1+1}{2} = 1$, $\overline{y}_2 = \overline{y}(X_1 = 1) = \frac{2+1+1+2}{4} = 1{,}5$ та $\overline{y}_3 = \overline{y}(X_1 = 2) = \frac{0+2}{2} = 1$.

Підставляючи отримані значення у формулу для розрахунку дисперсійного відношення, і з огляду на, що $\bar{y} = 1{,}25$, одержуємо:

$$\eta(X_1) = \sqrt{\frac{2\cdot(1-1{,}25)^2 + 4\cdot(1{,}5-1{,}25)^2 + 2\cdot(1-1{,}25)^2}{(0-1{,}25)^2 + (2-1{,}25)^2 + (1-1{,}25)^2 + (2-1{,}25)^2 + \ldots + (1-1{,}25)^2}} = \sqrt{\frac{0{,}5}{3{,}5}} = 0{,}38.$$

Для розрахунку дисперсійного відношення $\eta(X_3)$ ознаки X_3 необхідно попередньо її дискретизувати. Для цього визначимо кількість інтервалів розбивки його значень:

$$N_3 = \text{Ціле}(\log_2(m)) + 1 = \text{Ціле}(\log_2(8)) + 1 = 4,$$

і довжину кожного інтервалу:

$$\Delta = \frac{\max(X_3) - \min(X_3)}{N_3} = \frac{6{,}2 - 2{,}2}{4} = 1.$$

Таким чином, одержуємо 4 інтервали: [2,2; 3,2), [3,2; 4,2), [4,2; 5,2) і [5,2; 6,2].

В інтервал [2,2; 3,2) попадає одне значення X_3 ($X_3 = 2{,}2$), при цьому $Y = 0$. В інтервал [3,2; 4,2) попадає три значення X_3 ($X_3 = 3{,}3$; 4; 4,1), при цьому вихідний параметр три рази дорівнює одиниці. В інтервал [4,2; 5,2) попадає два значення X_3 ($X_3 = 4{,}3$; 4,7), при цьому вихідний параметр одного разу дорівнює одиниці й один раз дорівнює двом. В інтервал [5,2; 6,2] попадає два значення X_3 ($X_3 = 5{,}6$; 6,2), при цьому вихідний параметр два рази дорівнює двом.

Тому:

$$\bar{y}_1 = \bar{y}(X_3 \in [2{,}2;3{,}2)) = \frac{0}{1} = 0,$$

$$\bar{y}_2 = \bar{y}(X_3 \in [3{,}2;4{,}2)) = \frac{1+1+1}{3} = 1,$$

$$\bar{y}_3 = \bar{y}(X_3 \in [4{,}2;5{,}2)) = \frac{1+2}{2} = 1{,}5,$$

$$\bar{y}_4 = \bar{y}(X_3 \in [5{,}2;6{,}2]) = \frac{2+2}{2} = 2.$$

Таким чином:

$$\eta(X_3)=\sqrt{\frac{1\cdot(0-1{,}25)^2+3\cdot(1-1{,}25)^2+2\cdot(1{,}5-1{,}25)^2+2\cdot(2-1{,}25)^2}{(0-1{,}25)^2+(2-1{,}25)^2+(1-1{,}25)^2+(2-1{,}25)^2+\ldots+(1-1{,}25)^2}}=\sqrt{\frac{3}{3{,}5}}=0{,}93;$$

г) при розрахунку коефіцієнта зв'язку $\xi(X_3)$ для ознаки X_3 скористаємося результатами, отриманими вище, а також обчислимо середнє значення квадрата $\overline{y^2}$ вихідної змінної Y і середні значення квадратів $\overline{y^2}_k$ вихідного параметра для k-го інтервалу діапазону зміни третьої ознаки.

Середнє значення квадрата вихідної змінної Y:

$$\overline{y^2}=\frac{1}{m}\sum_{p=1}^{m}y_p^2=\frac{1}{8}\left(0^2+2^2+1^2+2^2+1^2+1^2+2^2+1^2\right)=2\,.$$

Середні значення квадратів $\overline{y^2}_k$ вихідного параметра для k-го інтервалу:

$$\overline{y^2}_1=\overline{y^2}\left(X_3\in[2{,}2;3{,}2)\right)=\frac{0^2}{1}=0\,,$$

$$\overline{y^2}_2=\overline{y^2}\left(X_3\in[3{,}2;4{,}2)\right)=\frac{1^2+1^2+1^2}{3}=1\,,$$

$$\overline{y^2}_3=\overline{y^2}\left(X_3\in[4{,}2;5{,}2)\right)=\frac{1^2+2^2}{2}=2{,}5\,,$$

$$\overline{y^2}_4=\overline{y^2}\left(X_3\in[5{,}2;6{,}2]\right)=\frac{2^2+2^2}{2}=4\,.$$

Дисперсія квадрата умовного математичного очікування:

$$D(M(Y/X_i))^2=\frac{1}{m}\sum_{k=1}^{N_i}n_k\left(\left(\overline{y}_k\right)^2-\overline{y^2}\right)^2=\frac{1}{8}\left(1\cdot\left(0^2-2\right)^2+3\cdot\left(1^2-2\right)^2+2\cdot\left(1{,}5^2-2\right)^2+2\cdot\left(2^2-2\right)^2\right)=1{,}391.$$

Дисперсія квадрата вихідного параметра Y:

$$DY^2=\frac{1}{8}\left(\left(0^2-2\right)^2+\left(2^2-2\right)^2+\left(1^2-2\right)^2+\left(2^2-2\right)^2+\left(1^2-2\right)^2+\left(1^2-2\right)^2+\left(2^2-2\right)^2+\left(1^2-2\right)^2\right)=2{,}5.$$

$$A=\frac{1}{8}\left(1\cdot\left(0-0^2\right)+3\cdot\left(1-1^2\right)+2\cdot\left(2{,}5-1{,}5^2\right)+2\cdot\left(4-2^2\right)\right)=0{,}0625.$$

$$B = \frac{1}{8}\left(1\cdot\left(0^2-2\right)\left(0-0^2-0{,}0625\right)+3\cdot\left(1^2-2\right)\left(1-1^2-0{,}0625\right)+\right.$$
$$\left.+2\cdot\left(1{,}5^2-2\right)\left(2{,}5-1{,}5^2-0{,}0625\right)+2\cdot\left(2^2-2\right)\left(4-2^2-0{,}0625\right)\right)=0{,}15625.$$

Підставляючи отримані значення у формулу для розрахунку коефіцієнта зв'язку, одержуємо:

$$\xi\left(X_3\right)=\sqrt[4]{\frac{1{,}391}{2{,}5-2\cdot 0{,}15625}}=0{,}893;$$

д) для розрахунку значення інформаційного критерію для ознаки X_4 необхідно врахувати, що він приймає три різних значення: 0, 1 і 2. При цьому ознака X_4 три рази дорівнює нулю, три рази – одиниці й два рази – двом. Вихідний параметр Y один раз приймає нульове значення, чотири рази дорівнює одиниці й три рази приймає значення, рівне двом.

Визначимо безумовні й умовні ймовірності різних подій, необхідні для обчислення значення інформаційного критерію:

– безумовні ймовірності віднесення вихідного параметра до класів 0, 1 і 2:

$$p_1 = p\left(Y=0\right)=\frac{1}{8};\qquad p_2 = p\left(Y=1\right)=\frac{4}{8};\qquad p_3 = p\left(Y=2\right)=\frac{3}{8};$$

– імовірності того, що ознака X_4 прийме у k-те з можливих значень за умови, що вихідний параметр y прийме l-те значення:

$$p\left(X_4=0/Y=0\right)=0;\qquad p\left(X_4=0/Y=1\right)=\frac{2}{4};\qquad p\left(X_4=0/Y=2\right)=\frac{1}{3};$$

$$p\left(X_4=1/Y=0\right)=0;\qquad p\left(X_4=1/Y=1\right)=\frac{1}{4};\qquad p\left(X_4=1/Y=2\right)=\frac{2}{3};$$

$$p\left(X_4=2/Y=0\right)=1;\qquad p\left(X_4=2/Y=1\right)=\frac{1}{4};\qquad p\left(X_4=2/Y=2\right)=0;$$

– імовірності того, що вихідний параметр y прийме q-*е* можливе значення за умови, що ознака X_4 прийме в k-*е* з можливих значень:

$$p\left(Y=0/X_4=0\right)=0;\qquad p\left(Y=0/X_4=1\right)=0;\qquad p\left(Y=0/X_4=2\right)=\frac{1}{2};$$

$$p(Y=1/X_4=0)=\frac{2}{3};\qquad p(Y=1/X_4=1)=\frac{1}{3};\qquad p(Y=1/X_4=2)=\frac{1}{2};$$

$$p(Y=2/X_4=0)=\frac{1}{3};\qquad p(Y=2/X_4=1)=\frac{2}{3};\qquad p(Y=2/X_4=2)=0.$$

Таким чином:

$$I(X_4)=-\left(\frac{1}{8}\log_2\frac{1}{8}+\frac{4}{8}\log_2\frac{4}{8}+\frac{3}{8}\log_2\frac{3}{8}\right)+\frac{1}{8}\left(0+0+1\cdot\left[\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{2}\log_2\frac{1}{2}+0\right]\right)+$$

$$+\frac{4}{8}\left(\frac{2}{4}\left[0+\frac{2}{3}\log_2\frac{2}{3}+\frac{1}{3}\log_2\frac{1}{3}\right]+\frac{1}{4}\left[0+\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}\right]+\frac{1}{4}\left[\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{2}\log_2\frac{1}{2}+0\right]\right)+$$

$$+\frac{3}{8}\left(\frac{1}{3}\left[0+\frac{2}{3}\log_2\frac{2}{3}+\frac{1}{3}\log_2\frac{1}{3}\right]+\frac{2}{3}\left[0+\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}\right]+0\right)=$$

$$=1{,}4056+\frac{1}{8}\cdot(-1)+\frac{4}{8}\cdot\left(\frac{2}{4}\cdot(-0{,}9183)+\frac{1}{4}\cdot(-0{,}9183)+\frac{1}{4}\cdot(-1)\right)+$$

$$+\frac{3}{8}\cdot\left(\frac{1}{3}\cdot(-0{,}9183)+\frac{2}{3}\cdot(-0{,}9183)\right)=1{,}4056-0{,}125-0{,}4069-0{,}3444=0{,}5293;$$

е) при розрахунку значення ентропії ознаки X_2 врахуємо, що вона приймає 3 різні значення (0, 1 і 2). При цьому ознака X_2 три рази приймає нульове значення, чотири рази дорівнює одиниці й один раз – двом. Тому:

$$e(X_2)=-\left(\frac{3}{8}\left(\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}+0\right)+\frac{4}{8}\left(0+\frac{1}{4}\log_2\frac{1}{4}+\frac{3}{4}\log_2\frac{3}{4}\right)+\frac{1}{8}\left(0+\frac{1}{1}\log_2\frac{1}{1}+0\right)\right)=$$

$$=-\left(\frac{1}{8}\log_2\frac{1}{3}+\frac{1}{4}\log_2\frac{2}{3}+\frac{1}{8}\log_2\frac{1}{4}+\frac{3}{8}\log_2\frac{3}{4}\right)=0{,}75.$$

Аналогічно міркуючи, одержуємо значення ентропії ознаки X_4:

$$e(X_4)=-\left(\frac{3}{8}\left(0+\frac{2}{3}\log_2\frac{2}{3}+\frac{1}{3}\log_2\frac{1}{3}\right)+\frac{3}{8}\left(0+\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}\right)+\frac{2}{8}\left(\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{2}\log_2\frac{1}{2}+0\right)\right)=$$

$$=-\left(\frac{1}{8}\log_2\frac{1}{3}+\frac{1}{4}\log_2\frac{2}{3}+\frac{1}{8}\log_2\frac{1}{4}+\frac{3}{8}\log_2\frac{3}{4}\right)=0{,}9387;$$

ж) для обчислення значення ентропії набору ознак $Xe = \{X_2, X_4\}$ необхідно врахувати, що набір ознак $\{X_2, X_4\}$ два рази приймає комбінацію значень $\{0, 2\}$, три рази – комбінацію $\{0, 0\}$, по одному разу – комбінації значень $\{0, 0\}$, $\{1, 0\}$ і $\{2, 0\}$. Тоді:

$$e(Xe)=-\left(\frac{2}{8}\left(\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{2}\log_2\frac{1}{2}+0\right)+\frac{3}{8}\left(0+\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}\right)+0+0+0\right)=0{,}59436;$$

з) оцінимо інформативність ознак X_5 і X_6 за допомогою імовірнісного підходу $Z(X_5)$ і $Z(X_6)$.

Вихідний параметр Y приймає 3 різні значення (0, 1 і 2), тому $N_y = 3$.

Визначимо безумовне й умовні математичні очікування ознаки X_5:

$$M_5 = M(X_5) = \frac{1+0+2+1+1+2+2+0}{8} = 1{,}125\,,\ m_{51} = m(X_5/y=0) = \frac{1}{1} = 1\,,$$

$$m_{52} = m(X_5/y=1) = \frac{2+1+2+0}{4} = 1{,}25$$

$$\text{та } m_{53} = m(X_5/y=2) = \frac{0+1+2}{3} = 1\,.$$

Підставляючи отримані значення у формулу для оцінювання інформативності за допомогою імовірнісного підходу, одержуємо:

$$Z(X_5) = \frac{\frac{1}{8}\left((1-1)^2+(2-1{,}25)^2+(1-1{,}25)^2+(2-1{,}25)^2+(0-1{,}25)^2+(0-1)^2+(1-1)^2+(2-1)^2\right)}{\frac{1}{3}\left((1-1{,}125)^2+(1{,}25-1{,}125)^2+(1-1{,}125)^2\right)} = 38.$$

Аналогічно знаходимо інформативність шостої ознаки $Z(X_6)$:

$$M_6 = M(X_6) = \frac{3+5+5+3+2+5+5+2}{8} = 3{,}75\,,$$

$$m_{61} = m(X_6/y=0) = \frac{3}{1} = 3\,,\ m_{62} = m(X_6/y=1) = \frac{5+2+5+2}{4} = 3{,}5$$

$$\text{та } m_{63} = m(X_6/y=2) = \frac{5+3+5}{3} = 4{,}33\,.$$

$$Z(X_5) = \frac{\frac{1}{8}\left((3-3)^2+(5-3{,}5)^2+(2-3{,}5)^2+(5-3{,}5)^2+(2-3{,}5)^2+(5-4{,}33)^2+(3-4{,}33)^2+(5-4{,}33)^2\right)}{\frac{1}{3}\left((3-3{,}75)^2+(3{,}5-3{,}75)^2+(4{,}33-3{,}75)^2\right)} = 4{,}53.$$

Оскільки $Z(X_6) < Z(X_5)$, то при використанні імовірнісного підходу можна зробити висновок про те, що ознака X_6 є більш інформативною у порівнянні з ознакою X_5;

и) оцінимо інформативність ознаки X_5 за допомогою критерію, заснованого на статистичному підході: $S(X_5)$. Будемо думати, що для 1-ий, 3-ий, 5-ий, 6-ий і 8-ий екземпляри відносяться до першого класу, а 2-ий, 4-ий, і 7-ий екземпляри – до другого класу.

Визначимо середні значення п'ятої ознаки, за умови, що екземпляр відноситься до першого й другого класів:

$$\overline{X_{5,1}} = \frac{1+2+1+2+0}{5} = 1{,}2 \text{ та } \overline{X_{5,2}} = \frac{0+1+2}{3} = 1 .$$

Розрахуємо квадрати середньоквадратичних відхилень ознаки X_5, за умови, що екземпляр відноситься до першого й другого класів:

$$\sigma^2_{5,1} = D(X_5/Y=1) = \frac{1}{5}\left((1-1{,}2)^2 + (2-1{,}2)^2 + (1-1{,}2)^2 + (2-1{,}2)^2 + (0-1{,}2)^2\right) = 0{,}56 ,$$

$$\sigma^2_{5,2} = D(X_5/Y=2) = \frac{1}{3}\left((0-1)^2 + (1-1)^2 + (2-1)^2\right) = 0{,}667 .$$

Інформативність ознаки X_5 за допомогою критерію, заснованого на статистичному підході:

$$S(X_5) = \frac{1}{2}(0{,}56 - 0{,}667)\left(\frac{1}{0{,}56} - \frac{1}{0{,}667}\right) + \frac{1}{2}\left(\frac{1}{0{,}56} + \frac{1}{0{,}667}\right)(1{,}2-1) = 0{,}344 ;$$

к) при використанні набору ознак $Xe = \{X_2, X_4\}$ однозначно можуть бути класифіковані третій, четвертий і шостий екземпляри. Тому: $POS_{Xe}(Y) = \{3, 4, 6\}$, $|POS_{Xe}(Y)| = 3$. Номера інших екземплярів не ввійшли в множину $POS_{Xe}(Y)$, оскільки комбінація з ознак X_2 і X_4 не дозволяє їх вірно класифікувати. Так, наприклад, перший і п'ятий екземпляри мають однакові значення ознак $x_{21} = x_{25} = 0$ і $x_{41} = x_{45} = 5$, але різні значення ознак: $y_1 = 0$, $y_5 = 1$. Таким чином:

$$\gamma_Y(Xe) = \frac{|POS_{Xe}(Y)|}{m} = \frac{3}{8} = 0{,}375 .$$

***Приклад* 3**. Задано вибірку, що складається з дев'яти екземплярів, що характеризуються трьома ознаками, що приймають нечіткі значення. При цьому ознака X_1 може приймати значення x_{11}, x_{12}, x_{13}, $X_2 - x_{21}, x_{22}, x_{23}$, $X_3 - x_{31}, x_{32}$, вихідний параметр Y приймає значення y_1, y_2, y_3. У табл. 1.5 наведені значення ймовірності того, що i-та ознака j-го екземпляра прийме значення x_{ijd}.

Таблиця 1.5 – Вибірка з нечіткими значеннями ознак і вихідного параметра

№ екземпляра	X_1			X_2			X_3		Y		
	x_{11}	x_{12}	x_{13}	x_{21}	x_{22}	x_{23}	x_{31}	x_{32}	y_1	y_2	y_3
1	0,3	0,7	0	0,2	0,7	0,1	0,3	0,7	0,1	0,9	0
2	1	0	0	1	0	0	0,7	0,3	0,8	0,2	0
3	0	0,3	0,7	0	0,7	0,3	0,6	0,4	0	0,2	0,8
4	0,8	0,2	0	0	0,7	0,3	0,2	0,8	0,6	0,3	0,1
5	0,5	0,5	0	1	0	0	0	1	0,6	0,8	0
6	0	0,2	0,8	0	1	0	0	1	0	0,7	0,3
7	1	0	0	0,7	0,3	0	0,2	0,8	0,7	0,4	0
8	0,1	0,8	0,1	0	0,9	0,1	0,7	0,3	0	0	1
9	0,3	0,7	0	0,9	0,1	0	1	0	0	0	1

Необхідно розрахувати значення критерію оцінювання інформативності ознак X_1, X_2, X_3, а також набору ознак $Xe = \{X_1, X_2\}$ на основі теорії нечітких множин: $\beta_Y(X_1)$, $\beta_Y(X_2)$, $\beta_Y(X_3)$ і $\beta_Y(Xe)$.

Розрахуємо значення критерію інформативності нечіткої ознаки X_1 за формулою:

$$\beta_Y(X_1) = \frac{\sum_{l=1}^{m} \mu_{POS_{X_1}(Y)}(l)}{m}.$$

Визначимо значення функції приналежності нечіткої множини $POS_{X1}(Y)$ першого екземпляра $\mu_{POS_{Xe}(Y)}(1) = \max\left(\mu_{X_1y_1}(1); \mu_{X_1y_2}(1); \mu_{X_1y_3}(1)\right)$.

$$\mu_{X_1y_1}(1) = \max_{b=1,2,3}\left[\min\left(\mu_{X_{1b}}(1); \min_{a=1,2,\ldots,9}\max\left(1-\mu_{X_{1b}}(a); \mu_{y_1}(a)\right)\right)\right].$$

При $b = 1$:

$$\min_{a=1,2,\ldots,9}\max\left(1-\mu_{x_{11}}(a); \mu_{y_1}(a)\right) = \min\left(\max(1-0{,}3;0{,}1); \max(1-1;0{,}8); \max(1-0;0);\right.$$
$$\left.\max(1-0{,}8;0{,}6); \max(1-0{,}5;0{,}6); \max(1-0;0); \max(1-1;0{,}7); \max(1-0{,}1;0); \max(1-0{,}3;0)\right) =$$
$$= \min(0{,}7;0{,}8;1;0{,}6;0{,}7;1;0{,}7;0{,}9;0{,}7;) = 0{,}6.$$

Аналогічно при $b = 2$: $\min\limits_{a=1,2,\ldots,9} \max\left(1-\mu_{x_{12}}(a);\mu_{y_1}(a)\right)=0{,}2$, при $b = 3$: $\min\limits_{a=1,2,\ldots,9} \max\left(1-\mu_{x_{13}}(a);\mu_{y_1}(a)\right)=0{,}2$.

Таким чином:

$$\mu_{X_1y_1}(1)=\max\left(\min\left(\mu_{x_{11}}(1);0{,}6\right);\min\left(\mu_{x_{12}}(1);0{,}2\right);\min\left(\mu_{x_{13}}(1);0{,}2\right)\right)=$$
$$=\max\left(\min(0{,}3;0{,}6);\min(0{,}7;0{,}2);\min(0;0{,}2)\right)=0{,}3.$$

Аналогічно $\mu_{X_1y_2}(1)=0{,}2$ й $\mu_{X_1y_3}(1)=0{,}3$.

Тому $\mu_{POS_{Xe}(Y)}(1)=\max(0{,}3;0{,}2;0{,}3)=0{,}3$.

Міркуючи аналогічно, одержуємо:

$$\mu_{POS_{Xe}(Y)}(2)=0{,}6;\ \mu_{POS_{Xe}(Y)}(3)=0{,}3;\ \mu_{POS_{Xe}(Y)}(4)=0{,}6;$$

$$\mu_{POS_{Xe}(Y)}(5)=0{,}5;\ \mu_{POS_{Xe}(Y)}(6)=0{,}3;\ \mu_{POS_{Xe}(Y)}(7)=0{,}6;$$

$$\mu_{POS_{Xe}(Y)}(8)=0{,}3;\ \mu_{POS_{Xe}(Y)}(9)=0{,}3.$$

Разом:

$$\beta_Y(X_1)=\frac{0{,}3+0{,}6+0{,}3+0{,}6+0{,}5+0{,}3+0{,}6+0{,}3+0{,}3}{9}=\frac{3{,}8}{9}=0{,}422.$$

У такий же спосіб знаходимо $\beta_Y(X_2)$ і $\beta_Y(X_3)$: $\beta_Y(X_2)=\frac{2{,}1}{9}=0{,}233$, $\beta_Y(X_3)=\frac{2{,}7}{9}=0{,}3$.

При оцінюванні набору ознак $Xe = \{X_1, X_2\}$ необхідно враховувати, що ознаки такої комбінації можуть приймати $B_{Xe}=\prod\limits_{X_i\in Xe}|X_i|=3\cdot 3=9$ значень: $\{x_{11}, x_{21}\}$, $\{x_{11}, x_{22}\}$, $\{x_{11}, x_{23}\}$, $\{x_{12}, x_{21}\}$, $\{x_{12}, x_{22}\}$, $\{x_{12}, x_{23}\}$, $\{x_{13}, x_{21}\}$, $\{x_{13}, x_{22}\}$, $\{x_{13}, x_{23}\}$. Для кожного з таких значень необхідно визначити $\mu_{x_{1i}x_{2j}y_k}(a)$, $i, j, k = 1, 2, 3$, $a = 1, 2,\ldots 9$,після чого обчислити $\mu_{POS_{Xe}(Y)}(a)$. У результаті одержимо:

$$\beta_Y(Xe)=\frac{4}{9}=0{,}444.$$

? 1.7 Контрольні питання

1. З якою метою виконують відбір інформативних ознак?
2. Наведіть постановку задачі відбору ознак.
3. Що є результатом виконання процедури відбору ознак?
4. Дайте визначення таких понять: інформативність ознаки, незначущі ознаки, надлишкові ознаки.
5. Проаналізуйте методи відбору інформативних ознак.
6. Порівняйте методи повного перебору.
7. Наведіть переваги та недоліки методів скороченого перебору.
8. У чому полягає основна ідея методу групового врахування аргументів?
9. Які стратегії використовують евристичні методи відбору ознак?
10. Наведіть послідовність виконання методів послідовного додавання та видалення ознак.
11. Які критерії використовуються у методі ранжирування ознак?
12. У яких випадках доцільним є використання методів кластеризації ознак?
13. У чому полягає основна ідея методу випадкового пошуку з адаптацією?
14. Яким чином можуть використовуватися методи еволюційного пошуку до відбору інформативних ознак?
15. Які множини можуть бути використані як початкова точка у методах відбору ознак?
16. Проаналізуйте процедури пошуку оптимального набору ознак.
17. Які стратегії використовуються для оцінювання інформативності набору ознак?
18. Порівняйте критерії оцінювання індивідуальної інформативності.
19. Для оцінювання якого типу зв’язку використовується коефіцієнт парної кореляції?
20. Які критерії використовуються для оцінювання спільного впливу набору ознак?

1.8 Практичні завдання

 *Завдання 1*. Проведіть порівняння методів відбору інформативних ознак. Сформуйте набір критеріїв порівняння та складіть порівняльну таблицю.

 *Завдання 2*. Критерії оцінювання індивідуальної інформативності.

2.1 За допомогою середовища Matlab згідно з номером студента за журналом для відповідного номера варіанту V сформувати навчаючу вибірку $<x, y>$ обсягом m екземплярів, $j = 1, 2, \ldots, m$ що характеризуються L ознаками x_{ij}, $i = 1, 2, \ldots, L$, та зіставити кожному екземпляру значення цільової ознаки y_j:

$$x_{ij} = \begin{cases} iV - 0{,}1j, \; i = 1, 5, 9, \ldots, \\ 0{,}01iV^{-1} + 0{,}3j, \; i = 2, 4, 6, \ldots, \\ i\,\mathrm{rand}, \; i = 3, 7, 11, \ldots; \end{cases} \quad y_j = 2x_{1j} + 0{,}1x_{2j};$$

$$m = \begin{cases} 10V, \; V < 10, \\ 5V, \; 10 \le V < 20, \\ 3V, \; V \ge 20; \end{cases} \quad L = \begin{cases} 5V, \; V < 7, \\ 4V, \; 7 \le V < 10, \\ 3V, \; 10 \le V < 20, \\ 2V, \; V \ge 20; \end{cases}$$

де rand – випадкове число в діапазоні [0, 1].

2.2 На алгоритмічній мові програмування пакету Matlab написати програму, що дозволяє оцінювати значення критеріїв індивідуальної інформативності ознак:

– коефіцієнт парної кореляції;
– коефіцієнт кореляції знаків;
– коефіцієнт кореляції Фехнера;
– дисперсійне відношення;
– коефіцієнт зв'язку;
– інформаційний критерій;
– теоретико-інформаційний критерій;
– ентропія ознаки;
– критерій, заснований на імовірнісному підході;
– критерій, заснований на статистичному підході.

2.3 За допомогою розробленої у п.2.2 програми оцінити інформативність ознак екземплярів вибірки, використовуючи різні критерії індивідуальної інформативності. При необхідності, значення вихідної ознаки можна дискретизувати.

2.4 Побудувати таблицю з оцінками інформативності ознак відносно вихідного параметру *y*, стовпці якої повинні мати назви:

1) номер ознаки;
2) коефіцієнт парної кореляції;
3) коефіцієнт кореляції знаків;
4) коефіцієнт кореляції Фехнера;
5) дисперсійне відношення;
6) коефіцієнт зв'язку;
7) інформаційний критерій;
8) теоретико-інформаційний критерій;
9) ентропія ознаки;
10) критерій, заснований на імовірнісному підході;
11) критерій, заснований на статистичному підході;

2.5 Проаналізувати за побудованими таблицями оцінки інформативності ознак. Зробити висновки щодо важливості ознак відповідно до вихідного параметру *y*.

 *Завдання 3*. Методи відбору інформативних ознак.

3.1 За допомогою мови пакету Matlab згідно з номером студента за журналом розробити програму, що реалізує два методи відбору інформативних ознак:

1) метод повного перебору;
2) пошук у глибину;
3) пошук завширшки;
4) метод гілок і границь або скорочений пошук у глибину;
5) метод групового врахування аргументів або скорочений пошук завширшки;
6) метод послідовного додавання ознак;
7) метод послідовного видалення ознак;
8) метод почергового додавання й видалення ознак;
9) ранжирування ознак;
10) кластеризація ознак;
11) випадковий пошук з адаптацією;

12) еволюційний пошук.

3.2 Для вибірки, сформованої у попередньому завданні, використовуючи розроблену програму, за допомогою різних методів виділити комбінацію інформативних ознак.

При цьому для оцінювання значущості набору ознак (групової інформативності) використовувати такі критерії:

1) середньоквадратична помилка;
2) сума квадратів відхилень;
3) середня абсолютна помилка;
4) сума значень абсолютних відхилень;
5) максимальне абсолютне відхилення;
6) середня відносна помилка;
7) сума відносних відхилень;
8) максимальне відносне відхилення.

В якості моделі для обчислення значущості набору ознак з використанням критеріїв, наведених вище, обрати лінійну багатовимірну регресію.

3.3 Проаналізувати отримані набори ознак. Зробити висновки щодо доцільності використання того чи іншого методу відбору ознак.

 *Завдання 4*. Використовуючи пакет Matlab, написати програму для синтезу інформативних ознак за допомогою методів:

– метод головних компонентів (Principal Component Analysis);

– метод незалежних компонентів (Principal Independent Analysis).

Для вибірки, сформованої у завданні № 2, використовуючи розроблену програму, синтезувати набір інформативних ознак.

Виконати аналіз отриманих результатів.

 *Завдання 5.* Написати реферат на одну з таких тем.

1. Методи відбору інформативних ознак.
2. Критерії оцінювання індивідуальної значущості.
3. Стохастичний підхід до відбору ознак.
4. Методи синтезу інформативних ознак.
5. Програмні засоби відбору та синтезу ознак.

1.9 Тестові завдання

Т *Завдання 1.* До чого призводить наявність надлишкових та неінформативних ознак?

А До прискорення класифікації за моделлю.
Б До спрощення процесу синтезу моделі.
В Збільшення часу класифікації за моделлю.
Г До підвищення якості математичної моделі.
Д Вірними є відповіді А та Г.
Е Серед відповідей Б–Г усі вірні.

Т *Завдання 2.* Яким з наступних способів може бути представлена постановка задачі відбору інформативних ознак?

А Виділити комбінацію ознак X^* з вихідного масиву даних, при якій досягається мінімум заданого критерію оцінювання набору ознак.
Б Відібрати з множини вхідних L ознак комбінацію, що складається не більш, ніж з L_0 ознак ($L_0 < L$), при якій досягається оптимум заданого критерію.
В Знайти набір ознак мінімального розміру, що забезпечує досягнення заданого значення критерію оцінювання значимості набору ознак.
Г Відібрати комбінацію ознак X^* з вихідного масиву даних, яка забезпечує досягнення заданого значення критерію оцінювання значимості набору ознак.
Д Вірними є відповіді Б та Г.
Е Серед відповідей А–В усі вірні.

Т *Завдання 3.* Що є результатом виконання процедури відбору ознак?

А Набір ознак, які є малоінформативними.
Б Набір ознак, що складається з незначущих ознак.
В Набір ознак, що складається з надлишкових ознак.
Г Оптимальний набір ознак, що має достатню інформативність.

Д Вірними є відповіді Б та В.
Е Вірними є відповіді А та Г.

Т *Завдання 4.* Який з методів виділення набору ознак відносяться до евристичних методів?
А Повний перебір.
Б Метод послідовного додавання ознак.
В Кластеризація ознак.
Г Ранжирування ознак.
Д Метод групового додавання ознак.
Е Вірними є відповіді Г та Д.

Т *Завдання 5.* Який з описаних недоліків має метод класичного повного перебору?
А Є ймовірність неврахування деяких ознак.
Б Має швидку збіжність.
В Великі обчислювальні витрати.
Г Потребує жорстко обумовленого порядку ознак.
Д Вірними є відповіді А–В.
Е Серед відповідей А–Г вірної немає.

Т *Завдання 6.* На скільки змінюється розмір наборів ознак на кожній ітерації у методі групового врахування аргументів?
А Зменшується на 2.
Б Зменшується на 1.
В Не змінюється.
Г Збільшується на 1.
Д Збільшується на 2.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 7.* Який критерії може буди використаний в якості критерію закінчення пошуку у методі послідовного додавання ознак?

А Виникнення ситуації, при якій всі згенеровані на поточній ітерації набори ознак гірше оптимального набору, отриманого раніше: $J_{\text{опт}} < \min(J(Xe_{t,k}))$

Б Виникнення ситуації, при якій значення критеріїв оцінювання всіх згенерованих на поточній ітерації наборів ознак гірше максимально припустимої величини ε.

В Досягнення максимально можливого розміру набору ознак L_0.

Г Перевищення припустимої кількості ітерацій T.

Д Вірними є відповіді Б та В.

Е Вірними є відповіді А, В, Г.

Т *Завдання 8.* Які критерії оцінювання використовуються у методі ранжирування ознак?

А Критерії оцінювання адаптивності ознаки.

Б Критерії оцінювання спільного впливу набору ознак на вихідний параметр.

В Критерії оцінювання індивідуальної значимості ознак.

Г Критерії оцінювання відстані на ознаках.

Д Критерії оцінювання ймовірності повторної появи ознаки у наборі.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 9.* Яким шляхом відбувається формування нових рішень у методі випадкового пошуку з адаптацією?

А Шляхом вибору найбільш інформативних ознак з наборів ознак, використовуваних на попередній ітерації.

Б Шляхом випадкової генерації наборів ознак.

В Шляхом додавання ознак до наборів ознак використовуваних на попередній ітерації.

Г Шляхом видалення малоінформативних ознак з наборів ознак, використовуваних на попередній ітерації.

Д Почергового додавання та видалення ознак з наборів ознак, використовуваних на попередній ітерації.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 10.* Якою вважається відповідна ознака, у методі еволюційного пошуку, коли біт хромосоми приймає нульове значення?

А Вважається інформативною.
Б Вважається не підходящою для схрещування.
В Вважається неінформативною.
Г Вважається не підлягаючою для мутації
Д Вважається елітною.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 11.* Що використовується в якості фітнесс-функції у методі еволюційного пошуку при відборі ознак?

А Будь-яка квадратична функція.
Б Значення критерію оцінювання індивідуальної значимості ознак.
В Будь-яка лінійна функція.
Г Значення критерію оцінювання набору ознак.
Д Будь-яка не лінійна функція.
Е Вірними є відповіді А, В, Д.

Т *Завдання 12.* Які критерії оцінювання інформативності ознак використовує метод гілок і границь при відборі ознак?

А Групові.
Б Індивідуальні.
В Метрики відстані.
Г Парні.
Д Вірними є відповіді А та В.
Е Серед відповідей А–Г вірної немає.

Т *Завдання 13.* Який критерії оцінювання індивідуальної інформативності ознак розраховується за формулою $r(X_i) = \sum_{p=1}^{m} \left(x_{ip} - \bar{x}_i\right)\left(y_p - \bar{y}\right)$?

А Коефіцієнт парної кореляції.
Б Коефіцієнт кореляції знаків.
В Коефіцієнт кореляції Фехнера.
Г Коефіцієнт зв'язку.
Д Інформаційний критерій.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 14.* Який критерій оцінювання індивідуальної інформативності ознак описує формула $r_{\Phi}(X_i)=\frac{C_i-D_i}{C_i+D_i}=\frac{2C_i}{m}-1$?

А Коефіцієнт парної кореляції.
Б Коефіцієнт кореляції знаків.
В Коефіцієнт кореляції Фехнера.
Г Коефіцієнт зв'язку.
Д Інформаційний критерій.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 15.* Що являє собою k у формулі коефіцієнта кореляції Пірсона $r_{\text{Пир}}(Xe)=\frac{k\overline{r_y}}{\sqrt{k+k(k-1)\overline{r_x}}}$?

А Коефіцієнта кореляції між ознаками.
Б Значення коефіцієнта кореляції між кожною ознакою й вихідним параметром.
В Кількість ознак у розглянутому наборі *Xe*.
Г Кількість комбінацій інтервалів діапазону зміни значень набору ознак *Xe*.
Д Кількість значень набору ознак *Xe*, що потрапили в k-*ий* інтервал групування.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 16.* Яким чином визначається середньоквадратична помилка, для розрахунку помилки синтезованої моделі у випадку відбору ознак при вирішенні задачі прогнозування?

А $MSE(Xe)=\frac{1}{m}\sum_{p=1}^{m}\left(y^p-y^p_{Xe}\right)^2$.

Б $MSE(Xe)=\sum_{p=1}^{m}\left(y^p-y^p_{Xe}\right)^2$.

В $MSE(Xe)=\frac{1}{m}\sum_{p=1}^{m}\left|y^p-y^p_{Xe}\right|$.

Г $MSE(Xe)=\sum_{p=1}^{m}\left|y^p-y^p_{Xe}\right|$.

Д $MSE(Xe)=\frac{1}{m}\sum_{p=1}^{m}\left|\frac{y^p-y^p_{Xe}}{y^p}\right|$.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 17.* Яким чином визначається середня відносна помилка, для розрахунку помилки синтезованої моделі у випадку відбору ознак при рішенні задачі прогнозування?

А $MARE(Xe)=m\cdot MSE(Xe)=\sum_{p=1}^{m}\left(y^p-y^p_{Xe}\right)^2,\ y^p\neq 0.$

Б $MARE(Xe)=\frac{1}{m}\sum_{p=1}^{m}\left|y^p-y^p_{Xe}\right|,\ y^p\neq 0.$

В $MARE(Xe)=\frac{1}{m}\sum_{p=1}^{m}\left|\frac{y^p-y^p_{Xe}}{y^p}\right|,\ y^p\neq 0.$

Г $MARE(Xe)=\frac{1}{m}\sum_{p=1}^{m}\left(y^p-y^p_{Xe}\right)^2,\ y^p\neq 0.$

Д $MARE(Xe)=\max_{p=1,2,\ldots,m}\left|\frac{y^p-y^p_{Xe}}{y^p}\right|,\ y^p\neq 0.$

Е Серед відповідей А–Д вірної немає.

Т *Завдання 18.* Який тип можуть мати значення ознак при оцінюванні інформативності ознак за допомогою коефіцієнта кореляції знаків?

А Безперервні.
Б Дискретні.
В Бінарні.
Г Нечіткі.
Д Вірними є відповіді А та Б.
Е Вірними є відповіді Б та В.

Т *Завдання 19.* Що позначає α_{jk} у формулі узагальненої відстані у просторі ознак $d_k = \left(\sum_{j=1}^{m} \alpha_{jk}^{v} \left| x_{aj} - x_{bj} \right|^{v} \right)^{\frac{\mu}{v}}$?

А Ступінь лінійності *j*-го екземпляра *k*-го класу.
Б Ступінь не лінійності *j*-го екземпляра *k*-го класу.
В Деяке ціле число, що задаються дослідником.
Г Коефіцієнт значимості *j*-го екземпляра для *k*-го класу.
Д Ступінь інформативності *j*-ої ознаки *k*-го класу.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 20.* Яку метрику, використовувану при кластеризації ознак, визначає формула $d = \sum_{j=1}^{m} \left(\alpha_j \right)^{p} (x_{aj} - x_{bj})^{p}$?

А Відстань у нелінійному просторі.
Б Діагностична міра відстані.
В Відстань Мінковского.
Г Евклідову відстань.
Д Відстань Камберра.
Е Серед відповідей А–Д вірної немає.

2 МОДЕЛІ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

2.1 Асоціативні правила

Асоціативні правила дозволяють виявляти закономірності між пов'язаними подіями і є одним з інструментів видобування знань з масивів даних.

2.1.1 Основні поняття про асоціативні правила

Нехай $I = \{i_1, i_2, ..., i_m\}$ – множина елементів, D – множина транзакцій, де кожна транзакція T – це набір елементів з I, $T \subseteq I$. Кожна транзакція – це множина елементів (подій), що відбулися одночасно, і являє собою бінарний вектор, де $t[k]=1$, якщо i_k елемент присутній у транзакції, інакше $t[k]=0$. Транзакція Т містить X, певний набір елементів з I, якщо $X \subseteq T$.

Розширеною транзакцією називається транзакція, розширена предками всіх елементів, що входять у цю транзакцію.

Асоціативним правилом (association rule) називається імплікація $X \rightarrow Y$, де $X \subset I$, $Y \subset I$ та $X \cap Y = \varnothing$.

Підтримкою (support) правила $X \rightarrow Y$ називається частка s, якщо s % транзакцій з D, містять $X \cup Y$, $\text{supp}(X \rightarrow Y) = \text{supp}(X \cup Y)$.

Вірогідність правила показує яка імовірність того, що з X випливає Y. Правило $X \rightarrow Y$ є справедливим з вірогідністю (confidence) с, якщо c % транзакцій з D, що містять X, також містять Y, $\text{conf}(X \rightarrow Y)=\text{supp}(X \rightarrow Y)/\text{supp}(X)$.

Іншими словами, метою аналізу є встановлення таких залежностей: якщо в транзакції зустрівся деякий набір елементів X, то на підставі цього можна зробити висновок про те, що інший набір елементів Y також повинний з'явитися в цій транзакції. Установлення таких залежностей дає нам можливість знаходити дуже прості й інтуїтивно зрозумілі правила.

Методи пошуку асоціативних правил призначені для знаходження всіх правил $X \rightarrow Y$, причому підтримка і вірогідність цих правил повинні бути вище деяких наперед визначених порогів,

називаних відповідно **мінімальною підтримкою** (minsupport) і **мінімальною вірогідністю** (minconfidence).

Перший метод пошуку асоціативних правил, що називався *AIS* був розроблений у 1993 році співробітниками дослідницького центра IBM Almaden.

Задача знаходження асоціативних правил розбивається на дві підзадачі:

1) знаходження всіх наборів елементів, що задовольняють порогові minsupport. Такі набори елементів називаються такими, що часто зустрічаються;

2) генерація правил зі знайдених наборів елементів з вірогідністю, що задовольняє порогові minconfidence.

Один з перших методів, що ефективно вирішують подібний клас задач, – це метод **APriori**. Крім цього методу останнім часом був розроблений ряд інших методів: **DHP**, **Partition**, **DIC** та інші.

Значення для параметрів minsupport та minconfidence вибираються таким чином, щоб обмежити кількість знайдених правил. Якщо підтримка має велике значення, то методи будуть знаходити правила, добре відомі аналітикам або настільки очевидні, що немає ніякого сенсу проводити такий аналіз. З іншого боку, низьке значення підтримки веде до генерації величезної кількості правил, що, звичайно, вимагає істотних обчислювальних ресурсів. Проте, більшість цікавих правил знаходиться саме при низькому значенні порога підтримки. Хоча занадто низьке значення підтримки веде до генерації статистично необґрунтованих правил.

Пошук асоціативних правил – це зовсім не тривіальна задача, як може показатися на перший погляд. Одна з проблем – алгоритмічна складність при знаходженні наборів елементів, що часто зустрічаються, оскільки з ростом числа елементів I експоненційно росте число потенційних наборів елементів.

Узагальненим асоціативним правилом (generalized association rule) називається імплікація $X \rightarrow Y$, де $X \subset I$, $Y \subset I$ та $X \cap Y = \varnothing$ і де жоден з елементів, що входять у набір Y, не є предком жодного елемента, що входить у X. Підтримка і вірогідність підраховуються так само, як і у випадку асоціативних правил. Ми називаємо правило $X \rightarrow Y$ узагальненим, тому що елементи, які

входять у нього можуть знаходитися на будь-якому рівні таксономії. Також будемо називати $\underline{x}$ предком x та x нащадком $\underline{x}$.

Основною відмінністю узагальнених асоціативних правил від асоціативних правил є те, що одержувані правила включають елементи, що є предками елементів, які входять у множину транзакцій.

Ієрархією (таксономією) елементів називається ліс спрямованих ациклічних дерев, листами яких є елементи транзакцій, а внутрішніми вузлами – групи елементів.

Уведення додаткової інформації про групування елементів у виді ієрархії допомагає установити асоціативні правила не тільки між окремими елементами, але і між різними рівнями ієрархії (групами).

Окремі елементи можуть мати недостатню підтримку, але в цілому група може задовольняти порогові minsupport. Це приводить до того, що раніше не виявлене потенційно цікаве правило, побудоване на елементах нижнього рівня ієрархії, може бути отримано, але його елементами будуть або елементи транзакції, або предки цих елементів.

Введення інформації про групування елементів може використовуватися для відсікання «нецікавих» правил.

Для знаходження таких правил можна використовувати кожний з вищезгаданих методів. Для цього кожну транзакцію потрібно доповнити всіма предками кожного елемента, що входить у транзакцію. Однак, пряме застосування цих методів неминуче приведе до таких проблем:

– елементи на верхніх рівнях ієрархії прагнуть до значно більших значень підтримки в порівнянні з елементами на нижніх рівнях;

– з додаванням у транзакції груп збільшується кількість атрибутів і, відповідно, розмірність вхідного простору. Це ускладнює задачу, а також веде до генерації більшої кількості правил;

– поява надлишкових правил, що суперечать визначенню узагальненого асоціативного правила. Отже, потрібні спеціальні оператори, що видаляють подібні надлишкові правила.

2.1.2 Виявлення узагальнених асоціативних правил

Нехай I – ліс спрямованих дерев. Дуги в I – це залежності між елементами. Нехай елементи, що належать I, розташовані в деякій ієрархії. Якщо є дуга від a до b, то говорять, що a – предок b та b – нащадок a (a – це узагальнення b).

Необхідно знайти закономірності, що є узагальненими асоціативними правилами виду $X \to Y$, причому $\text{supp}(X \to Y) \geq \text{minsupport}$ та $\text{conf}(X \to Y) \geq \text{minconfidence}$.

Це визначення задачі має одну проблему. Справа в тому, що при такому визначенні задачі, будуть знайдені «зайві» узагальнені асоціативні правила. Для вирішення цієї проблеми розглянемо такий параметр правила як **рівень інтересу**.

2.1.3 Визначення «цікавих» правил

Нехай $\underline{Z}$ – це предок Z, де Z та $\underline{Z}$ – множини елементів, що входять в ієрархію ($Z, \underline{Z} \subseteq I$). $\underline{Z}$ є предком Z, тільки в тому випадку, якщо $\underline{Z}$ можна одержати з Z шляхом підміни одного чи декількох елементів їхніми предками. Будемо називати правила $\underline{X} \to Y$, $X \to \underline{Y}$, $\underline{X} \to \underline{Y}$ предками правила $X \to Y$.

Правило $\underline{X} \to \underline{Y}$ є найближчим предком правила $X \to Y$, якщо не існує такого правила $X' \to Y'$, що $X' \to Y'$ – це предок $X \to Y$ та $\underline{X} \to \underline{Y}$ – це предок $X' \to Y'$.

Подібні визначення можна дати і для правил: $\underline{X} \to Y$, $X \to \underline{Y}$.

Нехай $\Pr(X)$ – це імовірність того, що всі елементи з X містяться в одній розширеній транзакції. Тоді $\text{supp}(X \cup Y) = \Pr(X \cup Y)$ та $\text{conf}(X \to Y) = \Pr(Y|X)$. Якщо підтримка $\{x,y\}$ більше значення мінімальної підтримки, то і підтримка $\{\underline{x},y\}$, і підтримка $\{x,\underline{y}\}$, і підтримка $\{\underline{x},\underline{y}\}$ будуть більше порога мінімальної підтримки. Однак якщо вірогідність правила $X \to Y$ більше мінімальної вірогідності, тільки правило $X \to \underline{Y}$ гарантовано буде мати вірогідність більшу ніж мінімальна. Підтримка елемента, узятого з внутрішнього рівня ієрархії, не дорівнює сумі підтримок елементів, що є безпосередніми нащадками цього елемента.

Розглянемо правило $X \to Y$. Нехай $Z = X \cup Y$. Помітимо, що $\text{supp}(X \to Y) = \text{supp}(Z)$. Назвемо $E_{\underline{Z}}[\Pr(Z)]$ очікуваним значенням

Pr(Z) відносно $\underline{Z}$. Нехай $Z=\{z_1, ..., z_n\}$, $\underline{Z}=\{\underline{z}_1, ..., \underline{z}_j, \underline{z}_{j+1}, ..., \underline{z}_n\}$, $1 \le j \le n$. Тоді можна визначити:

$$E_{\underline{Z}}\left[\Pr(Z)\right]=\frac{\Pr(z_1)}{\Pr(\underline{z}_1)}\times ...\times \frac{\Pr(z_j)}{\Pr(\underline{z}_j)}\times \Pr(\underline{Z}) .$$

Аналогічно $E_{\underline{X}\to\underline{Y}}[\Pr(Y|X)]$ визначимо як очікуване значення вірогідності правила $X\to Y$ відносно $\underline{X}\to\underline{Y}$. Нехай $Y=\{y_1, ..., y_n\}$, $\underline{Y}=\{\underline{y}_1, ..., \underline{y}_j, \underline{y}_{j+1}, ..., \underline{y}_n\}$, $1 \le j \le n$. Тоді можна визначити:

$$E_{\underline{X}\to\underline{Y}}\left[\Pr(Y \mid X)\right]=\frac{\Pr(y_1)}{\Pr(\underline{y}_1)}\times ...\times \frac{\Pr(y_j)}{\Pr(\underline{y}_j)}\times \Pr(Y \mid X) .$$

Правило $X\to Y$ називається ***R*-цікавим** щодо правила-предка, якщо підтримка правила $X \to Y$ у R разів більше очікуваної підтримки правила $X\to Y$ щодо предка або якщо вірогідність правила $X\to Y$ у R разів більше очікуваної вірогідності правила $X\to Y$ щодо правила-предка.

Цікавим називається правило, якщо в нього немає предків або воно є R-цікавим щодо усіх своїх найближчих предків.

Частково цікавим називається правило, якщо в нього немає предків або воно є R-цікавим щодо будь-якого свого найближчого предка.

Тепер задачу виділення узагальнених асоціативних правил можна сформулювати по новому: необхідно знайти закономірності, що є узагальненими асоціативними правилами виду $X\to Y$, причому підтримка правила $X\to Y$ більше або дорівнює деякому наперед заданому значенню мінімальної підтримки і вірогідність більше або дорівнює значенню мінімальної вірогідності і правила $X\to Y$ є цікавими або частково цікавими.

2.1.4 Обчислення узагальнених асоціативних правил

Метод обчислення узагальнених асоціативних правил можна розбити на кілька етапів.

Етап 1. Пошук множин елементів, що часто зустрічаються, підтримка яких більше ніж заданий поріг підтримки (мінімальна підтримка).

Етап 2. Обчислення правил на основі знайдених на попередньому етапі множин елементів, що часто зустрічаються. Основна ідея обчислення правил на основі множин, що часто зустрічаються, полягає в такому: якщо $ABCD$ – це множина елементів, що часто зустрічається, то на основі цієї множини можна побудувати правила $X \rightarrow Y$ (наприклад, $AB \rightarrow CD$), причому $X \cup Y = ABCD$. Підтримка правила дорівнює підтримці множини, що часто зустрічається. Вірогідність правила обчислюється за формулою $\text{conf}(X \rightarrow Y) = \text{supp}(X \rightarrow Y)/\text{supp}(X)$. Правило додається до результуючого списку правил, якщо вірогідність цього правила більше порога minconf.

Етап 3. З результуючого списку правил видаляються всі «нецікаві» правила.

2.1.5 Базовий метод пошуку часто зустрічаючихся множин

На першому кроці методу підраховуються 1-елементні набори, що часто зустрічаються. При цьому елементи можуть знаходитися на будь-якому рівні таксономії. Для цього необхідно переглянути весь набір даних і підрахувати для них підтримку, тобто скільки разів зустрічаються в базі.

Наступні кроки будуть складатися з двох частин: генерації потенційних наборів елементів, що часто зустрічаються, (їх називають кандидатами) і підрахунку підтримки для кандидатів.

Даний метод можна записати як послідовність кроків.

Крок 1. Виділити і занести в L_1 множини елементів і груп елементів, що часто зустрічаються. Установити: $k = 2$.

Крок 2. Якщо $L_{k-1} \neq \varnothing$, тоді перейти до кроку 3, у протилежному випадку – перейти до кроку 7.

Крок 3. Згенерувати C_k – множину кандидатів потужністю k на основі L_{k-1}.

Крок 4. Для всіх транзакцій $t \in D$ виконати кроки 4.1–4.3.

Крок 4.1 Розширити транзакцію t предками всіх елементів, що входять у транзакцію.

Крок 4.2 Видалити дублікати з транзакції t.

Крок 4.3 Для всіх кандидатів $c \in C_k$ виконати: якщо $c \subseteq t$, то установити: c.count = c.count+1.

Крок 5. Зробити відбір кандидатів: $L_k = \{c \in C_k \mid c.\text{count} \geq \text{minsupport}\}$.

Крок 6. Установити: $k = k+1$. Перейти до кроку 3.

Крок 7. Зупинення. Повернути як результат $\bigcup L_k$.

Функція генерації кандидатів. Для того щоб одержати k-елементні набори, скористаємося $(k–1)$-елементними наборами, які були визначені на попередньому кроці і є такими, що часто зустрічаються.

Метод генерації кандидатів буде складатися з двох кроків.

Крок 1. Об'єднання. Кожний кандидат C_k буде формуватися шляхом розширення набору, що часто зустрічається, розміром $(k–1)$ додаванням елемента з іншого $(k–1)$-елементного набору: включити до C_k ті елементи $a_1, a_2, ..., a_{k–1}, b_{k–1}$ з $L_{k–1}$: а, $b \in L_{k–1}$, для яких: $a_1 = b_1, a_2 = b_2, ... , a_{k–2} = b_{k–2}, a_{k–1} < b_{k–1}$.

Крок 2. Видалення надлишкових правил. На підставі властивості антимонотонності, варто видалити всі набори з C_k, якщо хоча б одна з його $(k–1)$ підмножин не є такою, що часто зустрічається.

Ґеш-дерево можна використовувати для ефективного підрахунку підтримки кандидатів.

Ґеш-дерево будується щоразу, коли формуються кандидати. Первісне дерево складається тільки з кореня, що є листом, і не містить ніяких кандидатів-наборів. Щоразу, коли формується новий кандидат, він заноситься в корінь дерева і так доти, поки кількість кандидатів у корні-листі не перевищить деякий поріг. Як тільки кількість кандидатів стає більше порога, корінь перетвориться в ґеш-таблицю, тобто стає внутрішнім вузлом, і для нього створюються нащадки-листи. І всі приклади розподіляються по вузлах-нащадках відповідно до ґеш-значення елементів, що входять у набір, і т. д. Кожний новий кандидат ґешується на внутрішніх вузлах, поки він не досягне першого вузла-листа, де він і буде зберігатися, поки кількість наборів знову ж не перевищить порога.

2.1.6 Поліпшений метод пошуку часто зустрічаючихся множин

До базового методу можна додати кілька оптимизацій, що збільшать швидкість роботи базового методу.

Доцільно один раз обчислити множини предків для кожного елемента ієрархії як для елемента нижнього рівня таксономії (лист дерева), так і для елемента внутрішнього рівня.

Необхідно видаляти кандидати, що містять елемент і його предка.

Підтримка множини X, що містить і елемент x і його предка $\underline{x}$, обчислюється за формулою: $\text{supp}(X)=\text{supp}(x–\underline{x})$. Беручи це до уваги, будемо видаляти кандидати, що містять і елемент і його предка зі списку кандидатів до початку процесу підрахунку підтримки.

Якщо L_k – це список множин, що часто зустрічаються, який не містить множини, що включають і елемент і його предка в одній множині, то C_{k+1} (список кандидатів, одержуваних з L_k) також не буде містити множини, що включають і елемент і його предка. З огляду на це, будемо видаляти кандидатів, що включають і елемент і його предка, зі списку кандидатів тільки на першій ітерації зовнішнього циклу.

Немає необхідності в додаванні всіх предків всіх елементів, що входять у транзакцію. Якщо який-небудь елемент, у якого є предок, не знаходиться в списку кандидатів, то в списку елементів із предками він позначається як вилучений. Отже, із транзакції видаляються елементи, позначені як вилучені, або здійснюється заміна цих елементів на їхніх предків. До транзакції додаються тільки невилучені предки.

Не будемо пропускати транзакцію через ґеш-дерево, якщо її потужність менше ніж потужність елементів, розташованих у ґеш-дереві.

Доцільно позначати транзакції як вилучені і не використовувати їх при підрахунку підтримки на наступних ітераціях, якщо на поточній ітерації в цю транзакцію не ввійшов жоден кандидат.

З огляду на написане вище, одержуємо такий метод.

Крок 1. Обчислити I^* – множини предків елементів для кожного елемента. Виділити і занести в L_1 множини елементів і груп елементів, що часто зустрічаються. Установити: $k = 2$.

Крок 2. Якщо $L_{k-1} \neq \varnothing$, тоді перейти до кроку 3, у протилежному випадку – перейти до кроку 9.

Крок 3. Згенерувати C_k – множину кандидатів потужністю k на основі L_{k-1}.

Крок 4. Якщо $k = 2$, то видалити ті кандидати з C_k, що містять елемент і його предка.

Крок 5. Позначити як вилучені множини предків елемента, що не міститься в списку кандидатів.

Крок 6. Для всіх транзакцій $t \in D$ виконати кроки 6.1–6.3.

Крок 6.1 Для кожного елемента $x \in t$: додати всіх предків x з I^* до t.

Крок 6.2 Видалити дублікати з транзакції t.

Крок 6.3 Якщо транзакція t не позначена як вилучена та $|t| \geq k$, то виконати кроки 6.3.1–6.3.2.

Крок 6.3.1 Для всіх кандидатів $c \in C_k$: якщо $c \in C_k$, то установити c.count = c.count+1.

Крок 6.3.2 Якщо в транзакцію t не ввійшов жоден кандидат, то позначити цю транзакцію як вилучену.

Крок 7. Виконати відбір кандидатів: $L_k = \{c \geq C_k \mid c.\text{count} \geq \text{minsupport}\}$.

Крок 8. Установити: $k = k + 1$. Перейти до кроку 2.

Крок 9. Зупинення. Повернути як результат $\bigcup L_k$.

2.1.7 Масштабувальний метод пошуку асоціативних правил Apriori

Для того, щоб було можливо застосувати метод Apriori, необхідно провести передобробку даних: по-перше, привести всі дані до бінарного виду; по-друге, змінити структуру даних – подати їх у виді таблиці. Кількість стовпців у таблиці дорівнює кількості елементів, що є присутніми у множині транзакцій D. Кожний запис відповідає транзакції, де у відповідному стовпці стоїть 1, якщо елемент присутній у транзакції, і 0 – у протилежному випадку. Всі елементи повинні бути упорядковані за абеткою (якщо це числа, вони повинні бути упорядковані в числовому порядку).

На першому кроці методу необхідно знайти набори елементів, що часто зустрічаються, а потім, на другому кроці, витягти з них правила. Кількість елементів у наборі будемо називати розміром набору, а набір, що складається з k елементів, – k-елементним набором.

Виявлення наборів елементів, що часто зустрічаються, – операція, що вимагає багато обчислювальних ресурсів і, відповідно, часу. Примітивний підхід до вирішення даної задачі – простий перебір усіх можливих наборів елементів. Це вимагає $O(2^{|I|})$ операцій, де $|I|$ – кількість елементів.

Властивість антимонотонності використовується в методі Apriori і слугує для зниження розмірності простору пошуку: підтримка будь-якого набору елементів не може перевищувати мінімальної підтримки кожної з його підмножин. Властивості антимонотонності можна дати й інше формулювання: з ростом розміру набору елементів підтримка зменшується, або залишається такою ж. З усього вищесказаного випливає, що будь-який k-елементний набір буде таким, що часто зустрічається, тоді і тільки тоді, коли всі його (k–1)-елементні підмножини будуть такими, що часто зустрічаються.

Усі можливі набори елементів з I можна уявити у вигляді ґрат, що починаються з порожньої множини, потім на 1-му рівні – 1-елементні набори, на 2-му – 2-елементні і т. д. На k-му рівні подані k-елементні набори, пов'язані з усіма своїми (k–1)-елементними підмножинами.

На першому кроці методу підраховуються одноелементні набори, що часто зустрічаються. Для цього необхідно пройтися по всьому набору даних і підрахувати для них підтримку, тобто скільки разів зустрічаються в базі.

Наступні кроки будуть складатися з двох частин: генерації потенційних наборів елементів, що часто зустрічаються, (їх називають кандидатами) і підрахунку підтримки для кандидатів.

Даний метод можна записати як таку послідовність кроків.

Крок 1. Задати F_1 – множину одноелементних наборів, що часто зустрічаються.

Крок 2. Установити: $k = 2$.

Крок 3. Якщо $F_{k-1} \neq \varnothing$, тоді перейти до кроку 4, у протилежному випадку – до кроку 8.

Крок 4. Виконати генерацію кандидатів C_k на основі F_{k-1}.

Крок 5. Для всіх транзакцій $t \in D$ виконати кроки 5.1–5.2.

Крок 5.1 Виконати видалення надлишкових правил з C_k, одержавши множини $C_t \subseteq C_k$.

Крок 5.2. Для всіх кандидатів $c \in C_t$: установити c.count = c.count +1.

Крок 6. Виконати відбір кандидатів: $F_k = \{c \in C_k \mid c.\text{count} \geq \text{minsupport}\}$.

Крок 7. Установити: $k = k+1$. Перейти до кроку 3.

Крок 8. Зупинення. Повернути як результат $\bigcup F_k$.

Функція генерації кандидатів. У даному випадку немає ніякої необхідності знову звертатися до бази даних. Для того, щоб одержати k-елементні набори, скористаємося $(k-1)$-елементними наборами, що були визначені на попередньому кроці і є такими, що часто зустрічаються.

Відзначимо, що вихідний набір зберігається в упорядкованому виді. Генерація кандидатів також буде складатися з двох кроків.

Крок 1. Об'єднання. Кожен кандидат C_k буде формуватися шляхом розширення набору, що часто зустрічається, розміром $(k-1)$ додаванням елемента з іншого $(k-1)$-елементного набору: включити в C_k ті елементи $p_1, p_2, ..., p_{k-1}, q_{k-1}$ з F_{k-1}: $p, q \in F_{k-1}$, для яких $p_1 = q_1, p_2 = q_2, ..., p_{k-2} = q_{k-2}, p_{k-1} < q_{k-1}$.

Крок 2. Видалення надлишкових правил. На підставі властивості антимонотонності, варто видалити всі набори $c \in C_k$, якщо хоча б одна з його $(k-1)$ підмножин не є такою, що часто зустрічається.

Після генерації кандидатів наступною задачею є підрахунок підтримки для кожного кандидата. Очевидно, що кількість кандидатів може бути дуже великою і потрібний ефективний спосіб підрахунку. Самий тривіальний спосіб – порівняти кожну транзакцію з кожним кандидатом. Але це далеко не краще рішення. Набагато швидше й ефективніше використовувати підхід, заснований на збереженні кандидатів у ґеш-дереві. Внутрішні вузли дерева містять ґеш-таблиці з покажчиками на нащадків, а листи – на кандидатів.

Підтримку для кожного кандидата легко обчислити використовуючи ґеш-дерево. Для цього потрібно пропустити кожну транзакцію через дерево і збільшити лічильники для тих кандидатів, чиї елементи також містяться й у транзакції, тобто $C_k \cap T_i = C_k$. На кореневому рівні ґеш-функция застосовується до кожного елемента з транзакції. Далі, на другому рівні, ґеш-функция застосовується до других елементів і т. д. На k-рівні ґешується k-й елемент. І так доти, поки не досягнемо листа. Якщо кандидат, що зберігається в листі, є підмножиною розглянутої транзакції, тоді збільшуємо лічильник підтримки цього кандидата на одиницю.

Після того, як кожна транзакція з вихідного набору даних пропущена через дерево, можна перевірити чи задовольняють значення підтримки кандидатів мінімальному порогові. Кандидати, для яких ця умова виконується, переносяться в розряд таких, що часто зустрічаються. Крім того, варто запам'ятати і підтримку набору, вона буде корисною при витягу правил. Ці ж дії застосовуються для знаходження (k+1)-елементних наборів і т. д.

Витягання правил виконують після того, як знайдені всі набори елементів, що часто зустрічаються. Це менш трудомістка задача. По-перше, для підрахунку вірогідності правила досить знати підтримку самого набору і множини, що лежить в умові правила.

Наприклад, є набір $\{A, B, C\}$, що часто зустрічається, і потрібно підрахувати вірогідність для правила $AB \rightarrow C$. Підтримка самого набору відома, але і його множина $\{A, B\}$, що лежить в умові правила, також є такою, що часто зустрічається, в силу властивості антимонотонності, і значить його підтримка є відомою. Тоді легко можна підрахувати вірогідність. Це рятує нас від небажаного перегляду бази транзакцій, що був би потрібний у тому випадку, якби ця підтримка була невідомою.

Щоб витягти правило з набору F, що часто зустрічається, варто знайти всі його непорожні підмножини. І для кожної підмножини s ми зможемо сформулювати правило $s \rightarrow (F–s)$, якщо вірогідність правила $\text{conf}(s \rightarrow (F–s)) = \text{supp}(F)/\text{supp}(s)$ не менше порога minconf.

Зазначимо, що чисельник залишається постійним. Тоді вірогідність має мінімальне значення, якщо знаменник має максима-

льне значення, а це відбувається в тому випадку, коли в умові правила є набір, що складається з одного елемента. Усі супермножини даної множини мають меншу або рівну підтримку і, відповідно, більше значення вірогідності. Ця властивість може бути використана при витягу правил. Якщо ми почнемо витягати правила, розглядаючи спочатку тільки один елемент в умові правила, і це правило має необхідну підтримку, тоді всі правила, де в умові стоять супермножини цього елемента, також мають значення вірогідності вище заданого порога.

Наприклад, якщо правило $A \rightarrow BCDE$ задовольняє мінімальному порогові вірогідності minconf, тоді $AB \rightarrow CDE$ також задовольняє.

Для того, щоб витягти всі правила використовується рекурсивна процедура. Важливе зауваження: будь-яке правило, складене з набору, що часто зустрічається, повинне містити всі елементи набору. Наприклад, якщо набір складається з елементів $\{A, B, C\}$, то правило $A \rightarrow B$ не повинне розглядатися.

2.2 Дерева рішень

Дерева рішень (дерева вирішальних правил) – один з методів автоматичного аналізу даних, що задає спосіб подання правил виду «Якщо – то» в ієрархічній послідовній структурі, де кожному об'єкту відповідає єдиний вузол, що дає рішення.

2.2.1 Основні поняття теорії дерев рішень

Основними поняттями теорії дерев рішень, є:

– **об'єкт** – приклад, шаблон, спостереження;

– **атрибут** – ознака, незалежна змінна, властивість;

– **мітка класу** – залежна (цільова) змінна, ознака, що визначає клас об'єкта;

– **вузол** – внутрішній вузол дерева, вузол перевірки;

– **лист** – кінцевий вузол дерева, вузол рішення;

– **перевірка** (test) – умова у вузлі.

Перші ідеї створення дерев рішень запропоновано у роботах П. Ховленда (P. Hoveland) та Е. Ханта (E. Hunt) наприкінці 50-х років XX століття.

Область застосування дерев рішень у даний час є дуже широкою, але всі задачі, розв'язувані цим апаратом, можуть бути об'єднані в такі три класи:

1) опис даних: дерева рішень дозволяють зберігати дані про об'єкти в компактній формі;

2) класифікація: дерева рішень відмінно справляються з задачами класифікації, тобто з віднесенням об'єктів до одного з заздалегідь відомих класів. Цільова змінна повинна мати дискретні значення;

3) регресія: якщо цільова змінна має неперервні значення, дерева рішень дозволяють установити залежність цільової змінної від незалежних (вхідних) змінних. Наприклад, до цього класу відносяться задачі чисельного прогнозування (передбачення значень цільової змінної).

2.2.2 Побудова дерева рішень

Нехай нам задана деяка навчаюча множина T, що містить об'єкти (приклади), кожний з яких характеризується m атрибутами, причому один з них указує на приналежність об'єкта до визначеного класу. Тоді дерево рішень може бути побудоване в результаті виконання таких дій.

Нехай через C_1, C_2, ..., C_k позначено класи (значення міток класів), тоді існують три ситуації:

– множина T містить один або більше прикладів, що відносяться до одного класу C_k. Тоді дерево рішень для T – це лист, що визначає клас C_k;

– множина T не містить жодного приклада, тобто порожня множина. Тоді це знову лист, і клас, асоційований з листом, вибирається з іншої множини відмінного від T, скажімо, з множини, асоційованої з батьком;

– множина T містить приклади, що відносяться до різних класів. У цьому випадку варто розбити множину T на деякі підмножини. Для цього вибирається одна з ознак, що має два і більше відмінних одне від одного значень O_1, O_2, ..., O_n. T розбиває-

ся на підмножини T_1, T_2, ..., T_n, де кожна підмножина T_i містить усі приклади, що мають значення O_i для обраної ознаки. Ця процедура буде рекурсивно продовжуватися доти, поки кінцева множина не буде складатися з прикладів, що відносяться до одного й того ж самого класу.

Вищеописана процедура лежить в основі багатьох сучасних методів побудови дерев рішень, цей метод відомий ще під назвою **поділ і захоплення** (divide and conquer). Очевидно, що при використанні цього методу, побудова дерева рішень буде відбувається зверху вниз.

Оскільки всі об'єкти були заздалегідь віднесені до відомих нам класів, такий процес побудови дерева рішень називається **навчанням із учителем** (supervised learning). Процес навчання також називають **індуктивним навчанням або індукцією дерев** (tree induction).

На сьогоднішній день існує значне число методів, що реалізують дерева рішень: **CART**, **C4.5**, **NewId**, **ITrule**, **CHAID**, **CN2** та ін. Але найбільше поширення і популярність одержали:

– **C4.5** – метод побудови дерева рішень, де кількість нащадків у вузла є необмеженою. Не вміє працювати з неперервними цільовими змінними, тому вирішує тільки задачі класифікації;

– **CART** (Classification and Regression Tree) – метод побудови **бінарного дерева рішень** – дихотомічної класифікаційної моделі. Кожен вузол дерева при розбитті має тільки двох нащадків. Як видно з назви методу, він вирішує задачі класифікації і регресії.

Більшість з відомих методів є «**жадібними методами**». Якщо один раз був обраний атрибут, і за ним було зроблено розбиття на підмножини, то метод не може повернутися назад і вибрати інший атрибут, що дав би краще розбиття. І тому на етапі побудови не можна сказати чи дасть обраний атрибут, в остаточному підсумку, оптимальне розбиття.

При побудові дерев рішень особлива увага приділяється вибору критерію атрибута, за яким піде розбиття, зупинки навчання і відсікання гілок.

Правило вибору ознаки для розбиття. Для побудови дерева на кожному внутрішньому вузлі необхідно знайти таку умову (перевірку), яка б розбивала множину, асоційовану з цим вузлом, на підмножини. У якості такої перевірки повинний бути обраний один з атрибутів. Загальне **правило для вибору атрибута** можна

сформулювати в такий спосіб: обраний атрибут повинний розбити множину так, щоб одержувані в підсумку підмножини складалися з об'єктів, що належать до одного класу, або були максимально наближені до цього, тобто кількість об'єктів з інших класів («домішок») у кожній з цих множин була якнайменшою. Відомі різні критерії:

– **теоретико-інформаційний критерій** для вибору найбільш придатного атрибута запропонований Р. Куінленом (R. Quinlean)і використовується в методі С4.5 – удосконаленій версії методу ID3 (Iterative Dichotomizer):

$$\text{Gain}(X) = \text{Info}(T) - \text{Info}_X(T),$$

$$\text{Info}_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \text{Info}(T_i),$$

де Info(T) – ентропія множини T, а $|T|$ – **потужність множини** T – кількість прикладів у множині T.

Множини T_1, T_2, ..., T_n, отримані при розбитті вихідної множини T за перевіркою X. Вибирається атрибут, що дає максимальне значення за критерієм Gain(X);

– **статистичний критерій** – **індекс Gini** (на честь італійського економіста С. Gini) оцінює «відстань» між розподілами класів і використовується в методі CART, запропонованому Л. Брейманом (L. Breiman):

$$\text{Gini}(c) = 1 - \sum_{j=1}^{k} p_j^2,$$

де c – поточний вузол, а p_j – імовірність класу j у вузлі c.

Правило зупинення визначає, чи розбивати далі вузол або позначити його як лист. На додаток до основного методу побудови дерев рішень були запропоновані такі правила:

– **раннє зупинення** (prepruning) – використання статистичних методів для оцінки доцільності подальшого розбиття. Рання зупинка процесу побудови приваблива в плані економії часу навчання, але цей підхід будує менш точні класифікаційні моделі і тому рання зупинка вкрай небажана. Рекомендується замість зупинки використовувати відсікання;

– **обмеження глибини дерева**: подальшу побудову зупиняють, якщо розбиття веде до дерева з глибиною, яка перевищує задане значення;

– **розбиття повинне бути нетривіальним**, тобто вузли, що вийшли в результаті, повинні містити не менш заданої кількості прикладів.

Цей список евристичних правил можна продовжити, але на сьогодні не існує такого правила, яке б мало велику практичну цінність. До цього питання варто підходити обережно, оскільки більшість цих правил є застосовними лише в якихось окремих випадках.

Правило відсікання визначає, яким чином гілки дерева повинні відтинатися.

Оскільки бажано мати дерево, що складається з малої кількості вузлів, яким би відповідала велика кількість об'єктів з навчаючої вибірки, виникає задача побудови всіх можливих дерев, що відповідають навчаючій множині, і вибору з них дерева з найменшою глибиною. На жаль, ця задача є NP-повною, що було показано Л. Хайфілем (L. Hyafill) та Р. Ривестом (R. Rivest), і, як відомо, цей клас задач не має ефективних методів рішення. Для вирішення даної проблеми часто застосовується **відсікання гілок** (pruning).

Під **точністю розпізнавання** для дерева рішень розуміється відношення правильно класифікованих об'єктів при навчанні до загальної кількості об'єктів з навчаючої множини, а під **помилкою** – кількість неправильно класифікованих. Припустимо, що нам відомий спосіб оцінки помилки для дерева, гілок і листів. Тоді, можливо використовувати таке просте правило: 1) побудувати дерево; 2) відітнути або замінити піддеревом ті гілки, що не приведуть до зростання помилки.

На відміну від процесу побудови, відтинання гілок відбувається знизу нагору, рухаючи з листів дерева, відзначаючи вузли як листи, або заміняючи їх піддеревом. Хоча відтинання не є панацеєю, але в більшості практичних задач дає гарні результати, що дозволяє говорити про правомірність використання подібної методики.

Іноді навіть усічені дерева можуть бути усе ще складними для сприйняття. У такому випадку можна удатися до методики витягання правил з дерева з подальшим створенням наборів пра-

вил, що описують класи. Для витягу правил необхідно досліджувати всі шляхи від кореня до кожного листа дерева. Кожен такий шлях дасть правило, де умовами будуть перевірки з вузлів, що зустрілися на шляху.

2.2.3 Метод C4.5

Метод C4.5 висуває такі вимоги до структури і значень даних. Дані, необхідні для роботи методу, повинні бути подані у вигляді плоскої таблиці. Вся інформація про об'єкти із предметної області повинна описуватися у виді кінцевого набору ознак. Кожний атрибут повинен мати дискретне або числове значення. Самі атрибути не повинні мінятися від приклада до приклада і кількість атрибутів повинна бути фіксованою для всіх прикладів. Кожен приклад повинний бути асоційований з конкретним класом, тобто один з атрибутів повинний бути обраний як мітка класу. Класи повинні бути дискретними, тобто мати кінцеве число значень. Кожен приклад повинний однозначно відноситися до конкретного класу. Випадки, коли приклади належать до класу з імовірнісними оцінками, виключаються. Кількість класів повинна бути значно меншою кількості прикладів.

Метод побудови дерева.

Нехай нам задана множина прикладів T, де кожний елемент цієї множини описується m атрибутами та мітка класу приймає значення C_1, C_2, ..., C_k.

Наша задача буде полягати в побудові ієрархічної класифікаційної моделі у виді дерева з множини прикладів T.

Процес побудови дерева буде відбуватися зверху вниз. Спочатку створюється корінь дерева, потім нащадки кореня і т. д. На першому кроці ми маємо порожнє дерево (є тільки корінь) і вихідну множину T (асоційовану з коренем). Потрібно розбити вихідну множину на підмножини. Це можна зробити, вибравши один з атрибутів як перевірку. Тоді в результаті розбиття виходять n (по числу значень атрибута) підмножин і, відповідно, створюються n нащадків кореня, кожному з яких співставлено у відповідність свою підмножину, отриману при розбитті множини T. Потім ця процедура рекурсивно застосовується до всіх підмножин (нащадків кореня) і т. д.

Розглянемо докладніше критерій вибору атрибута, за яким повинне піти розгалуження. Очевидно, що в нашому розпорядженні m (за числом атрибутів) можливих варіантів, з яких ми повинні вибрати самий придатний. Деякі методи виключають повторне використання атрибута при побудові дерева, але в нашому випадку ми таких обмежень накладати не будемо. Будь-який з атрибутів можна використовувати необмежену кількість разів при побудові дерева.

Нехай ми маємо перевірку X (у якості перевірки може бути обраний будь-який атрибут), що приймає n значень A_1, A_2, ..., A_n. Тоді розбиття T за перевіркою X дасть нам підмножини T_1, T_2, ..., T_n, при X рівному відповідно A_1, A_2, ..., A_n. Єдина доступна інформація – те, яким чином класи розподілені в множині T і її підмножинах, одержуваних при розбитті за X. Це і використовують при визначенні критерію.

Нехай freq(C_j,S) – кількість прикладів з деякої множини S, що відносяться до того ж самого класу C_j. Тоді імовірність того, що випадково обраний приклад із множини S буде належати до класу C_j:

$$P = \text{freq}(C_j,S) / |S|.$$

Відповідно до теорії інформації, кількість інформації, що міститься в повідомленні, залежить від її імовірності $\log_2(P^{-1})$. Оскільки ми використовуємо логарифм із двійковою основою, то цей вираз дає кількісну оцінку в бітах.

Вираз

$$\text{Info}(T) = -\sum_{j=1}^{k} \frac{\text{freq}(C_j,T)}{|T|} \log_2\left(\frac{\text{freq}(C_j,T)}{|T|}\right)$$

дає оцінку середньої кількості інформації, необхідної для визначення класу приклада з множини T. У термінології теорії інформації цей вираз називається ентропією множини T.

Ту ж оцінку, але тільки вже після розбиття множини T за X, дає вираз:

$$\text{Info}_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \text{Info}(T_i).$$

Тоді критерієм для вибору атрибута буде така формула:

$$\text{Gain}(X) = \text{Info}(T) - \text{Info}_X(T).$$

Даний критерій розраховується для всіх атрибутів. Вибирається атрибут, що максимізує даний вираз. Цей атрибут буде перевіркою в поточному вузлі дерева, а потім за цим атрибутом здійснюється подальша побудова дерева. Тобто у вузлі буде перевірятися значення за цим атрибутом і подальший рух по дереву буде здійснюватися в залежності від отриманої відповіді.

Таке ж міркування можна застосувати до отриманих підмножин T_1, T_2, ..., T_n і продовжити рекурсивно процес побудови дерева, доти, поки у вузлі не виявляться приклади з одного класу.

Якщо в процесі роботи методу отримано вузол, асоційований з порожньою множиною (тобто жоден приклад не потрапив в даний вузол), то він позначається як лист, і як рішення листа обирається клас, що найбільш часто зустрічається у безпосереднього предка даного листа.

З властивостей ентропії нам відомо, що максимально можливе значення ентропії досягається в тому випадку, коли всі його повідомлення є рівноімовірними. У нашому випадку, ентропія $\text{Info}_X(T)$ досягає свого максимуму, коли частота появи класів у прикладах множини T є рівноімовірною. Нам же необхідно вибрати такий атрибут, щоб при розбитті за ним один із класів мав найбільшу імовірність появи. Це можливо в тому випадку, коли ентропія $\text{Info}_X(T)$ буде мати мінімальне значення і, відповідно, критерій $\text{Gain}(X)$ досягне свого максимуму.

У випадку числових атрибутів варто обрати деякий поріг, з яким повинні порівнюватися всі значення атрибута. Нехай числовий атрибут має кінцеве число значень. Позначимо їх $\{v_1, v_2, ..., v_n\}$. Попередньо відсортуємо всі значення. Тоді будь-яке значення, що лежить між v_i і v_{i+1}, поділяє всі приклади на дві множини: ті, котрі лежать ліворуч від цього значення $\{v_1, v_2, ..., v_i\}$, і ті, що праворуч $\{v_{i+1}, v_{i+2}, ..., v_n\}$. Як поріг можна вибрати середнє між значеннями v_i і v_{i+1}: $\text{TH}_i = 0{,}5(v_i + v_{i+1})$.

Таким чином, ми істотно спростили задачу знаходження порога, і привели до розгляду усього $n-1$ потенційних граничних значень $\text{TH}_1, \text{TH}_2, ..., \text{TH}_{n-1}$. Послідовно для всіх потенційних граничних значень визначаються значення критерію $\text{Gain}(\text{TH}_i)$ і се-

ред них вибирається те, що дає максимальне значення критерію. Далі це значення порівнюється зі значеннями критерію, підрахованими для інших атрибутів. Якщо з'ясується, що серед всіх атрибутів даний числовий атрибут має максимальне значення критерію, то як перевірка вибирається саме він.

Слід зазначити, що всі числові тести є бінарними, тобто поділяють вузол дерева на дві гілки.

Класифікація нових прикладів.

Нехай ми маємо дерево рішень і хочемо використовувати його для розпізнавання нового об'єкта. Обхід дерева рішень починається з кореня дерева. На кожному внутрішньому вузлі перевіряється значення об'єкта Y за атрибутом, що відповідає перевірці в даному вузлі, і, у залежності від отриманої відповіді, знаходиться відповідне розгалуження, і по цій дузі рухаємося до вузла, що знаходиться на рівень нижче і т. д. Обхід дерева закінчується як тільки зустрінеться вузол рішення, що і дає назву класу об'єкта Y.

За допомогою розглянутого методу можна одержати дерево рішень, незначним недоліком якого буде гіллястість, якщо приклади подані дуже великою кількістю атрибутів. Цей недолік можна усунути, застосувавши відсікання (pruning) гілок.

Поліпшений критерій розбиття.

Критерій Gain(X) має один недолік – він надає переваги атрибутам, які мають багато значень. У випадку, якщо серед атрибутів зустрінеться такий, котрий для кожного приклада має унікальне значення, то при розбитті множини прикладів за цим атрибутом вийдуть підмножини, що містять тільки по одному прикладу. Оскільки всі ці множини «однозразкові», то і приклад відноситься, відповідно, до одного єдиного класу, тоді $\mathrm{Info}_X(T) = 0$.

Значить даний критерій приймає своє максимальне значення, і безсумнівно, що саме цей атрибут буде обраний методом. Однак, якщо розглянути проблему під іншим кутом – з погляду здібностей передбачення побудованої моделі, то стає очевидним уся марність такої моделі.

Проблема вирішується введенням деякої нормалізації. Нехай суть інформації повідомлення, що відноситься до прикладу, указує не на клас, до якого приклад належить, а на вихід. Тоді, за аналогією з визначенням Info(T), маємо

$$\text{splitInfo}(X) = -\sum_{i=1}^{n} \frac{|T_i|}{|T|} \log_2\left(\frac{|T_i|}{|T|}\right).$$

Цей вираз оцінює потенційну інформацію, одержувану при розбитті множини T на n підмножин.

Розглянемо вираз: gainratio(X) = Gain(X)/splitinfo(X).

Нехай цей вираз є критерієм вибору атрибута.

Очевидно, що атрибут, який має унікальне значення для кожного приклада, не буде високо оцінений критерієм даного виразу. Нехай є k класів, тоді чисельник даного виразу максимально буде дорівнювати $\log_2(k)$ і нехай n – кількість прикладів у навчаючій вибірці й одночасно кількість значень атрибута, тоді знаменник максимально дорівнює $\log_2(n)$. Якщо припустити, що кількість прикладів свідомо більше кількості класів, то знаменник росте швидше, ніж чисельник, і, відповідно, вираз буде мати невелике значення. Таким чином, ми можемо замінити критерій $\text{Info}_X(T)$ на новий критерій gainratio(X), і знову ж вибрати той атрибут, що має максимальне значення за критерієм. Критерій $\text{Info}_X(T)$ використовувався в методі ID3, критерій gainratio(X) введено у модифікованому методі C4.5.

Незважаючи на поліпшення критерію вибору атрибута для розбиття, метод може створювати вузли і листи, що містять незначну кількість прикладів. Щоб уникнути цього, варто скористатися ще одним евристичним правилом: при розбитті множини T, принаймні дві підмножини повинні мати не менше заданої мінімальної кількості прикладів k (k>1); звичайно вона дорівнює 2. У випадку невиконання цього правила, подальше розбиття цієї множини припиняється, і відповідний вузол відзначається як лист. Імовірно, що при такім обмеженні може виникнути ситуація коли приклади, асоційовані з вузлом, відносяться до різних класів. Як рішення для листа обирається клас, що найбільш часто зустрічається у вузлі, якщо ж прикладів рівна кількість з усіх класів, то рішення дає клас, що найбільш часто зустрічається у безпосереднього предка даного листа.

При виборі порога для числових атрибутів, можна ввести доповнення: якщо мінімальне число прикладів у вузлі – k, тоді є сенс розглядати тільки значення TH_1, TH_2, ..., TH_{n-1}, оскільки при

розбитті за першим і останнім k–1 порогам у вузол попадає менше k прикладів.

Пропущені дані.

Метод побудови дерев рішень припускає, що для атрибута, обраного як перевірка, існують усі значення, хоча явно це ніде не стверджувалося. Тобто для будь-якого приклада з навчаючої вибірки існує значення за цим атрибутом. На практиці дані далекі від ідеальних, і часто зустрічаються пропущені, суперечні й аномальні дані.

Одне з можливих рішень – не враховувати приклади з пропущеними значеннями. Варто підкреслити, що вкрай небажано відкидати весь приклад тільки тому, що за одним з атрибутів пропущене значення, оскільки ми ризикуємо втратити багато корисної інформації.

Тоді нам необхідно використовувати процедуру роботи з пропущеними даними.

Нехай T – множина навчальних прикладів та X – перевірка за деяким атрибутом A. Позначимо через U кількість невизначених значень атрибута A. Будемо враховувати тільки ті приклади, у яких існують значення за атрибутом A. Тоді:

$$\mathrm{Info}(T) = -\sum_{j=1}^{k} \frac{\mathrm{freq}(C_j, T)}{|T| - U} \log_2 \left(\frac{\mathrm{freq}(C_j, T)}{|T| - U} \right),$$

$$\mathrm{Info}_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T| - U} \mathrm{Info}(T_i).$$

У цьому випадку при підрахунку freq(C_j, T) враховуються тільки приклади з існуючими значеннями атрибута A.

Тоді критерій Gain(x) можна переписати:

$$\mathrm{Gain}(X) = (|T| - U)(\mathrm{Info}(T) - \mathrm{Info}_X(T)) / |T|.$$

Подібним чином змінюється і критерій gainratio. Якщо перевірка має n вихідних значень, то критерій gainratio розраховується як у випадку, коли вихідна множина розділена на n+1 підмножин.

Нехай тепер перевірка X з вихідними значеннями O_1, O_2, ..., O_n обрана на основі модифікованого критерію Gain.

Треба вирішити, що робити з пропущеними даними. Якщо приклад із множини T з відомим виходом O_i асоційований з під-

множиною T_i, імовірність того, що приклад із множини T_i, дорівнює 1. Нехай тоді кожний приклад з підмножини T_i має вагу, яка вказує імовірність того, що приклад належить T_i. Якщо приклад має значення за атрибутом A, тоді вага дорівнює 1, у протилежному випадку приклад асоціюється з усіма множинами T_1, T_2, ..., T_n, з відповідними вагами $|T_i| / (|T| - U)$, сума яких дорівнює одиниці.

Коротко, цей підхід можна сформулювати в такий спосіб: передбачається, що пропущені значення за атрибутом імовірнісно розподілені пропорційно частоті появи існуючих значень.

Класифікація нових прикладів із пропущеними даними.

Така ж методика застосовується, коли дерево використовується для класифікації нових прикладів. Якщо на якомусь вузлі дерева при виконанні перевірки з'ясовується, що значення відповідного атрибута приклада, який класифікується, є пропущеним, то метод досліджує всі можливі шляхи вниз по дереву і визначає з якою імовірністю приклад відноситься до різних класів. У цьому випадку, «класифікація» – це скоріше розподіл класів. Як тільки розподіл класів установлено, то клас, що має найбільшу імовірність появи, вибирається як відповідь дерева рішень.

Слід зазначити, що крім підходу, використаного в даному методі, застосовуються й інші методики. В остаточному підсумку, успіх від використання того чи іншого методу роботи з пропущеними даними, прямо залежить як від предметної області, так і від самих даних.

2.2.4 CART

CART (Classification And Regression Tree – дерево класифікації і регресії) – метод бінарного дерева рішень, призначений для вирішення задач класифікації і регресії. Вперше опубліковано у 1984 р. Л. Брейманом та ін.

Основними відмінностями методу CART від методів сімейства ID3 є: бінарне подання дерева рішень; функція оцінки якості розбиття; механізм відсікання дерева; метод обробки пропущених значень; побудова дерев регресії.

Бінарне подання дерева рішень – в методі CART кожен вузол дерева рішень має двох нащадків. На кожному кроці побудови дерева правило, формоване у вузлі, поділяє задану множину прик-

ладів (навчаючу вибірку) на дві частини: частину, у якій виконується правило (правий нащадок – right) і частину, у якій правило не виконується (лівий нащадок – left). Для вибору оптимального правила використовується функція оцінювання якості розбиття.

Кожен вузол (структура або клас) повинний мати посилання на двох нащадків Left і Right – аналогічні структури. Також вузол повинний містити ідентифікатор правила, певним чином описувати праву частину правила, містити інформацію про кількість або відношення прикладів кожного класу навчаючої вибірки, що пройшла через вузол, і мати ознаку термінального вузла – листа.

Функція оцінювання якості розбиття – критерій, що дозволяє оцінити якість поточного розбиття.

Навчання дерева рішень відноситься до класу навчання з учителем, тобто навчаюча і тестова вибірки містять **класифікований** набір прикладів. Оцінна функція, використовувана методом CART, базується на інтуїтивній ідеї зменшення нечистоти (невизначеності) у вузлі.

В методі CART ідея нечистоти формалізована в індексі Gini(T), де T – набір даних, що містить дані n класів.

Якщо набір T розбивається на дві частини T_1 і T_2 з числом прикладів у кожній N_1 і N_2 відповідно, тоді показник якості розбиття буде дорівнювати:

$$\mathrm{Gini}_{\mathrm{split}}(T) = \frac{N_1}{N}\mathrm{Gini}(T_1) + \frac{N_2}{N}\mathrm{Gini}(T_2).$$

Найкращим вважається те розбиття, для якого $\mathrm{Gini}_{\mathrm{split}}(T)$ є мінімальним.

Позначимо N – число прикладів у вузлі – предку, L, R – число прикладів відповідно в лівому і правому нащадках, l_i та r_i – число екземплярів i-го класу в лівому та правому нащадках, відповідно. Тоді якість розбиття оцінюється за формулою:

$$\mathrm{Gini}_{\mathrm{split}} = \frac{L}{N}\left(1 - \sum_{\mathrm{i}=1}^{\mathrm{n}}\left(\frac{l_i}{L}\right)^2\right) + \frac{R}{N}\left(1 - \sum_{i=1}^{n}\left(\frac{r_i}{R}\right)^2\right) \to \min.$$

Щоб зменшити обсяг обчислень формулу можна перетворити:

$$G_{\mathrm{split}} = \frac{1}{L}\sum_{i=1}^{n} l_i^2 + \frac{1}{R}\sum_{i=1}^{n} r_i^2 \to \max.$$

У результаті, кращим буде те розбиття, для якого величина G_{split} є максимальною.

Правила розбиття – визначають принцип прийняття рішень у вузлах дерева рішень.

Вектор предикторних змінних, подаваний на вхід дерева може містити як числові (порядкові) так і категоріальні змінні. У будь-якому випадку в кожному вузлі розбиття йде тільки за однією змінною. Якщо змінна числового типу, то у вузлі формується правило виду $x_i \le c$, де c – деякий поріг, що найчастіше вибирається як середнє арифметичне двох сусідніх упорядкованих значень змінної x_i навчаючої вибірки. Якщо змінна категоріального типу, то у вузлі формується правило $x_i \in V(x_i)$, де $V(x_i)$ – деяка непорожня підмножина множини значень змінної x_i у навчаючій вибірці. Отже, для n значень числового атрибута метод порівнює n–1 розбиттів, а для категоріального $(2^{n-1} - 1)$. На кожному кроці побудови дерева метод послідовно порівнює всі можливі розбиття для всіх атрибутів і вибирає найкращий атрибут і найкраще розбиття для нього.

Нехай джерело даних, необхідних для роботи методу, подане плоскою таблицею. Кожен рядок таблиці описує один приклад навчаючої / тестової вибірки.

Кожен крок побудови дерева фактично складається із сукупності трьох трудомістких операцій:

1) сортування джерела даних за стовпцем є необхідним для обчислення порога, коли розглянутий у поточний момент часу атрибут має числовий тип. На кожному кроці побудови дерева число сортувань буде як мінімум дорівнювати кількості атрибутів числового типу;

2) поділ джерела даних. Після того, як знайдене найкраще розбиття, необхідно розділити джерело даних відповідно до правила формованого вузла і рекурсивно викликати процедуру побудови для двох половинок джерела даних.

Обидві ці операції пов’язані (якщо діяти прямо) з переміщенням значних обсягів пам’яті. Тут навмисно джерело даних не називається таблицею, тому що можна істотно знизити часові витрати на побудову дерева, якщо використовувати індексоване джерело даних. Звертання до даних у такому джерелі відбувається не прямо, а за допомогою логічних індексів рядків даних. Сор-

тувати і розділяти таке джерело можна з мінімальною втратою продуктивності;

3) обчислення індексів G_{split} для всіх можливих розбиттів – операція, що займає 60–80 % часу виконання програми. Якщо є n – числових атрибутів і m – прикладів у вибірці, то виходить таблиця $n \times (m-1)$ – індексів, що займає великий обсяг пам'яті. Цього можна уникнути, якщо використовувати один стовпець для поточного атрибута й один рядок для кращих (максимальних) індексів для всіх атрибутів. Можна і зовсім використовувати тільки кілька числових значень, одержавши швидкий код, але такий, що погано читається. Значно збільшити продуктивність можна, якщо використовувати, що $L = N - R$, $l_i = n_i - r_i$, а l_i та r_i змінюються завжди і тільки на одиницю при переході на наступний рядок для поточного атрибута. Тобто підрахунок числа класів, а це основна операція, буде виконуватися швидко, якщо знати число екземплярів кожного класу усього в таблиці і при переході на новий рядок таблиці змінювати на одиницю тільки число екземплярів одного класу – класу поточного приклада.

Усі можливі розбиття для категоріальних атрибутів зручно подавати за аналогією з двійковим поданням числа. Якщо атрибут має n – унікальних значень, то буде 2^n – розбиттів. Перше (де всі нулі) і останнє (всі одиниці) нас не цікавлять, одержуємо 2^n-2. І оскільки порядок множин тут теж неважливий, одержуємо $(2^n-2)/2$ або $(2^{n-1} - 1)$ перших (з одиниці) двійкових подань. Якщо $\{A, B, C, D, E\}$ – усі можливі значення деякого атрибута X, то для поточного розбиття, що має подання, скажімо $\{0, 0, 1, 0, 1\}$, одержуємо правило X in $\{C, E\}$ для правої гілки та [not $\{0, 0, 1, 0, 1\}$ = $=\{1, 1, 0, 1, 0\} = X$ in $\{A, B, D\}$] для лівої гілки.

Часто значення атрибута категоріального типу подані в базі як строкові значення. У такому випадку швидше і зручніше створити кеш усіх значень атрибута і працювати не зі значеннями, а з індексами в кеші.

Механізм відсікання дерева (minimal cost-complexity tree pruning) – найбільш серйозна відмінність методу CART від інших методів побудови дерева. CART розглядає відсікання як одержання компромісу між двома проблемами: одержанням дерева оптимального розміру й одержанням точної оцінки імовірності помилкової класифікації.

Основна проблема відсікання – велика кількість усіх можливих відсічених піддерев для одного дерева. Більш точно, якщо бінарне дерево має $|T|$ – листів, тоді існує ~$[1{,}5028369^{|T|}]$ відсічених піддерев. І, якщо дерево має хоча б 1000 листів, тоді число відсічених піддерев стає просто величезним.

Базова ідея методу – не розглядати всі можливі піддерева, обмежуючись тільки кращими представниками відповідно до приведеної нижче оцінки.

Позначимо $|T|$ – число листів дерева, $R(T)$ – імовірність помилки класифікації дерева, що дорівнює відношенню числа неправильно класифікованих прикладів до числа прикладів у навчаючій вибірці. Визначимо повну вартість (оцінку / показник витрата – складність) дерева T як: $C_\alpha(T) = R(T)+\alpha|T|$, де α – деякий параметр, що змінюється від 0 до $+\infty$. Повна вартість дерева складається з двох компонентів – помилки класифікації дерева і штрафу за його складність. Якщо помилка класифікації дерева є незмінною, тоді зі збільшенням α повна вартість дерева буде збільшуватися. Тоді в залежності від α менш гіллясте дерево, що дає велику помилку класифікації, може коштувати менше, ніж те, що дає меншу помилку, але є більш гіллястим.

Визначимо $T_{\max}$ – максимальне за розміром дерево, що має бути обрізане. Якщо ми зафіксуємо значення α, тоді існує найменше мінімізоване піддерево α, що задовольняє таким умовам.

1. $C_\alpha(T(\alpha)) = \min\limits_{T \le T_\alpha} C_\alpha(T)$.

2. Якщо $C\alpha(T) = C\alpha(T(\alpha))$, то $T(\alpha) \subseteq T$.

Перша умова говорить, що не існує такого піддерева дерева $T_{\max}$, що мало б меншу вартість, ніж $T(\alpha)$ при цьому значенні α. Друга умова говорить, що якщо існує більше одного піддерева, що має дану повну вартість, тоді ми вибираємо найменше дерево.

Можна показати, що для будь-якого значення α існує таке найменше мінімізоване піддерево. Але ця задача не є тривіальною. Вона говорить, що не може бути такого, коли два дерева досягають мінімуму повної вартості і вони є непорівнянними, тобто жодне з них не є піддеревом іншого.

Хоча α має нескінченне число значень, існує кінцеве число піддерев дерева $T_{\max}$. Можна побудувати послідовність зменшуваних піддерев дерева $T_{\max}$: $T_1 > T_2 > T_3 > ... > \{t_1\}$, (де t_1 – корене-

вий вузол дерева) таку, що T_k – найменше мінімізоване піддерево для $\alpha \in [\alpha_k, \alpha_{k+1})$. Це важливий результат, тому що це означає, що ми можемо одержати наступне дерево в послідовності, застосувавши відсікання до поточного дерева. Це дозволяє розробити ефективний метод пошуку найменшого мінімізованого піддерева при різних значеннях α. Перше дерево в цій послідовності – найменше піддерево дерева $T_{\max}$, що має таку ж помилку класифікації, як і $T_{\max}$, тобто $T_1 = T(\alpha = 0)$. Якщо розбиття йде доти, поки в кожному вузлі залишиться тільки один клас, то $T_1 = T_{\max}$, але, оскільки часто застосовуються методи ранньої зупинки (prepruning), тоді може існувати піддерево дерева $T_{\max}$, що має таку ж помилку класифікації.

Метод обчислення T_1 з $T_{\max}$ є простим. Знайти будь-яку пару листів із загальним предком, що можуть бути об'єднані, тобто відсічені в батьківський вузол без збільшення помилки класифікації: $R(t) = R(l) + R(r)$, де r та l – листи вузла t. Продовжувати доти, поки таких пар більше не залишиться. Так ми одержимо дерево, що має таку ж вартість як $T_{\max}$ при $\alpha = 0$, але менш гіллясте, ніж $T_{\max}$.

Позначимо як T_t гілку дерева T з кореневим вузлом t.

Якщо ми відітнемо у вузлі t, тоді його внесок у повну вартість дерева $T - T_t$ стане $C_\alpha(\{t\}) = R(t) + \alpha$, де $R(t) = r(t)p(t)$, $r(t)$ – це помилка класифікації вузла t і $p(t)$ – пропорція випадків, що пройшли через вузол t. Альтернативний варіант: $R(t) = m/n$, де m – число прикладів, класифікованих некоректно, а n – загальне число класифікованих прикладів для всього дерева.

Внесок T_t у повну вартість дерева T складе $C_\alpha(T_t) = R(T_t) + \alpha|T_t|$, де $R(T_t) = \sum_{t' \in T_t} R(t')$.

Дерево $T - T_t$ буде кращим ніж T, коли $C_\alpha(\{t\}) = C_\alpha(T_t)$, оскільки при цій величині α вони мають однакову вартість, але $T - T_t$ найменше з двох. Коли $C_\alpha(\{t\}) = C_\alpha(T_t)$ ми одержуємо: $R(T_t) + \alpha|T_t| = R(t) + \alpha$, вирішуючи для α, одержуємо: $\alpha = (R(t) - R(T_t))/(|T_t| - 1)$.

Оскільки для будь-якого вузла t у T_1, якщо ми збільшуємо α, тоді коли $\alpha = (R(t) - R(T_{1,t}))/(|T_{1,t}| - 1)$, дерево, отримане відсіканням у вузлі t, буде кращим ніж T_1.

Обчислимо це значення α для кожного вузла в дереві T_1, і потім виберемо слабкі зв'язки (їх може бути більше одного), тобто вузли для яких величина $g(t)=(R(t)–R(T_{1,t}))/(|T_{1,t}|–1)$ є найменшою. Ми відтинаємо T_1 у цих вузлах, щоб одержати T_2 – наступне дерево в послідовності. Потім ми продовжуємо цей процес для отриманого дерева і так поки ми не одержимо кореневий вузол (дерево в якого є тільки один вузол).

Метод обчислення послідовності дерев.

Крок 1. Установити: $T_1 = T(\alpha=0)$, $\alpha_1 = 0$, $k = 1$.

Крок 2. Поки T_k більше ніж дерево, що складається тільки з одного вузла – кореня, виконувати кроки 2.1–2.4.

Крок 2.1 Для всіх нетермінальних вузлів у $t \in T_k$:

$$g_k(t) = (R(t) - R(T_{k,t}))/(|T_{k,t}| - 1).$$

Крок 2.2 Установити: $\alpha_{k+1} = \min_t g_k(t)$.

Крок 2.3 Обійти униз усі вузли й обрізати ті, де $g_k(t)=\alpha_{k+1}$, щоб одержати T_{k+1}.

Крок 2.4 Установити: $k = k + 1$. Перейти до кроку 2.

Вузли необхідно обходити вниз, щоб не відтинати вузли, що відсічуться самі собою, у результаті відсікання n-го предка.

Вибір фінального дерева.

Отже, ми маємо послідовність дерев і нам необхідно вибрати краще дерево з неї – те, що ми і будемо використовувати надалі. Найбільш очевидним є вибір фінального дерева через тестування на тестовій вибірці. Дерево, що дало мінімальну помилку класифікації, і буде кращим. Однак, це не єдиний можливий шлях.

Природно, якість тестування багато в чому залежить від обсягу тестової вибірки і рівномірності даних, що потрапили в навчаючу і тестову вибірки.

Часто можна спостерігати, що послідовність дерев дає помилки близькі одна до одної. Ця довга плоска послідовність є дуже чуттєвою до даних, що будуть обрані як тестова вибірка. Щоб зменшити цю нестабільність CART використовує **1–SE правило**: вибирається мінімальне за розміром дерево з R^{ts} у межах інтервалу [min R^{ts}, min R^{ts} + SE], де R^{ts} – помилка класифікації дерева, SE – стандартна помилка, що є оцінкою реальної помилки:

$SE(R^{ts}) = (R^{ts}(1–R^{ts}) / n_{test})^{0,5}$, де n_{test} – число прикладів у тестовій вибірці.

Перехресна перевірка (V-fold cross-validation) – найоригінальніша і найскладніша частина методу CART. Цей шлях вибору фінального дерева використовується, коли набір даних для навчання малий або кожний запис у ньому по своєму унікальний так, що ми не можемо виділити вибірку для навчання і вибірку для тестування.

У такому випадку будуємо дерево на всіх даних, обчислюємо α_1, α_2, ..., α_k та $T_1 > T_2 > ... > T_N$. Позначимо T_k – найменше мінімізоване піддерево для $\alpha \in [\alpha_k; \alpha_{k+1})$.

Тепер ми хочемо вибрати дерево з послідовності, але уже використовували всі наявні дані. Хитрість у тому, що ми збираємося обчислити помилку дерева T_k із послідовності непрямим шляхом.

Крок 1. Установимо: $\beta_1 = 0$, $\beta_2 = \sqrt{\alpha_2 \alpha_3}$, $\beta_3 = \sqrt{\alpha_3 \alpha_4}$, ..., $\beta_{N-1} = \sqrt{\alpha_{N-1} \alpha_N}$, $\beta_N = \infty$. Вважається, що β_k буде типовим значенням для $[\alpha_k; \alpha_{k+1})$ і, отже, як значення відповідає T_k.

Крок 2. Розділимо весь набір даних на V груп однакового розміру G_1, G_2, ..., G_V. Брейман рекомендує брати $V = 10$. Потім для кожної групи G_i:

Крок 2.1 Обчислити послідовність дерев за допомогою описаного вище механізму відсікання на всіх даних, крім G_i, і визначити $T^{(i)}(\beta_1)$, $T^{(i)}(\beta_2)$, ..., $T^{(i)}(\beta_N)$ для цієї послідовності.

Крок 2.2 Обчислити помилку дерева $T^{(i)}(\beta_k)$ на G_i. Тут $T^{(i)}(\beta_k)$ означає найменше мінімізоване піддерево з послідовності, побудоване на всіх даних, крім G_i для $\alpha = \beta_k$.

Крок 3. Для кожного β_k підсумовувати помилку $T^{(i)}(\beta_k)$ за всіма G_i ($i = 1, ..., V$). Нехай β_h буде з найменшою загальною помилкою. Оскільки β_h відповідає дереву T_h, ми вибираємо T_h з послідовності, побудованої на всіх даних як фінальне дерево. Показник помилки, обчислений за допомогою перехресної перевірки, можна використовувати як оцінку помилки дерева.

Альтернативний шлях – щоб вибрати фінальне дерево з послідовності на останньому кроці можна знову використовувати 1–SE правило.

Метод обробки пропущених значень.

Більшість методів інтелектуального аналізу даних припускає відсутність пропущених значень. У практичному аналізі це припущення часто є невірним. До пропущених даних можуть привести такі причини:

– респондент не бажає відповідати на деякі з поставлених питань;

– помилки при введенні даних;

– об'єднання не зовсім еквівалентних наборів даних.

Найбільш загальне рішення – відкинути дані, що містять один чи кілька порожніх атрибутів. Однак це рішення має свої недоліки:

1) зсув даних. Якщо викинуті дані лежать трохи осторонь від залишених, тоді аналіз може дати упереджені результати;

2) зменшення потужності. Може виникнути ситуація, коли прийдеться викинути багато даних. У такому випадку точність прогнозу сильно зменшується.

Якщо необхідно будувати і використовувати дерево на неповних даних, тоді необхідно вирішити питання:

1) як визначити якість розбиття?

2) у яку гілку необхідно послати спостереження, якщо пропущено змінну, на яку приходиться найкраще розбиття (побудова дерева і тренування)?

Помітимо, що спостереження з пропущеною міткою класу є непотрібним для побудови дерева і буде викинуто.

Щоб визначити якість розбиття CART просто ігнорує пропущені значення. Для вирішення, по якому шляху посилати спостереження з пропущеної змінної, утримуючої найкраще розбиття, CART обчислює так називане сурогатне розбиття. Воно створює найбільш близькі до кращої підмножини прикладів у поточному вузлі. Щоб визначити значення альтернативного розбиття як сурогатного ми створюємо таку крос-таблицю.

$p(l^*, l')$	$p(l^*, r')$
$p(r^*, l')$	$p(r^*, r')$

У цій таблиці $p(l^*,l')$ позначає пропорцію випадків, що будуть послані в ліву гілку при кращому s^* і альтернативній розбит-

ті $s^{'}$ і аналогічно для $p(r^{*},r^{'})$, оскільки $p(l^{*},l^{'}) + p(r^{*},r^{'})$ – пропорція випадків, що послані в ту саму гілку для обох розбиттів. Це міра подібності розбиттів або інакше це говорить, наскільки добре ми прогнозуємо шлях, по якому посланий випадок найкращим розбиттям, дивлячись на альтернативне розбиття. Якщо $p(l^{*},l^{'}) + p(r^{*},r^{'}) < 0{,}5$, тоді ми можемо одержати кращий сурогат, помінявши ліву і праву гілки для альтернативного розбиття. Крім того, необхідно помітити, що пропорції в таблиці обчислені, коли обидві змінні (сурогатна й альтернативна) є спостережними.

Альтернативні розбиття з $p(l^{*},l^{'}) + p(r^{*},r^{'}) > \max(p(l^{*}), p(r^{*}))$ відсортовані в спадному порядку подібності. Тепер, якщо пропущено змінну кращого розбиття, тоді використовуємо першу із сурогатних у списку, якщо пропущена вона, тоді наступну і т. д. Якщо пропущено всі сурогатні змінні, то використовуємо $\max(p(l^{*}), p(r^{*}))$.

Регресія.

Побудова дерева регресії багато в чому є схожою з деревом класифікації. Спочатку ми будуємо дерево максимального розміру, потім обрізаємо дерево до оптимального розміру.

Основна перевага дерев у порівнянні з іншими методами регресії – можливість працювати з багатомірними задачами і задачами, у яких є присутньою залежність вихідної змінної від змінних категоріального типу.

Основна ідея – розбиття всього простору на прямокутники, необов'язкового однакового розміру, у яких вихідна змінна вважається постійною. Помітимо, що існує сильна залежність між обсягом навчаючої вибірки і помилкою відповіді дерева.

Процес побудови дерева відбувається послідовно.

На першому кроці ми одержуємо регресійну оцінку просто як константу по всьому простору прикладів. Константу розраховуємо як середнє арифметичне вихідної змінної у навчаючій вибірці. Отже, якщо ми позначимо всі значення вихідної змінної як Y_1, Y_2, ..., Y_n, тоді регресійна оцінка виходить:

$$f(x_i) = \frac{I_R(x_i)}{n} \sum_{i=1}^{n} Y_i,$$

де R – простір навчальних прикладів, n – число прикладів, $I_R(x)$ – індикаторна функція простору – фактично, набір правил, що опи-

сують попадання змінної x_i у простір. Ми розглядаємо простір R як прямокутник. На другому кроці ми поділяємо простір на дві частини. Вибирається деяка змінна x_i і якщо змінна числового типу, тоді ми визначаємо: $R_1=\{x_i \in R: x_i \le a\}$, $R_2=\{x_i \in R: x_i > a\}$.

Якщо x_i категоріального типу з можливими значеннями A_1, A_2, ... , A_q, тоді вибирається деяка підмножина $I \subset \{A_1, ... A_n\}$ і ми визначаємо: $R_1=\{x_i \in R: x_i \in I\}$, $R_2=\{x_i \in R: x_i \in \{A_1, A_2, ..., A_q,\} \setminus I\}$.

Регресійна оцінка приймає вид:

$$f(x_i) = \frac{I_{R_1}(x_i)}{\left|I_1\right|} \sum_{I_1} Y_i + \frac{I_{R_2}(x_i)}{\left|I_2\right|} \sum_{I_2} Y_i,$$

де $I_1 = \{i, x_i \in R_1\}$, $|I_1|$ – число елементів у I_1, $I_2 = \{i, x_i \in R_2\}$, $|I_2|$ – число елементів у I_2.

Краще розбиття вибирається в такий спосіб. Як оцінка тут слугує сума квадратів різниць:

$$E = \sum_{i=1}^{n} \left(Y_i - f(x_i)\right)^2.$$

Вибирається розбиття з мінімальною сумою квадратів різниць.

Ми продовжуємо розбиття доти, поки в кожному підпросторі не залишиться мале число прикладів або сума квадратів різниць не стане менше деякого порога.

Відсікання, вибір фінального дерева відбуваються аналогічно дереву класифікації. Єдина відмінність – визначення помилки відповіді дерева: $R(f) = E/n$, чи, інакше кажучи, середньоквадратичної помилки відповіді.

Вартість дерева дорівнює: $C_\alpha(f) = R(f) + \alpha|f|$.

Інші операції відбуваються аналогічно дереву класифікації.

Метод CART успішно поєднує у собі якість побудованих моделей і, при вдалій реалізації, високу швидкість їхньої побудови. Містить у собі унікальні методики обробки пропущених значень і побудови оптимального дерева сукупністю методів cost-complexity pruning і V-fold cross-validation.

Існує також кілька модифікованих **версій методу CART**.

Метод IndCART, є частиною пакета Ind і відрізняється від CART використанням іншого способу обробки пропущених зна-

чень, не здійснює регресійну частину методу CART і має інші параметри відсікання.

Метод DB-CART (distribution based CART) базується на такій ідеї: замість того щоб використовувати навчаючий набір даних для визначення розбиттів, використовуємо його для оцінки розподілу вхідних і вихідних значень і потім використовуємо цю оцінку, щоб визначити розбиття. Стверджується, що ця ідея дає значне зменшення помилки класифікації, у порівнянні зі стандартними методами побудови дерева.

Перевагами дерев рішень є: швидкий процес навчання; генерація правил в галузях, де експерту важко формалізувати свої знання; витяг правил природною мовою; інтуїтивно зрозуміла класифікаційна модель; висока точність прогнозу, порівнянний з іншими методами (статистика, нейронні мережі); побудова непараметричних моделей.

У силу цих і багатьох інших причин, методологія дерев рішень є важливим інструментом у роботі кожного фахівця, що займається аналізом даних, поза залежністю від того практик він або теоретик.

2.3 Нейромережеві технології інтелектуального аналізу даних

2.3.1 Структура штучного нейрона

У загальному випадку нейронна мережа являє собою сукупність нейронів, зв'язаних певним чином. Основними відмінностями нейромережевих моделей є способи зв'язку нейронів між собою, функції нейроелементів, а також механізми та напрямки розповсюдження сигналів по мережі. Базовим елементом нейронної мережі є формальний нейрон (рис. 2.1), що має декілька входів і один вихід.

Кожному *i*-му входу ставиться у відповідність **ваговий коефіцієнт** (синоптична вага) w_i, що відповідає його значущості.

Дискримінантна функція φ (вагова функція) нейрона перетворює зважені входи і подає їх **на функцію активації** ψ (передатна функція), результат обчислення якої є виходом нейрона. Таким чином, формальний нейрон реалізує скалярну функцію векторного аргументу.

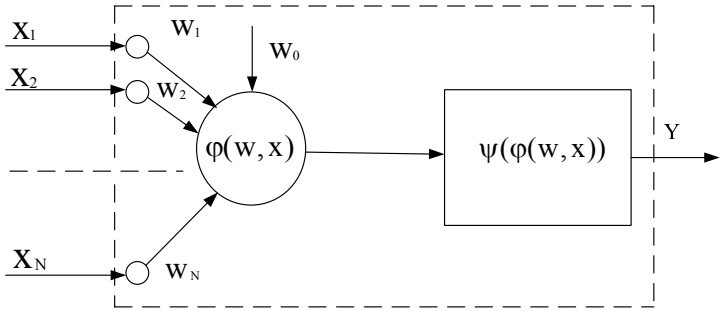


Рисунок 2.1 – Формальний нейрон

Отже, математична модель штучного нейрона описується співвідношенням:

$$y = \psi\ (\varphi(w, x)),$$

де y – значення виходу нейрона; ψ – функція активації; φ – дискримінантна функція; w – вектор, що містить значення вагових коефіцієнтів і значення зсуву (порогове значення); x – вектор вхідних аргументів.

2.3.1.1 Дискримінантні функції нейроелементів

В якості дискримінантних функцій, як правило, використовуються:

1) зважена сума: $\varphi(w,x) = \sum_{j=1}^{N} w_j x_j + w_0$;

2) зважений добуток: $\varphi(w,x) = w_0 \prod_{j=1}^{N} w_j x_j = \prod_{j=0}^{N} w_j \prod_{j=1}^{N} x_j$;

3) функція відстані: $\varphi(w,x) = w_0 \sum_{j=1}^{N} \left(w_j - x_j\right)^2$;

4) максимальне значення зважених входів:

$$\varphi(w,x) = \max_j \left(w_j x_j\right);$$

5) мінімальне значення зважених входів:

$$\varphi(w,x) = \min_j \left(w_j x_j\right).$$

2.3.1.2 Функції активації

Функція активації $\psi(x)$, де x – аргумент функції активації, повинна бути обмеженою (на інтервалі значень $x \in (-\infty, a]$ приймати значення y^0, на інтервалі $x \in [b,+\infty)$ – значення y^1, на інтервалі $x \in (a,b)$ приймати значення y: $y^0 \le y \le y^1$, де y^0 і y^1 – деякі постійні мінімальне та максимальне значення, a та b – деякі константи, причому: $a \le b$) і монотонної (на інтервалі $x \in (a,b)$ $\Delta\psi(x) = \psi(x+\Delta x) - \psi(x)$) і не повинна змінювати знак при $\Delta x > 0$ та $x, x+\Delta x \in (a,b)$.

В якості функції активації, як правило, застосовуються лінійна, порогова, сігмоідна, функція Гауса, гіперболічний тангенс та інші.

Лінійна функція активації приймає значення в діапазоні $(-\infty;+\infty)$:

$$\psi(x) = cx,$$

де c – деяка константа, як правило, $c = 1$.

Лінійна біполярна з насиченням:

$$\psi(x) = \begin{cases} 1 & \text{при} \quad x > a_2; \\ Kx & \text{при} \quad a_1 \le x \le a_2; \\ -1 & \text{при} \quad x < a_1. \end{cases}$$

Лінійна уніполярна з насиченням:

$$f(x) = \begin{cases} 1 & \text{при} \quad x \ge \dfrac{1}{2a}; \\ ax + 0{,}5 & \text{при} \quad |x| < \dfrac{1}{2a}; \\ 0 & \text{при} \quad x \le -\dfrac{1}{2a}. \end{cases}$$

Незважаючи на те, що лінійні функції є найбільш простими, їх застосування обмежене, в основному, найпростішими штучними нейронними мережами (ШНМ), що не містять прихованих шарів, в яких, крім того, існує лінійна залежність між вхідними і вихідними змінними. Такі мережі мають обмежені можливості. Для багатошарової ж лінійної мережі справедливо наступне.

Оскільки після вхідного оператора на оператор активації надходить сукупність зважених вхідних сигналів, записана, наприклад, в матричному вигляді (більш детально див. нижче) $w_i x$, використання лінійної активаційної функції призводить до того, що на виході другого шару з'явиться сигнал $w_2(w_1 x) = (w_2 w_1)x$. Це означає, що двошарова лінійна мережа еквівалентна одношарової з ваговою матрицею, що дорівнює добутку вагових матриць першого і другого шарів. Звідси випливає, що будь-яка багатошарова лінійна мережа може бути замінена еквівалентною одношарової.

Використання лінійних активаційних функцій не позбавлене сенсу, в багатошарових ШНМ для розширення можливостей мережі застосовують нелінійні функції активації.

Порогова (функція Хевісайда) функція активації приймає значення в діапазоні [0;1]:

$$\psi(x) = \begin{cases} 1, \text{якщо } x \geq 0, \\ 0, \text{якщо } x < 0. \end{cases} \text{ або } \psi(x) = \begin{cases} 1, \text{якщо } x \geq a, \\ 0, \text{якщо } x < a. \end{cases}$$

Порогова знакова (біполярна) функція активації приймає значення в діапазоні [–1; 1]:

$$\psi(x) = \begin{cases} 1, \text{якщо } x \geq 0, \\ -1, \text{якщо } x < 0. \end{cases} \text{ або } \psi(x) = \begin{cases} 1, \text{якщо } x \geq a, \\ -1, \text{якщо } x < a. \end{cases}$$

Дані функції активації застосовувалися в основному в класичних ШНМ. При побудові нових структур ШНМ часто доводиться працювати як із само. активаційною функцією, так і з її першою похідною. У цих випадках необхідним є використання в якості активаційної монотонної диференційованої та обмеженої функції. Особливо важливу роль відіграють такі функції при моделюванні нелінійних залежностей між вхідними і вихідними змінними. Це так звані **сигмоїдальні функції** (sigmoidal squashing functions).

Функція $f(\bullet)$ називається сигмоїдальною, якщо вона є монотонно зростаючою, диференційованою і задовольняє умові

$$\lim_{\lambda \to -\infty} f(\lambda) = k_1, \lim_{\lambda \to \infty} f(\lambda) = k_2, k_1 < k_2.$$

До числа таких функцій належать наступні.

Сигмоїдна логістична (уніполярна) функція приймає значення в діапазоні (0;1):

$$\psi(x) = \frac{1}{1+e^{-cx}},$$

де c – деяка константа, як правило, $c = 1$.

Функція гіперболічного тангенса (біполярна):

$$f_{th}(x) = \tanh(Kx) = \frac{e^{Kx} - e^{-Kx}}{e^{Kx} + e^{-Kx}}.$$

Синусоїдальна функція з насиченням (біполярна):

$$f_{sn}(z) = \begin{cases} 1 & \text{при} \quad x \geq a; \\ \sin x & \text{при} \quad |x| < a; \\ -1 & \text{при} \quad x \leq -a; \end{cases}$$

Косинусоїдальна функція з насиченням (уніполярна):

$$f_{cs}(z) = \begin{cases} 1 & \text{при} \quad x \geq \frac{\pi}{2}; \\ \frac{1}{2}(1+\cos(z-\frac{\pi}{2})) & \text{при} \quad |x| < \frac{\pi}{2}; \\ 0 & \text{при} \quad x \leq -\frac{\pi}{2}. \end{cases}$$

Сигмоїдальну функцію за аналогією з електронними системами можна вважати нелінійної підсилювальної характеристикою штучного нейрона. Центральна область такої функції, що має великий коефіцієнт підсилення, вирішує проблему обробки слабких сигналів, а області з падаючим посиленням на позитивному і від'ємному кінцях служать для обробки сильних збуджень. Таким чином, нейрон функціонує з великим посиленням у широкому діапазоні рівнів вхідного сигналу.

Радіально-базисна функція активації приймає значення в діапазоні $(0;+\infty)$:

$$\psi(x) = e^{-cx^2},$$

де c – деяка константа.

Таким чином, основними параметрами формальних нейронів є:

– кількість входів L;

– значення вагових коефіцієнтів $w_1, w_2, \ldots, w_L$;

– значення зсуву (порогу) w_0;

– використовувана дискримінантна функція;

– вид і параметри функції активації.

Як правило, кількість входів, дискримінантна функція і функція активації визначаються специфікою завдання. Тому настройкою параметрів формального нейрона є значення вагових коефіцієнтів і значення зсуву.

2.3.2 Побудова нейромережевих моделей

Нехай задана початкова вибірка вихідних даних у вигляді (див. п.1.1):

$$< X = \{X_1, X_2, ..., X_L\} = \{X_i\},\ \ Y = \{y_1, y_2, \ldots, y_m\} = \{y_p\} >.$$

Тоді завдання синтезу нейромережевої моделі за заданою вибіркою полягає в ідентифікації її параметрів і структури:

$$\text{НМ} = \text{НМ}(C, W, B, DF, TF),$$

таким чином, щоб значення критерію оптимальності ξ нейромоделі НМ було мінімальним:

$$\xi(\text{НМ}, X, Y) \to \min,$$

де C – матриця, що визначає наявність синаптичних зв'язків між елементами мережі (рецепторами, нейронами); $W = W(C)$ – матриця вагових коефіцієнтів, відповідних зв'язків, присутніх в мережі НМ; $B = B(C)$ – вектор зміщень нейронів мережі; $DF = DF(C)$ – вектор дискримінантних функцій нейроелементів; $TF = TF(C)$ – вектор функцій активації нейронів мережі; $\xi(\text{НС}, X, Y)$ – критерій, що визначає ефективність використання нейромережевої моделі НМ для апроксимації залежності між набором вхідних параметрів X і відповідним йому вектором значень вихідного параметра Y. Як правило, в якості критерію оптимальності нейромоделі використовується квадратичний критерій:

$$\xi = \sum_{p=1}^{m} \left(y_p - y\left(\text{НМ}, Z_p\right)\right)^2 ,$$

де Z_p – набір значень ознак для p-го екземпляру; y(НМ, Z_p) – значення виходу нейромоделі НМ, обчислене для набору значень Z_p.

Процес побудови ефективних нейромережних моделей на основі відомої вибірки вихідних даних може бути представлений у вигляді послідовності етапів:

– вибір системи інформативних ознак;

– структурний синтез;

– параметричний синтез;

– оптимізація побудованої нейромоделі.

2.3.2.1 Вибір системи інформативних ознак

Побудова нейромережевих моделей тісно пов'язана з вирішенням задачі вибору набору вхідних змінних, що володіють найбільшою інформацією про даний конкретний вихід.

Постановку задачі відбору інформативних ознак та методи її вирішення розглянуто у розділі 1.

2.3.2.2 Структурний синтез

На даному етапі формується топологія зв'язків, вибираються нейрони, що надалі визначають принцип функціонування мережі та її ефективність для розв'язання досліджуваної задачі. Так, нейромережі, що мають невелику кількість нейронів і лінійні функції активації, як правило, через свої обмежені апроксимаційні здібності не дозволяють вирішувати реальні практичні завдання. У той же час вибір надлишкової кількості нейронів у мережі призводить до проблеми перенавчання та втрати апроксимаційних властивостей нейромоделі. Оскільки для вирішення більшості практичних задач розпізнавання і оцінювання можуть бути використані нейромережі прямого поширення з нейроелементами, що використовують зважену суму як дискримінантну функцію, вибір виду вагових функцій при структурному синтезі нейромоделей, як правило, не здійснюють.

Головною метою дослідника на етапі структурного синтезу є визначення загального вигляду структури шуканого зв'язку, представленого у вигляді нейромережевої моделі, вихідного параметра і вектора керуючих змінних.

Завдання структурного синтезу нейромоделі полягає в пошуку структури моделі виду $НМ = НМ(C, TF)$, для якої $\xi(НМ, X, Y) \to \min$, де $C = C(L, A)$ – матриця, що визначає наявність синаптичних зв'язків між елементами мережі (рецепторами, нейронами); A – максимально допустима кількість нейронів у мережі.

В даний час не існує стандартних методів, які утворювали б сувору теоретичну базу для вирішення задачі структурного синтезу нейромоделей. Будучи вузловим у процесі синтезу нейромоделей реальних об'єктів, процесів або систем, цей етап не має суворих і закінчених математичних рекомендацій щодо його реалізації. Тому його реалізація вимагає спільної роботи фахівця у відповідній предметній області та аналітика, спрямованої на якомога більш глибоке проникнення у фізичний механізм досліджуваного зв'язку.

У ряді випадків структуру нейромоделі можна вибрати апріорі, виходячи з фізичних, хімічних та емпіричних міркувань про характер взаємодії досліджуваного об'єкта з зовнішнім середовищем. Однак для складних об'єктів, процесів і систем, що характеризуються багатомірністю і нелінійністю, загальних апріорних міркувань може виявитися недостатньо.

У цьому випадку завдання вибору структури зводиться до пошуку математичними методами оптимізації за апостеріорними даними з використанням наявної додаткової інформації. Використання таких методів передбачає формування певних гіпотез про топологію мережі, які, як правило, перевіряють на основі критеріїв, що базуються на наступних ідеях:

– досягнення компромісу між складністю моделі і точністю її оцінювання;

– пошук моделі, найбільш стійкої до варіювання складу вибіркових даних, на основі яких вона оцінюється.

Існуючі методи автоматичного пошуку оптимальної структури нейромережевих моделей, як правило, використовують жадібну стратегію пошуку. Так конструктивні методи починають пошук з мінімально можливою архітектури мережі (нейромере-

жа з мінімальною кількістю шарів, нейронів і міжнейронних зв'язків) і послідовно на кожній ітерації додають нові шари, нейрони і міжнейронні зв'язки. При використанні деструктивних методів на початковій ітерації оцінюється ефективність нейромоделі, що містить максимально допустиму кількість шарів, нейронів і міжнейронних зв'язків, потім у процесі пошуку структура такої моделі скорочується до найбільш прийнятною.

Однак такі методи внаслідок застосування жадібної стратегії досліджують незначну частину простору всіх можливих структур нейромоделей і схильні до потрапляння в локальні оптимуми.

2.3.2.3 Параметричний синтез

На етапі параметричного синтезу відбувається навчання нейромережевої моделі, тобто підбираються такі значення вагових коефіцієнтів мережі, а іноді і параметрів функцій активації, при яких мережа найбільш ефективним чином дозволяє вирішувати поставлену задачу.

Завдання параметричного синтезу нейромоделі заданої структури полягає в пошуку такого набору значень вагових коефіцієнтів і зміщень, при якому досягається мінімум критерію помилки нейромоделі. Навчання НМ полягає в зміні значень синаптичних ваг в результаті послідовного пред'явлення екземплярів навчальної вибірки.

Розрізняють дві концепції навчання нейромереж: навчання з вчителем і навчання без вчителя.

Навчання без учителя передбачає використання для налаштування вагових коефіцієнтів мережі тільки значення вхідних ознак екземплярів навчальної вибірки. При цьому реальні значення вихідного параметра невідомі, а процес навчання полягає в об'єднанні близько розташованих екземплярів у кластери. Таким чином, екземпляри, розташовані досить близько одне до одного в просторі ознак, будуть віднесені до одного класу при їх класифікації за допомогою налаштованої мережі.

При **навчанні з учителем** навчальна вибірка містить значення вхідних ознак, що характеризують даний об'єкт або процес, і значення вихідного параметра. При такому вигляді навчан-

ня мінімізується сума квадратів різниці між модельними та реальними виходами досліджуваного об'єкта, процесу або системи.

В даний час відомі різні методи навчання БНМ прямого поширення, більшість з яких заснований на **правилі Хебба**. Відповідно до цього правила у випадку одночасної активації двох нейронів вага їх зв'язку зростає, в результаті чого часто використовувані зв'язки в нейромережевої моделі посилюються.

Суттєвим недоліком такого підходу є відсутність механізму зменшення значень вагових коефіцієнтів, в результаті чого відбувається значне збільшення значень вагових коефіцієнтів з зростанням кількості екземплярів, які використовуються для навчання, що призводить до некоректної роботи мережі.

Іншим методом навчання нейромережевих моделей є **дельта-правило**, яке може бути використане для налаштування одношарових мереж прямого поширення і полягає в зміні значень вагових коефіцієнтів мережі пропорційно різниці між реальним виходом і виходом мережі.

Більшість методів, що використовуються в даний час для навчання БНМ прямого поширення, є **градієнтними**. Використання градієнтних методів навчання засноване на мінімізації ЦФ (квадратичної помилки), при цьому для корекції значень синаптичних ваг w_{ij} НМ використорують правило:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha \left. \frac{\partial \xi}{\partial w_{ij}} \right|_{w_{ij}=w_{ij}(t)},$$

де $w_{ij}(t+1)$ и $w_{ij}(t)$ – значення вагового коефіцієнта між i-им і j-им нейронами на t-ой та $(t+1)$-ій ітераціях, відповідно.

Таким чином, у загальному випадку навчання нейронних мереж зводиться до задачі багатовимірної нелінійної оптимізації, розв'язуваної, як правило, за допомогою градієнтних методів. Для обчислення значень приватних похідних може бути використаний метод зворотного поширення помилки, який, однак, може бути застосовний тільки для налаштування ваг нейромережевих моделей з диференційованими функціями активації.

У розглянутих вище детермінованих методах навчання використовується жорстка послідовність дій, що виконується за певними правилами.

Поряд з детермінованими методами також можуть бути використані **стохастичні методи** настроювання вагових коефіцієнтів нейромережевої моделі, які виконують параметричний синтез на основі дій, що підкоряються деякому випадковому закону. При використанні таких методів на початкових ітераціях виконують корекції значень вагових коефіцієнтів на великі величини, поступово зменшуючи їх у процесі пошуку. До стохастичних методів відносять: метод випадкового пошуку, метод імітації відпалу, а також методи еволюційного пошуку. Значною перевагою стохастичних методів навчання є відсутність необхідності обчислення значень похідних цільової функції. Однак, оскільки такі методи використовують імовірнісний підхід, не враховуючи при цьому інформації про поверхні функції помилки, рішення (набір параметрів нейромоделі), що забезпечує прийнятну точність, може бути не знайдено.

2.3.2.4 Оптимізація побудованої нейромоделі

На можливість застосування нейромережевих моделей на практиці істотний вплив роблять складність побудованої нейромережі та швидкість обчислення значення цільового параметра по набору даних, що не входить в навчальну вибірку.

Тому актуальним є спрощення структури синтезованої нейромоделі. Завдання оптимізації побудованої нейромережевої моделі полягає в пошуку таких нових значень $C' \subseteq C$, W', B', DF', TF', при яких досягаються оптимальні значення заданих критеріїв оптимальності $\xi_1, \xi_2, \ldots, \xi_K$, враховують основні характеристики нейромоделі, де K – кількість цільових критеріїв.

В даний час існує два основні підходи до оптимізації структури синтезованою нейромережевої моделі:

– методи видалення зв’язків;

– методи видалення нейронів.

Методи видалення зв’язків зменшують кількість елементів матриці ваг нейромережі, спрощуючи таким чином її структуру. При використанні таких методів видаляються зв’язки, які мають найменші значення синаптичних ваг або надають найменший вплив на ефективність функціонування нейромережевої моделі. Виділяють такі методи видалення зв’язків: метод обнуління най-

менших ваг, метод оптимального руйнування нейромережі, метод оптимальної розбудови мережі.

Методи другої групи спрощують структуру нейромоделі шляхом виключення з неї нейронів, видалення яких не призводить до значного збільшення помилки мережі. Таким чином, при використанні **методів видалення нейронів** додатково необхідно оцінювати їх значимість шляхом обчислення помилки спрощеної моделі. До таких методів відносять: метод видалення нейронів з урахуванням їх важливості і метод видалення нейронів з використанням вартісної функції.

Проте існуючі підходи спрощення нейромоделей, як правило, передбачають використання штрафних функцій, що в багатьох випадках не дозволяє отримувати оптимальну структуру нейромоделі, а іноді призводить до неможливості збіжності процесу спрощення нейромоделі або до видалення значущих зв'язків або нейронів.

2.3.3 Навчання одношарового персептрона

Одношаровий персептрон є одним з найпростіших варіантів НМ і містить лише один нейрон (див. рис. 2.1). Будучи самостійною моделлю НМ з одного боку, одношаровий персептрон (формальний нейрон) є основним конструкційним елементом для більшості моделей НМ, з іншого боку.

Метод синтезу моделей на основі одношарового дискретного персептрона має наступний вигляд.

Крок 1. Вагам w_i, $i = 1, 2, ..., N$, та порогу w_0 присвоюються випадкові значення.

Крок 2. Пред'являються черговий екземпляр $x^p=\{x_{1p}, ..., x_{Lp}\}$ з навчальної множини, $p = 1, 2, ..., m$, де m – кількість екземплярів у навчальній вибірці, і бажаний вихід y^{s*}.

Крок 3. Обчислюється реальне значення на виході персептрона $y = \psi(\varphi(w, x))$, де $w = \{w_0, w_1, w_2, ..., w_L\}$ – набір ваг, що утворюють пам'ять нейрона.

Крок 4. Коригуються ваги персептрона:

$$w_i = w_i + \alpha\ (y^{s*} - y^s)x_{ip},\ i = 0, 1, ..., L,\ x_0 = 1,$$

де α – позитивний коригуючий приріст (крок навчання).

Крок 5. Якщо досягнуто збіжність, то процедура корекції ваг закінчується, інакше – перехід до кроку 2.

Відповідно до даного методу спочатку здійснюється ініціалізація параметрів персептрона випадковими значеннями. Потім по черзі пред'являються образи з відомою класифікацією, вибрані з навчальної множини, і коригуються ваги відповідно до формул кроків 3 та 4. Величина корекції визначається позитивним коригуючим приростом, конкретне значення якого вибирається досить великим, щоб швидше проводилася корекція вагів, і в той же час досить малим, щоб не допустити надмірного зростання значень ваг. Процедура навчання триває до тих пір, поки не буде досягнута збіжність, тобто поки не будуть отримані ваги, що забезпечують правильну класифікацію для всіх образів з навчальної множини.

У тому випадку, коли навчальні вибірки розділити гіперплоскістю неможливо, для навчання персептрона можна використовувати метод Уїдроу-Хоффа, здатний мінімізувати середньоквадратичні помилки між бажаними і реальними виходами мережі для навчальних даних. Цей метод також можна застосовувати для навчання одношарового речового персептрона. Метод Уїдроу-Хоффа можна записати в тому ж вигляді, що й вищеописаний метод, припускаючи, що у вузлах персептрона нелінійні елементи відсутні, а коригуючий приріст в процесі ітерацій поступово зменшується до нуля.

Якщо для вирішення задачі розпізнавання образів використовується дискретний персептрон, вирішальне правило відносить вхідний образ до класу K_1, якщо вихід персептрона дорівнює 1, і до класу K_0 – в іншому випадку. У разі якщо для вирішення задач розпізнавання образів використовується дійсний персептрон, вирішальне правило відносить вхідний образ до класу K_1, якщо вихід мережі більше 0,5, і до класу K_0 – в іншому випадку.

Незважаючи на досить обмежені здібності кожного нейрона окремо, їх об'єднання в мережу дає можливість вирішувати більш складні завдання, непідвладні кожному окремому нейрону.

2.3.4 Класифікація нейромереж

Конкретний вид виконуваного НМ перетворення інформації обумовлюється характеристиками нейроподібних елементів і осо-

бливостями архітектури мережі: топологією міжнейронних зв'язків, вибором певних підмножин нейроподібних елементів для введення і виведення інформації, способами навчання мережі, наявністю або відсутністю конкуренції між нейронами, напрямком і способами управління і синхронізації передачі інформації між нейронами.

Класифікація ШНМ:

1) за **типом вхідної інформації** виділяють:

– аналогові НМ, які використовують інформацію у формі дійсних чисел;

– бінарні (виконавчі) НМ, які оперують з інформацією, описаною в двійковому вигляді;

2) за **типом функції активації** нейронів виділяють:

– неперервні (аналогові, речові, що диференціюються) мережі, кожен елемент яких реалізує неперервно диференційовану функцію;

– дискретні (бінарні, порогові, що не диференціюються) мережі, кожен елемент яких реалізує недиференційовану функцію;

– дискретно-неперервні мережі – містять елементи з диференційованими і недиференційованими функціями;

3) за **типом графа міжнейронних сполук** виділяють:

– мережі без циклів (ациклічні мережі);

– мережі з циклами, які поділяються на рівноважні мережі з циклами і мережі з обмеженими циклами;

4) за **типом структур нейронів** виділяють:

– гомогенні (однорідні) НМ, які складаються з нейронів одного типу з єдиною функцією активації;

– гетерогенні (неоднорідні) НМ, що містять нейрони з різними функціями активації;

5) за **числом шарів нейронів** виділяють:

– одношарові НМ (містять один шар нейронів);

– БНМ (містять більше одного шару взаємопов'язаних нейронів);

6) за **типом дискримінантної функції**;

7) за **принципом синтезу** виділяють:

– навчаємі НМ (графи міжнейронних зв'язків та ваги входів змінюються при виконанні алгоритму навчання);

– конструюємі НМ (число і тип нейронів, граф міжнейронних зв'язків, ваги входів нейронів визначаються при створенні НМ, виходячи з розв'язуваної задачі);

8) за **топологією зв'язків** виділяють (див. рис. 2.2):

– повнозв'язні (інтерактивні) мережі, в яких усі нейрони пов'язані за принципом «кожний з кожним»: кожен нейрон передає свій вихідний сигнал іншим нейронам, включаючи самого себе, і вихідні сигнали мережі можуть бути всі або деякі вихідні сигнали нейронів після декількох тактів функціонування мережі ; всі вхідні сигнали подаються всім нейронам;

– неповнозв'язні (шаруваті, багатошарові, ієрархічні) мережі мають зазвичай шарувату організацію. У них різняться латеральні (бічні) зв'язки, які охоплюють нейрони одного шару, і проекційні (аферентні), що з'єднують шари нейронів. Частина нейронів має додаткові зовнішні входи, які утворюють рецепторне поле. Нейрони першого шару отримують вхідні сигнали, перетворять їх і через точки галуження передають нейронам наступного шару і так далі до останнього шару, який видає вихідні сигнали. У цьому випадку інформація рухається по висхідній від вхідного до вихідного прошарку і кожен наступний шар забезпечує як би більш високий рівень її обробки, ніж попередній. Число нейронів у кожному шарі може бути будь-яким і ніяк заздалегідь не пов'язано з кількістю нейронів в інших шарах;

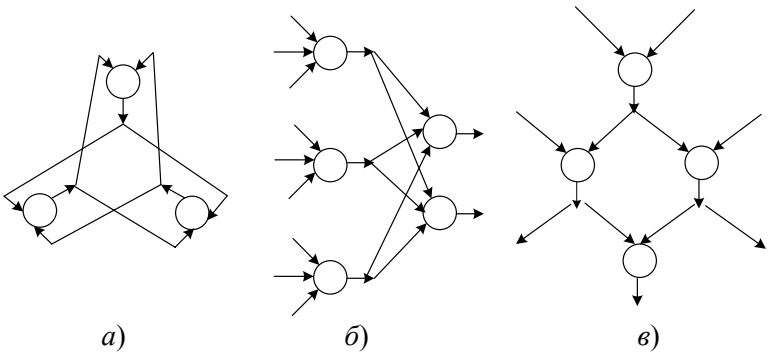


Рисунок 2.2 – НМ з різною топологією зв'язків:

а) повнозв'язна мережа; *б*) шарувата мережа; *в*) слабозв'язана мережа

– ієрархічно інтерактивні мережі: шари різного рівня пов'язані двосторонніми зв'язками і наявні зв'язки між елементами одного шару, але структура зв'язків не є однорідною, як в повністю інтерактивній мережі;

– слабозв'язні мережі (мережі з локальними зв'язками): нейрони розташовуються у вузлах прямокутної або гексагональної решітки, кожен нейрон зв'язаний з чотирма (окіл фон Неймана), шістьма (окіл Голео) або вісьмома (окіл Мура) своїми найближчими сусідами;

9) за **характером зв'язків** виділяють:

– мережі прямого поширення (мережі без зворотних зв'язків, feedforward) – сигнал по мережі проходить тільки в одному напрямку: від входу до виходу;

– рекурентні мережі (НМ із зворотними зв'язками, мережі зі зворотним поширенням інформації, feedforward / feedback): характеризуються як прямим, так і зворотним поширенням інформації між шарами НМ.

Серед рекурентних мереж, у свою чергу, виділяють:

а) релаксаційні, в яких циркуляція інформації відбувається до тих пір, поки не перестануть змінюватися вихідні значення НМ (стан рівноваги);

б) шаруваті мережі із зворотними зв'язками, в яких відсутній процес релаксації. Серед шаруватих мереж із зворотними зв'язками виділяють: шарувато-циклічні мережі, що відрізняються тим, що шари замкнуті в кільце (останній шар передає свої вихідні сигнали першому; всі верстви рівноправні і можуть, як отримувати вхідні сигнали, так і видавати вихідні); шарувато-повнозв'язні мережі (складаються з шарів, кожен з яких представляє собою повнозв'язну мережу, а сигнали передаються як від шару до шару, так і всередині шару; в кожному шарі цикл роботи розпадається на три частини: прийом сигналів з попереднього шару, обмін сигналами всередині шару, вироблення вихідного сигналу і передача до подальшого шару); повнозв'язно-шаруваті (за своєю структурою аналогічні шарувато-повнозв'язним, але функціонуючи по-іншому: у них не розділяються фази обміну всередині шару і передачі наступного шару; на кожному такті нейрони всіх шарів приймають сигнали від нейронів як свого шару, так і наступних); рециркуляційні мережі (характеризуються

прямим $y = f(x)$, і зворотнім $x = f(y)$ перетворенням інформації; в рециркуляційних мережах навчання проводиться без вчителя, тобто вони є самоорганізуючимися в процесі роботи); мережі з бічними зворотними зв'язками (laterally connected);

10) за **характером поділу зв'язків** виділяють:

– немонотонні мережі, які не дозволяють визначити, як вплине зміна будь-якого внутрішнього параметра мережі на вихідний сигнал;

– монотонні мережі: кожен шар мережі крім останнього поділяється на два блоки: збудливий і гальмуючий. При цьому всі зв'язки в мережі влаштовані так, що елементи збудливої частині шару збуджують елементи збудливою частини наступного шару і гальмують елементи наступного шару. Аналогічно, які гальмуючі елементи збуджують гальмуючі елементи і гальмують збуджуючі елементи наступного шару (назви «гальмуючий» і «збуджуючий» відносяться до впливу елементів обох частин на вихідні елементи). Для нейронів монотонних мереж необхідна монотонна залежність вихідного сигналу нейрона від параметрів вхідних сигналів. Відзначимо, що для мереж з сігмоідними елементами вимога монотонності означає, що ваги всіх зв'язків повинні бути не негативні.

11) за **типом методу навчання** (характеру налаштування синапсів) виділяють:

– мережі з динамічними зв'язками і ітеративним навчанням, заснованому на принципі корекції помилок, які базуються на запропонованій фізіологами моделі повторення шляхів нервового збудження. Основний метод навчання – зворотне поширення помилки широко використовується в сучасних нейрокомп'ютерів. Головною його перевагою є асимптотична збіжність процесу навчання, гарантуюча потенційну досяжність необхідної точності реакції НМ. Недолік методу полягає в необхідності багаторазового повторення ітерацій навчання на всьому обсязі запам'ятовуємих даних. Велика витрата часу й обчислювальних ресурсів при навчанні обмежували можливість застосування в адаптивних системах реального часу методів, заснованих на методі зворотного поширення;

– НМ з фіксованими зв'язками і неітеративним навчанням – використовують методи прямого обчислення значення матриці зв'язків, які забезпечують запам'ятовування інформації без повторення, що прискорює процес навчання у порівнянні з ітерати-

вними методами. Однак вони характеризується надмірною кількістю нейронів і низьким обсягом інформації, що запам'ятовується. Тому вони поки не отримали широкого поширення і застосовуються, в основному, в дослідницьких проектах.

12) за **характером навчання** виділяють:

– мережі з контрольованим (піднаглядним, з вчителем, з супервизором) навчанням, коли відомим є вихідний простір рішень НМ (при навчанні порівнюють заздалегідь відомий вихід з отриманими значеннями);

– мережі з неконтрольованим (без нагляду, без вчителя, без супервізора) навчанням, коли НМ формує вихідний простір рішень тільки на основі вхідних впливів, навчається, не знаючи заздалегідь правильних вихідних значень, але групуючи «близькі» вхідні вектори так, щоб вони формували один і той же вихід мережі (неспостережене навчання використовується, зокрема, при вирішенні задачі кластеризації); мережі зі змішаним навчанням, коли частина ваг визначається при спостеріганні, а частина – при неспостерігаємому навчанні (навчання здійснюється шляхом пред'явлення прикладів, що складаються з наборів вхідних даних у сукупності з відповідними результатами при спостерігаємому навчанні і без останніх при неспостерігаємому);

13) за **методом навчання** виділяють:

– мережі з неітеративним навчанням,

– мережі з методом зворотного поширення помилки,

– мережі з конкурентним навчанням,

– мережі, що використовують правило Хебба,

– мережі з гібридним навчанням, в яких застосовуються різні методи навчання;

– інші;

14) за **типом часу функціонування** виділяють:

– мережі з неперервним часом (аналогові мережі),

– мережі з дискретним асинхронним часом (в кожен момент часу лише один нейрон змінює свій стан);

– мережі з дискретним часом, що функціонують синхронно (стан міняється відразу у цілої групи нейронів).

15) **в залежності від врахування попереднього стану мережі** виділяють мережі:

– статичні (не мають у своїй структурі ні зворотних зв'язків, ні динамічних елементів, а вихід залежить від заданої множини на вході і не залежить від попередніх станів мережі);

– нечіткі (поєднують в собі БНМ і нечітку систему, їх перший шар нейронів реалізує етап введення нечіткості (фаззифікації), другий шар відображає сукупність нечітких правил, третій шар реалізує дефаззіфікацію – функцію приведення до чіткості), нетрадиційні НМ.

16) за **типом розв'язуваних задач** виділяють мережі для:

– обробки та фільтрації даних;

– категоризації і таксономії даних;

– пошуку закономірностей у даних;

– заповнення прогалин у таблицях даних;

– візуалізації та картографування даних;

– розпізнавання (класифікації) образів;

– непараметричної апроксимації залежностей по крапковим даним;

– побудови баз даних великої ємності з швидким асоціативним пошуком інформації за неповними вхідними даними;

– розробки пристроїв асоціативної пам'яті;

– побудови експертних систем, яких навчають, на прикладах;

– здобуття знань з даних;

– адаптивного управління складними об'єктами і процесами;

– трудомістких задач оптимізації (типу задач про комівояжера і т.п.), шифрування, дешифрування, стиснення інформації, перекладу тексту.

17) за **областю застосування** виділяють мережі для:

– розпізнавання сигналів, мови, зображень і тексту;

– технічної та біомедичної діагностики;

– моделювання залежностей в природничих науках і техніці;

– соціально-економічного прогнозування;

– управління в техніці та економіці;

– побудови інформаційно-пошукових систем;

– криптографії.

2.3.5 Властивості нейронних мереж

Основними властивостями нейромереж є:

1) **однорідність нейроелементів** і базових операцій, а також технологічна простота різних способів їх фізичної реалізації: НМ за аналогією з мозком будуються з множини простих уніфікованих типових елементів (нейронів), що виконують елементарні дії (множення, додавання, обчислення найпростішої нелінійної функції) і з'єднаних між собою різними зв'язками;

2) **можливість реалізації нелінійних відображень** шляхом використання нелінійних функцій активації нейронів (це важливо для вирішення завдань управління з істотними нелінійностями, для яких традиційні підходи поки не дають практично реалізованих рішень);

3) **пластичність** – обумовлює складність поведінки НМ, яка розглядається як результат взаємодії багатьох елементів, кожен з яких обмежує дію інших і сам обмежується іншими на шляху до формування глобального спостережуваного поведінки. Розрізняють нейронну пластичність (як пластичних елементів розглядаються нейрони), а також синаптичну пластичність (модифікація сили синаптичного зв'язку між нейронами) НМ. Оскільки кількість синапсів, як правило, на кілька порядків перевищує кількість нейронів, то мережу з пластичними синапсами буде володіти великими можливостями, ніж мережа еквівалентного розміру з пластичними нейронами;

4) **адаптація** – сукупність пристосувальних, що протікають на різних ієрархічних рівнях, реакцій на зміну зовнішнього середовища та організації впливають на організм сигналів зовнішнього середовища. НМ є адаптивними системами, оскільки можуть пристосовуватися до змін зовнішнього середовища за допомогою зміни своєї структури і значень параметрів;

5) **здатність до навчання**: НМ здатні удосконалювати свою роботу (навчатися або адаптуватися), використовуючи приклади для налаштування на вирішення певної задачі;

6) **узагальнення** – здатність інтегрувати приватні дані для визначення закономірностей та пролонгації результатів, що дозволяє після навчання на одних даних застосовувати отримані знання для інших даних. Для НМ під властивістю узагальнення

розуміється здатність генерувати правильні виходи для вхідних сигналів, які не були враховані в процесі навчання. Однак НМ не завжди може узагальнювати дані. Якщо НМ була перетренована, тобто вона запам'ятала тренувальні дані як таблицю і видає коректні дані лише при точній вказівці «адреси» в ній, то в цьому випадку мережа не зможе коректно інтерполювати вхідні дані між тренувальними. При низькому рівні узагальнення перетворення, здійснюване НМ, недостатньо гладко. Гладкість перетворення в НМ безпосередньо пов'язана з необхідністю вибрати найпростішу модель в умовах відсутності апріорної інформації. У даному випадку під найпростішої мається на увазі найбільш гладка функція, найкращим чином апроксимує розглядається перетворення. Вимога гладкості забезпечує гарні інтерполяційні властивості НМ і гарантує їм мінімальну складність структури, що прямо впливає на обсяг обчислень, які виконуються при навчанні. Іншою вимогою є достатня репрезентативного тренувального набору даних: НМ тим краще узагальнює, чим щільніше і рівномірніше розташовані тренувальні дані у вхідному просторі. Якщо тестові дані завжди будуть надаватися між близько розташованими тренувальними шаблонами, то мережа зможе проводити коректне узагальнення, інтерполючи вхідні дані;

7) **універсальна апроксимація**: НМ здійснюють наближення функцій багатьох змінних за допомогою лінійних операцій і суперпозицій функцій однієї змінної. Радянські вчені О. М. Колмогоров і В. І. Арнольд показали, що будь-яку неперервну функцію N змінних можна отримати за допомогою операцій додавання, множення і суперпозиції з неперервних функцій однієї змінної. На основі цього доведено ряд теорем про апроксимації неперервних функцій багатьох змінних НМ з використанням довільної неперервної функції однієї змінної. НМ дозволяють з будь-якою точністю обчислювати довільну неперервну функцію. Отже, з їх допомогою можна як завгодно точно апроксимувати функціонування будь-якої неперервної системи. У 1987 році Р. Хехт-Нільсеном була запропонована теорема, яка доводить представимість функції багатьох змінних досить загального вигляду за допомогою НМ з прямими повними зв'язками, N нейронами вхідного шару, $(2N+1)$ нейронами прихованого шару з наперед відомими обмеженими функціями активації (наприклад, сигмоїдальними) і N_M нейронами вихі-

дного шару з невідомими функціями активації. З теореми Хехт-Нільсена представимість будь-якої багатовимірної функції декількох змінних за допомогою НМ фіксованого розмірності. Таким чином, НМ є універсальними структурами, що дозволяють реалізувати будь-який обчислювальний алгоритм;

8) **самоорганізація** являє собою процес динамічної перебудови системи з метою адаптації до зовнішнього середовища. Самоорганізація на рівні організму відбувається за допомогою навчання, в результаті якого динамічно перебудовуються нейронні структури головного мозку. Самоорганізація НМ називається також здатністю до навчання: НМ можуть автономно «вивчати» статичні і динамічні властивості ОУ на основі результатів вимірювань, які проводилися в минулому, а потім діяти таким чином, щоб прийняти найкраще рішення при невідомому стані зовнішнього середовища. Звичайні комп'ютери повинні бути попередньо запрограмовані, щоб мати можливість обробляти дані, вони не можуть працювати за межами рішень, що задаються програмою. Таким чином, інженерія знань не може бути повною мірою реалізована на звичайних комп'ютерах, так як вони не можуть приймати рішення в нових зовнішніх умовах;

9) **асоціативна пам'ять**: НМ розглядається як мультистабільна система, стійкий стан якої (атрактори) відповідає запам'ятованим станам. Якщо зовнішній вплив встановлює мережу в стан, близький до одного з атракторів, то, будучи надана сама собі, мережа за одну або декілька ітерацій конвергенції перейде в стан аттрактора. Таким чином, якщо початковий стан відображає неповний або спотворений набір раніше запам'ятованих даних, то після конвергенції весь цей набір даних буде відновлений. На цьому базується властивість асоціативної пам'яті, що дозволяє застосовувати нейрокомп'ютери для розпізнавання образів, прогнозування складних процесів і т. п. Обсяг асоціативної пам'яті НМ залежить від кількості стійких станів, положення яких визначається значенням матриці зв'язків, отриманих в результаті навчання. Число таких станів не може бути як завгодно великим. Якщо число запам'ятованих станів перевищує деяку межу, відбувається насичення пам'яті і з'являються неправдиві стійкі стани, що спотворюють вміст асоціативної пам'яті;

10) **розподіленість пам'яті** – інформація зберігається за багатьма адресами, розподілених таким чином, що кожен елемент даних представляється шаблоном активності, розподіленим по багатьом обчислювальним елементам, і кожен обчислювальний елемент бере участь в уявленні багатьох різних елементів даних. У звичайних комп'ютерах реалізується локальна пам'ять, або локальне подання, в якому використовується один обчислювальний елемент для кожного елемента даних. На основі розподіленої архітектури подання інформація в НМ може дробитися і оброблятися по частинах;

11) **ізоморфізм НМ топології** навчальних даних і структурної інформації про модельовану залежність. В основі принципів побудови штучних НМ закладені: наївний ізоморфізм Келера, що вимагає, щоб подання події в НМ, мало просторову структуру, що нагадує структуру відображуваного реального об'єкта, а також ізоморфізм Гріна, що допускає, щоб подання події в НМ мало таку логічну структуру (не обов'язково фізично реалізовану в просторовій формі), яка дозволяла б розбивати це подання, розчленовувати його і знову відновлювати за допомогою відповідних процедур або процесів направленої уваги так, що встановлюється зв'язок між частинами, поверхні або аспектами явищ реального світу. Відомо, що існують ділянки мозку – «сенсорні проекційні поля», – в яких справді зберігається просторова структура образів. Однак залишається невідомим, наскільки далеко має поширюватися код уявлень в сенсі просторового ізоморфізму, щоб у ньому враховувалися організуючі властивості досвіду. Показано, що в ряді завдань необхідне структурне подання може бути отримано за допомогою адаптивних процесів і не потрібно, щоб уявлення були повністю ізоморфні піднаглядним явищам;

12) **паралельна архітектура, масовий паралелізм і розподіленість обчислень** – обробка інформації в НМ виконується кількома процесорними елементами, в той час як у звичайних комп'ютерах, що мають тільки один центральний процесор, інформація обробляється послідовно, крок за кроком;

13) **ієрархічна організація структури, цілісність і дробимість елементів НМ**: багато завдань штучного інтелекту неможливо вирішити без використання ієрархічних структур, що дозволяють будувати моделі складних об'єктів з простіших; робота

ієрархічної структури вимагає, щоб інформаційний елемент у кожному ієрархічному рівні вів себе як єдине ціле, але при переході з рівня на рівень допускав дроблення, причому при переході з верхнього ієрархічного рівня на нижній це дроблення відповідає виділенню складових його елементів, а при переході з нижнього рівня на верхній воно відповідає включенню певної частини цього елемента в більш складний об'єкт;

14) **предорганізація в навчанні**: незважаючи на те, що НМ мають більш широкі можливості до адаптації і навчання на прикладах порівняно з традиційними методами штучного інтелекту, вирішити на їх основі складні задачі штучного інтелекту за рахунок одного лише навчання, мабуть, неможливо. Необхідно, щоб вихідна структура НМ була попередньо організована. Для виконання цієї умови необхідно мати достатньо розвинені інструментальні засоби, що дозволяють формувати структуру НМ до початку її навчання;

15) **функціональна блочність** – полягає в побудові архітектури НМ на стандартизованих функціональних блоках. Для вирішення задач штучного інтелекту необхідно будувати НМ, які мають велику кількість нейронів, тому базовими елементами для інструментальних засобів предорганізації НМ повинні бути великі функціональні блоки, внутрішня організація і властивості яких визначені заздалегідь і відомі розробнику;

16) **поліалгоритмічність** – властивість, що дозволяє одній і тій же НМ одночасно переробляти вхідну інформацію за різними алгоритмами. Поліалгоритмічність пов'язана з наявністю в мережі різних вихідних нейронів, з'єднаних з нейронами інших ієрархічних рівнів. При цьому ланцюжки, по яких збудження передається від входу до вихідних нейронів, мають різну структурну конфігурацію, яка залежно від сполучень схем проміжних нейронів, що включають збуджуючі або гальмують синапси, відповідає різним математичним формулам;

17) **логічна прозорість**: логічно прозорої називають НМ, яка вирішує задачу зрозумілим для людини способом, для якої легко сформулювати словесний опис у вигляді явного алгоритму. Така мережа повинна мати мінімальну структурну складністю і при цьому задовольняти вимогам (перевагам) користувача і (або) вимогам алгоритму автоматизованого здобуття знань до виду ре-

зультуючої мережі. Процес отримання логічно прозорої мережі полягає в тому, що спочатку задаються необхідні критерії логічної прозорості, для кожного критерію вводиться дискретна шкала, по якій відраховується віддаленість поточного стану НМ до класу логічно прозорих мереж як число сигналів або елементів мережі, не задовольняє вимогам критерію, проводиться процедура спрощення НМ, яка будується так, щоб мінімізувати «відстань» від поточної НМ до класу логічно прозорих функцій. «Відстань» обчислюється як зважена (за встановленими користувачем вагами для кожного критерію логічної прозорості) сума тих критеріїв, за якими мережа не задовольняє вимогам логічної прозорості. На кожному кроці спрощення серед всіх доступних елементарних операцій вибирається така операція, яка призводить до найбільшого зменшення «відстані» (причому, деяка спрощуюча операція може зменшувати значення відразу за кількома критеріями). Якщо вибрана операція не може бути виконана (наприклад, через досягнення мінімуму за цим критерієм логічної прозорості), то вибирається наступна операція. Якщо досягнуті всі умови зупинка, процес спрощення закінчується. Структура НМ при цьому найкращим чином задовольняє набору сформованих користувачем вимог. Спрощення НМ будується як послідовний процес виключення з мережі найменш значущого елемента і подальшого довчання мережі. Якщо після кроку спрощення неможливо довчання мережі до необхідної точності, то повертаємося до мережі, отриманої на попередньому кроці, і завершуємо процес спрощення. Для видалення або модифікації одного структурного елементу з НМ запропонована номенклатура спрощує операції: видалення вхідного сигналу мережі, видалення синапсу мережі, видалення нейрона мережі, бінаризація синапсу мережі, модифікація нелінійного перетворювача нейрона мережі. Для кожного з заданих критеріїв простоти НМ зіставляється алгоритм досягнення цього критерію, що виражається через послідовність виконання елементарних спрощуючих операцій і додаткових специфікацій. Скорочення множини параметрів мережі і вхідних сигналів може переслідувати кілька цілей: спрощення архітектури та поліпшення інтер- та екстраполяційних здатностей навченої НМ (мережа з мінімальним числом нейронів повинна краще апроксимувати функцію, але з’ясування цього мінімального числа

нейронів вимагає великих обчислювальних витрат і експериментів з навчання мереж), скорочення обсягу використовуваної пам'яті і підвищення швидкодії програм-нейроімітаторів; спрощення подальшої апаратної реалізації НМ (якщо мережа реалізується не на дискретних елементах, а як програма для цифрового сигнального процесора або мікроконтролера, то в цьому випадку спрощена мережа, в порівнянні з вихідною, вимагає меншого обсягу пам'яті і має при тій же тактовій частоті процесора підвищену в кілька разів швидкість роботи); здешевлення процесу збору та зберігання даних за рахунок скорочення їх обсягу шляхом рішенням задачі відбору найбільш інформативних вхідних ознак;

18) **здатність отримувати знання з даних**: в процесі навчання НМ засвоює (апроксимує) найбільш загальні закономірності, присутні в навчальних даних, витягуючи тим самим неявні для людини знання з даних. Труднощі використання таких мереж складаються в нашій обмеженій здатності пояснити, як НМ вирішує завдання: внутрішні уявлення НМ, що виходять в результаті навчання, настільки складні, що їх неможливо проаналізувати, за винятком найпростіших випадків. Спрощення НМ, що досягається контрастуванням, дозволяє отримувати логічно прозорі НМ, на основі структури яких можна побудувати вербальний опис алгоритму отримання відповіді за наявною нейромережевої моделі – витягти явні знання з даних;

19) **точність рішення задач** НМ визначається помилкою розпізнавання (апроксимації залежності) для навчальних даних, а також помилкою розпізнавання (апроксимації залежності) для тестових даних;

20) **ефективність (якість) вирішення завдань** НМ визначається точністю (помилкою) рішення задачі для навчальних і тестових даних, простотою, логічної прозорістю і швидкодією отриманої нейромережевої моделі, а також витратами на побудову нейромережевої моделі (вимоги до апаратних засобів, ітераційність і витрати часу методу навчання);

21) **варіативність моделей апроксимуємої залежності**: для однієї і тієї ж таблиці даних і різних мереж або однієї мережі, але з різною початковою генерацією вихідних значень набору параметрів, що настроюються, після навчання, спрощення за єдиною схемою та вербалізацією може вийти трохи різних НМ (нейроме-

режевих моделей) і, відповідно, кілька алгоритмів рішення однієї і тієї ж задачі. За кінцевою таблицею даних завжди будується кілька алгоритмів рішення. Далі алгоритми починають перевірятися і конкурувати між собою. Комбінуючи фрагменти декількох алгоритмів (моделей), можна сконструювати нову теорію. У силу цього неєдиність одержуваного знання (варіативність, неєдиність нейромережевої моделі) не представляється недоліком;

22) **складність НМ** визначається кількістю і топологією міжнейронних зв'язків, складністю виконуваних перетворень і кількістю нейроелементів;

23) **надійність та стійкість мережі до відмов окремих елементів**, складових НМ, проявляється в тому, що відмова одного або декількох нейроелементів мережі не призводить до відмови всієї НМ і не може істотно впливати на роботу мережі в цілому.

2.3.6 Нейрокомп'ютери

Для використання на практиці НМ реалізують у вигляді нейрокомп'ютера – обчислювальної системи, архітектура якої спеціалізована на виконанні операцій, адекватних структурі НМ.

Нейрокомп'ютери якісно відрізняються від усіх попередніх поколінь ЕОМ тим, що в них відсутні заздалегідь створені алгоритмічні програми і що вони, аналогічно людському мозку, здатні навчатися на окремих прикладах. У звичайних ЕОМ елементи схем з'єднані послідовно, кожен елемент з'єднаний тільки з двома-трьома елементами, так що сигнал обробляється поетапно, крок за кроком. Однак у НМ елементи мають множину паралельних з'єднань, причому кожен елемент з'єднаний майже з кожним. Через це вхідний сигнал поширюється по всій мережі, і всі елементи мережі працюють паралельно, реалізуючи, як кажуть, масовано-паралельні обчислення. Цим пояснюється можливість вирішувати складні обчислювальні задачі в реальному часі, справлятися з непередбаченими ситуаціями і навіть синтезувати знання з даних майже без участі людини.

Виділяють **три рівні нейрокомп'ютерів**:

– рівень 1: імітаційна модель НМ на звичайній ЕОМ;

– рівень 2: плата або приставка до персонального комп'ютера;

– рівень 3: повнофункціональний нейрокомп'ютер на мікрочіпах.

Переваги НМ і нейрокомп'ютерів полягають в тому, що вони дають стандартний спосіб рішення багатьох нестандартних завдань (неважливо, що спеціалізована машина краще вирішить один клас задач, важливіше, що один нейрокомп'ютер вирішить і цю задачу, і другу, і третю – і не треба кожен раз проектувати спеціалізовану ЕОМ – нейрокомп'ютер зробить все сам і майже не гірше); замість програмування використовується навчання (нейрокомп'ютер вчиться – потрібно лише формувати навчальні задачники, праця програміста, розпорядчого машині всі деталі роботи, заміщається працею вчителя, що створює «освітнє середовище», до якого пристосовується нейрокомп'ютер); нейрокомп'ютери особливо ефективні там, де необхідно подобу людської інтуїції і важко створити явний алгоритм.

2.3.7 Нейромережі прямого поширення

2.3.7.1 Багатошаровий персептрон

Багатошарова нейронна мережа (БНМ) прямого поширення складається з формальних нейронів і характеризується наступними параметрами і властивостями: M – число шарів мережі, N_μ – число нейронів μ-го шару, зв'язки між нейронами в шарі відсутні. Виходи нейронів μ-го шару, μ=1,2,..., M–1, надходять на входи нейронів тільки наступного μ +1-го шару. Зовнішній векторний сигнал x надходить на входи нейронів тільки першого шару, виходи нейронів останнього M-го шару утворюють вектор виходів мережі $y(M)$. Структура мережі показана на рис. 2.3.

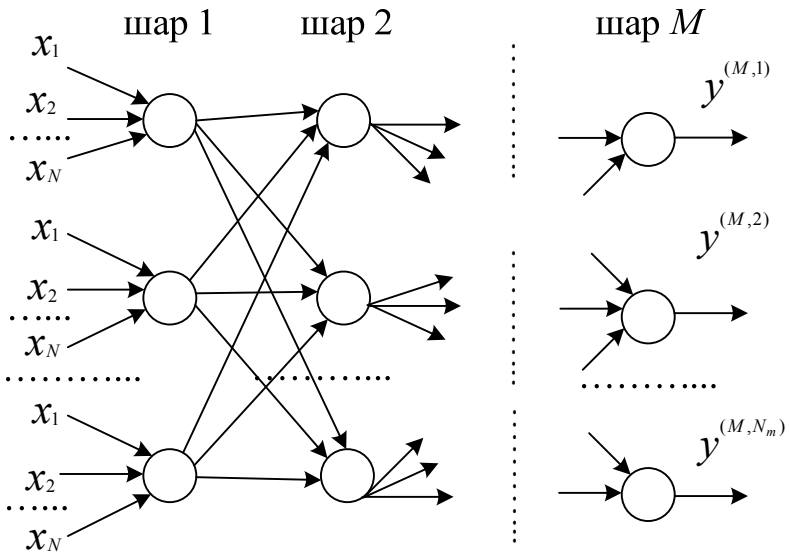


Рисунок 2.3 – Структура БНМ

Кожен *i*-й нейрон μ-го шару ((μ, *i*)-й нейрон) перетворює вхідний вектор $x^{(\mu,i)}$ у вихідну скалярну величину $y^{(\mu,i)}$. Це перетворення складається з двох етапів: спочатку обчислюється дискримінантна функція $\phi^{(\mu,i)}(w^{(\mu,i)}, x^{(\mu,i)})$, яка далі перетворюється на вихідну величину $y^{(\mu,i)} = \psi^{(\mu,i)}(\varphi^{(\mu,i)}(w^{(\mu,i)}, x^{(\mu,i)}))$, де $w^{(\mu,i)}=(w_0^{(\mu,i)}, w_1^{(\mu,i)}, \ldots, w_N^{(\mu,i)})^T$ – вектор вагових коефіцієнтів нейрона, $x_j^{(\mu,i)}$ – *j*-а компонента *N*-мірного вхідного вектора $x^{(\mu,i)}$.

Якість рішення, одержуваного БНМ, буде істотно залежати від кількості шарів, кількості нейронів у кожному шарі і кількості зв'язків між шарами.

Для різних класів задач, що вирішуються за допомогою НМ, запропоновані евристичні оцінки числа шарів і нейронів. Наприклад, для оцінки кількості нейронів у прихованих шарах однорідних двошарових БНМ з сігмоідними функціями активації можна використовувати формулу для оцінки необхідного числа синаптичних ваг N_w:

$$\frac{N_M S}{1+\log_2 S} \le N_w \le N_M\left(\frac{S}{N_M}+1\right)\left(N+N_M+1\right)+N_M,$$

де N – розмірність вхідного сигналу (кількість ознак); N_M – розмірність вихідного сигналу.

Оцінивши необхідну кількість ваг, можна розрахувати $N_н$ – число нейронів у прихованих шарах. Наприклад, для двошарової мережі це число складе: $N_н = N_w/\left(N+N_M\right)$. Відомі й інші формули для оцінки, наприклад: $2(N + N_н + N_M) \le S \le 10(N + N_н + N_M)$; $0{,}1S - N - N_M \le N_н \le 0{,}5S - N - N_M$.

2.3.7.2 Навчання БНМ методом зворотного поширення помилки

Процес навчання БНМ здійснюється в результаті мінімізації цільової функції – деякого критерію якості $E(Q(\varepsilon,s))$, що характеризує інтегральну міру близькості виходів мережі $y^{(M)}$ і вказівок вчителя $y^* = \{y^{s*}\}$, s = 1,2, ..., S, де s – номер поточного екземпляра навчаючої вибірки, S – кількість екземплярів у навчальній вибірці, $Q(\varepsilon,s)$ – миттєвий критерій якості, що залежить від вектора помилки мережі для i-ої вихідної змінної s-го екземпляра $\varepsilon(s,i,w) = y^{(M,i)} - y_i^s$, i =1,2, ..., N_M, де w – множина ваг мережі, $y^{(M,i)}$ та y_i^{s*} – відповідно, розраховане мережею і бажане значення i-ої вихідної змінної для s-го екземпляра навчаючої вибірки. Як миттєвого критерію якості найбільш часто використовують суму квадратів помилки або суму модулів помилки для s-го екземпляра вибірки:

$$Q(\varepsilon,s) = \sum_{i=1}^{N_M}\varepsilon(s,i,w)^2,$$

$$Q(\varepsilon,s) = \sum_{i=1}^{N_M}\left|\varepsilon(s,i,w)\right|.$$

Інтегральну міру близькості E визначають за формулою:

$$E = \frac{1}{\upsilon}\sum_{s=1}^{S} Q(\varepsilon,s),$$

де υ – деякий коефіцієнт (якщо υ=1, то E визначає сумарну миттєву помилку; якщо $\upsilon = S$, то E визначає середню миттєву помилку).

Для кожного вхідного вектора x^s з навчальної множини повинен бути визначений вектор бажаних виходів мережі y^{s*}. Якщо навчальна БНМ використовується в якості класифікатора, то зазвичай бажані виходи мають низький рівень (0 або менше 0,1), крім виходу вузла, відповідного класу, до якого належить х; цей вихід у даному випадку має високий рівень (1 або більше 0, 9).

Для мінімізації цільової функції навчання вирішують завдання багатомірної нелінійної оптимізації, для чого традиційно використовують градієнтні методи. Ці методи носять ітеративний характер, так як компоненти градієнта виявляються нелінійними функціями. Градієнтні методи засновані на ітераційній процедурі корекції значень ваг, реалізованої у відповідності з формулою:

$$w_{k+1} = w_k + \alpha_k p(w_k),\ k = 0,1,2,\ ...$$

де w_k , w_{k+1} – поточне і нове наближення значень ваг і порогів НМ до оптимального рішення, відповідно; α_k – крок збіжності; $p(w_k)$ – напрям пошуку в багатовимірному просторі ваг.

Спосіб визначення $p(w_k)$ та α_k на кожній ітерації залежить від особливостей конкретного методу.

Після ініціалізації ваг мережу навчають за допомогою методів, що мінімізують цільову функцію помилки, які, як правило, носять ітеративний характер і вимагають обчислення похідних цільових функції для визначення напрямку пошуку. Для розрахунку приватних похідних цільової функції за вагами мережі використовують **метод зворотного поширення помилки** (Error backpropagation method), який був розроблений як узагальнення методу Уїдроу-Хоффа для БНМ з нелінійними диференційовними функціями активації.

Метод зворотного поширення помилки для настроювання ваг НМ використовує градієнт функції помилки таким чином, щоб мінімізувати помилку на навчальній вибірці. Для цього мережі послідовно надаються вхідні вектора (екземпляри) з навчальної вибірки, і, починаючи з останнього шару, обчислюються градієнти функції помилки для кожного нейрона:

$$\frac{\partial E}{\partial w_j^{(\mu,i)}} = \frac{\partial E}{\partial \psi^{(\mu,i)}} \frac{\partial \psi^{(\mu,i)}}{\partial \phi^{(\mu,i)}} \frac{\partial \phi^{(\mu,i)}}{\partial w_j^{(\mu,i)}},$$

де $\frac{\partial \psi^{(\mu,i)}}{\partial \phi^{(\mu,i)}} = \psi'^{(\mu,i)}\left(\phi^{(\mu,i)}\right)$.

Якщо в якості дискримінантної функції нейронів мережі використовується зважена сума, то $\frac{\partial \phi^{(\mu,i)}}{\partial w_j^{(\mu,i)}} = x_j^{(\mu,i)} = \psi^{(\mu-1,j)}$.

Для нейронів вихідного шару: $\frac{\partial E}{\partial \psi^{(M,i)}} = y_i^s - \psi^{(M,i)}$, для нейронів інших шарів: $\frac{\partial E}{\partial \psi^{(\mu,i)}} = \sum_j \frac{\partial E}{\partial \psi^{(\mu+1,j)}} \frac{\partial \psi^{(\mu+1,j)}}{\partial \phi^{(\mu+1,j)}} \frac{\partial \phi^{(\mu+1,j)}}{\partial \psi^{(\mu,i)}}$,

де $\frac{\partial \phi^{(\mu+1,j)}}{\partial \psi^{(\mu,i)}} = w_j^{(\mu,i)}$.

У методі зворотного поширення помилки ваги і пороги змінюються у напрямку, протилежному градієнту. Тому базовий метод зворотного поширення часто називають методом градієнтного спуску (gradient descent). Базовий метод зворотнього поширення помилки складається з наступних кроків.

Крок 1. Задати навчальну вибірку значень ознак екземплярів x і зіставлених їм значень цільовий параметр y^*. Ініціалізувати всі ваги і пороги мережі. Задати значення прийнятного рівня помилки. Встановити покажчик поточного екземпляра вибірки $s = 1$.

Крок 2. Подати на вхід БНМ s-ий екземпляр з навчальної вибірки $x^s = \{x^s_i\}$, а на вихід – співставлений йому бажаний вихідний вектор $y^{s*} = \{y^{s*}_j\}$, $s = 1,2,...,S$; $i = 1,2,...,N$; $j = 1,2,...,N_M$.

Крок 3. Рухаючись від першого шару до останнього, розрахувати значення на виходах нейронів і фактичний вихід мережі $y^{(M)} = \{y^{(M,i)}\}$, $i = 1, 2, ..., N_M$, при подачі на вхід мережі екземпляра x^s.

Крок 4. Розрахувати помилку мережі E. Якщо $E > \xi$ (умова закінчення навчання для s-го екземпляра не виконується), тоді перейти на крок 5; в іншому випадку – перейти на крок 6.

Крок 5. Рухаючись від вихідного шару до першого шару мережі провести настроювання ваг і порогів за формулою $w_{k+1} = w_k +$

$+\alpha_k p(w_k)$, визначивши напрямок пошуку в багатовимірному просторі ваг по формулі:

$$p(w_{j,k}^{(\mu,i)}) = -g_{j,k}^{(\mu,i)},$$

де $g_{j,k}^{(\mu,i)} = \dfrac{\partial E}{\partial w_j^{(\mu,i)}}$ – градієнт помилки по *j*-ій вазі *i*-го нейрона μ-го шару МНС на *k*-ій ітерації навчання, визначається за формулою:

$$g_{j,k}^{(M,i)} = \psi_k'^{(M,i)}\left(\phi_k^{(M,i)}\right)\left(y_i^s - \psi_k^{(M,i)}\left(\phi_k^{(M,i)}\right)\right)\psi_k^{(M-1,j)}\left(\phi_k^{(M-1,j)}\right),$$

а для нейронів інших шарів по рекурентній формулі:

$$g_{p,k}^{(\mu,i)} = \psi_k'^{(\mu,i)}\left(\phi_k^{(\mu,i)}\right)\left(\sum_{j=1}^{N_{\mu+1}} g_{i,k}^{(\mu+1,j)} w_{i,k}^{(\mu+1,j)}\right) x_{p,k}^{(\mu,i)},\ \mu = M-1, \ldots, 1;$$

$$p = 0, \ldots, N_{\mu-1}.$$

Якщо в мережі використовуються нейрони з сігмоідною функцією активації, то для них:

$$\psi'^{(\mu,i)} = \left(\frac{1}{1+e^{-\phi(w^{(\mu,i)},x^{(\mu,i)})}}\right)' = \frac{e^{-\phi(w^{(\mu,i)},x^{(\mu,i)})}}{\left(1+e^{-\phi(w^{(\mu,i)},x^{(\mu,i)})}\right)^2} =$$

$$= \frac{1}{1+e^{-\phi(w^{(\mu,i)},x^{(\mu,i)})}}\left(1 - \frac{1}{1+e^{-\phi(w^{(\mu,i)},x^{(\mu,i)})}}\right) = \psi^{(\mu,i)}(1-\psi^{(\mu,i)}).$$

Перейти на крок 4.

Крок 6. Встановити: $s = s + 1$. Якщо $s > S$, то перейти на крок 7, у противному випадку – перейти на крок 2.

Крок 7. Зупинення.

Стандартний метод зворотного поширення часто збігається дуже повільно, рухаючись вздовж плоских ділянок поверхні помилки. Тому на практиці його часто використовують лише для розрахунку приватних похідних цільової функції помилки за вагами НМ, а пошук в просторі ваг НМ виробляють на основі більш швидких методів (градієнтного спуску з моментом, еластичного методу зворотного поширення, методу Ньютона, методу Левен-

берга-Марквардта, методів спряжених градієнтів Флетчера-Рівса і Полака-Ріб'єра, методу Quickprop, методу адаптивної зміни швидкості навчання Delta-Delta, методу Delta-Bar-Delta), серед яких найбільш швидким є **метод Левенберга-Марквардта**, для якого напрямок пошуку визначається за формулою:

$$p(w_k) = -\left[J_k{}^T J_k + \lambda_k I\right]^{-1} J_k{}^T e_k ,$$

де J – якобіан функції помилки за вагами БНМ; I – одинична матриця (матриця, всі елементи якої дорівнюють нулю, за винятком діагональних елементів, рівних +1); λ_k – параметр, що дозволяє регулювати довжину кроку; e – вектор помилки мережі.

На початковій стадії пошуку параметру λ_0 приписується велике значення. Якщо після першого кроку отримана точка з меншим значенням цільової функції, то слід встановити: $\lambda_{k+1} = \lambda^{(-)}\lambda_k$, в іншому випадку – встановити: $\lambda_{k+1} = \lambda^{(+)}\lambda_k$ і знову реалізувати попередній крок. Тут $\lambda^{(-)}$ та $\lambda^{(+)}$ деякі константи, такі що: $\lambda^{(-)} < 1$, $\lambda^{(+)} > 1$.

Основним недоліком традиційно використовуваних градієнтних методів навчання БНМ є обумовлена ітераційною корекцію ваг низька швидкість збіжності навчання, яка серйозно обмежує практичне застосування нейронного керування. Іншим не менш важливим недоліком даних методів є невизначеність у виборі початкової точки пошуку, параметрів архітектури та топології мережі.

2.3.7.3 Радіально-базисні нейронні мережі

Радіально-базисні мережі (Radial Basis Function Nets. RBF) були запропоновані для апроксимації функцій багатьох змінних. За допомогою радіально-базисних функцій можна як завгодно точно аппроксимувати задану функцію. Як і багатошаровий персептрон, радіально-базисна мережа (РБМ) є універсальним аппроксиматором.

Математичну основу РБ мережі складає метод потенційних функцій, розроблений М. А. Айзерманом, Е. М. Бравермaном і Л. І. Розоноером, що дозволяє представити деяку функцію $y(x)$ у вигляді суперпозиції потенційних або базисних функцій $f_i(x)$:

$$y(x) = \sum_{i=1}^{N} a_i f_i(x) = a^T f(x),$$

де $a_i = (a_1, a_2,...,a_N)^T$ – вектор підлягаючих визначенню параметрів; $f(x) = (f_1(x), f_2(x),\ldots,f_N(x))^T$ – вектор базисних функцій.

В РБМ в якості базисних вибираються деякі функції відстані між векторами

$$f_i(x) = f(\| x - c_i \|).$$

Вектори c_i називають центрами базисних функцій. Функції $f(x)$ вибираються невід'ємними і зростаючими при збільшенні $\|x - c_i\|$. В якості міри близькості векторів x та c_i вибираються, як правило, такі метрики:

– евклідова метрика $\|x - c_i\| = \left(\sum_{j=1}^{N} (x_j - c_{ij})^2 \right)^{\frac{1}{2}}$;

– манхеттенська метрика $\| x - c_i \| = \sum_{i=1}^{N} | x_j - c_{ij} |$,

де $| x_j - c_{ij} | = (x_j - c_{ij}) sign(x_j - c_{ij})$,

$$sign(x_i - c_{ij}) = \begin{cases} 1 & \text{якщо} \quad (x_i - c_{ij}) > 0; \\ 0 & \text{якщо} \quad (x_i - c_{ij}) = 0; \\ -1 & \text{якщо} \quad (x_i - c_{ij}) < 0. \end{cases}$$

Архітектура РБМ

Особливістю цих мереж є наявність радіально-симетричного шаблонного шару.

Структура РБМ відповідає мережі прямого поширення першого порядку (рис.2.4).

Інформація про образи передається з вхідного шару на прихованого, що є шаблонним і містить p нейронів. Кожен нейрон шаблонного шару, отримуючи повну інформацію про вхідні сигнали x, обчислює функцію

$$f_i(x) = f((x - c_i)^T R^{-1} (x - c_i)), i = \overline{1, p},$$

де x – вектор вхідних сигналів (Nx1); c_i – вектор центрів (Nx1); R – вагова матриця.

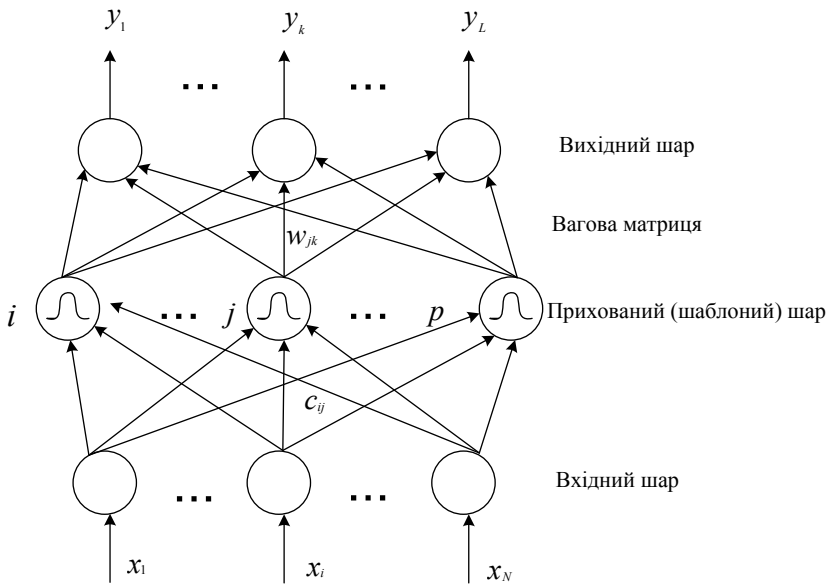


Рисунок 2.4 – Структура радіально-базисної мережі

Як вже зазначалося, особливістю даних мереж є наявність радіально-симетричного шаблонного шару, в якому аналізується відстань $(x–c_i)^T R^{-1}(x–c_i)$ між вхідним вектором і центром, представленим у вигляді вектора у вхідному просторі. Вектор центрів визначається за навчальною вибіркою і зберігається в просторі ваг від вхідного шару до шару шаблонів.

На рис. 2.5 представлено *i*-ий **нейрон шаблонного шару** РБ мережі. Обробку інформації, що надходить на нього умовно можна розділити на два етапи: на першому – обчислюється відстань між пред'явленими x і вектором центрів C_j з урахуванням обраної метрики і норми матриці R, на другому ця відстань перетвориться нелінійною активаційною функцією $f(x)$.

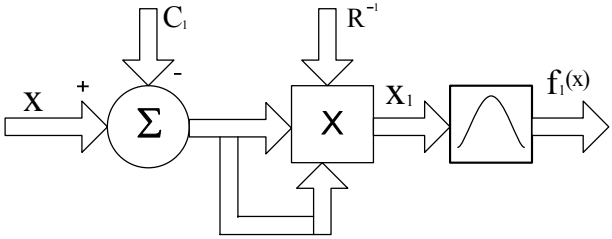


Рисунок 2.5 – Нейрон шаблонного шару РБМ

Як функції перетворення f (•) найбільш часто вибираються такі:

– гауссова функція:

$$f(x) = \exp\left\{-\frac{(x-c)^2}{2\sigma^2}\right\};$$

– мультиквадратична функція:

$$f(x) = \left(\frac{(x-c)^2}{\sigma^2} + a^2\right)^{\frac{1}{2}};$$

– зворотня мультиквадратична функція:

$$f(x) = \left(\frac{(x-c)^2}{\sigma^2} + a^2\right)^{\frac{1}{2}};$$

– сплайн-функція:

$$f(x) = x^2 \log(x);$$

– функція Коши:

$$f(x) = (1+x)^{-1}.$$

Слід зазначити, що зазначені функції допускають різні модифікації.

Таким чином, радіально-базисні нейронні мережі (**РБНМ**) являють собою **окремий випадок двошарових НМ** прямого

розповсюдження, **нейрони першого шару** які виробляють локалізовану реакцію на вхідний стимул і використовують в якості дискримінантної функції Евклідіву відстань, а в якості функції активації – радіально-базисну (ядерну) функцію Гауса, а **нейрони другого шару** формують зважену лінійну комбінацію з радіально-базисних функцій, обчислених нейронами першого шару. Нейрони другого шару РБНМ відповідають вихідним класам, у той час як нейрони першого шару відповідають кластерам, на які розбивається вхідний простір. Кількість кластерів заздалегідь, як правило, невідомо. На практиці її визначають на основі методів кластер-аналізу або задають рівним двом і далі нарощують, навчаючи щоразу мережу заново, поки помилка мережі не досягне прийнятного значення.

Навчання радіально-базисної мережі

Як випливає з вищевикладеного, мережа РБ характеризує три типи параметрів:

1) лінійні вагові параметри вихідного шару w_{ij} (входять в опис мережі лінійно);

2) центри c_j – нелінійні (входять в опис нелінійно) параметри прихованого шару;

3) відхилення (радіуси базисних функцій) σ_{ij} – нелінійні параметри прихованого шару.

Навчання мережі, що складається у визначенні цих параметрів, може зводиться до одного з наступних варіантів:

1) задаються центри і відхилення, а обчислюються тільки ваги вихідного шару;

2) визначаються шляхом самонавчання (за допомогою методів кластеризації) центри і відхилення, а для корекції ваг вихідного шару використовується навчання з учителем. Тобто алгоритмом кластеризації формується фіксована множина центрів кластерів. Потім мінімізацією помилки E отримують асоціації центрів кластерів з виходом;

3) визначаються всі параметри мережі за допомогою навчання з учителем, наприклад за допомогою методу зворотного поширення помилки.

Перші два варіанти застосовуються в мережах, що використовують базисні функції з жорстко заданим радіусом (відхилен-

ням). Третій же варіант, будучи найбільш складним і трудомістким в реалізації, передбачає використання будь-яких базисних функцій.

Таким чином, навчання мережі полягає в тому, що

– визначаються центри c_j;

– вибираються параметри σ_i;

– обчислюються елементи матриці ваг W.

2.4 Нейро-нечіткі моделі

Різні типи інтелектуальних систем мають свої особливості, наприклад, за можливостями навчання, узагальнення і отримання результатів, що робить їх найбільш придатними для вирішення одних класів задач і менш придатними – для інших.

Нейронні мережі, наприклад, є зручними для задач розпізнавання образів, але дуже незручні для пояснення, як вони таке розпізнавання здійснюють. Вони можуть автоматично здобувати знання, але процес їхнього навчання найчастіше відбувається досить повільно, а аналіз навченої мережі є дуже складним (навчена мережа є звичайно «чорною скринею» для користувача). При цьому яку-небудь апріорну інформацію (знання експерта) для прискорення процесу навчання в нейронну мережу ввести складно.

Системи з нечіткою логікою, напроти, є зручними для пояснення одержуваних за їхньою допомогою висновків, але вони не можуть автоматично здобувати знання для використання їх у механізмах виведень. Необхідність розбиття універсальних множин на окремі області, як правило, обмежує кількість вхідних змінних у таких системах невеликим значенням.

Хаяши (Y. Hayashi) та Імура (A. Imura) показали, що нейромережа прямого поширення може апроксимувати будь-яку систему, що заснована на нечітких правилах, та будь-яка нейромережа прямого поширення може бути апроксимована системою, що заснована на нечітких правилах.

Узагалі говорячи, теоретично, системи з нечіткою логікою і штучні нейронні мережі подібні одні одним, однак, відповідно до викладеного вище, на практиці в них є власні переваги і недоліки.

Дане розуміння лягло в основу створення апарату **нечітких нейронних мереж**, у яких виведення робляться на основі апарату

нечіткої логіки, але відповідні функції приналежності підбудовуються з використанням методів навчання нейронних мереж, наприклад, методу зворотного поширення помилки. Такі системи не тільки використовують апріорну інформацію, але можуть здобувати нові знання, будучи **логічно прозорими**.

Нейро-нечітка мережа – це подання системи нечіткого виведення у вигляді нейронної мережі, зручної для навчання, поповнення, аналізу та використання. Структура нейро-нечіткої мережі відповідає основним блокам систем нечіткого виведення.

2.4.1 Загальна характеристика та властивості нейро-нечітких мереж

Основними **властивостями нейро-нечітких мереж** є те, що:

– нейро-нечіткі мережі засновані на нечітких системах, які навчаються за допомогою методів, використовуваних у нейромережах;

– нейро-нечітка мережа зазвичай є багатошаровою (частіше – тришаровою) нейронною мережею. Перший шар становить вхідні змінні, середній становить нечіткі правила, а третій – вихідні змінні. Ваги підключення відповідають нечітким множинам вхідних і вихідних змінних. Іноді використовується п'ятишарова архітектура. В загальному випадку нечітка система необов'язково повинна бути подана в такому вигляді, однак це є зручною моделлю для застосування навчаючих методів;

– нейро-нечітка мережа завжди (до, під час і після навчання) може бути інтерпретована як система нечітких правил;

– процедура навчання враховує семантичні властивості нечіткої системи. Це виражається в обмеженні можливих модифікацій, які застосовуються до параметрів, що налагоджуються. Потрібно, однак, сказати, що не всі методи навчання нейро-нечітких мереж враховують семантику системи;

– нейро-нечітка система апроксимує $N \times M$ – розмірну невідому функцію, що частково описана навчаючими даними.

Типи поєднання нечіткої логіки і нейронних мереж за способом взаємодії виділяють такі:

– **нечіткі нейронні системи** (fuzzy neural systems). В цьому випадку в нейронних мережах застосовуються принципи нечіткої ло-

гіки для прискорення процесу налагодження або поліпшення інших параметрів. При такому підході нечітка логіка є лише інструментом нейронних мереж і така система не може бути інтерпретована в нечіткі правила, оскільки являє собою «чорну скриню»;

– **конкуруючі нейро-нечіткі системи** (concurrent neuro-fuzzy systems). У таких моделях нечітка система і нейронна мережа працюють над однією задачею, не впливаючи на параметри одна одної. Можлива послідовна обробка даних спочатку однією системою, потім іншою;

– **паралельні нейро-нечіткі системи** (cooperative neuro-fuzzy systems). У таких системах налагодження параметрів виконується за допомогою нейронних мереж. Далі нечітка система функціонує самостійно. Виділяють такі **типи паралельних нейро-нечітких моделей**: 1) нечітка асоціативна пам'ять (fuzzy associative memory); 2) системи із виділенням нечітких правил шляхом використання карт, що самоорганізуються (fuzzy rule extraction using self-organizing maps); системи, здатні навчати параметри нечітких множин (systems capable of learning fuzzy set parameters);

– **інтегровані (гібридні) нейро-нечіткі системи** (integrated neuro-fuzzy systems) – системи з тісною взаємодією нечіткої логіки і нейронних мереж. Під терміном «нейро-нечіткі мережі» частіше за все мають на увазі системи саме цього типу. Як правило інтегровані системи є системами типу Мамдані або Такагі-Сугено.

За способом відображення нечітких множин в структурі мережі нейро-нечіткі мережі бувають трьох основних типів:

– **системи, побудовані на вибіркових нечітких множинах**. В таких системах ступені приналежності описані лише для деяких значень з області визначення і функція приналежності подана у вигляді вектору. Кожному ступеню приналежності відповідає лише один вхідний або вихідний нейрон. Існує два підходи до реалізації таких систем. В першому система просто апроксимує відповідність виходів входам, така система є «чорною скринею». В другому створюється система із спеціальною архітектурою, в якій втілюються нечіткі правила;

– **системи, параметризовані функції** приналежності яких зберігаються в нейронах. Прикладом таких систем є ANFIS;

– системи, в яких параметризовані функції приналежності використовуються як ваги зв'язків між нейронами. Таку систему інакше можна назвати персептроном з нечіткими зв'язками або **нечітким персептроном**. Прикладами таких систем є NEFCON, NEFCLASS, NEFPROX.

За характером навчання виділяють такі **типи нейро-нечітких мереж**:

– самоналагоджувані нейро-нечіткі мережі – з адаптацією структури та параметрів;

– адаптивні нейро-нечіткі мережі – із жорсткою структурою та адаптацією параметрів мережі.

Адаптивні нейро-нечіткі мережі **за видом методу оптимізації** поділяють на такі, що використовують детерміновані методи типу градієнтного пошуку, та такі, що використовують стохастичні методи, зокрема еволюційні.

Адаптивні нейро-нечіткі мережі **за типом параметрів адаптації** поділяють на мережі з адаптацією параметрів функцій приналежності, мережі з адаптацією ваг правил та мережі з адаптацією параметрів оператора агрегації.

2.4.2 Формування бази знань нейро-нечіткої мережі

Розглянемо об'єкт виду $y = f(x_1, x_2, ...,x_n)$ для якого зв'язок «входи x_i – вихід y» можна подати у вигляді **експертної матриці знань** (табл. 2.1).

Цій матриці відповідає нечітка база знань:

$$\text{Якщо } [(x_1 = a_1^{j1}) \text{ та } (x_2 = a_2^{j1}) \text{ та} \ldots (x_n = a_n^{j1})] \text{ (з вагою } w_{j1}) \ldots$$

$$\ldots \text{ або } [(x_1 = a_1^{jk_j}) \text{ та } (x_2 = a_2^{jk_j}) \text{ та} \ldots (x_n = a_n^{jk_j})] \text{ (з вагою } w_{jk_j}),$$

$$\text{то} \quad y = d_j, \quad j = 1, 2, ..., m, \text{p} = 1, 2, \ldots, k_j,$$

де a_i^{jp} – лінгвістичний терм, що оцінює змінну x_i у рядку p; $p = 1, 2,\ldots,k_i$; k_i – кількість рядків-кон'юнкцій, що відповідають класу d_j вихідної змінної y, w_{jp} – число в діапазоні [0, 1], що характеризує суб'єктивну міру впевненості експерта щодо висловлення з номером p.

Таблиця 2.1 – Експертна матриця знань

Номер правила	Якщо (входи)				То (вихід)	Вага правила
	x_1	x_2	...	x_n		
11	a_1^{11}	a_2^{11}	...	a_n^{11}	d_1	w_{11}
12	a_1^{12}	a_2^{12}	...	a_n^{12}		w_{12}
...	...	...	...	...		...
$1k_1$	$a_1^{1k_1}$	$a_2^{1k_1}$	...	$a_n^{1k_1}$		w_{1k_1}
...	...	...	...	...	...	...
$m1$	a_1^{m1}	a_2^{m1}	...	a_n^{m1}	d_m	w_{m1}
$m2$	a_1^{m2}	a_2^{m2}	...	a_n^{m2}		w_{m2}
...	...	...	...	...		...
mk_m	$a_1^{mk_m}$	$a_2^{mk_m}$	...	$a_n^{mk_m}$		w_{mk_m}

Якщо вихідна змінна (консеквент) є дискретною, то класи d_j, $j = 1, 2, ..., m$, формуються як номери дискретних значень вихідної змінної.

Якщо вихідна змінна є дійсною, то класи d_j, $j = 1, 2, ..., m$, формуються шляхом квантування діапазону $[\underline{y}, \overline{y}]$ вихідної змінної y на m рівнів:

$$[\underline{y}, \overline{y}] = \underbrace{[\underline{y}, y_1]}_{d_1} \cup \ldots \cup \underbrace{[y_{j-1}, y_j]}_{d_j} \cup \ldots \cup \underbrace{[y_{m-1}, \overline{y}]}_{d_m}.$$

Основні принципи формування бази знань нейро-нечітких систем полягають у наступному:

1) **копіювання навчаючої вибірки в базу знань** – для кожного екземпляра навчаючої вибірки формується окреме правило. Перевагою даного методу є простота та висока швидкість роботи, недоліком – відсутність узагальнюючих властивостей і громіздкість одержуваної мережі;

2) **оптимізація кількості продукційних правил** – знаходження такого значення кількості продукційних правил S, при якому значення помилки $E(S)$ є мінімальним, для чого при різних значеннях S навчають мережу і вимірюють значення помилки, після чого оптимізують функцію $E(S)$ за параметром S. Недоліком

даного методу є дуже високі вимоги до обчислювальних ресурсів, обумовлені необхідністю заново навчати мережу на кожному кроці;

3) **спільна оптимізація ваг мережі та кількості продукційних правил** шляхом вирішення багатоекстремальної оптимізаційної задачі або автоматичне визначення числа кластерів у навчаючій вибірці та встановлення центрів функцій приналежності в їхні центри на основі кластер-аналізу;

4) **скорочення (редукція) правил**. В методах скорочення при ініціалізації формується нечітка система, що містить свідомо надлишкове число продукційних правил. У процесі роботи методу зайві продукційні правила виключаються.

Основні принципи редукції правил:

– скорочення нечітких правил відповідно до їхніх логічних функцій: а) виключення правил, для яких результуюча функція приналежності менше визначеного порога, як таких, що мало впливають на остаточний результат; б) виключення суперечливих правил, які взаємно компенсуються; в) виключення одного з двох співпадаючих правил, як таких, що не несуть нової інформації;

– ортогоналізація: видалення тих продукційних правил, вплив яких на точність виявляється мінімальним після оцінки індивідуального внеску кожного продукційного правила у вихідний сигнал мережі, одержуваної шляхом використання ортогонального методу найменших квадратів.

Істотним недоліком методів скорочення є необхідність спочатку працювати зі свідомо надлишковою за розміром базою знань, що обумовлює в ряді випадків повільну роботу методів;

5) **Нарощування (конструювання) правил**: формується початкова база продукційних правил (вона може бути і порожньою), що потім послідовно поповнюється нечіткими правилами.

Виділяють два різновиди методів нарощування:

а) При надходженні чергової навчаючої пари $<x^s, y^s>$ визначають відстані $d(x^s, C^v)$ між центрами передумов наявних правил і точкою x^s. Якщо $\min(d(x^s, C^v)) > d$, $v = 1, 2, \ldots, V$, де d – деякий параметр, що можливо змінюється в процесі роботи методу, то додається ще одне правило з центрами функцій приналежності у точці $C^{V+1} = x^s$ та висновком y^s. У випадку невиконання зазначеної нерівності додавання продукційного правила не відбувається.

Недоліком даного методу є відсутність явного зв'язку між процедурою додавання продукційних правил і точністю апроксимації, що повинна визначатися окремо;

б) нове правило додається при виконанні нерівності $|y^{s*} - y^s| > \varepsilon$, де y^{s*} – значення виходу мережі при подачі на вхід x^s, розраховане за наявними продукційними правилами, ε – постійний параметр, що характеризує точність апроксимації.

Прикладом цього підходу є метод формування бази знань шляхом генерації нових продукційних правил, що не суперечать правилам з бази знань системи, виходячи з аналізу експериментальних даних про об'єкт.

Нехай ми маємо вибірку S пар значень $<x^s, y^s>$, $s = 1, 2, ..., S$; значення S при необхідності допускає модифікацію. Тоді метод формування бази знань може бути описаний у такий спосіб.

Крок 1. З m ($m < S$) довільних значень $<x^s, y^s>$ складається початкова база знань моделі, відображувана матрицею $U_{m\times(N+1)}$ з рядками $<x^s, y^s> = <x^s_1, x^s_2, ... x^s_N, y^s>$. Таке подання еквівалентно набору продукційних правил:

Π_s: якщо $x_1 \in A^s_1$ та $x_2 \in A^s_2$ та ... та $x_N \in A^s_N$, то $y = y^s$, $s = 1, ..., m$.

Крок 2. Для кожної нової експериментальної точки $<x^*, y^*>$ розраховується прогнозоване значення за формулою, що відповідає центроїдному методу:

$$y^*_{\text{p.}} = \frac{\sum_{s=1}^{m} y^s \mu(\left\| x^s - x^* \right\|)}{\sum_{s=1}^{m} \mu(\left\| x^s - x^* \right\|)},$$

де μ – функція колоколообразної або експонентної форми:

$$\mu(\left\| x^s - x^* \right\|) = \exp(-\lambda \sum_{j=1}^{N} \left| x^s_j - x^*_j \right|),$$

де λ – параметр функції.

Крок 3. Якщо $|y^* - y^*_{\text{p.}}| > \varepsilon$, де ε – задана константа, що визначає погрішність апроксимації, тоді база знань поповнюється шляхом розширення матриці U (додаванням рядка $<x^*, y^*>$), у протилежному випадку – матриця U залишається без змін.

Крок 4. Перевіряється правило зупинення. У даному варіанті методу побудова моделі вважається закінченою, якщо відповідно до кроків 2 та 3 перебрані всі S експериментальних точок (без урахування значень початкової бази знань). Якщо не всі експериментальні точки використані, то здійснюється перехід до кроку 2, у протилежному випадку – зупинення.

У процесі реалізації методу параметри λ та ε вважаються наперед заданими. При використанні системи заданими вважаються матриця U (на етапі використання моделі вона не змінюється) та параметри λ й ε, а розрахунок $y^*_{p.}$ здійснюється відповідно до кроку 2 наведеного методу.

Легко бачити, що описаний метод, по суті, відповідає спрощеному методу нечіткого логічного виведення, але відрізняється від останнього тім, що база знань не залишається фіксованою, а модернізується в міру надходження експериментальних даних. Причому несуперечність нового продукційного правила щодо набору правил з бази знань гарантується процедурою її поповнення.

2.4.3 Елементи нейро-нечітких мереж

Розглянемо просту нейронну мережу, що складається з одного нейрона з двома входами (рис. 2.6).

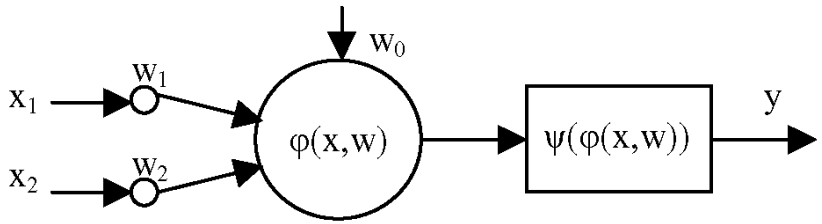


Рисунок 2.6 – Схема штучного нейрона

Вхідні сигнали x_j «взаємодіють» із **синаптичними вагами** w_j, утворюючи значення рівня збудження нейрона $\varphi(x,w) = w_1x_1+w_2x_2+w_0$, де $\varphi(x,w)$ – **функція постсинаптичного потенціалу** (дискримінантна, вагова функція) нейрона, w_0 – **поріг нейро-**

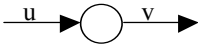
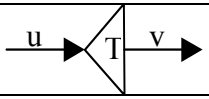
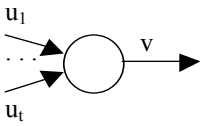
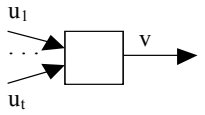
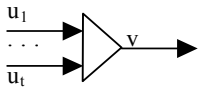
на. Вихід нейрона у утворюється в результаті перетворення значення $\varphi(x,w)$ **функцією активації** ψ:

$$y = \psi(\varphi(x,w)) = \psi(w_1 x_1 + w_2 x_2 + w_0).$$

Розглянута однонейронна мережа, у якій використовуються операції множення, підсумовування і функція активації є **стандартною нейронною мережею**. У випадку застосування замість операцій множення, підсумовування й функцій активації таких операцій, як *t*-норма і *t*-конорма дану нейронну мережу будемо називати **нечіткою**. Входи, виходи і ваги нечіткої нейронної мережі – дійсні числа, що належать відрізку [0, 1].

У табл. 2.2 наведено опис основних елементів нечітких нейромереж. Тут використовуються такі позначення: $\mu_T(x)$ – функція приналежності змінної x терму T; d_j – центр j-го класу правил, $d_j \in [\underline{y}, \overline{y}]$, де $\underline{y}$ та $\overline{y}$ – нижня і верхня межі значень вихідної змінної.

Таблиця 2.2 – Елементи нечітких нейромереж

Назва елемента	Зображення елемента	Функція елемента
Вхід		$v = u$
Нечіткий терм		$v = \mu_T(u)$
Нечітке правило		$v = \min\limits_{i=1,2...,t} u_i$; $v = \prod\limits_{i=1}^{t} u_i$
Клас правил		$v = \max\limits_{i=1,2...,t} u_i$ або $v = \sum\limits_{i=1}^{t} u_i$
Дефазифікація		$v = \sum\limits_{j=1}^{m} u_j d_j \Big/ \sum\limits_{j=1}^{m} u_j$

Розглянемо приклади **елементів нечітких нейронних мереж**.

Нечіткий нейрон «ТА». Сигнали x_j та ваги w_j у даному випадку поєднуються за допомогою t-конорми: $p_j = S(w_j, x_j)$, $j = 1, 2$, а вихід утворюється з застосуванням t-норми: $y = \text{AND}(p_1, p_2) = T(p_1, p_2) = T(S(w_1, x_1), S(w_2, x_2))$. Якщо прийняти $T = \min$, $S = \max$, то нечіткий нейрон «ТА» реалізує композицію min-max: $y = \min(\max(w_1, x_1), \max(w_2, x_2))$.

Нечіткий нейрон «АБО». Сигнали x_j і ваги w_j тут поєднуються за допомогою t-норми: $p_j = T(w_j, x_j)$, $j = 1, 2$, а вихід утворюється з застосуванням t-конорми: $y = \text{OR}(p_1, p_2) = S(p_1, p_2) = S(T(w_1, x_1), T(w_2, x_2))$. Якщо прийняти $T = \min$, $S = \max$, то нечіткий нейрон «АБО» реалізує композицію max-min: $y = \max(\min(w_1, x_1), \min(w_2, x_2))$.

2.4.4 Паралельні нейро-нечіткі системи

У найпростішому випадку паралельна модель (рис. 2.7) може розглядатися як передоброблювач, де механізм навчання штучної нейронної мережі визначає функції приналежності або нечіткі правила системи нечіткого виведення за навчаючими даними. Як тільки параметри системи нечіткого виведення визначено, нейронна мережа припиняє використовуватися.

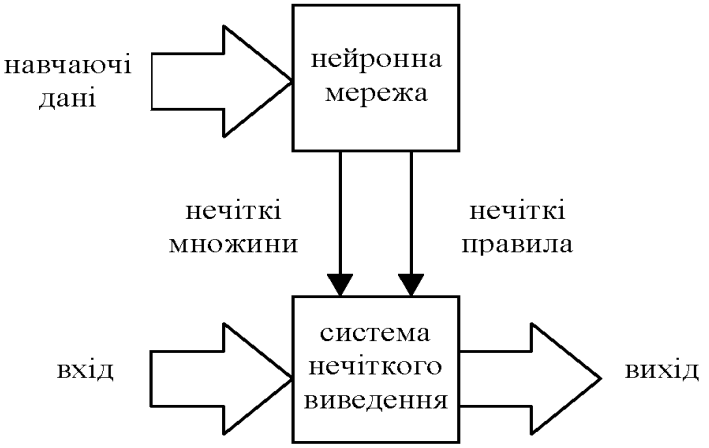


Рисунок 2.7 – Паралельна нейро-нечітка модель

База правил нечіткої системи звичайно визначається на основі карт, що самоорганізуються, або методів нечіткої кластеризації. Функції приналежності звичайно апроксимуються нейронною мережею за навчаючими даними.

Найбільш відомими прикладами паралельних нейро-нечітких систем є: нечітка асоціативна пам'ять Коско (B. Kosko), виділення нечітких правил на основі карт Педрича (W. Pedrycz), що самоорганізуються, та системи Номури (H. Nomura), здатні до навчання параметрів нечітких множин.

Нейро-нечітка мережа FAM (Kosko Fuzzy Associative Memory – нечітка асоціативна пам'ять Коско) заснована на інтерпретації нечіткого правила як асоціації між антецедентом і консеквентом. Якщо нечітка множина подається як точка в одиничному гіперкубі, а правила є асоціаціями, то можливо використовувати нейронну асоціативну пам'ять для збереження нечітких правил. Нейронна асоціативна пам'ять може бути подана її матрицею зв'язків.

Асоціативний пошук еквівалентний множенню ключового фактора на цю матрицю. Ваги зберігають кореляції між ознаками ключа й інформаційною частиною. Через обмежений обсяг асоціативної пам'яті і через небажаність об'єднання багатьох матриць зв'язків в одну матрицю через серйозну втрату інформації необхідно зберігати кожне нечітке правило в окремій мережі FAM. Правила з N змінними, з'єднаними за допомогою кон'юнкцій в антецедентах, можуть бути подані за допомогою N мереж FAM, де кожна мережа зберігає одне правило. Побудова мережі FAM для всього набору правил завершується об'єднанням усіх виходів операцією максимуму і дефазифікацією результату.

Навчання може бути включене в FAM як навчання ваг, зв'язаних з виходом FAM, або шляхом створення FAM цілком за допомогою навчання. Метод навчання нейронної мережі визначає ваги правил для нечітких правил. Такі показники часто інтерпретуються як вплив правил та перемножуються з виходами правил.

Ваги правил можуть бути замінені еквівалентною модифікацією функцій приналежності. Однак це може привести до невірної інтерпретації нечітких множин та того, що ідентичні лінгвістичні значення могли б бути по різному подані в різних правилах. Коско запропонував метод адаптивного квантування вектора для

навчання FAM. Цей метод названий диференціальним конкурентним навчанням і подібний навчанню карт самоорганізації.

Адаптивна пам'ять Коско є паралельною нейро-нечіткою моделлю, оскільки вона використовує навчаючий метод для визначення правил і їхніх ваг. Головною незручністю FAM є зважування правил. Те, що деякі правила не мають сильного впливу, не означає, що вони є малозначимими. Отже, надійність FAM для деяких застосуваньдодатків є сумнівною. Однак через простоту реалізації FAM широко використовується в багатьох застосуваннях.

Виділення нечітких правил з використанням карт, що самоорганізуються (Fuzzy Rule Extraction Using Self Organizing Maps). Педрич запропонував використання карт самоорганізації із змагаючимся шаром для кластеризації навчаючих даних і розробив засоби для інтерпретації результатів навчання.

Результати навчання можуть показувати, чи є два вхідних вектори подібними один одному або чи належать вони до одного й того ж самого класу. Однак, у випадку багатомірних вхідних векторів, структура задачі навчання може рідко виявлятися в двомірній карті. Передбачено процедуру інтерпретації результатів навчання з використанням лінгвістичних змінних.

Після процесу навчання матриця ваг w подає вагу кожної вхідної ознаки стосовно до виходу. Така матриця визначає карту тільки для однієї ознаки. Для кожної вхідної ознаки відповідно до лінгвістичного опису B визначаються нечіткі множини (одна нечітка множина для кожної змінної). Вони застосовуються до матриці ваг w для одержання множини перетворених матриць.

Кожна комбінація лінгвістичних термів – це можливий опис підмножини або екземплярів кластера. Щоб перевірити лінгвістичний опис B на придатність перетворені карти схрещуються і виходить матриця D, що визначає сумісність результату навчання з лінгвістичним описом B. $D(B)$ – це нечітке відношення, яке інтерпретується як ступінь підтримки B.

Описуючи відношення $D(B)$ його a-розтинами $D_a(B)$ одержують підмножини вихідних вузлів, ступінь приналежності яких є принаймні a, так, що достовірність приналежності всіх екземплярів x_a до класу, описаному B, скорочується зі зменшенням a. Кожний опис B є придатним описом кластера, якщо $D(B)$ має непорожній a-розтин $D_a(B)$.

Якщо ознаки розподілені на вхідні і вихідні, то кожний опис B подає лінгвістичне правило, і, досліджуючи кожну комбінацію лінгвістичних значень, можна створити повну базу нечітких правил. Цей метод також показує, які зразки належать нечіткому правилу, оскільки вони не містяться в жодній підмножині x_a.

Важливою перевагою даного методу порівняно з FAM є те, що правила не зважуються. Проблемою є визначення кількості вихідних нейронів і значень a для кожної задачі навчання. Оскільки форма функції приналежності сильно впливає на якість роботи мережі дані можуть бути краще використані в порівнянні з FAM. Оскільки процедура навчання Коско не бере до уваги відношення сусідства між вихідними нейронами, гарне топологічне відображення вхідних даних до вихідного не завжди може бути отримано. Таким чином, навчаюча процедура FAM більше залежить від послідовності навчаючих даних ніж навчаюча процедура Педрича. Спочатку визначається структура простору ознак і потім, використовуючи доступні нечіткі розбиття, отримують лінгвістичні описи, що найкраще відповідають результатам навчання. Якщо велика кількість екземплярів не відповідає жодному з описів, це може пояснюватися невірним вибором функцій приналежності, і вони можуть бути перевизначені. Отже для навчання нечітких правил цей підхід є кращим у порівнянні з FAM.

Ефективність даного методу усе ще залежить від кроку навчання і розміру околиці для модифікації ваг, що у залежності від задачі визначаються евристично.

Системи, здатні навчати нечіткі множини. Номура (H. Nomura), Хаяши (I. Hayashi) та Вакамі (N. Wakami) запропонували метод контрольованого навчання для настроювання параметрів нечітких множин нечіткої системи Такагі-Сугено.

Метод навчання використовує процедуру градієнтного спуску, що використовує міру помилки E (різниця між фактичним і цільовим значенням виходу) для настроювання параметрів функцій приналежності. Дана процедура подібна дельта-правилу багатошарових персептронів. Навчання здійснюється в режимі реального часу. Для вхідного вектора обчислюється результуюча помилка E, заснована на зміні консеквентів правил. Потім для тих же екземплярів налагоджуються параметри функцій приналежно-

сті. Це робиться для того, щоб врахувати зміни в консеквентах, коли модифікуються антецеденти.

Серйозним недоліком даного методу є те, що подання лінгвістичних значень вхідних змінних залежить від правил, у яких вони з'являються. Спочатку ідентичні лінгвістичні терми подаються ідентичними функціями приналежності. Протягом процесу навчання вони можуть по різному змінюватися так, щоб ідентичні лінгвістичні терми були подані різними нечіткими множинами. Даний метод застосовується тільки до систем нечіткого виведення типу Такагі-Сугено.

Використовуючи подібний підхід, Мійоши (T. Miyoshi), Тано (S.Tano), Като (Y. Kato) та Арнольд (T. Arnould), налагоджували оператори *t*-норми і *t*-конорми, у той час, як Ягер (R. Yager) налагоджував оператор дефазифікації, використовуючи метод контрольованого навчання.

2.4.5 Конкурентні нейро-нечіткі системи

У конкурентній моделі нейронна мережа допомагає системі нечіткого виведення безупинно визначати необхідні параметри, особливо, якщо вхідні змінні не можуть бути безпосередньо вимірювані. Така комбінація мереж не оптимізує нечітку систему, але допомагає поліпшити експлуатаційні якості всієї конкуруючої системи. Навчання застосовується тільки для нейронної мережі, а нечітка система залишається незмінною протягом цієї стадії.

У деяких випадках виходи нечіткої системи безпосередньо не можуть бути застосовні до керованого процесу. У цьому випадку нейронна мережа може діяти як постпроцесор виходів нечіткої системи.

На рис. 2.8 зображено конкурентну нейро-нечітку систему, де вхідні дані подаються на входи нейронної мережі, а вихід нейронної мережі далі оброблюється нечіткою системою.

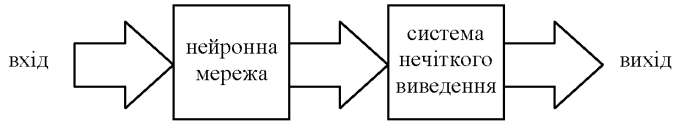


Рисунок 2.8 – Конкурентна нейро-нечітка система

2.4.6 Інтегровані нейро-нечіткі системи

В інтегрованій моделі для визначення параметрів системи нечіткого виведення використовуються методи навчання нейронних мереж. Інтегровані нейро-нечіткі системи розподіляють структури даних і подання знань.

Система нечіткого виведення може використовувати людські експертні знання, зберігаючи їхні істотні компоненти в базі правил і базі даних, та виконувати нечітке виведення для одержання вихідного значення. Формування правил і відповідних функцій приналежності сильно залежить від апріорного знання щодо розглянутої системи. Однак універсальний спосіб перетворення дослідних знань людських експертів у базу знань системи нечіткого виведення є невідомим. Є також потреба в адаптованості або деяких методах навчання для одержання виходів з необхідним рівнем точності.

З іншого боку, механізм навчання нейронних мереж не покладається на людську експертизу. Через однорідну структуру нейронних мереж, з них складно витягти структуроване знання. Ваги нейронної мережі являють собою коефіцієнти гіперплощини, що розподіляє вхідний простір на області з різними значеннями виходу. Якщо ми можемо візуалізувати цю структуру гіперплощини за навчаючими даними, тоді наступні процедури навчання в нейронній мережі можуть бути виключені. Однак, у дійсності, апріорне знання звичайно виходить від людських експертів, що більш відповідає уявленню знань у вигляді набору нечітких правил «якщо, то», і це складно закодувати в нейронну мережу.

У значній мірі недоліки цих двох підходів компенсуються. Тому, природно розглянути створення інтегрованої системи, що сполучить концепції нейромереж і систем нечіткого виведення.

Найбільш загальний спосіб застосування методу навчання до нечіткої системи полягає в тому, щоб подати її у вигляді архітектури, подібної нейронній мережі. Однак звичайні (градієнтні) методи навчання нейронних мереж не можуть безпосередньо застосовуватися до такої системи, оскільки функції, використовувані в процесі виведення звичайно є недиференційовані. Ця проблема може бути вирішена шляхом використання диференційованих

функцій у системі виведення або не використання стандартних методів навчання нейромереж.

Очевидно, що паралельні та конкуруючі моделі не є цілком інтерпретованими через присутність нейронної мережі («чорна скриня»). У той же час інтегрована нейро-нечітка модель є інтерпретованою і здатна до контрольованого навчання (чи навіть до навчання з підкріпленням, подібно NEFCON). У FALCON, GARIC, ANFIS, NEFCON, SONFIN, FINEST і FUN процес навчання сконцентрований тільки на настроюванні значень параметрів у межах установлених структур. Для багатомірних задач буде складним визначити оптимальні структури «передумованаслідок», кількість правил і т. д. Користувач повинний визначити деталі архітектури: тип і кількість функцій приналежності для вхідної і вихідної змінних, тип нечітких операторів і т. д. FINEST містить механізм, заснований на поліпшеному узагальненому modus ponens для настроювання нечітких предикатів і комбінуючих функцій, а також настроювання функції імплікації. Важливою особливістю EFUNN та dmEFuNN є однопрохідне навчання в реальному режимі часу.

Серед інтегрованих нейро-нечітких моделей ANFIS має найбільшу точність, а NEFPROX – найменшу. Це пояснюється тим, що в ANFIS реалізовані правила Такагі-Сугено, а NEFPROX є нечіткою системою типу Мамдані. Однак, NEFPROX швидше працює в порівнянні з ANFIS. Через меншу кількість правил SONFIN, EFUNN і dmEFuNN також здатні працювати швидше ніж ANFIS. Отже, є протиріччя між інтерпретабельністю і точністю. Системи виведення типу Такагі-Сугено є більш точними, але вимагають більше обчислювальних витрат. У той час як системи виведення типу Мамдані є більш інтерпретабельними і вимагають менше обчислювальних витрат, але часто з компромісом за точністю.

2.4.6.1 Нейро-нечіткий апроксиматор Мамдані

Нейро-нечіткий апроксиматор Мамдані (Mamdani neuro-fuzzy approximator) є найбільш узагальненою моделлю нейро-нечіткої мережі, побудованої на правилах Мамдані. Структуру мережі Мамдані зображено на рис. 2.9.

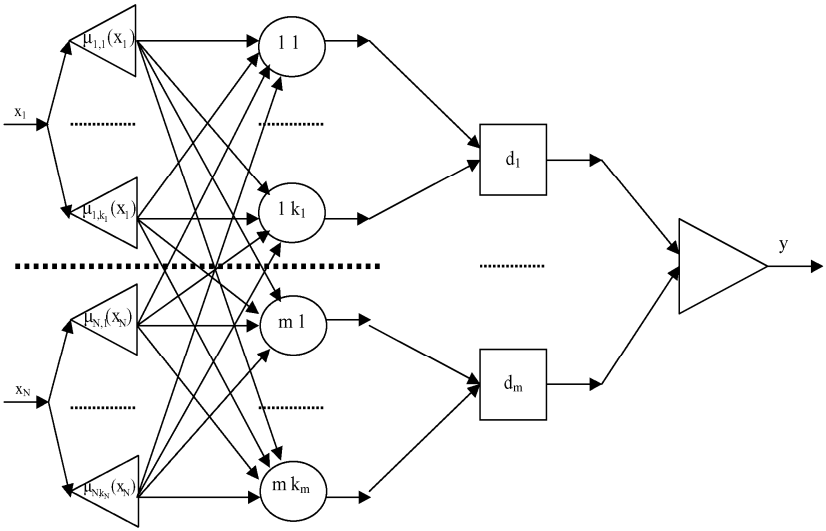


Рисунок 2.9 – Структура нейро-нечіткої мережі Мамдані

Нейро-нечітка мережа Мамдані має п’ять шарів.

Перший (вхідний) шар утворюють елементи вхідного вектора.

Другий шар містить нечіткі терми, які відповідають вхідним змінним, та обчислює приналежності вхідного вектора до кожного з нечітких термів:

$$\mu_{jp}(x_i) = \frac{1}{1 + \left(\dfrac{x_i - b_i^{jp}}{c_i^{jp}} \right)^2}, \quad i = \overline{1,N},\ p = \overline{1,k_j},\ j = \overline{1,m},$$

де $\mu_{jp}(x_i)$ – функція приналежності змінної x_i терму p-го рядка j-го правила, a_i^{jp}, b_i^{jp}, c_i^{jp} – параметри настроювання функцій приналежності.

Третій шар утворюють рядки-кон’юнкції антецедентів правил нечіткої бази знань: кожний вузол цього шару реалізує оператор T-норми та видає ступінь відповідності вхідного вектора до співставленого вузлу правила.

Четвертий шар поєднує правила в класи d_j, j = 1, 2, …, m, та обчислює ступені приналежності вхідного вектора до відповідних

термів вихідної змінної; кожний вузол цього шару реалізує оператор *T*-конорми.

Функція приналежності вхідного вектора до *j*-го терму d_j вихідної змінної у визначається як:

$$\mu_{d_j}(y) = \max_{p=1,k_j} \{w_{jp} \min_{i=1,N} [\mu_{jp}(x_i)]\}.$$

П'ятий шар містить один нейрон, що поєднує приналежності до нечітких термів вихідної змінної та виконує операцію дефазифікації:

– для дискретного виходу: $y = \dfrac{\sum_{j=1}^{m} d_j \mu_{d_j}(y)}{\sum_{j=1}^{m} \mu_{d_j}(y)}$;

– для неперервного виходу: $y = \dfrac{y\mu_{d_1}(y) + y_1\mu_{d_2}(y) + \ldots + y_{m-1}\mu_{d_m}(y)}{\sum_{j=1}^{m} \mu_{d_j}(y)}$.

Число вузлів у нейро-нечіткій мережі визначається так: шар 1 – за кількістю елементів вхідного вектора; шар 2 – за кількістю нечітких термів у базі знань (n); шар 3 – за кількістю рядків-кон'юнкцій у базі знань; шар 4 – за кількістю класів, на які розбивається діапазон вихідної змінної (m).

Дугам графа присвоюються такі ваги: одиниця (дуги між першим і другим шарами); функції приналежності входу до нечіткого терму (дуги між другим і третім шарами); ваги правил (дуги між третім і четвертим шарами); одиниця (дуги між четвертим і п'ятим шарами). Пороги нейронів вважають відсутніми або такими, що дорівнюють нулю.

Навчання нейро-нечіткої мережі Мамдані здійснюють методом зворотного поширення помилки з метою мінімізації **критерію помилки**

$$E = \frac{1}{V} \sum_{s=1}^{S} (y^{s*} - y^s)^2 ,$$

що застосовується в теорії нейронних мереж, де y^{s*} та y^s – бажане та розрахункове значення виходу мережі для *s*-го екземпляра, *V* – певна константа, як правило *V* = 2 або *S*.

Доти, поки помилка E не досягне максимального прийнятного рівня, послідовно для кожного s-го екземпляра x^s, s = 1, 2, …, S, виконують два етапи. На першому етапі (прямий хід) для s-го екземпляра обчислюють розрахункове значення виходу мережі y^s. На другому етапі (зворотний хід) обчислюють значення миттєвої помилки E_t та перераховують ваги зв'язків:

$$E_t = \frac{1}{2}(y^{s*}(t) - y^s(t))^2, \quad w_{jp}(t+1) = w_{jp}(t) - \alpha \frac{\partial E_t}{\partial w_{jp}(t)},$$

$$c_i^{jp}(t+1) = c_i^{jp}(t) - \alpha \frac{\partial E_t}{\partial c_i^{jp}(t)},$$

$$b_i^{jp}(t+1) = b_i^{jp}(t) - \alpha \frac{\partial E_t}{\partial b_i^{jp}(t)}, \quad j = \overline{1,m}, \quad i = \overline{1,N}, \quad p = k_j,$$

де $y^{s*}(t)$ та $y^s(t)$ – бажане та розрахункове значення виходу мережі для s-го екземпляра на t-му кроці навчання, $w_{jp}(t)$, $c_i^{jp}(t)$, $b_i^{jp}(t)$ – ваги правил і параметри функцій приналежності на t-му кроці навчання; α – величина кроку навчання.

Часткові похідні, що використовуються у правилах корегування параметрів мережі, характеризують чуттєвість помилки E_t до зміни параметрів та обчислюються таким чином:

$$\frac{\partial E_t}{\partial w_{jp}} = \varepsilon_1\varepsilon_2\varepsilon_3 \frac{\partial \mu_{d_j}}{\partial w_{jp}}, \quad \frac{\partial E_t}{\partial c_i^{jp}} = \varepsilon_1\varepsilon_2\varepsilon_3\varepsilon_4 \frac{\partial \mu_{jp}(x_i)}{\partial c_i^{jp}},$$

$$\frac{\partial E_t}{\partial b_i^{jp}} = \varepsilon_1\varepsilon_2\varepsilon_3\varepsilon_4 \frac{\partial \mu_{jp}(x_i)}{\partial b_i^{jp}},$$

де $\varepsilon_1 = \frac{\partial E_t}{\partial y} = y_t^{s*} - y_t^s$, $\varepsilon_2 = \frac{\partial y}{\partial \mu_{d_j}} = \frac{d_j \sum_{q=1}^{m} \mu_{d_q} - \sum_{q=1}^{m} d_q \mu_{d_q}}{\left(\sum_{q=1}^{m} \mu_{d_q}\right)^2}$, $\varepsilon_3 = \frac{\partial \mu_{d_j}}{\partial \left(\prod_{i=1}^{N} \mu_{jp}(x_i)\right)} = w_{jp}$,

$$\varepsilon_4 = \frac{\partial \left(\prod_{i=1}^{N} \mu_{jp}(x_i)\right)}{\partial \mu_{jp}(x_i)} = \frac{1}{\mu_{jp}(x_i)} \prod_{i=1}^{N} \mu_{jp}(x_i), \quad \frac{\partial \mu_{d_j}}{\partial w_{jp}} = \prod_{i=1}^{N} \mu_{jp}(x_i),$$

$$\frac{\partial \mu_{jp}(x_i)}{\partial c_i^{jp}} = \frac{2c_i^{jp}\left(x_i - b_i^{jp}\right)^2}{\left(\left(c_i^{jp}\right)^2 + \left(x_i - b_i^{jp}\right)^2\right)^2}, \quad \frac{\partial \mu_{jp}(x_i)}{\partial b_i^{jp}} = \frac{2\left(c_i^{jp}\right)^2\left(x_i - b_i^{jp}\right)}{\left(\left(c_i^{jp}\right)^2 + \left(x_i - b_i^{jp}\right)^2\right)^2}.$$

2.4.6.2 Нейро-нечітка мережа FALCON

Нейро-нечітка мережа FALCON (Fuzzy Adaptive Learning Control Network – нечітка мережа керування з адаптивним навчанням) має п'ятишарову архітектуру, як показано на рис. 2.10, і реалізує систему нечіткого виведення Мамдані. Для кожної вихідної змінної в мережі є два лінгвістичних вузли. Один вузол – для навчаючих даних (бажаний вихід), а інший – для фактичного виходу FALCON. Перший схований шар виконує фазифікацію вхідних змінних. Кожен вузол може бути єдиним вузлом, що реалізує просту функцію приналежності, або складатися з декількох шарів вузлів, що обчислюють складну функцію приналежності. Другий схований шар визначає антецеденти правил, що супроводжуються консеквентами правил, які реалізує третій схований шар.

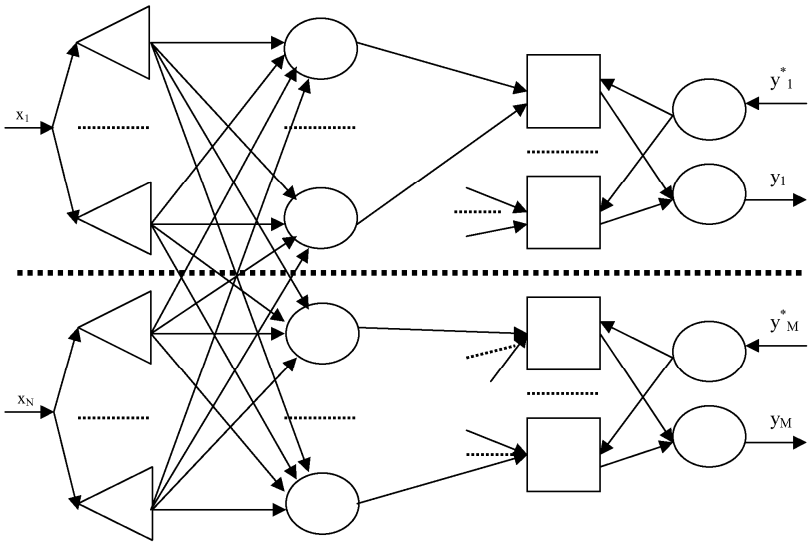


Рисунок 2.10 – Мережа FALCON

FALCON використовує гібридний метод навчання на основі неконтрольованого навчання і градієнтного спуску для оптимального настроювання параметрів з метою одержання бажаних виходів. Гібридне навчання здійснюється на двох стадіях.

У початковій стадії центри і ширина функцій приналежності визначаються методами навчання, що самоорганізуються, аналогічними статистичним методам кластеризації. Як тільки початкові параметри визначені формулюються антецеденти правил. Метод конкурентного навчання використовується для визначення зв'язків консеквентів правил для кожного вузла правила. Після того, як база нечітких правил отримана, уся структура мережі є відомою.

Потім мережа входить у другу стадію навчання для оптимального настроювання параметрів (вхідних і вихідних) функцій приналежності. Метод зворотного поширення помилки використовується для контрольованого навчання.

Таким чином, метод FALCON забезпечує основу для структурної і параметричної адаптації при проектуванні нейро-нечітких систем.

2.4.6.3 Нейро-нечітка мережа Такагі-Сугено-Канга

Нейро-нечітка мережа Такагі-Сугено-Канга (Takagi-Sugeno-Kang Neuro-fuzzy network, TSK) виконує нечітке виведення із використанням N змінних x_j та m правил:

Π_k: Якщо $x_1 \in A^{(k)}{}_1$ та $x_2 \in A^{(k)}{}_2$ та … та $x \in A^{(k)}{}_N$, тоді

$$y_k = w_0^k + \sum_{j=1}^{N} w_j^k x_j,$$

де $A^{(k)}{}_j$ – нечіткий терм, до якого має належати j-та вхідна змінна, щоб активізувати k-те правило, w^k_j – ваговий коефіцієнт, k = 1, 2, …, m.

Мережа Такагі-Сугено-Канга (рис. 2.11) складається з п'яти шарів.

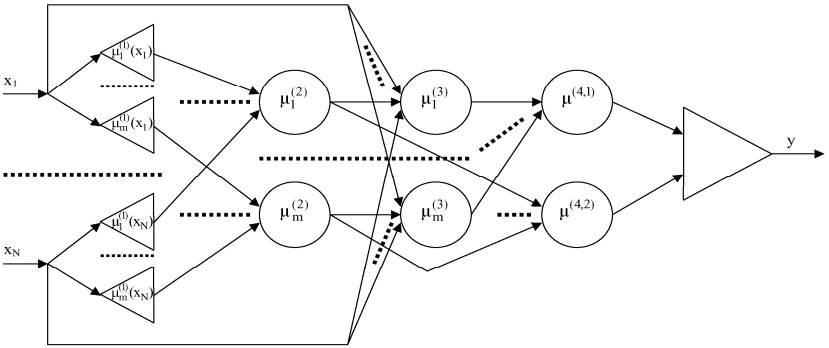


Рисунок 2.11 – Мережа Такагі-Сугено-Канга

Перший шар виконує окрему фазифікацію кожної вхідної змінної x_j, $j = 1, 2, \ldots, N$, визначаючи для кожного k-го правила значення функції приналежності

$$\mu_k^{(1)}(x_j) = \frac{1}{1 + \left(\frac{x_j - c_{jk}}{\sigma_{jk}} \right)^{2b_{jk}}},$$

параметри якої $c_{jk}, \sigma_{jk}, b_{jk}$ підлягають адаптації в процесі навчання.

Другий шар виконує агрегування функцій приналежності вхідних змінних до термів антецедентів нечітких правил, визначаючи приналежність вхідного вектора до k-го правила (рівень активації правила):

$$\mu_k^{(2)} = \prod_{j=1}^{N} \mu_k^{(1)}(x_j, c_{jk}, \sigma_{jk}, b_{jk}), \; k = 1, 2, \ldots, m.$$

Третій шар генерує значення функцій консеквентів нечітких правил з урахуванням рівнів активації правил:

$$\mu_k^{(3)} = \mu_k^{(2)} \left(w_0^k + \sum_{j=1}^{N} w_j^k x_j \right).$$

Це параметричний шар, у якому адаптації підлягають лінійні ваги w^k_j, $k = 1, 2, \ldots, m$, $j = 1, 2, \ldots, N$.

Четвертий шар агрегує m правил виведення (перший нейрон) і генерує нормалізуючий сигнал (другий нейрон):

$$\mu^{(4,1)} = \sum_{k=1}^{m} w_k^{(4)} \mu_k^{(3)}, \ \mu^{(4,2)} = \sum_{k=1}^{m} \mu_k^{(3)}.$$

П'ятий (вихідний) шар містить один нейрон і здійснює нормалізацію, формуючи вихідний сигнал y: $y(x_1, x_2, ..., x_N) = \mu^{(4,1)} \Big/ \mu^{(4,2)}$.

Мережа Такагі-Сугено-Канга містить тільки два параметричних шари (перший і третій), параметри яких уточнюються в процесі навчання. Параметри першого шару будемо називати нелінійними параметрами, а параметри третього шару – лінійними вагами.

Якщо прийняти, що в конкретний момент часу параметри умов зафіксовані, то функція $y(x_1, x_2, \ldots, x_N)$ є лінійною щодо змінних $x_1, x_2, \ldots, x_N$.

При наявності N вхідних змінних кожне правило формує N+1 змінних w^k_j лінійної залежності $y_k(x_1, x_2, \ldots, x_N)$. При m правилах виведення це дає $m(N+1)$ лінійних параметрів мережі. У свою чергу, кожна функція приналежності використовує три параметри $c_{jk}, \sigma_{jk}, b_{jk}$, що підлягають адаптації. Якщо прийняти, що кожна змінна x_j, характеризується власною функцією приналежності, то при m правилах виведення ми одержимо $3m$ нелінійних параметрів. У сумі це дає $m(4N+1)$ лінійних і нелінійних параметрів, значення яких повинні підбиратися в процесі навчання мережі.

На практиці для зменшення кількості параметрів, що адаптуються, оперують меншою кількістю незалежних функцій приналежності для окремих змінних, керуючись правилами, у яких комбінуються функції приналежності різних змінних. Якщо прийняти, що кожна змінна x_j має k_j різних функцій приналежності, то максимальна кількість правил, яку можна створити при їхньому комбінуванні, складе: $m = k_j$. У такий спосіб сумарна кількість нелінійних параметрів мережі при m правилах висновку зменшується з $3m$ у загальному випадку до $3Nm^{1/N}$. Кількість лінійних

параметрів при подібній модифікації залишається без змін, тобто $m(N+1)$.

Завдання мережі полягає в такому відображенні пар даних $<x, y>$, щоб для вхідного вектора x^s розрахункове значення вихідної ознаки $y^{s*} = y(x^s)$ не сильно відрізнялося б від співставленого x^s фактичного значення цільової ознаки y^s.

Навчання мережі засновано на мінімізації цільової функції

$$E = \frac{1}{2}\sum_{s=1}^{S}\left(y^{s*} - y^s\right)^2,$$

де S – кількість навчаючих пар $<x^s, y^s>$, та є контрольованим.

Гібридний метод навчання застосовується для мережі Такагі-Сугено-Канга й інших нейро-нечітких мереж, подібних їй. У гібридному методі підлягаючі адаптації параметри розподіляються на дві групи: одна група складається з лінійних параметрів w^k_j третього шару, а інша група – з параметрів нелінійних функцій приналежності першого шару. Уточнення параметрів проводиться в два етапи.

На першому етапі при фіксації визначених значень параметрів функцій приналежності (у першому циклі – це значення, отримані в результаті ініціалізації) шляхом розв'язку системи лінійних рівнянь розраховуються лінійні параметри w^k_j. При відомих значеннях функції приналежності залежність $y(x_1, x_2, \ldots, x_N)$ можна подати в лінійній формі:

$$y(x_1, x_2, \ldots, x_N) = \sum_{k=1}^{m} w'_k\left(w^k_0 + \sum_{j=1}^{N} w^k_j x_j\right), \quad w'_k = \left(\sum_{p=1}^{m}\left(\prod_{j=1}^{N}\mu^{(1)}_p(x_j)\right)\right)^{-1}\prod_{j=1}^{N}\mu^{(1)}_k(x_j).$$

При S екземплярах навчаючої вибірки $<x^s, y^s>$, $s = 1, 2, \ldots, S$, одержимо систему з лінійних рівнянь у матричній формі: $Aw = y$, де $w = (w^1, w^2, \ldots, w^m)^T$, $w^k = (w^k_0, w^k_1, \ldots, w^k_N)^T$, $y = (y_1, y_2, \ldots, y_S)^T$, $A = (a_1, a_2, \ldots, a_S)^T$, $a_s = (a_{s1}, a_{s2}, \ldots, a_{sm})^T$, $a_{sk} = (\mu^{(2)}_k(x^s), \mu^{(2)}_k(x^s)x^s_1, \mu^{(2)}_k(x^s)x^s_2, \ldots, \mu^{(2)}_k(x^s)x^s_N)^T$. Розмірність матриці A дорівнює $S(N+1)m$, при цьому звичайно кількість рядків S значно більше кількості стовпців $(N+1)m$. Розв'язок цієї системи: $w = A^{-1}y$.

На другому етапі після фіксації значень лінійних параметрів w^k_j розраховуються вихідні сигнали мережі $y = \{y^{s*}\}$ ($s = 1, 2, \ldots, S$): $y = Aw$, а слідом за ними – вектор помилки $e = \{e^s\}$, $e^s = y^{s*} - y^s$. Сигнали помилок направляються через підключену мережу за напрямком до входу мережі (зворотне поширення) аж до першого шару, де можуть бути розраховані компоненти градієнта цільової функції щодо конкретних параметрів $c_{jk}, \sigma_{jk}, b_{jk}$.

Після формування вектора градієнта параметри уточнюються з використанням одного з градієнтних методів навчання. Якщо застосовується найпростіший метод найшвидшого спуску, то відповідні формули адаптації приймають форму:

$$c_{jk}(t+1) = c_{jk}(t) - \alpha_c \frac{\partial E(t)}{\partial c_{jk}},\ \ \sigma_{jk}(t+1) = \sigma_{jk}(t) - \alpha_\sigma \frac{\partial E(t)}{\partial \sigma_{jk}},\ \ b_{jk}(t+1) = b_{jk}(t) - \alpha_b \frac{\partial E(t)}{\partial b_{jk}},$$

де t – номер ітерації, α_c, α_σ, α_b – коригувальні прирости (кроки навчання).

Після уточнення нелінійних параметрів знову запускається процес адаптації лінійних параметрів (перший етап) і нелінійних параметрів мережі (другий етап). Цей цикл повторюється аж до стабілізації всіх параметрів процесу навчання.

Остаточний вид формул настроювання параметрів залежить як від використовуваного визначення функції помилки на виході мережі, так і від виду функції приналежності. Так при використанні функцій Гауса як функцій приналежності та середньоквадратичної помилки, часткові похідні цільової функції приймають вигляд:

$$\frac{\partial E}{\partial c_{jk}} = \left(y^{s*} - y^s\right) \sum_{p=1}^{m} \left(w_0^p + \sum_{i=1}^{N} w_i^p x_i \right) \frac{\partial w'_p}{\partial c_{jk}},\ \ \frac{\partial E}{\partial \sigma_{jk}} = \left(y^{s*} - y^s\right) \sum_{p=1}^{m} \left(w_0^p + \sum_{i=1}^{N} w_i^p x_i \right) \frac{\partial w'_p}{\partial \sigma_{jk}},$$

$$\frac{\partial E}{\partial b_{jk}} = \left(y^{s*} - y^s\right) \sum_{p=1}^{m} \left(w_0^p + \sum_{i=1}^{N} w_i^p x_i \right) \frac{\partial w'_p}{\partial b_{jk}},$$

$$\frac{\partial w'_p}{\partial c_{jk}} = \beta_{pk} \left(\prod_{i=1, i\neq j}^{N} \mu_k^{(1)}(x_i) \right) \left(\frac{2b_{jk}}{\sigma_{jk}} \right) \left(\frac{x_j - c_{jk}}{\sigma_{jk}} \right)^{2b_{jk}-1} \left(1 + \left(\frac{x_j - c_{jk}}{\sigma_{jk}} \right)^{-2b_{jk}} \right)^{-2},$$

$$\frac{\partial w'_p}{\partial \sigma_{jk}} = \beta_{pk}\left(\prod_{i=1,i\neq j}^{N}\mu_k^{(1)}(x_i)\right)\left(\frac{2b_{jk}}{\sigma_{jk}}\right)\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{2b_{jk}}\left(1+\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{-2b_{jk}}\right)^{-2},$$

$$\frac{\partial w'_p}{\partial c_{jk}} = \beta_{pk}\left(\prod_{i=1,i\neq j}^{N}\mu_k^{(1)}(x_i)\right)\left(-2\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{2b_{jk}}\ln\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)\right)\left(1+\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{-2b_{jk}}\right)^{-2},$$

$$\beta_{pk} = \left(\delta_{pk}\sum_{q=1}^{m}\left(\prod_{j=1}^{N}\mu_q^{(1)}(x_j)\right) - \prod_{j=1}^{N}\mu_k^{(1)}(x_j)\right)\left(\sum_{q=1}^{m}\left(\prod_{j=1}^{N}\mu_q^{(1)}(x_j)\right)\right)^{-2}, \quad \delta_{pk} = \begin{cases}1, p = k,\\ 0, p \neq k.\end{cases}$$

Незважаючи на складну структуру наведених формул, що виражають компоненти вектора градієнта, вони дозволяють аналітично визначити величини, необхідні для уточнення параметрів нечіткої мережі.

При практичній реалізації гібридного методу навчання нечітких мереж домінуючим фактором їхньої адаптації вважається перший етап, на якому ваги w_j^k підбираються з використанням псевдоінверсії за один крок. Для зрівноважування його впливу другий етап (підбір нелінійних параметрів градієнтним методом) багаторазово повторюється в кожному циклі.

Гібридний метод є одним з найбільш ефективних способів навчання нейро-нечітких мереж. Його головна особливість полягає в розподілі процесу навчання на два відособлених у часі етапи. На кожному етапі уточнюється тільки частина параметрів мережі. Якщо взяти до уваги, що обчислювальна складність кожного методу оптимізації пропорційна (нелінійно) кількості параметрів, то зменшення розмірності задачі оптимізації істотно скорочує кількість математичних операцій і збільшує швидкість виконання методу. Завдяки цьому гібридний метод є значно більш ефективним, ніж звичайний градієнтний метод фронтального типу, відповідно до якого уточнення всіх параметрів мережі робиться паралельно й одночасно.

2.4.6.4 Нейро-нечітка мережа ANFIS

Нейро-нечітка мережа ANFIS (Adaptive Neuro Fuzzy Inference System – адаптивна система нейро-нечіткого виведення) запропонована Янгом (R. Jang) і реалізує систему нечіткого виведення Такагі-Сугено у вигляді п'ятишарової нейронної мережі прямого поширення сигналу (рис. 2.12).

Мережа ANFIS є ізоморфною базі правил Такагі-Сугено.

Перший шар містить нейрони, які відповідають нечітким термам вхідних змінних із функціями приналежності $\mu_{i,j}^{(1)}(x_i)$, де x_i – i-ий вхід, j – номер нечіткої множини, визначеної для i-го входу. Як функція приналежності зазвичай обирається колоколообразна функція або функція Гауса. Входи мережі з'єднані тільки зі своїми термами. Кількість вузлів першого шару дорівнює сумі потужностей терм-множин вхідних змінних.

Другий шар містить m нейронів, на входи яких поступають значення з виходів вузлів першого шару, що формують антецеденти правил. Кожний k-ий нейрон другого шару знаходить ступінь виконання відповідного правила:

$$\mu_k^{(2)} = \min_{\substack{i=1,2,\ldots,N \\ j=1,2,\ldots,k_i}} (w_{i,j}^{(2,k)} \mu_{i,j}^{(1)}) \text{ або } \mu_k^{(2)} = \prod_{i=1}^{N} \prod_{j=1}^{k_i} \mu_{i,j}^{(1)}, w_{i,j}^{(2,k)} = 1,$$

де $w_{i,j}^{(2,k)} = 1$, якщо j-ий терм i-ої ознаки входить в умову k-го правила, $w_{i,j}^{(2,k)} = 0$ – у протилежному випадку.

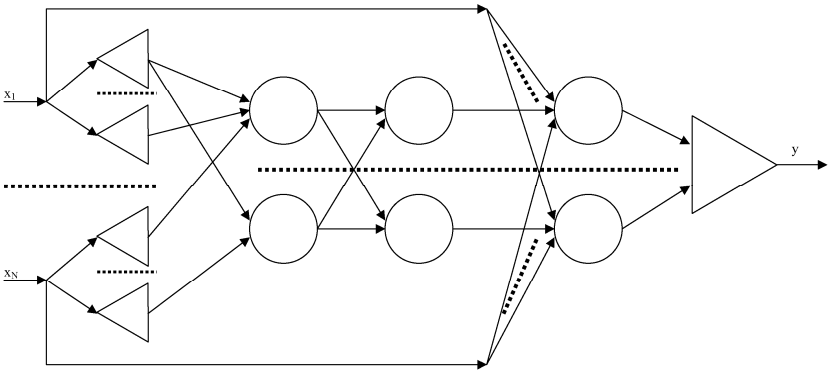


Рисунок 2.12 – Мережа ANFIS

Третій шар містить m нейронів, які знаходять нормалізовані ступені виконання правил:

$$\mu_j^{(3)} = \frac{\mu_j^{(2)}}{\sum_{i=1}^{m} \mu_i^{(2)}}, j = 1, 2, ..., m.$$

Четвертий шар містить m нейронів, які обчислюють консеквенти (лінійні комбінації вхідних сигналів з урахуванням ступенів виконання) правил:

$$y_j = \mu_j^{(3)} \sum_{i=1`}^{N} w_i^{(4,j)} x_i, j = 1, 2, ..., m.$$

Кожний вузол четвертого шару з'єднаний з одним вузлом третього шару а також із усіма входами мережі.

П'ятий шар містить один нейрон, який обчислює загальний вихід мережі, складаючи консеквенти правил:

$$y = \sum_{j=1}^{m} y_j \ .$$

Для налагодження параметрів ANFIS застосовують комбінацію градієнтного спуску у вигляді методу зворотного поширення помилки і методу найменших квадратів. Метод зворотного поширення помилки налагоджує параметри антецедентів правил (функцій приналежності). Методом найменших квадратів оцінюються коефіцієнти висновків правил, тому що вони лінійно пов'язані з виходом мережі.

Кожна ітерація процедури настроювання виконується в два етапи. На першому етапі на входи подається навчаюча вибірка, і за відхилом між бажаною і дійсною поведінкою мережі ітераційним методом найменших квадратів знаходяться оптимальні параметри вузлів четвертого шару. На другому етапі залишковий відхилення передається з виходу мережі на входи, і методом зворотного поширення помилки модифікуються параметри вузлів першого шару. При цьому знайдені на першому етапі коефіцієнти висновків правил не змінюються. Ітераційна процедура настроювання продовжується поки відхилення перевищує заздалегідь установлене значення.

2.5 Програмні засоби для подання й обробки інтелектуальних моделей

Операції з **асоціативними правилами** у пакеті Matlab дозволяє виконувати модуль *ARMADA*. Він дозволяє створювати асоціативні правила по заданим користувачем даним в рамках середовища Matlab.

Для запуску модулю *ARMADA* необхідно зробити папку, де знаходиться цей модулю, поточною, після чого в командному рядку середовища Matlab написати наступну команду:

```
armada
```

В результаті з'явиться головна інтерфейсна форма модулю, в якій необхідно обрати файл з вхідними даними та параметри для аналізу даних (рис. 2.13).

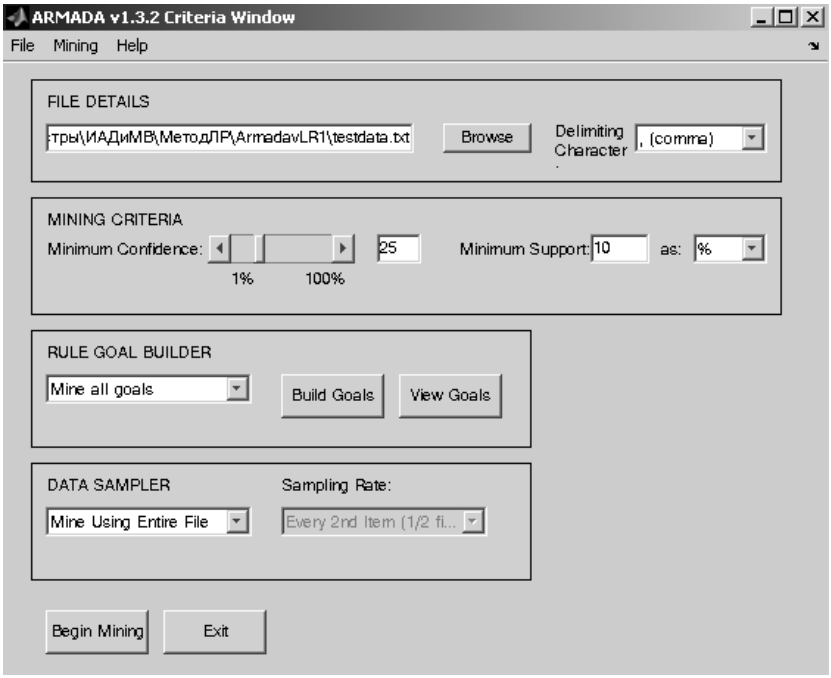


Рисунок 2.13 – Головна інтерфейсна форма модулю *ARMADA*

Для інтелектуального аналізу даних за допомогою асоціативних правил в полі FILE DETAILS необхідно ввести шлях до файлу з даними для аналізу. Файл з даними для аналізу можна також обрати у віконному режимі, натиснувши кнопку Browse. В полі зі списком Delimiting Character необхідно вказати символ, що відокремлює дані одне від одного.

Компоненти головної форми Minimum Confidence Minimum Support призначені для введення параметрів minsupport та minconfidence, відповідно.

Для виконання аналізу даних та отримання вихідної інформації у вигляді асоціативних правил необхідно натиснути кнопку Begin Mining, після чого відбудеться процес аналізу даних, та з'явиться форма з результатами роботи програми (рис. 2.14). Вихід з програми виконується за допомогою кнопки Exit.

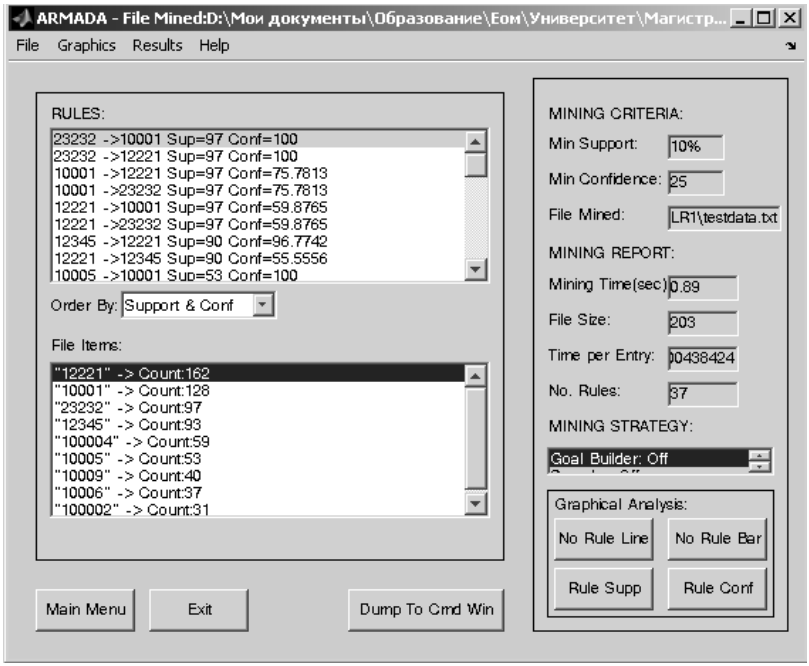


Рисунок 2.14 – Результати аналізу даних за допомогою модулю *ARMADA*

В області RULES наведені отримані асоціативні правила, в області File Items – елементи файлу для аналізу та кількість разів, які вони зустрічаються у файлі.

Компоненти в області MINING CRITERIA відображають параметри аналізу: значення параметрів minsupport та minconfidence, а також файл із даними для аналізу.

В області MINING REPORT відображається інформація про хід виконання аналізу:

– час, витрачений на виконання аналізу (Mining Time);

– розмір файлу (File Size);

– час, затрачений на один запис (Time per Entry);

– кількість отриманих асоціативних правил (No. Rules)

За допомогою головного меню отримані дані можна зберегти на диск (File –> Save) та отримати графіки, що відображають процес аналізу даних (Graphics).

Дерева рішаючих правил у пакеті Matlab реалізовані у модулі Statistics Toolbox. Основними функціями для роботи з деревами є: treefit, treeprune, treedisp, treetest, treeval (табл. 2.3).

Таблиця 2.3 – Функції пакету Matlab для роботи із деревами рішаючих правил

№	Формат виклику	Призначення функції
1	`t = treefit(X,y)`	створює дерево рішаючих правил t на основі заданих значень незалежних змінних (матриця X) та значень вихідного параметру (вектор y)
2	`t1 = treeprune(t,'level',n)`	створює дерево рішаючих правил t1 на основі заданого дерева t, скороченого до n-го рівня
	`t1 = treeprune(t,'nodes',nod)`	створює дерево рішаючих правил t1 на основі заданого дерева t, видаляючи при цьому вузли, вказані в змінній nod

Продовження табл. 2.3

№	Формат виклику	Призначення функції
3	`treedisp(t)`	відображує дерево t у графічному вигляді
4	`c = treetest(t,'test',X,y)`	виконує тестування дерева t за допомогою тестової вибірки(X, y)
5	`Ycalc = treeval(t,X)`	за допомогою дерева t для масиву незалежних змінних X розраховує значення вихідного параметру Ycalc

Нижче наведено приклад створення дерева рішаючих правил на основі масивів даних meas та species, що зберігаються в структурі fisheriris.

```
load fisheriris; % завантажити з файлу fisheriris.mat змінні meas та species для створення дерева рішаючих правил.
t = treefit(meas,species); % створити дерево на основі змінних meas та species.
treedisp(t,'names',{'SL' 'SW' 'PL' 'PW'}); % вивести на екран побудоване дерево t у графічному вигляді. При цьому для підпису значень незалежних змінних, на основі яких було побудоване дерево t, використовується параметр names.
[c,s,n,best] = treetest(t,'cross',meas, species); % протестувати дерево t.
tmin = treeprune(t,'level',best); % мінімізувати дерево t.
% розрахунок точності класифікації побудованого дерева
sfit = treeval(t,meas); % за допомогою дерева t отримати відповідні числові значення класів для екземплярів з масиву meas.
sfit = t.classname(sfit); % отримати відповідні назви класів.
mean(strcmp(sfit,species)) % розрахунок точності класифікації.
```

Для моделювання **нейронних мереж** за допомогою пакета Matlab необхідно встановити і використовувати бібліотеку Neural Network Toolbox. Розглянемо приклади побудови моделей і навчання нейромереж мовою пакета Matlab (табл. 2.4–2.5).

Таблиця 2.4 – Приклад моделювання і навчання персептрона

Команда Matlab	Опис команди
x = [0.1 0.5 0.2 0.4 0.3 0.9; 0.9 0.5 0.8 0.6 0.7 0.1; 0.3 0.0 0.6 0.1 0.2 0.9];	Задаємо значення ознак екземплярів навчальної вибірки: 6 екземплярів (стовпці), 3 ознаки (рядки)
y = [1 0 1 0 1 0];	Задаємо номери класів для 6 екземплярів навчальної вибірки
net=newff(*repmat*([0 1], 3, 1), [2,1], {'*logsig*', '*logsig*'}, '*trainlm*');	Створюємо нейронну мережу net і визначаємо її топологію: діапазон зміни значень ознак [0 1], кількість ознак – 3, кількість вихідних змінних – 1, на першому шарі – 2 нейрони, на другому шарі 1 – нейрон, нейрони 1 і 2 шарів мають сигмоїдні функції активації (*logsig*), для навчання мережі використовується метод Левенберга-Марквардта (*trainlm*)
net.trainparam. show=25;	Задаємо період відображення інформації про процес навчання на екрані в циклах (епохах) навчання
net.trainparam.lr= 0.01;	Задаємо крок навчання
net.trainparam. epochs=500;	Задаємо максимально припустиму кількість циклів навчання (епох)
net.trainparam. goal=0.01;	Задаємо максимально припустиме значення критерію навчання (помилки навчання)
ct=cputime;	Визначаємо і запам'ятовуємо поточне значення лічильника часу в змінній *ct*

Продовження табл. 2.4

Команда Matlab	Опис команди
net=train(*net*, *x*, *y*);	Навчаємо нейронну мережу *net* на основі навчальної вибірки, представленої набором значень ознак екземплярів *x* і набором значень відповідних їм номерів класів *y*
ct=cputime-ct	Визначаємо поточне значення лічильника часу, віднімаємо від нього значення змінної *ct* – визначаємо час навчання НМ, що заносимо в змінну *ct* і видаємо на екран (ознака друку на екран – відсутність символу “;” наприкінці оператора)
a=round(*sim*(*net*, *x*));	Обчислюємо по навченій мережі *net* номера класів для екземплярів, що характеризуються набором значень ознак *x*

Таблиця 2.5 – Приклад моделювання і навчання радіально-базисних нейронних мереж

Команда Matlab	Опис команди
x = [1 2 3];	Задаємо значення ознак екземплярів навчальної вибірки: 3 екземпляри (стовпці), 1 ознака (рядки)
y = [2.0 4.1 5.9];	Задаємо значення параметра, що прогнозується, для 3 екземплярів навчальної вибірки
net = *newrb*(*x*,*y*);	Створюємо і навчаємо радіально-базисну НМ
a = *sim*(*net*,*x*)	Обчислюємо по навченій мережі *net* значення параметра, що прогнозується, для екземплярів, що характеризуються набором значень ознак *x*

Модуль Fuzzy Logic Toolbox пакету Matlab поряд із засобами для побудови нечітких моделей містить засоби для синтезу **нейро-нечітких мереж**.

ANFIS-редактор дозволяє автоматично синтезувати з експериментальних даних нейро-нечіткі мережі. Завантаження ANFIS-редактора здійснюється командою `anfisedit`. ANFIS-редактор містить меню: File, View (аналогічні меню FIS-редактору) та Edit, а також області керування створенням нейро-нечіткої мережі (рис. 2.15).

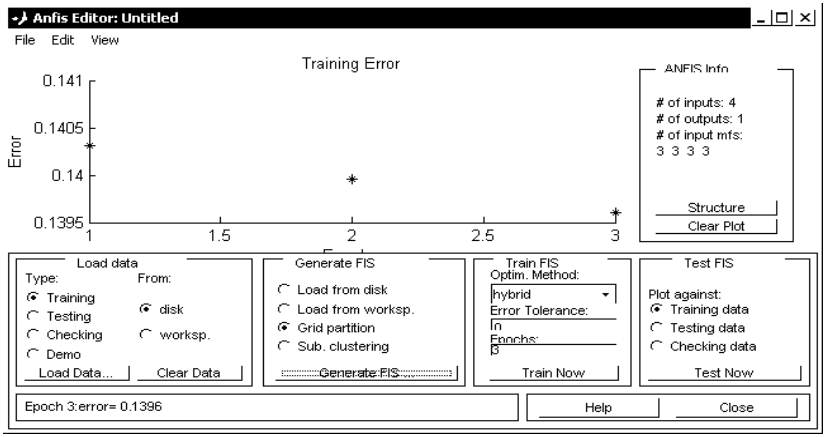


Рисунок 2.15 – ANFIS-редактор

Меню Edit. Команда Undo скасовує раніше зроблену дію. Команда FIS Properties відкриває FIS-редактор. Команда Membership Functions відкриває редактор функцій приналежності. Команда Rules відкриває редактор бази знань. Команда Anfis відкриває ANFIS-редактор.

Область візуалізації містить два типи інформації: при навчанні системи – крива навчання у вигляді графіка залежності помилки навчання від порядкового номера ітерації; при завантаженні даних і тестуванні системи – експериментальні дані і результати моделювання. Експериментальні дані і результати моделювання виводяться як множина точок у двовимірному просторі. При цьому по вісі абсцис відкладається порядковий номер рядка даних у вибірці (навчаючої, тестової або контрольної), а по осі ординат – значення вихідної змінної для даного рядка вибірки. Використовуються такі маркери: блакитна крапка (**.**) – тестова вибірка; блакитна окружність (o) – навчаюча вибірка; блакитний плюс (+) – контрольна вибірка; червона зірочка (*) – результати моделювання.

В області властивостей (ANFIS info) виводиться інформація про кількість вхідних і вихідних змінних, про кількість функцій приналежності для кожної вхідний змінний, а також про кількість рядків у вибірках. Натискання кнопки Structure відчиняє нове

графічне вікно, у якому система нечіткого логічного виведення представляє у виді нейро-нечіткої мережі. Натискання кнопки Clear Plot дозволяє очистити область візуалізації.

В області завантаження даних (Load data) розташовані: меню вибору типу даних Type, що містить альтернативи: Traning – навчаюча вибірка, Testing – тестова вибірка, Checking – контрольна вибірка, Demo – демонстраційний приклад, меню вибору джерела даних From, що містить альтернативи: disk – диск, worksp. – робоча область Matlab; кнопка завантаження даних Load Data, по натисканню якої з'являється діалогове вікно вибору файлу, якщо завантаження даних відбувається з диска, або вікно введення ідентифікатора вибірки, якщо завантаження даних походить з робочої області; кнопка очищення даних Clear Data.

В області генерування FIS (Generate FIS) розташовані меню вибору способу створення вихідної системи нечіткого логічного виведення. Меню містить альтернативи: Load from disk – завантаження FIS з диска, Load from worksp. – завантаження системи з робочої області Matlab, Grid partition – генерування системи за методом ґрат (без кластеризації), Sub. clustering – генерування системи за методом субкластеризації. В області також розташована кнопка Generate, по натисканню якої генерується вихідна система нечіткого логічного виведення.

При виборі Grid partition з'являється вікно введення параметрів методу ґрат, у якому потрібно вказати кількість термів для кожної вхідної змінної і тип функцій приналежності для вхідних і вихідної змінних.

При виборі Sub. clustering з'являється вікно введення параметрів методу субкластеризації: Range of influence – рівні впливу вхідних змінних, Squash factor – коефіцієнт пригнічення, Accept ratio – коефіцієнт, що встановлює у скільки разів потенціал даної точки повинний бути вище потенціалу центра першого кластера для того, щоб центром одного з кластерів була призначена розглянута точка, Reject ratio – коефіцієнт, що встановлює у скільки разів потенціал даної точки повинний бути нижче потенціалу центра першого кластера, щоб розглянута точка була виключена з можливих центрів кластерів.

В області навчання (Train FIS) розташовані: Optim. method – меню вибору методу оптимізації, Error tolerance – поле введення

необхідної точності навчання, Epochs – поле введення кількості ітерацій навчання та кнопка Train Now, натискання якої запускає режим навчання. Проміжні результати навчання виводяться в область візуалізації й у робочу область Matlab. У ANFIS-редакторі реалізовані два методи навчання: backprora – метод зворотного поширення помилки та hybrid – гібридний метод, що поєднує метод зворотного поширення помилки з методом найменших квадратів.

В області тестування (Test FIS) розташовані меню вибору вибірки і кнопка Test Now, по натисканню якої відбувається тестування нечіткої системи з виведення результатів в область візуалізації.

Поряд із функціями для побудови систем нечіткого виведення модуль Fuzzy Logic Toolbox пакету Matlab містить функції для створення й використання нейро-нечітких мереж: anfis – функція навчання для системи Сугено, функція genparam – генерує початкові параметри передумов для навчання ANFIS, функція evalfis – виконує обчислення для вибірки екземплярів за допомогою вказаної нейро-нечіткої мережі.

Вибірки даних та нейро-нечіткі мережі, з якими працюють описані функції повинні зберігатися в робочій області Matlab (в оперативній пам'яті). В стовпцях матриць, відповідних вибіркам, зберігаються значення входів (ознак) та виходів, в рядках – екземпляри вибірки.

Приклад використання функцій створення нейро-нечітких мереж модуля Fuzzy Logic Toolbox пакету Matlab.

Апроксимувати залежність: $y = \frac{\sin 2x}{e^{x/5}}$, $x \in [0, 10]$, $\Delta x = 0{,}1$.

```
x = (0:0.1:10)'; % задаємо масив x
y = sin(2.*x)./exp(x./5); % визначаємо значення елементів ма-
сиву y
trnData = [x y]; % створюємо масив навчаючої вибірки
numMFs = 5; % кількість функцій приналежності
mfType = 'gbellmf'; % тип функцій приналежності
epoch_n = 20;% кількість ітерацій
% створюємо FIS-структуру типу Сугено
in_fismat = genfis1(trnData,numMFs,mfType);
% навчаємо FIS-структуру
out_fismat = anfis(trnData,in_fismat,20);
% будуємо графіки:
```

```
% цільової залежності – синій колір, суцільна лінія
% апроксимуючої залежності – зелений колір, пунктир
plot(x,y,x,evalfis(x,out_fismat));
% підписуємо легенду до графіків
% Training Data – навчаюча вибірка – цільові дані
% ANFIS Output – вихід ANFIS – розрахункові дані
legend('Training Data','ANFIS Output');
```

? 2.6 Контрольні питання

Асоціативні правила

1. Що таке асоціативне правило? Для чого призначені асоціативні правила?
2. Дати визначення понять підтримки та достовірності правила.
3. Яке призначення алгоритмів пошуку асоціативних правил?
4. На які підзадачі розбивається задача знаходження асоціативних правил?
5. Які методи використовуються для знаходження асоціативних правил?
6. Яким чином обираються значення параметрів minsupport та minconfidence?
7. Що таке числові асоціативні правила?
8. Поясніть поняття «узагальнене асоціативне правило».
9. Що називається ієрархією елементів?
10. Які переваги дає введення додаткової інформації про групування елементів?
11. Поясніть, які проблеми можуть виникнути при безпосередньому застосуванні алгоритмів знаходження асоціативних правил.
12. В чому полягає сутність виявлення узагальнених асоціативних правил?
13. Яким чином визначають «цікаві» правила? В чому полягає актуальність такого процесу?

14. Дати визначення понять батьківського правила (предка) та найближчого батьківського правила.

15. Порівняйте поняття цікавого та частково цікавого правила.

16. Які проблеми усуває алгоритм обчислення узагальнених асоціативних правил?

17. З яких етапів складається процес обчислення узагальнених асоціативних правил?

18. Проаналізуйте базовий алгоритм пошуку множин, що зустрічаються часто.

19. Опишіть алгоритм генерації кандидатів.

20. Яким чином використовується хеш-дерево для підрахунку підтримки кандидатів? Як відбувається процес побудови такого дерева?

21. Виконайте порівняльний аналіз базового та покращеного алгоритмів пошуку множин, що зустрічаються часто.

22. За рахунок яких оптимізацій відбувається покращення базового алгоритму пошуку множин, що зустрічаються часто?

23. В чому полягає сутність масштабуємого алгоритму пошуку асоціативних правил Apriori?

24. Яким чином перетворюються дані для можливості використання алгоритму Apriori?

25. Яка властивість використовується в алгоритмі Apriori? Для чого вона використовується?

26. Наведіть послідовність виконання алгоритму Apriori.

27. Опишіть функцію генерації кандидатів в алгоритмі Apriori.

28. Як відбувається підрахунок підтримки для кожного кандидату в алгоритмі Apriori? Для чого в цій процедурі використовують хеш-дерево?

29. Як здійснити добування правил з набору, що часто зустрічається?

30. Проаналізуйте внутрішню структуру модулю Armada пакету Matlab: основні змінні, параметри, методи та функції, їх призначення та використання.

Дерева рішень

31. Що таке дерево рішаючих правил? Який спосіб подання правил в них використовується?

32. Дати означення основних понять, що відносяться до теорії дерев рішаючих правил: об'єкт, атрибут, мітка класу, вузол, лист, перевірка.

33. Навести основні класи задач, до яких можуть бути застосовані дерева рішаючих правил.

34. Яким чином відбувається побудова дерева рішаючих правил? Який метод використовується для цього?

35. В чому полягає процес навчання з учителем?

36. Порівняйте методи, що реалізують дерева рішаючих правил: CART та C4.5.

37. Поясніть принцип роботи «жадібних» алгоритмів.

38. Перелічіть основні аспекти, яким приділяється увага при побудові дерев рішаючих правил.

39. В чому полягає правило відбору ознаки для розбиття? Сформулюйте загальне правило для відбору атрибуту.

40. Виконайте порівняльний аналіз критеріїв оцінки якості розбиття множини на класи.

41. Що визначає правило зупину? Дайте порівняльну характеристику відомих критеріїв зупину побудови дерева рішаючих правил?

42. Для чого використовується правило відсіку?

43. Що розуміють під точністю та помилкою розпізнавання для дерева рішаючих правил?

44. Що необхідно зробити для добування правил з дерева рішаючих правил?

45. Які вимоги до структури та значень даних висуває метод C4.5?

46. Проаналізуйте алгоритм побудови дерева рішаючих правил за допомогою методу C4.5.

47. Яким чином визначається критерій вибору атрибуту в методі C4.5?

48. В якому випадку в процесі роботи методу C4.5 вузол помічається як лист? Що обирається в якості розв'язку листа?

49. Коли ентропія досягає свого максимуму (мінімуму) при використанні методу C4.5?

50. В яких випадках необхідно обрати поріг для порівняння значень атрибуту?

51. В чому полягає класифікація нових об'єктів? Звідки починається обхід дерева?

52. Порівняйте покращений критерій розбиття з класичним.

53. Яке евристичне правило використовується для зменшення ймовірності створення вузлів та листя, які містять незначну кількість об'єктів?

54. Проаналізуйте процедуру роботи з пропущеними даними.

55. Яким чином відбувається класифікація нових об'єктів у випадку відсутності значення певного атрибуту об'єкту, що класифікується?

56. Які переваги використання дерев рішаючих правил?

57. Який напрямок побудови дерева рішаючих правил використовується при використанні методу ID3?

58. Наведіть послідовність побудови дерева рішаючих правил за допомогою методу ID3.

59. В чому полягає вибір властивості на основі теорії інформації?

60. Проаналізуйте основні функції пакету Matlab для роботи з деревами рішаючих правил: внутрішня структура, параметри, основні змінні, методи, їх призначення та використання.

Нейронні мережі

61. Поняття: нейрон, нейромережа, нейрокомп'ютер, нейроінформатика.

62. Класифікація та види моделей нейромереж.

63. Властивості штучних нейромереж.

64. Загальне уявлення про навчання нейромереж.

65. Лінійна роздільність і лінійна нерозділеність класів.

66. Нейронні мережі у пакеті Matlab.

67. Моделі нейроелементів у пакеті Matlab.

68. Математичні моделі нейроелементів.

69. Поняття: формальний нейрон, синапс, вага (ваговий коефіцієнт), поріг, дискримінантна (вагова) функція, функція актива-

ції, бажаний і реальний вихід нейромережі, навчання нейромережі, класифікація, апроксимація, оцінювання, помилка навчання / класифікації, час навчання / класифікації, цільова функція навчання.

70. Метод найменших квадратів як основа алгоритму Уідроу-Хоффа. Чи завжди збігаються алгоритми навчання одношарового персептрона?

71. Можливості і властивості одношарових персептронів. Які задачі можна вирішувати на основі одношарових персептронів, а які не можна? Чи доцільно застосовувати одношаровий персептрон для класифікації складно (нелінійно) роздільних образів? Обґрунтуйте і доведіть відповідь. Приведіть приклади.

72. Чи впливає величина кроку навчання на час навчання одношарового персептрона, багатошарової мережі? Відповідь обґрунтуйте теоретично та доведіть експериментально.

73. Які функції активації нейронів найчастіше використовують і чому? Чи впливає вид функції активації нейрона на тривалість навчання і роботи персептрона, величину помилки навчання і класифікації (оцінювання) і, якщо впливає, то яким чином?

74. Чи впливає кількість використаних ознак на швидкість навчання персептрона? Відповідь обґрунтуйте теоретично та доведіть експериментально.

75. У чому подібність і відмінність біологічних і формальних нейронів?

76. Чи можна навчити дискретний одношаровий персептрон обчислювати значення функцій: $y = (x1 \text{ and } x2)$, $y = (x1 \text{ xor } x2)$, $y = \text{not } ((\text{not } x1) \text{ and } x2)$, а дійсний одношаровий персептрон – обчислювати значення функцій: $y = 3x1 - 0{,}05x2$, $y = \sin(x1) + 0{,}3x2$, $y = 0{,}5x1 + 2x2 - 2{,}5(x1 - x2) + 5{,}5$? Відповіді обґрунтуйте і, якщо можливо, поясніть рисунками.

77. Багатошаровий персептрон: модель і принципи побудови архітектури.

78. Які задачі можна вирішувати на основі багатошарових персептронів, а які не можна? Обґрунтуйте і доведіть відповідь. Приведіть приклади.

79. Чи може дійсний багатошаровий персептрон моделювати функцію $y = x1x2 + 0{,}5x1 + x2$, якщо число його шарів дорівнює: а) 1, б) 2, в) 3? Відповіді обґрунтуйте і, якщо можливо, поясніть рисунками.

80. Чи можливе використання неградієнтних методів багатовимірної безумовної оптимізації для настроювання ваг багатошарових нейромереж, і, якщо можливе, то наскільки це доцільно робити?

81. Чи можна навчити дискретний багатошаровий персептрон обчислювати значення функцій: 1) $y = x1$ and $x2$, 2) $y = x1$ xor $x2$, 3) $y =$ not ((not $x1$) and $x2$), а дійсний багатошаровий персептрон – обчислювати значення функцій: 1) $y = 3x1 - 0{,}05x2$, 2) $y = \sin(x1) + 0{,}3x2$, 3) $y = 0{,}5x1 + 2x2 - 2{,}5(x1 - x2) + 5{,}5$, 4) $y = x1 / (x2\sin(\pi))$? Відповіді обґрунтуйте і, якщо можливо, поясніть рисунками.

82. У чому подібність і відмінність радіально-базисних нейромереж і багатошарових нейромереж?

83. Які задачі можна вирішувати на основі радіально-базисних нейромереж, а які не можна? Обґрунтуйте і доведіть відповідь. Наведіть приклади.

84. Радіально-базисні нейромережі у пакеті Matlab.

85. Чи доцільно застосовувати радіально-базисні нейромережі для класифікації складно (нелінійно) роздільних образів?

86.Чи можна навчити радіально-базисну нейромережу обчислювати значення функцій: 1) $y = x1$ and $x2$, 2) $y = x1$ xor $x2$, 3) $y =$ not ((not $x1$) and $x2$), 4) $y = 3x1 - 0{,}05x2$, 5) $y = \sin(x1) + 0{,}3x2$, 6) $y = 0{,}5x1 + 2x2 - 2{,}5(x1 - x2) + 5{,}5$, 7) $y = x1 / (x2\ \sin(\pi))$? Відповіді обґрунтуйте і, якщо можливо, поясніть рисунками.

87. Переваги і недоліки радіально-базисних мереж.

88. Що таке генеральна сукупність, вибірка, екземпляр, ознака? Вимоги до навчальних вибірок даних. Що таке репрезентативна вибірка даних? Чи повинна навчальна вибірка бути репрезентативною? Чи повинна тестова вибірка бути репрезентативною?

89. Чи впливає обсяг навчальної вибірки на швидкість навчання нейромереж? Чи впливає репрезентативність навчальної вибірки на точність класифікації екземплярів тестової вибірки? Чи впливає репрезентативність тестової вибірки на точність класифікації екземплярів тестової вибірки?

90.Чи впливає репрезентативність тестової вибірки на точність навчання персептрона по навчальній вибірці? Чи залежить якість навчання нейромереж від якості та обсягу навчальної вибірки?

Нейро-нечіткі моделі

91. Порівняйте властивості нечітких систем та нейронних мереж.

92. Що таке нейро-нечітка мережа?

93. Які властивості мають нейро-нечіткі мережі?

94. Які існують типи нейро-нечітких мереж?

95. Як формують базу знань нейро-нечіткої мережі?

96. З яких елементів складаються нейро-нечіткі мережі?

97. У чому полягають особливості паралельних нейро-нечітких мереж?

98. Що таке нейро-нечітка асоціативна пам'ять Коско?

99. Як відбувається виділення нечітких правил за допомогою карт, що самоорганізуються?

100. Опишіть особливості систем, що здатні навчати нечіткі множини.

101. У чому полягають особливості конкурентних нейро-нечітких систем?

102. У чому полягають особливості інтегрованих нейро-нечітких систем?

103. Опишіть архітектуру та методи навчання нейро-нечітких мереж: апроксиматора Мамдані, мережі Такагі-Сугено-Канга, мережі ANFIS.

104. У чому полягають основні принципи використання методу зворотного поширення помилки для навчання нейро-нечітких мереж?

105. Які існують основні етапи гібридного алгоритму навчання мереж Такагі-Сугено-Канга?

106. Опишіть функції пакету Matlab для створення нейро-нечітких мереж.

107. Опишіть редактор anfisedit.

108. Який алгоритм кластер-аналізу призводить до отримання нейро-нечіткої мережі меншої складності?

109. Як впливає задана кількість циклів навчання на точність навчання? Як впливає задана точність навчання на тривалість навчання?

110. Які вимоги мають пред'являтися до навчаючої вибірки та як це вплине на процес навчання нейро-нечітких мереж?

2.7 Практичні завдання

 *Завдання 1*. Розробити за допомогою середовища Matlab програмне забезпечення для видобування асоціативних правил з великих масивів даних або вивчити рекомендоване програмне забезпечення (пакет Armada модулю Matlab) та здійснити обробку набору даних з метою виділення асоціативних правил.

 *Завдання 2*. Розробити програму для побудови дерев рішаючих правил. Виконати обробку набору тестових даних з метою синтезу дерева рішень. Проаналізувати побудоване дерево рішень.

 *Завдання 3*. Використовуючи пакет Matlab, синтезувати одношарову та багатошарову нейронні мережі. Параметри багатошарового персептрону (кількість шарів, кількості нейронів у шарах, вид функції активації кожного нейрону) встановити відповідно до номеру варіанту (табл. 2.6). Навчаючу вибірку для побудови нейромоделей сформувати самостійно.

Таблиця 2.6 – Параметри нейромереж для варіантів

Номер варіанту	Багатошаровий персептрон		Назва функції активації
	Кількість шарів	Кількості нейронів у шарах	
1	2	3–1	логістична сигмоїдна
2	3	3–3–1	тангенційна сигмоїдна
3	4	3–3–3–1	логістична сигмоїдна
4	2	2–1	тангенційна сигмоїдна
5	3	2–2–1	логістична сигмоїдна
6	4	2–2–2–1	тангенційна сигмоїдна
7	2	5–1	логістична сигмоїдна
8	3	5–5–1	тангенційна сигмоїдна
9	4	5–5–5–1	логістична сигмоїдна
10	2	4–1	тангенційна сигмоїдна
11	3	4–4–1	логістична сигмоїдна

Продовження табл. 2.6

Номер варіанту	Багатошаровий персептрон		Назва функції активації
	Кількість шарів	Кількості нейронів у шарах	
12	4	4–4–4–1	тангенційна сигмоїдна
13	2	6–1	логістична сигмоїдна
14	3	6–3–1	тангенційна сигмоїдна
15	4	6–4–3–1	логістична сигмоїдна
16	2	7–1	тангенційна сигмоїдна
17	3	7–7–1	логістична сигмоїдна
18	4	7–3–4–1	тангенційна сигмоїдна
19	2	8–1	логістична сигмоїдна
20	3	8–8–1	тангенційна сигмоїдна
21	4	8–7–6–1	логістична сигмоїдна
22	2	2–1	тангенційна сигмоїдна
23	3	2–3–1	логістична сигмоїдна
24	4	2–4–4–1	тангенційна сигмоїдна
25	2	3–1	логістична сигмоїдна
26	3	6–2–1	тангенційна сигмоїдна
27	4	7–6–3–1	логістична сигмоїдна
28	2	4–1	тангенційна сигмоїдна
29	3	9–2–1	логістична сигмоїдна
30	4	9–3–2–1	тангенційна сигмоїдна

 *Завдання 4*. Змінюючи значення кроку навчання, для одношарового та багатошарового персептронів з заданою функцією активації, що відповідає номеру варіанту студента (табл. 2.6) дослідити, як впливає величина кроку навчання на час навчання. Побудувати графіки залежності часу навчання персептронів від величини кроку навчання.

 *Завдання 5*. На основі згенерованої у завданні № 3 вибірки даних побудувати радіально-базисну мережу.

 *Завдання 6*. Порівняти результати навчання одношарового, багатошарового персептронів та радіально-базисної мережі (завдання 3–5).

Результати порівняння нейромоделей занести в таблицю, стовпці якої повинні мати назви: назва архітектури нейромережі, кількість шарів, кількість нейронів у шарах, функції активації нейронів у шарах, метод навчання, час навчання, час класифікації (оцінювання) навченої мережі для навчальної вибірки, помилка класифікації (оцінювання) для навчальної вибірки, час класифікації (оцінювання) навченої мережі для тестової вибірки, помилка класифікації (оцінювання) для тестової вибірки.

 *Завдання 7*. Згідно з номером індивідуального варіанта студента за журналом згенерувати навчаючу та тестову вибірки даних.

Нехай V – номер студента за журналом, а rand – функція, що генерує псевдовипадкові числа у діапазоні [0; 1]. Визначимо кількість екземплярів у навчаючій вибірці $S_{н.}$, кількість екземплярів у тестовій вибірці $S_{т.}$, кількість вхідних змінних (ознак) вибірок N, значення ознак вибірок: навчаючої – $x_{н.}$ та тестової – $x_{т.}$, а також значення цільових ознак для вибірок: навчаючої – $y_{н.}$ та тестової – $y_{т.}$:

$$N = \begin{cases} V, \text{якщо } V < 15, \\ V - 14, \text{якщо } V > 15; \end{cases}$$

$$S_{н.} = \begin{cases} 10\,\text{V}, \text{якщо } V < 5, \\ 5V, \text{якщо } 5 \le V < 10, \\ 2V, \text{якщо } 10 \le V < 20, \\ V, \text{інакше}; \end{cases} \quad S_{т.} = \begin{cases} 8V, \text{якщо } V < 5, \\ 6V, \text{якщо } 5 \le V < 10, \\ 3V, \text{якщо } 10 \le V < 20, \\ 2V, \text{інакше}; \end{cases}$$

$x_{н.} = \{x_j^s\}$, $y_{н.} = \{y^s\}$, $s = 1, 2,...,S_{н.}$; $x_{т.} = \{x_j^s\}$, $y_{т.} = \{y^s\}$, $s = 1, 2,..., S_{т.}$;

$$x_j^s = \frac{V}{s \cdot j} \cdot \text{rand}, j = 1,2,...,N,$$

$$y^s = \begin{cases} 0{,}1(x^s{}_1 + x^s{}_2), x^s{}_1 < V, \\ 0{,}3(x^s{}_1 - x^s{}_2), x^s{}_1 > V, \\ 0{,}5x^s{}_1, x^s{}_1 = V. \end{cases}$$

Для згенерованих вибірок за допомогою редактору anfisedit побудувати нейро-нечітку модель. Спробувати використати різні методи кластеризації, різні кількості функцій приналежності для входів, різні кількості циклів навчання та різні методи навчання. Протестувати побудовану модель. Проаналізувати отримані результати та відповісти на питання: який метод кластер-аналізу призводить до отримання мережі меншої складності; як впливає задана кількість циклів навчання на точність навчання; як вливає задана точність навчання на тривалість навчання; які вимоги мають пред'являтися до навчаючої вибірки та як це вплине на процес навчання.

 *Завдання 8*. Виконати завдання 7 у режимі командного вікна без застосування редактору anfisedit, використовуючи функції модуля Fuzzy Logic Toolbox.

 *Завдання 9*. На алгоритмічній мові програмування високого рівня (наприклад, Сі, Паскалі, мові пакету Matlab) створити програму, що реалізує одну з моделей нейро-нечітких мереж.

2.8 Тестові завдання

Т *Завдання 1*. Що визначає поняття листа в теорії дерев рішень?

А Змінна, ознака, що визначає клас об'єкта.

Б Кінцевий вузол дерева, вузол рішення.

В Внутрішній вузол дерева, вузол перевірки.

Г Внутрішній вузол дерева, умова переходу.

Д Внутрішній вузол дерева, атрибут об'єкту.

Е Вірними є відповіді В та Г.

Т *Завдання 2*. Який процес побудови дерев рішень називається навчанням з учителем?

А Процес побудови дерев, коли класи, до яких віднесені об'єкти, заздалегідь невідомі.

Б Процес побудови дерев, коли хоча б один з об'єктів віднесений до заздалегідь відомого класу.

В Процес побудови дерев, коли всі об'єкти віднесені до заздалегідь відомих классів.

Г Процес побудови дерев, коли навчаюча множина не містить жодного приклада.

Д Процес побудови дерев, коли навчаюча множина містить приклади, віднесені тільки до одного класу.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 3.* Як називається метод побудови дерева рішень, де кількість нащадків у вузла є необмеженою, та який не вміє працювати з неперервними цільовими змінними, тому вирішує тільки задачі класифікації?

А NewId.

Б CART.

В ITrule.

Г C4.5.

Д ID3.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 4.* Який критерій для вибору найбільш придатного атрибута використовується в методі C4.5?

А Критерій, заснований на теорії чітких множин.

Б Критерій, заснований на імовірнісному підході.

В Статистичний критерій.

Г Теоретико-інформаційний критерій.

Д Коефіцієнт парної кореляції.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 5.* Який критерії для вибору найбільш придатного атрибута використовується в методі CART?

А Критерій, заснований на теорії чітких множин.

Б Критерій, заснований на імовірнісному підході.

В Статистичний критерій.

Г Теоретико-інформаційний критерій.
Д Коефіцієнт парної кореляції.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 6.* Яка з вимог не відноситься до вимог, що висуває метод С4.5 до структури і значені даних?
А Дані, необхідні для роботи методу, повинні бути подані у вигляді плоскої таблиці.
Б Кожний атрибут повинен мати дискретне або числове значення.
В Самі атрибути не повинні мінятися від приклада до приклада.
Г Кількість атрибутів повинна бути фіксованою для всіх прикладів.
Д Кожен приклад повинний бути асоційований з конкретним класом.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 7.* Яким чином визначається стандартна помилка, що є оцінкою реальної помилки, при виборі фінального дерева в методі CART, якщо n_{test} – число прикладів у тестовій вибірці?
А $SE(R^{ts}) = (n_{test} / R^{ts}(1 - R^{ts}))^{0,5}$.
Б $SE(R^{ts}) = (R^{ts}(1 - R^{ts}) / n_{test})^{0,5}$.
В $SE(R^{ts}) = (n_{test} / R^{ts}(1 - R^{ts}))^{2}$.
Г $SE(R^{ts}) = (R^{ts}(1 - R^{ts}))^{2}$
Д $SE(R^{ts}) = (R^{ts}(1 - n_{test}))^{2}$
Е Серед відповідей А–Д вірної немає.

Т *Завдання 8.* Який шлях вибору фінального дерева використовується в методі CART, коли набір даних для навчання малий або кожний запис у ньому по своєму унікальний так, що ми не можемо виділити вибірку для навчання і вибірку для тестування?
А Простий вибір фінального дерева.
Б Правило 1–*SE*.

В Перехресна перевірка.
Г Адаптований метод обходу та відсікання піддерев.
Д Вірними є відповіді В та Г.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 9.* Що являють собою асоціативні правила?

А Асоціативні правила не дозволяють виявляти залежність між пов'язаними подіями і є одним з інструментів видобування знань з масивів даних.
Б Асоціативні правила дозволяють виявляти залежність між пов'язаними подіями і є одним з інструментів побудови дерев рішень.
В Асоціативні правила дозволяють виявляти закономірності між пов'язаними подіями і є одним з інструментів видобування знань з масивів даних.
Г Асоціативні правила дозволяють виявляти закономірності між подіями і є одним з методів програмування.
Д Асоціативні правила є різновидом правил розбиття при побудові дерев рішень.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 10.* До чого призводить велике значення підтримки при знаходженні асоціативних правил?

А Методи будуть знаходити правила, добре відомі аналітикам.
Б Генерації величезної кількості правил.
В Методи будуть знаходити дуже очевидні правила.
Г Генерації статистично необґрунтованих правил.
Д Вірними є відповіді Б та В.
Е Вірними є відповіді А та В.

Т *Завдання 11.* До чого, при відсіканні «нецікавих» правил призводить пряме застосування методів пошуку асоціативних правил?

А Елементи на верхніх рівнях ієрархії прагнуть до значно більших значень підтримки в порівнянні з елементами на нижніх рівнях.

Б З додаванням у транзакції груп збільшується кількість атрибутів і, відповідно, розмірність вхідного простору.

В Поява надлишкових правил, що суперечать визначенню узагальненого асоціативного правила.

Г Знищуються асоціативні правила між різними групами.

Д Вірними є відповіді А, Б, В.

Е Серед відповідей А–Д вірної немає

Т *Завдання 12.* У якому випадку правило $X \to Y$ називається *R*-цікавим щодо правила-предка?

А Якщо підтримка правила $X \to Y$ у *R* разів більше очікуваної підтримки правила $X \to Y$ щодо предка.

Б Якщо підтримка правила $X \to Y$ у *R* разів менше очікуваної підтримки правила $X \to Y$ щодо предка.

В Якщо вірогідність правила $X \to Y$ у *R* разів більше очікуваної вірогідності правила $X \to Y$ щодо правила-предка.

Г Якщо вірогідність правила $X \to Y$ у *R* разів менше очікуваної вірогідності правила $X \to Y$ щодо правила-предка.

Д Вірними є відповіді А та В.

Е Вірними є відповіді Б та Г.

Т *Завдання 13.* Пошук яких множин виконується на першому етапі методу обчислення узагальнених асоціативних правил?

А Множин елементів, що рідко зустрічаються, підтримка яких більше ніж заданий поріг підтримки.

Б Множин елементів, що часто зустрічаються, підтримка яких більше ніж заданий поріг підтримки.

В Множин елементів, що рідко зустрічаються, підтримка яких менше ніж заданий поріг підтримки.

Г Множин елементів, що часто зустрічаються, підтримка яких менше ніж заданий поріг підтримки.

Д Множин елементів, що рідко зустрічаються, підтримка яких більше ніж середнє значення порогу підтримки.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 14.* Як повинні бути подані дані, щоб можна було застосувати метод Apriori?

А Повинні бути пронормовані.
Б Повинні бути переведені до бінарного виду.
В Представлені у вигляді таблиці.
Г Повинні бути упорядковані за абеткою.
Д Вірними є відповіді Б та Г.
Е Вірними є відповіді Б, В, Г.

Т *Завдання 15.* Що є базовим елементом нейронної мережі?

А Формальний нейрон, що має кілька входів і кілька виходів.
Б Формальний нейрон, що має один вхід і один вихід.
В Формальний нейрон, що має кілька входів і один вихід.
Г Формальний нейрон, що має один вхід і кілька виходів.
Д Вектор вхідних аргументів.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 16.* Математична модель штучного нейрона описується співвідношенням: $y = \psi\ (\varphi(w, x))$. Що у даному співвідношенні є ψ*?*

А Дискримінантна функція.
Б Вектор, що містить значення вагових коефіцієнтів і значення зсуву.
В Значення виходу нейрона.
Г Функція активації.
Д Вектор вхідних аргументів.
Е Серед відповідей А–Д вірної немає.

Т *Завдання 17.* Виберіть вірну формулу дискримінантної функції «зважена сума»:

А $\varphi(w,x) = \sum_{j=1}^{N} w_j x_j + w_0$.

Б $\varphi(w,x)=\sum_{j=1}^{N} w_j x_j - w_0$.

В $\varphi(w,x)=\sum_{j=1}^{N} w_j x_j - w_0 x_0$.

Г $\varphi(w,x)=\sum_{j=1}^{N} w_j x_j + w_0 x_0$.

Д $\varphi(w,x)=w_0 \sum_{j=1}^{N} \left(w_j - x_j\right)^2$.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 18.* Які нейромережі виділяють за типом функції активації нейронів?

А Мережі без циклів, мережі з циклами.

Б Гомогенні та гетерогенні.

В Безперервні, дискретні та дискретно-безперервні мережі.

Г Вірними є відповіді А–В.

Д Серед відповідей А–В вірної немає.

Е Вірними є відповіді Б та Г.

Т *Завдання 19.* Як називається принцип формування бази знань нейро-нечітких систем, коли формується початкова база продукційних правил, що потім послідовно поповнюється нечіткими правилами?

Б Оптимізація кількості продукційних правил.

А Копіювання навчаючої вибірки в базу знань.

В Нарощування (конструювання) правил.

Г Скорочення (редукція) правил.

Д Нормування правил.

Е Серед відповідей А–Д вірної немає.

Т *Завдання 20.* За допомогою чого поєднуються вхідні сигнали та синаптичні ваги у елементі нечіткої нейронної мережі «Нечіткий нейрон «ТА»»?

А Функції активації.

Б *t*-норми.

В *t*-конорми.

Г Нечіткого правила.

Д Нечіткого терму.

Е Серед відповідей А–Д вірної немає.

3 СИНТЕЗ ІНТЕЛЕКТУАЛЬНИХ МОДЕЛЕЙ

Для побудови інтелектуальних моделей, як правило, застосовуються оптимізаційні методи, сутність яких полягає в пошуку екстремуму цільового критерію, в якості якого звичайно використовується середньоквадратична похибка.

3.1 Градієнтні методи

Нехай задано вигляд (правило обчислення) деякої функції $y = f(w, x)$, де w та x – складені вектори-стовпці змінних і відомі реалізації функції y при відомих значеннях x. Потрібно оцінити невідомі параметри w, таким чином, щоб значення функції y при цих параметрах незначно відрізнялися від відомих реалізацій y^* при однакових заданих значеннях x, тобто необхідно мінімізувати деяку цільову функцію $F(w)$, де w – складений вектор-стовпець змінних, що керуються, за кінцеву кількість ітерацій.

Представимо функцію $F(w)$ у вигляді інтегрального критерію якості

$$F(w) = \frac{1}{k}\sum_{m=1}^{k} Q(\varepsilon(w,m)),$$

де k – номер поточної ітерації, m=1, 2,..., k – номера попередніх ітерацій, а $Q(\varepsilon(w,k))$ – деякий миттєвий критерій якості, що залежить від вектора помилки $Q(\varepsilon(w,m))$:

$$\varepsilon(w,m) = y(m) - y^*(m)$$

і який має вигляд квадратичної форми:

$$Q(\varepsilon(w,m)) = \varepsilon^T(w,m) R \varepsilon(w,m),$$

де R – позитивно визначена матриця.

Градієнтні методи мінімізації інтегрального критерію якості засновані на використанні градієнта цільової функції $F(w)$. Ці методи носять ітеративний характер, тому що компоненти градієнта виявляються нелінійними функціями.

Усі далі розглянуті методи засновані на ітераційній процедурі, що реалізована відповідно до формули:

$$w_{k+1} = w_k + \alpha_k s(w_k),$$

де w_k, w_{k+1} – поточне і нове наближення значень керувальних змінних до оптимального рішення, відповідно, α_k – крок збіжності, $s(w_k)$ –напрямок пошуку в N-вимірному просторі змінних, що керуються. Спосіб визначення $s(w_k)$ і α_k на кожній ітерації залежить від особливостей конкретного методу.

3.1.1 Метод Коші

Метод Коші (метод найшвидшого спуску, у навчанні нейронних мереж відомий, як алгоритм зворотного поширення помилки першого порядку, Backpropagation) полягає в реалізації правила:

$$w_k = w_{k-1} - \gamma \nabla_w Q\left(\varepsilon\left(w_{k-1}, k\right)\right) = \mathrm{w}_{k-1} - \gamma \frac{\partial Q\left(\varepsilon\left(w_{k-1}, k\right)\right)}{\partial w}.$$

Позначивши поточний градієнт $g = \dfrac{\partial Q}{\partial w}$, одержимо:

$$w_{k+1} = w_k - \gamma_k g_k,$$

де γ_k – швидкість навчання.

Величина γ або покладається постійною, і тоді звичайно послідовність w_k сходиться в околицю оптимального значення w, або вона є спадаючою функцією часу так, як це робиться в стохастичних алгоритмах оптимізації й адаптації.

Дана процедура може виконуватися доти, доки значення керованих змінних не стабілізуються, або доки помилка не зменшиться до прийнятного значення.

Слід зазначити, що дану процедуру характеризує повільна швидкість збіжності і можливість влучення в локальні мінімуми функціонала.

3.1.2 Метод Ньютона

Метод Ньютона або алгоритм зворотного поширення помилки другого порядку відносно до задачі мінімізації цільової функції $F(w)$ має вигляд:

$$w_k = w_{k-1} - \left(\sum_{m=1}^{k} \frac{\partial}{\partial w} \left(\frac{\partial Q(m)}{\partial w} \right)^{\mathrm{T}} \right)^{-1} \gamma \nabla_w Q\left(\varepsilon\left(w_{k-1}, k\right)\right).$$

Обчислення градієнта $\nabla_w Q(\varepsilon(w_{k-1},k))$ може бути виконано за допомогою методу Коші.

У результаті задача зводиться до обчислення матриці других похідних $\sum_{m=1}^{k}\frac{\partial}{\partial w}\left(\frac{\partial Q(m)}{\partial w}\right)^T$ мінімізуючого функціонала.

Метод Ньютона за певних умов має істотно більшу швидкість збіжності, ніж градієнтний метод.

3.1.3 Алгоритми спряжених градієнтів

Алгоритми спряжених градієнтів представляють собою підклас методів, що збігаються квадратично. Алгоритми спряжених градієнтів використовують метод Коші для обчислення похідних цільової функції щодо керувальних змінних.

Всі алгоритми спряжених градієнтів на першій ітерації починають роботу з пошуку у найкрутішому напрямку спуску – протилежному градієнту цільової функції:

$$p_0 = -g_0.$$

Це напрямок, у якому цільова функція зменшується найбільш швидко. Однак зазначимо, що це не обов'язково забезпечує найшвидшу збіжність.

Після цього виконується лінійний пошук, щоб визначити оптимальну відстань для руху у поточному напрямку пошуку:

$$w_{k+1} = w_k + \alpha_k p_k.$$

Наступний напрямок пошуку визначається так, щоб він був спряженим з попередніми напрямками. Пошук відбувається за спряженим напрямком градієнта, щоб визначити розмір кроку, що мінімізує цільову функцію. Загальна процедура для визначення нового напрямку пошуку поєднує новий найкрутіший напрямок спуску з попереднім напрямком пошуку:

$$p_{k+1} = -g_k + \beta_k p_{k-1}.$$

У більшості алгоритмів спряжених градієнтів розмір кроку корегується при кожній ітерації, на відміну від інших алгоритмів, де швидкість навчання використовується для визначення розміру кроку.

Різні версії алгоритмів спряжених градієнтів відрізняються способом, за яким обчислюється константа β.

Для **алгоритму Флетчера-Рівса** правило обчислення константи β має вигляд:

$$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}},$$

де β – відношення квадрата норми поточного градієнта до квадрата норми попереднього градієнта.

Для **алгоритму Полака-Ріб'єра** правило обчислення константи β має вигляд:

$$\beta_k = \frac{\Delta g_{k-1}^T g_k}{g_{k-1}^T g_{k-1}},$$

де β – внутрішній добуток попередньої зміни в градієнті і поточного градієнта, розділений на квадрат норми попереднього градієнта.

3.1.4 Партан-метод

Партан-метод призначений для мінімізаціх квадратичної цільової функції $F(w) = -\frac{1}{2}(w, Aw) + bw$, де A – матриця, що задає позитивно визначену квадратичну форму (w, Aw), b – деякий вектор. Партан-методу складається з таких етапів.

Етап 1. Ініціалізація.

У довільній точці w_0 знаходиться градієнт g_1 функції $F(w)$. На прямій $w = w_0 + \lambda g_1$ знаходиться точка w_1, що доставляє максимум функції $F(w)$.

Етап 2. Основна ітерація.

Нехай вже знайдені точки w_0, w_1, ...,w_{k-1}, w_k. Тоді у точці w_k знаходиться градієнт g_{k+1}, після чого на прямій $w = w_k + \lambda\ g_{k+1}$ визначається точка w'_{k+1}, у якій функція $F(w)$ досягає умовного екстремуму за формулою:

$$w'_{k+1} = \lambda'_k g_{k+1} + w_k, \text{ де } \lambda'_k = \frac{g_{k+1}^2}{\left(g_{k+1}, \mathrm{A} g_{k+1}\right)}.$$

На прямій $w = w_{k-1} - \lambda\ (w'_{k+1} - w_{k-1})$ знаходиться точка w_{k+1}, що доставляє умовний екстремум $F(w)$, за формулою:

$$w'_{k+1} = w_{k-1} + \lambda''_k \left(w'_{k+1} - w_{k-1}\right),$$

де $\lambda''_k = \dfrac{\left(\left(w'_{k+1} - w_{k-1}\right), g_k\right) g^2_{k+1}}{\left(\left(w'_{k+1} - w_{k-1}\right), \mathrm{A}\left(w'_{k+1} - w_{k-1}\right)\right)}$.

Етап 3. Зупинення. Процес припиняється, коли на черговому кроці виявиться, що $g_{m+1} = 0$. Тоді w_m – точка оптимуму функції $F(w)$.

3.1.5 Метод Заутендайка

Метод Заутендайка (метод проекцій) реалізується у відповідності з наступними кроками.

Крок 1. Почати в w_0 і покласти проектуючу матрицю $P_0=I$.

Крок 2. Для k-го кроку проектуюча матриця обчислюється таким чином:

$$P_k = P_{k-1} - P_{k-1}\Delta g_k \left[\Delta g_k{}^T P_{k-1} \Delta g_k\right]^{-1} \Delta g_k P_{k-1}.$$

Крок 3. Якщо $P_k g_k \neq 0$, покласти $s_k = - P_k g_k$ і мінімізувати функцію $F(w)$ у напрямку s_k; точка мінімуму w_{k+1}. Повторити крок 2. Після того, як пройдено n напрямків пошуку, де n – кількість керувальних змінних, почати знову з кроку 1 при $x_0 = x_n$.

Крок 4. Якщо $P_k g_k = 0$ і $g_k = 0$, закінчити пошук.

Крок 5. Якщо $P_k g_k = 0$ і $g_k \neq 0$, знову почати з кроку 1, поклавши $x_0 = x_k$.

3.1.6 Багатопараметричний пошук

В **методі Міля-Кентрелла** на етапі ініціалізації задається значення w_0, параметр Δw_{k-1} установлюється рівним нулю: $\Delta w_{k-1} = 0$.

На k-ій ітерації методу виконується така послідовність дій:

1) обчислюються w_k, g_k та Δw_{k-1};

2) використовуючи один з ефективних способів двовимірного пошуку, знаходять з необхідною точністю λ^1_k та λ^2_k.

3) обчислюють x_{k+1}: $x_{k+1}=x_k - \lambda_k^1 g_k + \lambda_k^2 \Delta w_{k-1}$ та переходять до першого кроку.

Кожний (n+1)-й крок, де n – кількість керувальних змінних, починається з $\Delta w_{k-1} = 0$.

Процес закінчується, коли $|\Delta F(w)| < \varepsilon$.

У **методі Крегга-Леві** на кожному кроці здійснюється багатовимірний пошук за більшим числом параметрів, ніж у методі Міля-Кентрелла. Кожний наступний вектор w знаходиться за формулою:

$$w_{k+1} = w_k - \lambda_k^0 g_k + \sum_{i=1}^{m} \lambda_k^i \Delta w^{i-1} \quad , \text{при } m \le n-1,$$

де w^i – i-та керувальна змінна.

На початкових кроках роботи методу Δw^i можна покласти рівними нулю.

3.1.7 Квазіньютонівські методи

Ці методи подібні методам спряжених градієнтів, оскільки також засновані на властивостях квадратичних функцій. Відповідно до викладених вище методів пошуку рішення здійснюється за системою спряжених напрямків, тоді як квазіньютонівські методи мають позитивні риси методу Ньютона, однак використовують тільки перші похідні. В усіх методах зазначеного класу побудова векторів напрямків пошуку здійснюється за допомогою формули:

$$w_{k+1}=w_k - \alpha_k A_k g_k,$$

де A_k – матриця порядку N x N, що називається **метрикою**. Методи пошуку уздовж напрямків, обумовлених цією формулою, називаються **методами змінної метрики**, оскільки матриця A змінюється на кожній ітерації.

Градієнт g_k може бути знайдений за допомогою методу Коші і, отже, квазіньютонівські методи полягають у знаходженні матриці A_k.

Метод Давідона-Флетчера-Пауелла для знаходження A_k використовує такі формули:

$$\Delta w_k = w_{k+1} - w_k ,$$

$$\Delta g_k = g_{k+1} - g_k \,,$$

$$A_k = A_{k-1} + \frac{\Delta w_{k-1} \Delta w_{k-1}^T}{\Delta g_{k-1}^T \Delta g_{k-1}} - \frac{A_{k-1} \Delta g_{k-1} \Delta g_{k-1}^T A_{k-1}}{\Delta g_{k-1}^T A_{k-1} \Delta g_{k-1}} \,,$$

причому звичайно зручно вибирати $A_0 = I$, де I – одинична матриця.

Метод Бройдена-Флетчера-Шенно (ранг 2) реалізується відповідно до рекурентної формули:

$$\mathrm{A}_{k+1} = \left[I - \frac{\Delta w_k \Delta g_k^T}{\Delta w_k^T \Delta g_k} \right] A_k \left[I - \frac{\Delta w_k \Delta g_k^T}{\Delta w_k^T \Delta g_k} \right] + \frac{\Delta w_k \Delta w_k^T}{\Delta w_k^T \Delta g_k} \,.$$

До числа головних переваг цього методу варто віднести не завжди обов'язкову необхідність повернення до початкової ітерації алгоритму і відносно слабку залежність від точності обчислень при проведенні одновимірного пошуку.

Метод Бройдена (ранг 1) реалізується у відповідності з формулою:

$$A_{k+1} = \mathrm{A}_k + \frac{\left[\Delta w_k - A_k \Delta g_k\right]\left[\Delta w_k - A_k \Delta g_k\right]^T}{\left[\Delta w_k - A_k \Delta g_k\right]^T \Delta g_k} \,.$$

Метод Пірсона № 2 для знаходження матриці A_{k+1} використовує рекурентний вираз:

$$A_{k+1} = A_k + \frac{\left[\Delta w_k - A_k \Delta g_k\right] \Delta w_k^T}{\Delta w_k^T \Delta g_k} \,, \; A_0 = R_0,$$

де R_0 – довільна позитивно визначена симетрична матриця.

Метод Пірсона № 3 реалізується у відповідності з формулою:

$$A_{k+1} = A_k + \frac{\left[\Delta w_k - A_k \Delta g_k\right]\left[A_k \Delta g_k\right]^T}{\Delta g_k^T A_k \Delta g_k} \,, \; A_0 = R_0.$$

Проективний метод Ньютона-Рафсона реалізується у відповідності з формулою:

$$A_{k+1} = A_k - \frac{\left(A_k \Delta g_k\right)\left(A_k \Delta g_k\right)^T}{\Delta g_k^T A_k \Delta g_k} \,, \; A_0 = R_0.$$

3.1.8 Метод Левенберга-Марквардта

В методі Левенберга-Марквардта використовується метод Коші, щоб обчислити якобіан J цільової функції щодо керувальних змінних. Керувальні змінні змінюються відповідно до коригувального правила, що у матричній формі має вигляд:

$$H=J^TJ,\ g=J^Te,\ w_{k+1}=w_k-[H_k+\eta I]^{-1}g_k,$$

де J – якобіан; e – вектор помилок; η – скаляр; I – одинична матриця.

Адаптивне значення η збільшується в η^+ раз доти, доки значення цільової функції не зменшиться. Після чого зміни вносяться в мережу і η зменшується в η^- раз.

Метод Левенберга-Марквардта полягає у виконанні такої послідовності кроків.

Крок 1. Ініціалізація: задаються початкові значення керувальних змінних, а також граничні значення параметрів закінчення роботи алгоритму.

Лічильник числа циклів навчання $k = 1$.

Крок 2. Перевірка збіжності та умов закінчення роботи. Якщо збіжність досягнута, $k >$ Epochs або робота повинна бути припинена – закінчення роботи.

Крок 3. Якщо $\eta \le \eta_{max}$, то перехід до кроку 4, інакше до кроку 7.

Крок 4. Обчислити значення помилки і скорегувати відповідним чином керувальні змінні:

$$H=J^TJ,\ \ g=J^Te,$$
$$w_{k+1}=w_k-[H_k+\eta I]^{-1}g_k.$$

Крок 5.Обчислити нове значення цільової функції. Якщо воно менше поточного, то повернути попередні значення керувальним змінним, змінити η: $\eta = \eta\eta^-$ і вийти з циклу, інакше зафіксувати значення керувальних змінних і змінити η: $\eta = \eta\eta^+$.

Крок 6. Перехід до кроку 4.

Крок 7. $k=k+1$.

Крок 8. Перехід до кроку 2.

Навчання зупиняється, якщо виконується хоча б одна з умов:

1) досягнуто максимально припустиме число циклів навчання Epochs;

2) цільова функція мінімізована (ціль досягнута);

3) градієнт цільової функції менше мінімального припустимого значення;

4) η перевищує максимальне припустиме значення η_{max};

5) цільова функція збільшилась більше ніж у MAX_FAIL разів, починаючи з останнього разу зменшення.

Метод Левенберга-Марквардта широко використовується для побудови нейро- та нейро-нечітких моделей. Проте цей метод вимагає наявності інформації про значення других похідних цільової функції, що в окремих випадках ускладнює його застосування на практиці.

3.2 Еволюційні методи

Градієнтні методи оптимізації передбачають обчислення значень цільової функції, що ускладнює, а в деяких випадках і унеможливлює їх застосування на практиці.

Еволюційні методи, на відміну від градієнтних, не вимагають диференційованості цільової функції та не накладають обмежень на її вигляд.

Еволюційний (генетичний) пошук включає групу багатовимірних, стохастичних, евристичних оптимізаційних методів, заснованих на ідеї еволюції за допомогою природного відбору, висунутої Ч. Дарвіном в 1857 р. Методи генетичного пошуку отримані в процесі узагальнення та імітації в штучних системах таких властивостей живої природи, як природний відбір, пристосовність до змінюваних умов середовища, спадкоємність нащадками життєво важливих властивостей від батьків і т. ін.

Формально методи генетичного пошуку можуть бути описані у вигляді такої функції:

$$\text{ГМ} = \text{ГМ}(P_0, N, L, f, \Omega, \Psi, \vartheta, T),$$

де $P_0 = \{H_1^0, H_2^0, \ldots, H_N^0\}$ – початкова популяція – множина рішень задачі, поданих у вигляді хромосом; $H_j^0 = \{h_{1j}^0, h_{2j}^0, \ldots, h_{Lj}^0\}$ – j-та хромосома популяції P_0 – набір значень незалежних змінних, по-

даних у вигляді генів; h_{ij}^{0} – i-ий ген j-ої хромосоми популяції P_0 – значення i-го оптимізованого параметру задачі, що входить в j-те рішення; N – кількість хромосом в популяції (розмір популяції); L – довжина хромосом (кількість генів); f – цільова функція (фітнес-функція, функція пристосованості, функція здоров'я, функція придатності); Ω – оператор відбору; Ψ – оператор схрещування; ϑ – оператор мутації; T – критерії зупинення.

Таким чином, на кожній ітерації генетичного пошуку метод працює не з єдиним рішенням, а із деякою множиною рішень (сукупністю хромосом), за рахунок чого забезпечується паралельність пошуку. При цьому кожна нова множина рішень залежить лише від попередньої і, в загальному випадку, є кращою за попередню.

Оскільки генетичні методи в процесі пошуку використовують деяке кодування множини параметрів замість самих параметрів, то вони можуть ефективно застосовуватися для вирішення задач дискретної оптимізації, визначених як на числових множинах, так і на скінчених множинах довільної природи.

Для роботи генетичних методів як інформація про функцію, що оптимізується, використовуються її значення в даних точках простору пошуку і не потрібно обчислень похідних або інших характеристик. Тому дані методи можуть бути застосовані до широкого класу функцій, зокрема до тих, що не мають аналітичного опису. Таким чином, методи генетичного пошуку є достатньо гнучкими і можуть бути застосовані до широкого кола задач, в тому числі до задач, для розв'язування яких не існує загальновідомих методів.

Генетичні методи є більш ефективним інструментом пошуку в порівнянні з класичними методами оптимізації за таких **умов**:

– досліджуваний простір пошуку є великим, негладким (існують точки розриву) і неунімодальним (є декілька оптимумів);

– цільова функція пошуку може мати шуми;

– задача не вимагає знаходження надточного глобального оптимуму. Тобто необхідно достатньо швидко знайти прийнятне рішення, що досить часто має місце в реальних задачах.

Таким чином, генетичний пошук може успішно використовуватися для вирішення комбінаторних задач, а також для пошуку оптимальних значень полімодальних функцій.

Генетичні методи мають такі **переваги***:*

– відсутність необхідності в специфічних знаннях про вирішувану задачу. Проте у випадку, якщо додаткова інформація про досліджувану систему, об'єкт або процес є відомою, то вона може бути використана в процесі пошуку;

– концептуальна простота та прозорість реалізації;

– можливість розпаралелювання;

– простота кодування вхідної і вихідної інформації. Некритичність до виду параметрів досліджуваних систем (можливість використання експертної, емпіричної, довідкової та іншої інформації про об'єкт, поданої різними типами даних);

– можливість застосування до великого кола задач без внесення серйозних змін у внутрішню структуру методу;

– можливість адаптивності параметрів генетичного пошуку до особливостей вирішуваної задачі;

– менша ймовірність попадання і зациклення в локальному оптимумі, яка досягається за рахунок використання популяційного підходу;

– можливість застосування в методі інших пошукових процедур.

До **недоліків** генетичного пошуку відносять:

– високу ітеративність;

– сильну залежність ефективності генетичного пошуку від його параметрів (розмір популяції, початкова точка пошуку, імовірнісні характеристики генетичних операторів і т. ін.);

– епістазис – внутрішню залежність між змінними (генами), закодованими в хромосомі. Якщо гени не пов'язані один з одним, то вважається, що епістазис малий або не існує. У випадку, якщо гени є взаємозалежними, то епістазис може створювати проблеми для генетичного пошуку, спричинені тим, що при схрещуванні ланцюжки взаємозалежних генів будуть зруйновані, що призводить до появи низько пристосованих нащадків. Вирішення проблеми епістазису полягає в тому, щоб зберігати в хромосомі взаємозалежні гени, розташовуючи їх близько один до одного. При групуванні залежних генів істотно знижується ймовірність того, що вони будуть зруйновані при застосуванні схрещування;

– передчасна збіжність, пов'язана з недостатньою різноманітністю хромосом в популяції. Найпоширенішою причиною пе-

редчасної збіжності є недостатній розмір популяції. Таким чином, вирішенням такої проблеми може бути збільшення кількості хромосом в популяції.

Враховуючи особливості, переваги і недоліки оптимізаційних методів, можна надати такі **рекомендації щодо вибору пошукового методу** при вирішенні практичних задач:

– якщо простір пошуку є дискретним і невеликим за розміром, то можна скористатися методом повного перебору, який знайде найкраще рішення. Генетичний метод, на відміну від методу повного перебору, може з більшою ймовірністю зійтися до локального оптимуму, а не до глобального. Проте генетичний пошук швидше знайде субоптимальне рішення, що знаходиться недалеко від дійсного оптимуму;

– якщо цільова функція в пошуковому просторі є гладкою і унімодальною, то будь-який градієнтний метод (наприклад, метод найшвидшого спуску) буде ефективнішим, ніж генетичний пошук;

– якщо про простір пошуку відома деяка додаткова інформація (наприклад, для задачі комівояжера), то методи пошуку, що використовують апріорні відомості про пошуковий простір, часто перевершують будь-який універсальний метод, у тому числі й генетичний метод;

– при достатньо складному рельєфі цільової функції методи пошуку, що працюють із єдиним рішенням на кожній ітерації (наприклад, простий метод спуску), можуть зациклитися в локальному рішенні. Генетичні методи працюють з набором із декількох рішень, тому вони мають менше шансів зійтися до локального оптимуму і надійно функціонують на багатоекстремальних поверхнях.

3.2.1 Узагальнена схема роботи генетичних методів

Сутність генетичного пошуку полягає в циклічній заміні однієї популяції наступною, більш пристосованою. Таким чином, популяція існує не тільки в просторі, але і в часі. Часто можна вважати, що вся популяція складається в просторі і в часі з дискретних поколінь (генерацій, епох) P_0, P_1, P_2, ..., P_T.

Покоління P_{t+1} – це сукупність особин, батьки яких належать поколінню P_t. Покоління P_0 є **початковою популяцією**. Процес формування покоління P_0 називається **ініціалізацією**. Кожне покоління є результатом циклу роботи генетичного методу.

Кожна хромосома (особина, точка в просторі пошуку) оцінюється мірою її пристосованості відповідно до того, наскільки є гарним відповідне їй рішення задачі. **Пристосованість** визначається як обчислена цільова функція (**фітнес-функція**) для кожної з хромосом. Правила **відбору** (**селекції**) прагнуть залишити лише ті точки-рішення, де досягається оптимум цільової функції. Найбільш пристосовані особини дістають можливість відтворювати нащадків за допомогою перехресного **схрещування** з іншими особинами популяції. Це призводить до появи нових особин, які поєднують в собі деякі характеристики, успадковані ними від батьків. Найменш пристосовані особини з меншою ймовірністю зможуть відтворити нащадків, внаслідок чого ті властивості, якими вони володіли, поступово зникатимуть з популяції в процесі еволюції.

Таким чином, з покоління в покоління, гарні характеристики розповсюджуються по всій популяції. Комбінація гарних характеристик від різних батьків іноді може призводити до появи суперпристосованого нащадка (або мутанта), чия пристосованість більша, ніж пристосованість будь-якого з його батьків. Схрещування найбільш пристосованих особин призводить до того, що досліджуються найбільш перспективні ділянки простору пошуку. Зрештою, популяція збігатиметься до оптимального рішення задачі.

Після схрещування інколи відбуваються **мутації** – спонтанні зміни в генах, які випадковим чином розкидають точки по всій множині пошуку.

Процес збіжності за рахунок відбору повинен бути більш виражений порівняно з розкидом точок за рахунок мутації і інвертування, інакше збіжність до екстремумів не матиме місця. Збіжність за рахунок відбору не повинна бути занадто швидкою, інакше всі точки можуть зібратися поблизу локального екстремуму, а інший, можливо глобальний, так і не буде знайдено.

Після отримання нащадків за допомогою схрещування та мутації розмір популяції збільшується. Для подальших перетворень число хромосом поточної популяції зменшується до заданого розміру популяції.

Подальша робота генетичного методу є ітераційним процесом застосування **генетичних операторів** до особин наступного покоління. Генетичні оператори необхідні, щоб застосувати принципи спадковості і мінливості до популяції. Такі оператори володіють **властивістю ймовірності**, тобто вони не обов'язково застосовуються до всіх особин, що вносить додатковий елемент невизначеності в процес пошуку рішення. Невизначеність не має на увазі негативного чинника, а є своєрідною мірою свободи роботи генетичного методу.

Схему роботи узагальненого генетичного методу наведено на рис. 3.1.

Рисунок 3.1 – Схема роботи узагальненого генетичного метода

Узагальнений метод генетичного пошуку можна записати таким чином.

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$. Виконати ініціалізацію (initialization) початкової популяції особин: $P_t = \{H_1, H_2, \ldots, H_N\}$.

Крок 2. Оцінити особини поточної популяції (evaluating) шляхом обчислення їх фітнес-функції $f(H_j)$, $j = 1, 2, \ldots, N$.

Крок 3. Перевірити умови закінчення пошуку (termination criteria). Як такі умови можуть бути використані: досягнення максимально допустимого часу функціонування методу, числа ітерацій, значення функції пристосованості і т. ін. Якщо критерії закінчення пошуку задовільнено, тоді виконати перехід до кроку 12.

Крок 4. Збільшити лічильник ітерацій (часу): $t = t + 1$.

Крок 5. Вибрати частину популяції (батьківські особини) для схрещування (selection of parents) *P'*.

Крок 6. Сформувати батьківські пари (mating) з особин, що відібрані на попередньому кроці.

Крок 7. Схрестити (crossover) вибрані батьківські особини.

Крок 8. Застосувати оператор мутації (mutation) до особин *P'*.

Крок 9. Обчислити нову функцію пристосованості $f(H_j)$ особин, отриманих в результаті схрещування та мутації.

Крок 10. Сформувати нове покоління шляхом вибору особин, що вижили, виходячи з рівня їх пристосованості (replacing, selection of survivors).

Крок 11. Перейти до кроку 3.

Крок 12. Зупинення.

Таким чином, при реалізації генетичних методів необхідно:

– визначити параметри, що оптимізуються, залежно від вирішуваної задачі, вибрати спосіб кодування (подання в хромосомі) параметрів, що оптимізуються;

– задати цільову функцію;

– визначити правила ініціалізації початкової популяції;

– вибрати оператори відбору, схрещування і мутації, а також задати їх параметри;

– визначити критерії зупинення.

3.2.2 Ініціалізація еволюційного пошуку

Будь-який організм може бути поданий своїм **фенотипом**, який фактично визначає, чим є об'єкт в реальному світі, і **генотипом**, який містить всю інформацію про об'єкт на рівні хромосомного набору. При цьому кожен ген, тобто елемент інформації генотипу, має своє відображення у фенотипі. Таким чином, для розв'язання задач необхідно подати кожну ознаку (атрибут, вхідну змінну, параметр) об'єкту у певному закодованому вигляді для використання в генетичному методі. Все подальше функціонування механізмів генетичного пошуку проводиться на рівні генотипу, що дозволяє обійтися без інформації про внутрішню структуру об'єкту, що і обумовлює широке застосування генетичного пошуку в різних задачах.

За методами подання генів хромосоми можна умовно поділити на три групи.

1. **Бінарні хромосоми** – хромосоми, гени яких можуть приймати значення 0 або 1.

2. **Числові хромосоми** – гени можуть приймати значення в заданому інтервалі.

Числові хромосоми можна поділити на гомологічні і негомологічні.

Гомологічними називають хромосоми, що мають загальне походження, є морфологічно і генетично схожими, і тому не створюють неприпустимих рішень при застосуванні стандартних генетичних операторів. У гомологічних числових хромосомах кожен ген може приймати будь-які значення в заданому інтервалі. Для різних генів можуть бути задані різні інтервали. Бінарна хромосома є гомологічною числовою хромосомою, кожен ген якої може приймати цілі значення в інтервалі [0; 1].

У **негомологічних** хромосомах гени можуть приймати значення в заданому інтервалі; при цьому інтервал однаковий для всіх генів, але в хромосомі не може бути двох генів з однаковим значенням. Для негомологічних хромосом застосовуються різні спеціальні генетичні оператори, що не створюють неприпустимих рішень.

3. **Векторні хромосоми** – хромосоми, гени яких є вектором цілих чисел.

Ген у векторних хромосомах має властивості негомологічної хромосоми, тобто числа у векторі можуть приймати значення в заданому інтервалі, і вектор не може містити двох однакових чисел. Проте, хоча гени у векторних хромосомах є негомологічними, самі векторні хромосоми є гомологічними.

Як правило, генетичні методи починають свою роботу з ініціалізації, тобто формування **початкової популяції** P_0 – скінченого набору допустимих рішень задачі: $P_0 = \{H_1, H_2, ..., H_N\}$; де N – розмір популяції; $H_j = \{h_{1j}, h_{2j}, ..., h_{Lj}\}$ – j-та хромосома, що складається з L генів; h_{ij} – значення i-го гену j-ої хромосоми, $w_{\min, i} \le h_{ij} \le w_{\max, i}$; $w_{\min, i}$ та $w_{\max, i}$ – мінімальне й максимальне значення i-го параметру у вирішуваній за допомогою генетичного методу задачі.

Хромосоми початкової популяції можуть бути вибрані випадковим чином або введені користувачем. Вибір початкової популяції не має значення для збіжності процесу в асимптотиці, проте, формування гарної початкової популяції (наприклад, з множини локальних оптимумів) може помітно скоротити час досягнення глобального оптимуму. Таким чином, за наявності необхідної інформації задавання початкової популяції користувачем є переважним.

Найчастіше розмір початкової популяції вибирається в інтервалі 20–100 особин.

Використовуються такі **стратегії ініціалізації початкової популяції**:

– випадкове формування початкової популяції;

– рівномірне формування початкової популяції (сіткова ініціалізація). При такій стратегії генерується множина рішень, рівно віддалених одне від одного;

– формування початкової популяції, засноване на знаннях. Така стратегія застосовується в тих випадках, коли є припущення, що деяка точка простору пошуку розташовується поблизу глобального оптимуму цільової функції;

– сіткова ініціалізація, заснована на знаннях, – об’єднує основні положення попередніх двох стратегій.

3.2.3 Оператор відбору

Оператор відбору (вибору, селекції) є механізмом, на основі якого відбувається вибір хромосом для застосування до них того або іншого еволюційного оператору, а також для формування нової популяції. При різних моделях еволюційного пошуку оператор відбору може бути використаний для таких цілей:

– вибір хромосом-батьків для схрещування;

– формування пар хромосом для схрещування;

– вибір хромосом для формування нового покоління з хромосом-батьків старої популяції і хромосом-нащадків, одержаних в результаті схрещування і мутації.

Як правило, оператор відбору використовується для вибору хромосом-батьків для схрещування. При цьому хромосоми вибираються на основі значення цільової функції так, щоб з ненульовою ймовірністю будь-який елемент популяції міг би бути вибраний як один з батьків для схрещування.

Найбільш поширеними є такі оператори відбору:

– пропорційний відбір (пропорційно-імовірнісний);

– відбір ранжируванням;

– турнірний відбір;

– відбір з використанням порогу.

3.2.3.1 Пропорційний відбір

Даний вид відбору виконується у такій послідовності кроків.

Крок 1. Обчислити значення фітнес-функції кожної особини f_j.

Крок 2. Знайти середнє значення фітнес-функції $f_{\text{ср.}}$ популяції як середнє арифметичне значень фітнес-функцій всіх особин:

$$f_{\text{ср.}} = \frac{1}{N}\sum_{j=1}^{N} f_j \ .$$

Крок 3. Для кожної особини обчислити відношення:

$$P_s(j) = \frac{f_j}{f_{\text{ср.}}}.$$

Крок 4. Залежно від величини $P_s(j)$ сформувати масив особин, допущених до схрещування.

Формування масиву допущених до схрещування особин (крок 4) можна здійснити двома способами.

Перший спосіб (стохастичний залишковий відбір): якщо $P_s(j) > 1$, тоді особина вважається добре пристосованою і допускається до схрещування.

Наприклад, якщо $P_s(j) = 2{,}36$, то дана особина має подвійний шанс на схрещування і матиме ймовірність, що дорівнює 0,36, третього схрещування. Якщо ж пристосованість дорівнює 0,54, то особина візьме участь в єдиному схрещуванні з ймовірністю 0,54.

Реалізується це таким чином. Нехай є масив всіх особин популяції і додатковий масив для особин, допущених до схрещування. Для кожної особини популяції визначається значення описаного вище відношення. Далі, записуються рядки в проміжний масив згідно такого правила: необхідно узяти цілу частину відношення $P_s(j)$ і рівно стільки разів записати дану особину в допоміжний масив, після цього за допомогою випадкової величини визначити, чи буде вноситься дана особина ще раз: якщо випадкова величина більше дробової частини відношення, то буде, в іншому випадку – не буде. Надалі, особини для схрещування вибираються лише з проміжного масиву випадковим чином.

Для вже розглянутих прикладів з числами 2,36 і 0,54 ситуація виглядатиме таким чином: перша хромосома запишеться в проміжний масив двічі і з ймовірністю 0,36 запишеться втретє. Друга ж хромосома має ймовірність, що дорівнює 0,54 того, що вона взагалі буде присутня в допоміжному масиві.

Другий спосіб полягає в тому, що після знаходження відношення $P_s(j)$ відбувається відбір (із заміщенням) всіх N особин для подальшої генетичної обробки згідно величини $P_s(j)$.

Простий пропорційний відбір – *рулетка* – відбирає особини за допомогою N запусків рулетки. Колесо рулетки містить по одному сектору для кожного члена популяції. Розмір j-го сектора пропорційний відповідній величині $P_s(j)$. Особина одержує нащадків, якщо випадково згенероване число в межах від 0 до 2π потрапляє в сектор, що відповідає цій особині. При такому відборі члени популяції з вищою пристосованістю з більшою ймовірністю частіше вибиратимуться, ніж особини з низькою пристосованістю.

При реалізації відбору рулеткою доцільно замінити колесо рулетки інтервалом [0; 1] у зв'язку з тим, що в такому разі немає необхідності обчислювати ширину кожного сектора – в цьому випадку кожній особині зіставляється напівінтервал $[x_{j-1}; x_j)$, де $x_{j-1} - x_j = P_s(j)$, а $x_0 = 0$ (при цьому $x_N = 1$). Допускається до схрещування особина з номером j, де j: $x_{\text{rnd}} \in [x_{j-1;}\, x_j)$, а число x_{rnd} генерується кожного разу випадковою функцією з рівномірним розподілом щільності ймовірності на відрізку [0; 1].

Схема рулетки може давати дуже великі відхилення між одержаною кількістю нащадків даної особини і очікуваною. Одержане число нащадків наближається до очікуваного лише в популяціях дуже великих розмірів.

3.2.3.2 Відбір ранжируванням

Відбір ранжируванням виконується за чотири кроки.

Крок 1. Обчислити пристосованість кожної особини f_j.

Крок 2. Відсортувати (ранжирувати) популяцію за збільшенням пристосованості особин.

Крок 3. Для кожної особини обчислити величину $P_s(j)$. Для цього використовувати один з двох видів ранжирування:

а) лінійне ранжирування: $P_s(j) = \frac{1}{N}\left(\eta_{\max} - (\eta_{\max} - \eta_{\min})\frac{j-1}{N-1}\right)$,

де $\eta_{\max} \in [1; 2]$, $\eta_{\min} = 2 - \eta_{\max}$;

б) рівномірне ранжирування: $P_s(j) = \begin{cases} \frac{1}{\mu}, \text{ якщо } 1 \le j \le \mu; \\ 0, \text{ якщо } \mu < j \le N, \end{cases}$

де μ – деяке фіксоване число перших членів популяції.

Крок 4. Залежно від величини $P_s(j)$ відібрати певну частину особин для схрещування.

3.2.3.3 Турнірний відбір

Турнірний відбір реалізує k турнірів, щоб вибрати k особин. Кожен турнір складається з двох етапів.

Етап 1. Вибір m елементів з популяції.

Етап 2. Вибір кращої особини серед особин, відібраних на попередньому етапі.

Розмір групи особин, відібраних для турніру, часто дорівнює двом. В цьому випадку говорять про **двійковий** (**парний**) **турнір**. Взагалі ж m називається **чисельністю турніру**.

Турнірний відбір має певні переваги перед пропорційним відбором, оскільки не втрачає своєї вибірковості у випадку, коли в ході еволюції всі елементи популяції стають приблизно рівними по значенню цільової функції.

3.2.3.4 Відбір з використанням порогу

Відбір з використанням порогу (відбір усіканням) виконується таким чином.

Крок 1. Обчислити пристосованість кожної особини f_j.

Крок 2. Відсортувати популяцію за збільшенням пристосованості особин.

Крок 3. Задати поріг $tr \in [0; 1]$. Поріг визначає, яка частка особин, починаючи з найпершої (самої пристосованої), братиме участь в схрещуванні. В принципі, поріг можна задати і числом, більшим за 1, тоді він буде просто дорівнювати числу особин з поточної популяції, допущених до схрещування.

Крок 4. Серед особин, що потрапили під значення порогу, випадковим чином N разів вибрати саму везучу і записати її в проміжний масив, з якого потім вибираються особини безпосередньо для схрещування.

Через те, що в цій стратегії використовується відсортована популяція, час її роботи може бути великим для популяцій великого розміру і залежати також від методу сортування.

3.2.4 Оператор схрещування

У теорії еволюції важливу роль відіграє те, яким чином ознаки батьків передаються нащадкам. У генетичних методах за передачу ознак батьків нащадкам відповідає оператор схрещування (кросинговер, кроссовер, рекомбінація). Цей оператор моделює процес схрещування особин і визначає передачу ознак батьків нащадкам.

Метою оператора схрещування є породження з наявної множини рішень нового, в якому кожна хромосома буде нащадком деяких двох елементів попередньої популяції, тобто нести в собі частково інформацію кожного батька. Допускається ситуація, коли обидва батька подані одним і тим же елементом популяції.

3.2.4.1 Вибір батьківської пари

Вибираючи кожного разу для схрещування найбільш пристосовані особини, можна з певним ступенем впевненості стверджувати, що нащадки будуть або не набагато гірші, ніж батьки, або кращі за них.

Існує декілька способів вибору батьківської пари.

Випадковий вибір батьківської пари («**панміксія**») – це найпростіший підхід, коли обидві особини, які складуть батьківську пару, випадковим чином вибираються зі всієї популяції, причому будь-яка особина може стати членом декількох пар.

Крок 1. Для вибору пари батьків задати ймовірність схрещування P_c.

Крок 2. Довільним чином пронумерувати всіх представників поточної популяції.

Крок 3. Вибрати першого батька. Для цього, починаючи з першої, перебрати всі хромосоми популяція до тих пір, поки випадково вибране число з інтервалу [0; 1] не буде меншим за P_c. Коли це відбудеться для одного з елементів популяції, цей елемент стане першим батьком.

Крок 4. Продовжити перегляд популяції, починаючи з наступного після першого батька рішення. Перегляд продовжувати до тих пір, поки знову випадково обране число не буде меншим, ніж P_c. Хромосома, для якої виконується така умова, буде другим батьком.

Описаним способом складаються пари до тих пір, поки не вибереться потрібна кількість пар батьків.

Конкретне значення P_c залежить від вирішуваної задачі, і в загальному випадку лежить в інтервалі [0,6; 0,99].

Не зважаючи на простоту, такий підхід універсальний для розв'язання різних класів задач. Проте, він достатньо критичний

до чисельності популяції, та його ефективність знижується із зростанням чисельності популяції.

Інший метод випадкового вибору батьківської пари може бути поданий у вигляді такої послідовності кроків.

Крок 1. Розбити популяцію випадковим чином на два масиви (підпопуляції) одного розміру.

Крок 2. Відсортувати кожну підпопуляцію.

Крок 3. Сформувати пари для схрещування з особин, що мають однаковий ранг (номер) в підпопуляціях.

Крок 4. Допустити до схрещування пари, для яких число, що випадково згенероване в інтервалі [0; 1], перевищуватиме задану ймовірність схрещування.

Селективний спосіб формування батьківської пари полягає в тому, що «батьками» можуть стати лише ті особини, значення пристосованості яких не менше середнього значення пристосованості по популяції, при рівній імовірності таких кандидатів скласти батьківську пару.

Такий підхід забезпечує більш швидку збіжність генетичного пошуку. Проте через швидку збіжність селективний вибір батьківської пари не підходить тоді, коли ставиться задача визначення декількох екстремумів, оскільки для таких задач метод, як правило, швидко сходиться до одного з рішень.

Крім того, для деякого класу задач зі складним ландшафтом фітнес-функції швидка збіжність може перетворитися на передчасну збіжність до квазіоптимального рішення. Цей недолік може бути частково компенсований використанням відповідного механізму відбору, який би сповільнював дуже швидку збіжність методу.

Інші два способи формування батьківської пари – це **інбридинг** і **аутбридинг**. Обидва ці методи побудовані на формуванні пари на основі близької і далекої «спорідненості» відповідно. Під «спорідненістю» тут розуміється відстань між членами популяції як в сенсі Евклідової (геометричної) відстані особин в просторі параметрів (для фенотипів), так і в сенсі відстані Хеммінга між хромосомними наборами особин (для генотипів).

Евклідова відстань $d_E(H_j; H_k)$ між j-ою і k-ою особинами популяції визначається за формулою:

$$d_E\left(H_j; H_k\right) = \sqrt{\sum_{i=1}^{p}\left(x_i^{(j)} - x_i^{(k)}\right)^2},$$

де p – кількість параметрів (генів) особини; $x_i^{(j)}$ – i-ий параметр в незакодованому вигляді j-ої особини.

Відстань Хеммінга $d_H(H_j; H_k)$ між j-ою і k-ою особинами популяції визначається як кількість неспівпадаючих бітів в однакових позиціях j-ої і k-ої хромосом:

$$d_H\left(H_j; H_k\right) = \sum_{i=1}^{L}\left|h_{ij} - h_{ik}\right|.$$

Інбридинг складається з двох етапів. На першому етапі випадковим чином обирається перший член пари, на другому етапі вибирається другий батько. При цьому другою батьківською особиною з більшою ймовірністю буде максимально близька до першої особина.

Один з варіантів процедури інбридингу може бути реалізований таким чином.

Крок 1. Вибрати випадковим чином першого батька.

Крок 2. Вибрати з поточної популяції випадковим чином групу із C хромосом (C = 1 %–15 % від розміру популяції).

Крок 3. Розрахувати Евклідову відстань від хромосоми, отриманої на першому кроці, до кожної з C відібраних на другому кроці хромосом.

Крок 4. Як другого батька вибрати найближчу до першого батька хромосому.

Аутбридинг формує батьківські пари з максимально далеких особин.

Використання генетичних інбридингу і аутбридингу є ефективним для багатоекстремальних задач. Проте, ці два способи порізному впливають на поведінку генетичного методу.

Інбридинг можна охарактеризувати властивістю концентрації пошуку в локальних вузлах, що фактично призводить до розбиття популяції на окремі локальні групи навколо підозрілих на екстремум ділянок ландшафту фітнес-функції.

Аутбридинг, навпаки, спрямований на попередження збіжності методу до вже знайдених рішень, примушуючи метод проглядати нові, недосліджені області.

3.2.4.2 Точкове схрещування

Точкове схрещування виконується таким чином.

Крок 1. Випадково вибрати n точок розриву, внаслідок чого одержати розбиття початкових хромосом на $(n + 1)$ частин різної довжини.

Крок 2. Обміняти у початкових хромосом ділянки з парними номерами, а ділянки з непарними залишити без змін.

Класичним варіантом такого схрещування є **одноточкове схрещування**, при якому:

– випадковим чином визначається точка всередині хромосоми. Ця точка називається **точкою розриву** (**точкою схрещування**, crossover point);

– у вибраній точці обидві хромосоми діляться на дві частини і обмінюються ними. В результаті утворюються два нащадки.

Даний тип схрещування називається одноточковим, оскільки при його застосуванні батьківські хромосоми розрізаються лише в одній випадковій точці.

При **двоточковому схрещуванні** в хромосомі випадково вибираються вже дві точки схрещування. Ліву точку вважають першою, а праву – другою. Перший нащадок формується з частин першого батька, розташованих лівіше першої точки схрещування і правіше другої точки, і частини другого батька, розташованої між першою і другою точками схрещування. Другий нащадок отримується з лівої і правої частин другого батька і центральної частини першого батька.

3.2.4.3 Однорідне схрещування

Однорідне схрещування (uniform crossover) генерує нащадка шляхом випадкової передачі йому генетичної інформації від батьків. Генерація нащадка виконується таким чином.

Крок 1. Встановити лічильник бітів (генів) нащадка: $j = 1$.

Крок 2. Визначити хромосому з батьківської пари, яка передасть значення свого j-го гена нащадку: n = round(rand[0; 1]), де rand[0; 1] – число, що випадково згенероване в інтервалі [0; 1]; round(A) – округлене значення числа A.

Крок 3. Виконати: $h_{jo} = h_{jn}$, де h_{jo} – значення j-го гена нащадка, h_{jn} – значення j-го гена n-го батька.

Крок 4. Виконати: $j = j + 1$.

Крок 5. Якщо $j > L$, де L – довжина хромосоми, тоді виконати перехід до кроку 6. Інакше перейти до кроку 2.

Крок 6. Зупинення.

3.2.4.4 Рівномірне схрещування

Рівномірне схрещування зручно застосовувати у тому випадку, коли вже одержані індивіди з гарними спадковими ознаками, і їх необхідно закріпити в поточній популяції. Рівномірне схрещування виконується як послідовність кроків 1–3.

Крок 1. Випадковим чином задати **маску схрещування**, що є рядком з нулів і одиниць, довжиною, що дорівнює довжині хромосом.

Крок 2. Сформувати першого нащадка.

Одиниця на конкретній позиції в масці схрещування означає, що елемент, який стоїть на тому ж місці в першого батька, необхідно помістити на це місце в першого нащадка. Нуль на цій позиції в масці схрещування означає, що елемент, який стоїть на тому ж місці другого батька, необхідно помістити на це місце першого нащадка.

Крок 3. Сформувати другого нащадка.

Якщо першого батька вважати другим, а другого – першим, то аналогічно кроку 2 можна одержати другу хромосому-нащадка.

Варто відзначити, що маска схрещування може бути одна для всіх хромосом, або своя для кожної пари батьків.

3.2.4.5 Порівняльне схрещування

При порівняльному схрещуванні в хромосомах батьків порівнюються всі біти. Біти, що співпали (0 і 0 або 1 і 1), залишають-

ся без зміни, а ті, що не співпали визначаються через генератор випадкових чисел.

3.2.4.6 Арифметичне схрещування

Арифметичне схрещування (arithmetical crossover) є найбільш вдалим для пошуку оптимуму функції багатьох дійсних змінних. Цей оператор схрещування може застосовуватися при використанні гомологічних числових хромосом.

При використанні арифметичного схрещування на основі значень генів хромосом-батьків H_1 та H_2 створюються два нащадки $H_{1п}$ й $H_{2п}$, значення i-их генів $h_{1пi}$ та $h_{2пi}$ яких розраховуються за формулами:

$$h_{1пi} = k \cdot h_{1i} + (1 - k) h_{2i},$$
$$h_{2пi} = k \cdot h_{2i} + (1 - k) h_{1i},$$

де $k \in [0; 1]$ – деякий дійсний коефіцієнт, який може залежати від $d(H_1; H_2)$ – відстані між хромосомами.

Залежно від способу завдання коефіцієнта k розрізняють:

– **рівномірне арифметичне схрещування**, при якому коефіцієнт k задається користувачем на етапі ініціалізації генетичного пошуку і є постійним впродовж функціонування генетичного методу;

– **нерівномірне арифметичне схрещування** – коефіцієнт k генерується випадковим чином на кожній ітерації або змінюється залежно від номера поточної ітерації.

Якщо початкова функція є неперервною, то ця модель працює гарно, а якщо вона ще і є гладкою, тоді – відмінно.

Арифметичне схрещування може бути модифіковано таким чином. Дві хромосоми-батьки H_1 та H_2 породжують чотирьох нащадків $H_{1п} - H_{4п}$, значення i-их генів $h_{1пi} - h_{4пi}$ яких розраховуються за формулами:

$$h_{1пi} = \frac{h_{1i} + h_{2i}}{2},$$

$$h_{2пi} = w_{\max,i}(1-k) + k \max\left(h_{1i}, h_{2i}\right),$$

$$h_{3пi} = w_{\min,i}(1-k) + k \min\left(h_{1i}, h_{2i}\right),$$

$$h_{4\text{п}i} = \frac{\left(w_{\max i} + w_{\min i}\right)(1-k) + \left(h_{1i} + h_{2i}\right)k}{2},$$

де $w_{\min,\, i}$ та $w_{\max,\, i}$ – мінімальне й максимальне значення i-го параметра у вирішуваній за допомогою генетичного методу задачі; $k \in [0; 1]$.

В якості нащадку вибирається хромосома з найкращим значенням фітнес-функції.

Лінійне схрещування (linear crossover) є іншою модифікацією арифметичного схрещування. При використанні лінійного схрещування двоє хромосом-батьків H_1 та H_2 формують трьох нащадків $H_{1\text{п}} - H_{3\text{п}}$, значення i-их генів $h_{1\text{п}i} - h_{3\text{п}i}$ яких розраховуються за формулами:

$$h_{1\text{п}i} = \frac{h_{1i} + h_{2i}}{2},$$

$$h_{2\text{п}i} = 1{,}5h_{1i} - 0{,}5h_{2i},$$

$$h_{3\text{п}i} = -0{,}5h_{1i} + 1{,}5h_{2i}.$$

Модифікацією нерівномірного арифметичного схрещування є **розширене лінійчате схрещування** (extended line crossover), при якому значення i-го гена $h_{\text{п}i}$ хромосоми-нащадка $H_{\text{п}}$ визначається за формулою:

$$h_{\text{п}i} = h_{1i} + k\cdot(h_{1i} - h_{2i}).$$

При цьому інтервал можливих значень генів нащадка розширюється за рахунок збільшення діапазону можливих значень коефіцієнта k. Так, якщо $k \in [-0{,}25; 1{,}25]$, то інтервал можливих значень гена хромосоми-нащадка збільшується таким чином.

3.2.5 Оператор мутації

Оператор мутації полягає в зміні генів у випадково вибраних позиціях. На відміну від операторів відбору і схрещування, які використовуються для поліпшення структури хромосом, метою оператора мутації є *диверсифікація*, тобто підвищення різноманітності пошуку і введення нових хромосом в популяцію для того, щоб більш повно досліджувати простір пошуку. Оскільки число членів популяції P, як правило, вибирається значно меншим в по-

рівнянні із загальним числом (2^L) можливих хромосом в просторі пошуку, то через це вдається досліджувати лише частину пошукового простору. Отже, мутація ініціює різноманітність в популяції, дозволяючи проглядати більше точок в просторі пошуку і долати таким чином локальні екстремуми в ході пошуку.

В ході мутації з ненульовою ймовірністю чергове рішення може перейти в будь-яке інше рішення.

Вибір хромосом для мутації відбувається таким чином.

Крок 1. Довільно нумеруються всі хромосоми H_j поточної популяції.

Крок 2. Починаючи з першої хромосоми, переглядається вся популяція, при цьому кожній хромосомі H_j ставляться у відповідність випадкові числа x_j з інтервалу [0; 1).

Крок 3. Якщо число x_j виявляється меншим за ймовірність мутації $P_м$, то поточна хромосома H_j піддається мутації.

Серед рекомендацій по вибору ймовірності мутації нерідко можна зустріти варіанти $1/L$ або $1/N$, де L – довжина хромосоми, N – розмір популяції. Ймовірність мутації значно менше ймовірності схрещування і рідко перевищує 1 %.

Необхідно відзначити, що оператор мутації є основним пошуковим оператором і існують методи, що не використовують інших операторів окрім мутації.

3.2.5.1 Проста мутація

Проста мутація використовується для бінарних, гомологічних числових і векторних хромосом. Суть її полягає у внутрішньогенній мутації. При цьому в хромосомі випадковим чином вибирається ген, а потім проводиться його випадкова зміна.

Алгоритм простої мутації для бінарних і гомологічних числових хромосом може складатися з таких кроків.

Крок 1. Скопіювати батьківську хромосому в хромосому-нащадка.

Крок 2. Вибрати випадковим чином ген, що мутує.

Крок 3. У заданому інтервалі допустимих значень гена вибрати нове значення гена, що не дорівнює поточному.

Для векторних хромосом проста мутація проводиться шляхом внесення змін в порядок елементів усередині обраного гена.

Крок 1. Скопіювати батьківську хромосому в хромосому-нащадка.

Крок 2. Вибрати випадковим чином ген для мутації.

Крок 3. Вибрати випадковим чином точку мутації усередині гена, що мутує.

Крок 4. Провести обмін значеннями між розрядами гена, що мутує, які знаходяться безпосередньо ліворуч і праворуч від точки мутації.

Відомий також модифікований оператор простої мутації, що називається **точковою мутацією**. При такій мутації в хромосомі мутує не один, а декілька генів із заданою ймовірністю. Послідовність виконання даного оператора наведено нижче.

Крок 1. Скопіювати батьківську хромосому в хромосому-нащадка.

Крок 2. Встановити: $i = 1$.

Крок 3. Випадковим чином згенерувати число x_i з інтервалу [0; 1).

Крок 4. Якщо число x_i виявляється менше ймовірності мутації гена $P_{\text{мг}}$, то виконати мутацію гена h_i.

Крок 5. Встановити: $i = i + 1$.

Крок 6. Якщо $i < (L + 1)$, де L – довжина хромосоми, то виконати перехід до кроку 3.

Крок 7. Зупинення.

В наш час відома велика кількість різних варіантів операторів мутації. Більшість з них використовують комбінації таких ідей:

– так само як і схрещування, мутація може проводитися не лише по одній випадковій точці. Можна обирати деяку кількість точок в хромосомі для зміни значень генів, причому їх число також може бути випадковим;

– піддається мутації відразу деяка група генів, що йдуть підряд;

– сумісне застосування випадкової мутації і часткової перебудови рішення алгоритмами локального пошуку. Застосування локального спуску дозволяє генетичному методу зосередитися

лише на локальному оптимумі. Множина локальних оптимумів може опинитися експоненціально великою і на перший погляд здається, що такий варіант методу не матиме великих переваг. Проте експериментальні дослідження розподілу локальних оптимумів свідчать про високу концентрацію їх в безпосередній близькості від глобального оптимуму.

3.2.5.2 Мутація гомологічних числових хромосом

Такі види мутацій полягають в зміні вибраного для мутації гена h_{ij} (або всієї хромосоми H_j) на деяку величину Δh_{ij}, розраховану за певними методами:

$$h_{ij}' = h_{ij} + \Delta h_{ij},$$

де h_{ij} – ген до мутації; h_{ij}' – ген після мутації.

1. **Нерівномірна (non-uniform) мутація** до вибраного для мутації i-го гена h_{ij} хромосоми H_j застосовується за формулою:

$$h_{ij}' = \begin{cases} h_{ij} + \Delta\left(t, w_{\max,i} - h_{ij}\right), & \text{якщо } r \le 0{,}5; \\ h_{ij} - \Delta\left(t, h_{ij} - w_{\min,i}\right), & \text{якщо } r > 0{,}5, \end{cases}$$

де $\Delta(t, y) = y \cdot \left(1 - r^{(1-t/T)^k}\right)$; r = rand[0; 1] – випадково згенероване число в інтервалі [0; 1]; t – номер поточної ітерації; T – максимальна кількість ітерацій; k – параметр, що визначає ступінь однорідності (рівномірності); $w_{\min,\,i}$ та $w_{\max,\,i}$ – мінімальне і максимальне значення i-го параметра у вирішуваній за допомогою генетичного методу задачі.

Крім того, нерівномірна мутація i-го гена j-ої хромосоми h_{ji} може бути виконана за формулою:

$$h_{ij}' = \begin{cases} h_{ij} + \Delta\left(t, w_{\max,i} - h_{ij}\right), & \text{якщо } f\left(H_j + \Delta\left(t, w_{\max,i} - h_{ij}\right)\right) \ge f\left(H_j - \Delta\left(t, h_{ij} - w_{\min,i}\right)\right); \\ h_{ij} - \Delta\left(t, h_{ij} - w_{\min,i}\right), & \text{якщо } f\left(H_j + \Delta\left(t, w_{\max,i} - h_{ij}\right)\right) < f\left(H_j - \Delta\left(t, h_{ij} - w_{\min,i}\right)\right), \end{cases}$$

де $\Delta(t, y) = y \cdot w_i(t)$; $w_i(t)$ – коефіцієнт, що залежить від відношення t/T. Наприклад, коефіцієнт $w_i(t)$ може бути заданий такою формулою: $w_i(t) = r\left(1 - \frac{t}{T}\right)^{k_i}$, де r = rand[0; 1] – число, що згенероване

випадковим чином в інтервалі [0; 1]; $k_i > 0$ – параметр, що задається користувачем.

2. **Випадкова мутація** вибраного гена h_{ij} полягає в зміні його значення на величину Δh_{ij}, розраховану за формулою:

$$\Delta h_{ij} = \text{rand}[w_{\min,\,i} \cdot r \cdot q(t);\ w_{\max,\,i} \cdot r \cdot q(t)],\ \ q(t) = \frac{\ln T - \ln t}{\ln T},$$

де rand[a; b] – випадкове число в інтервалі [a; b]; r = rand[0; 1].

3.2.5.3 Мутація обміну

Мутація обміну використовується для бінарних і числових негомологічних хромосом.

1. При **класичній мутації обміну** в хромосомі випадковим чином вибираються два гени, які міняються місцями.

Крок 1. Створити хромосому-нащадок як копію батьківської хромосоми $H = \{h_1, h_2, ..., h_L\}$.

Крок 2. Вибрати два числа y_1 і y_2 випадковим чином з множини Y = {0, 1, 2, …, L + 1}, причому $y_1 \neq y_2$.

Крок 3. Сформувати нову хромосому H шляхом обміну елементів, що стоять на позиціях y_1 і y_2.

Таким чином, після застосування класичної мутації обміну одержуємо хромосому H':

$$H' = \{h_1, h_2, ..., h_{y_1-1}, h_{y_2}, h_{y_1+1}, ..., h_{y_2-1}, h_{y_1}, h_{y_2+1}, ..., h_L\}.$$

2. **Одноточкова мутація обміну**. У даному операторі, на відміну від попереднього, обмінюються місцями лише сусідні гени, і точка мутації вибирається між двома генами.

Крок 1. Створити хромосому-нащадок як копію батьківської хромосоми $H = \{h_1, h_2, ..., h_L\}$.

Крок 2. Вибрати точку мутації випадковим чином з множини Y = {1, 2, ..., L – 1}.

Крок 3. Сформувати нову хромосому H шляхом обміну елементів, що стоять на позиціях y і y + 1.

Таким чином, після застосування одноточкової мутації обміну одержуємо хромосому H':

$$H' = \{h_1, h_2, ..., h_{y+1}, h_y, ..., h_L\}.$$

3.2.6 Формування нового покоління

Після схрещування і мутації необхідно створити нову популяцію. Види операторів формування нового покоління (репродукції, редукції) практично співпадають з видами операторів відбору батьків, що передбачають формування проміжного масиву особин, допущених до схрещування.

Необхідно визначити, які з нових особин увійдуть до наступного покоління, а які – ні. Для цього застосовують один з двох способів.

Перший спосіб полягає в тому, що **нові особини (нащадки) займають місця своїх батьків**. Після чого настає наступний етап, в якому нащадки оцінюються, відбираються, дають потомство і поступаються місцем своїм нащадкам.

Недоліком даного способу є можливість втрати найбільш пристосованої особини попереднього покоління. Одним із способів вирішення даної проблеми може бути використання **принципу елітизму**, який полягає в тому, що хромосоми з найбільшою пристосованістю гарантовано переходять в нову популяцію. Кількість елітних особин k_e, які гарантовано перейдуть в наступну популяцію, може бути обчислена за формулою:

$$k_e = (1 - S_O) \cdot N,$$

де S_O – ступінь оновлення популяції, що знаходиться в діапазоні [0,95; 1,0]; N – розмір популяції.

Використання принципу елітизму зазвичай дозволяє прискорити збіжність генетичного пошуку. Недолік використання даної стратегії полягає в тому, що підвищується ймовірність попадання методу в локальний оптимум.

Другий спосіб передбачає створення проміжної популяції, яка включає як батьків, так і їх нащадків. Члени цієї популяції оцінюються, а потім з них вибираються ***N* найкращих, які й увійдуть до наступного покоління**. Другий варіант є більш оптимальним, але він вимагає сортування масиву розміром 2N.

Другий варіант формування нового покоління можна реалізувати за допомогою **принципу витіснення**, який носить двокритеріальний характер: включення особин в нове покоління залежить не лише від величини їх фітнес-функції, але і від того, чи

існує вже у формованій популяції наступного покоління особина з аналогічним хромосомним набором. З усіх особин з однаковими генотипами перевага спочатку віддається тим, чия пристосованість вище. Таким чином, досягаються дві мети: по-перше, не втрачаються кращі знайдені рішення, що мають різні хромосомні набори, а по-друге, в популяції постійно підтримується достатня генетична різноманітність. Витіснення в даному випадку формує нову популяцію скоріше з далеко розташованих особин, замість особин, що групуються біля поточного знайденого рішення. Цей метод особливо добре себе показав при вирішенні багатоекстремальних задач, при цьому крім визначення глобальних екстремумів з'являється можливість виділити і ті локальні оптимуми, значення яких близькі до глобальних.

3.2.7 Критерії зупинення

Одним з важливих елементів генетичного пошуку є критерії зупинення. Очевидно, що еволюція – нескінченний процес, в ході якого пристосованість особин поступово підвищується. Примусово зупинивши цей процес через достатньо довгий час після його початку і вибравши найбільш пристосовану особину в поточному поколінні, можна одержати не абсолютно точне, але близьке до оптимального рішення.

Як правило, як критерій зупинення застосовується **обмеження на максимальну кількість ітерацій** функціонування методу (тобто обмеження на кількість поколінь). Кількість популяцій може бути будь-якою, але зазвичай встановлюється 50–100 популяцій. Якщо в якості критерію зупинення обирається максимальна кількість ітерацій, то задається кількість ітерацій T, внаслідок чого цикл генетичного пошуку виконується T разів.

Зупинення роботи генетичного методу може відбутися також у випадку, якщо **популяція вироджується**, тобто, якщо практично немає різноманітності в генах особин популяції. Вироджения популяції називають передчасною збіжністю.

Якщо в якості критерію зупинення вибирається вироджения популяції, то задаються кількість ітерацій $t_п$ та поріг коефіцієнта поліпшення значень фітнес-функції кращої хромосоми $\rho_п$.

Починаючи з циклу ($t_п$ + 1), обчислюються значення цільової функції для кожної хромосоми поточної t-ої популяції, і вибирається з цих значень найкраще значення цільової функції f_{best_t}.

З попередніх (t – 1) поколінь вибирається найкраще значення цільової функції $f_{best_{t-1}}$.

Після чого обчислюється коефіцієнт поліпшення ρ за формулою:

$$\rho = \frac{f_{best_t} - f_{best_{t-1}}}{f_{best_{t-1}}}.$$

Якщо значення ρ виявляється меншим, ніж $\rho_п$, то критерій зупинення вважається досягнутим, в іншому випадку виконується наступний цикл генетичного пошуку.

Прийнятне значення цільової функції $f_п$ також може використовуватися як критерій зупинення. Якщо в процесі функціонування генетичного методу значення цільової функції f деякої особини досягло значення $f_п$ з певною наперед заданою точністю ε, то метод зупиняється. При цьому рішенням задачі є набуте значення цільової функції $f_п$.

Останнім часом методи еволюційного пошуку знайшли широке використання для вирішення завдань, пов'язаних з синтезом інтелектуальних моделей, зокрема нейромережевих, нечітких та нейронечітких моделей.

3.3 Інтелектуальні методи мультиагентної оптимізації

При вирішенні оптимізаційних задач, що виникають при побудові інтелектуальних моделей, досить перспективним є використання мультиагентних методів. При реалізації даних методів використовується парадигма агентно-орієнтованого програмування, що сприяє підвищенню продуктивності обчислень.

До інтелектуальних методів мультиагентної оптимізації, заснованих на моделюванні суспільного інтелекту, відносяться: метод мурашиних колоній (Ant Colony Optimization, ACO), метод бджолиної колонії (Bee Colony Optimization, BCO), оптимізація за допомогою рою часток (Particle Swarm Optimization, PSO) та інші методи.

3.3.1 Мультиагентні системи

Мультиагентну систему (МАС) в загальному випадку можна подати у вигляді множини з трьох елементів: Агенти, Середовище, Зв'язки між Середовищем та Агентами:

$$МАС=<Агенти, Середовище, Зв'язки>.$$

Кожен *i*-ий $Агент_i$ описується за допомогою множини чотирьох елементів: $Стан_i$, $Вхід_i$, $Вихід_i$, $Процес_i$:

$$Агент_i = <Стан_i, Вхід_i, Вихід_i, Процес_i>,$$

де $Стан_i$ – це множина змінних, що повністю визначають агента; $Вхід_i$ та $Вихід_i$ – підмножина $Стану_i$, елементи яких пов'язані із середовищем.

$Процес_i$ – автономний метод, що виконує відповідні зміни над *Станом*. «Автономний метод» означає, що даний метод викликається без будь-якого зовнішнього впливу.

Середовищем являється множина з двох елементів:

$$Середовище = <Стан_c, Процес_c>,$$

де індекс *c* інформує про те, що дані елементи відносяться до середовища, а не до будь-якого іншого агента.

Важливою особливістю такого подання *Середовища* є те, що *Середовище* є саме по собі активним, оскільки воно містить свій власний $Процес_c$, котрий може змінювати $Стан_c$ незалежно від агентів, що входять в це *Середовище*.

$Вхід_i$ та $Вихід_i$ різноманітних агентів пов'язані з елементами $Стану_c$, але середовище не відрізняє, які з елементів $Стану_c$ знаходяться в залежності з ними. Відсутність *Входу* та *Виходу* у *Середовища* означає, що воно, на відміну від агентів, є необмеженим. Якби в *Середовище* були додані *Вхід* та *Вихід*, то це означало б, що середовище обмежене та є фактично агентом високого рівня. Таким чином, може моделюватися взаємодія агентів високого та низького рівнів.

У відповідності до наведеної концепції подання мультиагентних систем можуть створюватися різноманітні типи мультиагентних систем, при цьому така концепція організації систем може бути використана не тільки в програмних (штучних) системах, а й в природних системах.

До мультиагентних розподілених систем штучного інтелекту відносяться мультиагентні методи інтелектуальної оптимізації (методи колективного інтелекту, Swarm Intelligence). Дані методи мають біонічну природу, тобто вони засновані на моделюванні поведінки комах, птахів, тварин і т.п., поведінка яких носить колективний характер, за рахунок чого досягається, так званий, колективний інтелект.

Тому для визначення основних принципів колективного інтелекту досить розглянути принципи поведінки різних колективних комах. До колективних комах відносяться комахи, які живуть колоніями, наприклад, мурахи, бджоли, терміти, деякі види ос і т. п. В основі поведінки колоній таких комах лежить самоорганізація.

Самоорганізація – множина динамічних механізмів, відповідно до яких система регулюється на глобальному рівні за рахунок взаємодії її компонентів на нижньому рівні без прямої взаємодії між цими компонентами. Самоорганізація базується на чотирьох складових.

1. **Позитивний зворотний зв'язок**: досягається за рахунок виконання простих поведінкових емпіричних прийомів, які забезпечують створення рішень. Наприклад, при фуражировці мурах збільшення феромонів на шляху до джерела їжі – це різновид позитивного зворотного зв'язку, оскільки в такий спосіб створюється мережа з можливих шляхів до джерела їжі. У бджіл позитивний зворотний зв'язок полягає в тому, що бджоли, ґрунтуючись на отриманій інформації від інших бджіл, починають летіти до зазначеного джерела ресурсів.

2. **Негативний зворотний зв'язок**: урівноважує позитивний зворотний зв'язок, що веде до стабілізації колективної поведінки. При фуражировці мурах негативний зворотний зв'язок може бути викликаний такими факторами: обмежена кількість фуражирів, виснаження джерела їжі, випаровування феромону. У бджіл негативний зворотний зв'язок полягає в тому, що бджола, ґрунтуючись на інформації, отриманій від інших бджіл, може вирішити, що знайдене нею джерело гірше.

3. **Нестійкість позитивного зворотного зв'язку**: поведінка соціальних комах заснована на стохастичних правилах. Випадкові відхилення в рішеннях – основа для нових рішень. Крім того,

хаотичність може бути корисною, оскільки це дозволяє колонії виявити нові рішення. Наприклад, фуражир, що збився з курсу, може знайти нові, неопрацьовані джерела харчування, і потім привести за собою інших робочих особин для розробки цього джерела їжі.

4. **Вимога множинної взаємодії між особинами**, за рахунок чого досягається поява нових гарних і витривалих рішень. Множинність взаємодії у бджіл полягає в тому, що інформація про джерело ресурсів, знайденого однією бджолою, доступна для всіх інших у вулику за допомогою виконання, так званого, виляючого танцю.

Крім основних складових, самоорганізація характеризується певними властивостями.

1. **Динамічність**: формування рішень вимагає постійної взаємодії між особинами колонії й навколишнім середовищем. Ця взаємодія забезпечує позитивний зворотний зв'язок, за рахунок якого створюються колективні групи й можливості для їхнього проживання, у той час як негативний зворотний зв'язок знижує вплив прямого зворотного зв'язку.

2. **Емерджентність**: властивості систем, що самоорганізуються, є більш складними, ніж властивості окремих агентів, що входять у неї. Такі властивості системи є результатом нелінійної комбінації взаємодій між агентами.

3. **Нелінійні взаємодії між агентами** ведуть також до розгалуження в поведінці систем, що самоорганізуються. Розгалуження (біфуркація) – поява нових стійких рішень внаслідок зміни певних параметрів системи. За рахунок цього досягається якісне поліпшення в колективній поведінці.

4. **Мультистійкість** – для заданих встановлених параметрів система може досягти різних стійких станів в залежності від початкового стану й випадкових відхилень.

Для розуміння колективного інтелекту необхідно визначити функції, які виконують соціальні комахи в процесі вирішення різних задач. Можна виділити чотири функції такого виду: координація, кооперація, колективне прийняття рішень, спеціалізація. Вони не є взаємно виключними, а навпаки – спільно вносять вклад у досягнення поставлених перед колонією цілей.

Координація – відповідна організація задач окремих індивідів у часі й просторі, що дозволяє вирішити виниклу перед колонією проблему. Дана функція приводить до певного просторово-часового розподілу особин, їхніх дій та/або результатів їхніх дій для досягнення поставленої мети.

Кооперація досягається за рахунок того, що особини виконують разом загальну задачу, що не могла б бути вирішена окремим індивідом. Індивіди повинні об'єднувати свої зусилля, щоб успішно вирішити загальну задачу, що перебуває поза межами можливостей окремих індивідів.

Колективне прийняття рішень відноситься до механізмів, які спрацьовують, коли колонія зіштовхується із проблемою вибору. Цей механізм завершується колективним вибором одного з можливих рішень.

Спеціалізація полягає в тому, що різні дії виконуються окремими спеціалізованими групами індивідів.

Таким чином, організація колективної поведінки у соціальних комах може бути розглянута як сукупність чотирьох складових: координація, кооперація, колективне прийняття рішень і спеціалізація. Кожна із цих складових проявляється на колективному рівні за рахунок непрямої взаємодії між особинами. Така взаємодія забезпечує можливість оцінювання зовнішньої інформації відповідно до двох основних напрямків.

1. **Координація й спеціалізація** забезпечують формування просторових, часових і соціальних зв'язків, що виникають у процесі роботи колонії. Координація регулює просторово-часове розміщення індивідів, у той час як за рахунок спеціалізації досягається розподіл їхніх дій.

2. **Кооперація й колективне прийняття рішень** є інструментами, за допомогою яких колонія може реагувати на зміни в навколишньому середовищі. Колективне прийняття рішень надає механізми, які забезпечують рішення колонії, у той час як кооперація надає механізми, які дозволяють вийти за рамки можливості окремих індивідів.

Разом чотири складові в поведінці соціальних комах формують уявлення, що колонія в цілому планує свою роботу для досягнення поставлених цілей.

3.3.2 Мультиагентний метод з непрямим зв'язком між агентами

Мультиагентний метод з непрямим зв'язком між агентами (метод мурашиних колоній) базується на моделюванні взаємодії декількох штучних аналогів мурах, що програмно подаються у вигляді інтелектуальних агентів, які є членами великої колонії. Модельовані агенти, переміщуючись по графу рішень, спільно вирішують проблему й допомагають іншим агентам у подальшій оптимізації рішення. Таким чином, оптимізаційна задача вирішується агентами, що знаходяться у непрямому зв'язку одне з одним. У методі мурашиних колоній такий зв'язок забезпечується завдяки моделюванню виділення феромонів агентам при переміщенні.

Нехай кількість феромонів $\tau_{ru}(t)$, що залишає агент, відповідає дузі (r, u) – це кількість, що характеризує перевагу вибору даного ребра в порівнянні з іншими при переміщенні. Інформація про феромони на ребрах змінюється в процесі складання рішень. При цьому кількість феромонів, що залишається агентами, пропорційна якості рішення, сформованого відповідним агентом: чим менше шлях, тим більше буде залишено феромону, і навпаки, чим довше шлях, тим менше буде залишено феромону на відповідних ребрах. Такий підхід дозволяє забезпечити безпосередній пошук у напрямку знаходження кращого рішення.

Пам'ять про вузли, які були відвідані агентом, забезпечується шляхом введення, так званого списку табу *tList* – у ньому зберігається бітовий масив, за допомогою якого визначаються відвідані й не відвідані вузли. Таким чином, агент повинен проходити через кожний вузол тільки один раз. Вузли в списку «поточної подорожі» *Path* розташовуються в тому порядку, у якому агент відвідував їх. Пізніше список використовується для визначення довжини шляху між вузлами.

Метод мурашиних колоній включає такі **основні кроки**.

Крок 1. Задати параметри методу: α – коефіцієнт, що визначає відносну значущість шляху; β – параметр, що показує значимість відстані; ρ – коефіцієнт кількості феромону, що агент залишає на шляху, де $(1–\rho)$ показує коефіцієнт випаровування феромону на шляху після його завершення; Q – константа, що відноситься до кількості феромону, що було залишено на шляху;

startPheromone – початкове значення феромону, що знаходиться на шляхах до початку моделювання.

Крок 2. Ініціалізація методу. Створення популяції агентів. Після створення популяції агенти рівномірно розподіляється у вузлах мережі. Необхідно розподіляти агенти рівномірно між вузлами, щоб всі вузли мали однакові шанси стати відправною точкою. Якщо всі агенти почнуть рух з однієї точки, це б означало, що дана точка вважається оптимальною для старту, а насправді вона такою може й не бути. Але при цьому, якщо кількість агентів не кратна кількості вузлів, то кількість агентів у вузлах буде різною, але ця різниця не повинна перевершувати 1.

Крок 3. Рух агентів. Якщо агент ще не закінчив шлях, тобто не відвідав всі вузли мережі, для визначення наступного ребра шляху розраховується ймовірність переходу в *u*-ий вузол, коли агент перебуває в *r*-му вузлі, за формулою:

$$P_{ru} = \frac{\tau_{ru}(t)^{\alpha} \cdot \eta_{ru}(t)^{\beta}}{\sum_{k \in J} \tau_{rk}(t)^{\alpha} \cdot \eta_{rk}(t)^{\beta}} > rand(1),$$

де P_{ru} – імовірність того, що агент переміститися в *u*-ий вузол з *r*-го вузла; *rand*(1) – випадкове число в інтервалі (0; 1); J – множина вузлів, ще не відвіданих агентом; $\tau_{ru}(t)$ – інтенсивність феромону на ребрі між вузлами *r* та *u* у момент часу *t*; $\eta_{ru}(t)$ – функція, що репрезентує вимір зворотної відстані для ребра.

Агент переміщується тільки по тим вузлам, які ще не були відвідані (відзначені списком табу *tList*). Тому ймовірність розраховується тільки для ребер, які ведуть до ще не відвіданих вузлів.

Крок 3 повторюється доти, поки кожний агент не завершить шлях. Цикли заборонено, оскільки в метод включений список табу *tList*.

Крок 4. Після завершення переміщень агентів може бути підрахована довжина шляху. Вона дорівнює сумі всіх ребер, по яких подорожував агент. Кількість феромону, що була залишена на кожному ребрі шляху *i*-го агента, визначається за формулою:

$$\Delta\tau^{i}(t) = \frac{Q}{L^{i}(t)},$$

де $\Delta\tau^{i}(t)$ – кількість феромону, що залишив *i*-ий агент; $L^{i}(t)$ – довжина шляху *i*-го агенту.

Результат є засобом виміру шляху: короткий шлях характеризується високою концентрацією феромону, а більш довгий шлях – більш низькою. Потім отриманий результат використовується для збільшення кількості феромону уздовж кожного ребра пройденого *i*-им агентом шляху за формулою:

$$\tau_{ru}(t) = \tau_{ru}(t-1) + \rho \cdot \sum_{i=1}^{N_{ru}} \Delta\tau^{i}(t),$$

де *r*, *u* – вузли, що утворюють ребра, які відвідав *i*-ий агент; N_{ru} – загальна кількість агентів, що відвідали ребро *ru*.

Дана формула застосовується до всього шляху, при цьому кожне ребро позначається феромоном пропорційно довжині шляху. Тому варто дочекатися, поки агент закінчить переміщення і тільки потім оновити рівні феромону, у протилежному випадку справжня довжина шляху залишиться невідомою. Константа ρ приймає значення між 0 та 1.

На початку шляху в кожного ребра є шанс бути обраним. Щоб поступово видалити ребра, які входять у гірші шляхи в мережі, до всіх ребер застосовується процедура випару феромону. Використовуючи константу ρ з попереднього виразу, можна скласти формулу:

$$\tau_{ru}(t) = \tau_{ru}(t) \cdot (1-\rho).$$

Крок 5. Перевірка на досягнення оптимального результату. Перевірка може виконуватися відповідно до обмеження на максимальну кількість ітерацій або перевірка може вважатися успішною, коли протягом декількох ітерацій не було відзначено змін у виборі найкращого шляху. Якщо перевірка дала позитивний результат, то відбувається закінчення роботи методу (перехід до кроку 7), у протилежному випадку – перехід до кроку 6.

Крок 6. Повторний запуск. Після того як шляхи агентів завершено, ребра оновлено відповідно до довжин шляхів й відбувся випар феромону на всіх ребрах, метод виконується повторно. Список табу очищається, довжини шляхів онуляються. Перехід до кроку 3.

Крок 7. Зупинення. Визначається кращий шлях, що і є рішенням.

Переваги та недоліки методу. Ґрунтуючись на розглянутих основних принципах методу мурашиних колоній, його різновидах та областях застосування можна виділити переваги й недоліки методу мурашиних колоній.

До переваг методу мурашиних колоній можна віднести:

– можливість використання методу в динамічних додатках (адаптуються до змін навколишнього середовища);

– використання пам'яті всієї колонії, що досягається за рахунок моделювання виділення феромонів;

– гарантування збіжності до оптимального рішення;

– стохастичність, тобто випадковість пошуку, за рахунок чого виключається можливість зациклення в локальному оптимумі;

– мультиагентність;

– вища швидкість знаходження оптимального рішення, ніж у традиційних методів;

– можливість застосування до множини різних задач оптимізації.

Можна виділити такі недоліки методу мурашиних колоній:

– складність теоретичного аналізу, оскільки підсумкове рішення формується в результаті послідовності випадкових рішень; розподіл ймовірностей змінюється по ітераціях; дослідження є більш експериментальними, ніж теоретичними;

– невизначеність часу збіжності при тому, що збіжність гарантується;

– висока ітеративність методу;

– сильна залежність результатів роботи методу від початкових параметрів пошуку, які підбираються експериментально.

Виходячи з наведених галузей застосування, переваг та недоліків методу мурашиних колоній можна зробити висновок, що метод рекомендовано використовувати при вирішення дискретних оптимізаційних задач та в динамічних застосуваннях, оскільки він здатен пристосовуватися до змін навколишнього середовища. Варто також відзначити, що ефективність методу є більшою у випадках, коли задача характеризується великою розмірністю, оскільки традиційні методи зазвичай дають гарні результати, якщо розмірність є малою, а от у випадках з великою розмірністю вони можуть або зациклюватися в локальних оптимумах,

або занадто довго працювати, що є небажаним при вирішенні практичних задач.

Не рекомендується застосовувати базовий метод мурашиних колоній до вирішення задач безперервної оптимізації, проте, оскільки метод характеризується гарною розширюваністю, то в нього можна легко вводити додаткові процедури, ідеї яких взяті з інших методів, заснованих на імовірнісному підході, внаслідок чого створюються гібридні системі на основі методу мурашиних колоній для вирішення задач й безперервної оптимізації.

3.3.3 Мультиагентний метод з прямим зв'язком між агентами

Однією з задач, що часто виникають у процесі моделювання складних об'єктів і систем, є знаходження глобального оптимуму багатовимірної функції. Для вирішення цієї задачі застосовується ряд методів (наприклад, методи Коші, Ньютона, Левенберга-Марквардта й т. п.), які, однак, вимагають безперервності, диференційованості та унімодальності цільових функцій. Тому пропонується застосовувати мультиагентний метод з прямим зв'язком між агентами (метод бджолиної колонії) для вирішення задачі оптимізації багатовимірної функції, оскільки він не накладає ніяких вимог до цільової функції.

Запропонований метод можна подати як послідовність таких кроків.

Крок 1. Задаються початкові параметри роботи методу: початкова кількість агентів-розвідників B_s, максимальна енергія, коефіцієнт α, початкова температура T_{init}, кінцева температура T_{final}, максимальна кількість ітерацій $iter_{\max}$. Також задаються функція f, що оптимізується, та параметри, пов'язані з нею: кількість змінних *argCnt*; мінімальна $Range^{\min}$ і максимальна $Range^{\max}$ межі для кожної зі змінних, тобто область, у якій можна шукати можливі рішення; напрямок оптимуму *optOrient* (визначає максимізацію або мінімізацію функції); у випадку використання локальної оптимізації задається один із традиційних методів локальної оптимізації, що буде використовуватися, і задається клас агентів, до рішень яких буде застосовуватися локальна оптимізація – агент із кращим рішенням або агенти, що виконали моделювання виляючого танцю.

Крок 2. Створюються початкові агенти-розвідники.

Крок 2.1. Для кожного початкового агента-розвідника створюється випадкове рішення:

$$bee_i.x_j = rand \cdot (Range_j^{\max} - Range_j^{\min}) + Range_j^{\min},$$
$$\forall i = \overline{1, B_s}, j = \overline{1, argCnt},$$

де bee_i – i-ий агент-розвідник; *rand* – випадкове число в інтервалі [0; 1].

Крок 2.2. Для отриманих випадкових рішень розраховується корисність даного джерела нектару як значення оптимізованої функції:

$$bee_i.profitability = optOrient \cdot f(bee_i.x_1, ..., bee_i.x_{argCnt}), \forall i = \overline{1, B_s},$$

де *OptOrient* = 1, якщо виконується максимізація, у протилежному випадку – встановлюють: *OptOrient* = –1.

Встановлюється поточна кількість ітерацій: *iter*=1; встановлюється кількість агентів-розвідників: $B=B_s$; встановлюється поточна температура: $T=T_{init}$.

Крок 3. Обираються робочі агенти, тобто такі агенти, на базі яких будуть створюватися нові агенти за допомогою процедури схрещування.

Крок 3.1. Визначається агент *best* з найбільшою корисністю.

Крок 3.2. Процедура імітації відпалу. Агент відноситься до робочих агентів *workBee*, якщо виконується умова:

$$\exp\left(-\frac{|bee_i.profitability - best.profitability|}{T}\right) > rand, \forall i = \overline{1, B}.$$

Крок 4. Схрещування. Оскільки реальні бджоли-розвідники при виборі джерела нектару користуються також генетичним матеріалом (у біології ще не вивчено, яким саме чином розвідники обирають одні квітки й пропускають інші, тобто вважається, що розвідники ґрунтуються на генетичному досвіді), то за допомогою процедури схрещування моделюється саме цей момент поведінки бджіл. Для схрещування використовуються раніше відібрані за допомогою процедури імітації відпалу робочі агенти *workBee* і кращий агент за всі ітерації *best*. Нові агенти створюються у два етапи: на базі рішень робочих агентів і на базі рішення кращого агента.

Шаг 4.1. Створення нових агентів на базі робочих агентів *workBee*:

$$newWorkBee_i.x_j = workBee_i.x_j \pm rand \cdot (workBee_i.x_j - best.x_j), \forall i = \overline{1, B_w},$$

де B_w – кількість робочих агентів *workBee*; знак перетворення «+» або «–» обирається випадковим чином.

Крок 4.2. Створення нових агентів на базі кращого агента *best*:

$$newWorkBee_i.x_j = best.x_j \pm rand \cdot (workBee_i.x_j - best.x_j).$$

Крок 4.3. Для всіх нових агентів виконується корегування отриманих рішень, оскільки отримані в такий спосіб значення змінних можуть виходити за межі [$Range^{\min}$; $Range^{\max}$].

Крок 4.4. Розраховується корисність отриманих рішень:

$$newWorkBee_i.profitability = f(newWorkBee_i.x_1, ..., newWorkBee_i.x_{argCnt}), \forall i = \overline{1, B_n},$$

де B_n – кількість створених при схрещуванні агентів *newWorkBee*.

Крок 4.5 Обирається новий кращий агент *best*.

Крок 5. Моделювання виконання виляючого танцю. До можливого виконання танцю допускаються робочі агенти *workBee*, агенти, створені шляхом схрещування, *newWorkBee*, кращий агент за всі ітерації *best*. Моделювання виконання виляючого танцю відбувається в кілька етапів. У результаті цього моделювання обираються ті агенти, які за рахунок виконання танцю виконають вербування інших агентів для дослідження знайденого ними рішення.

Шаг 5.1. Виконується нормування корисностей агентів, допущених до можливості виконання танцю. При цьому нормування враховує напрямок оптимізації *optOrient*.

Крок 5.2. Додавання шумів до отриманих нормованих корисностей і їхнє корегування:

$$np_i = \begin{cases} 1, & \text{якщо } np_i + w_i > 1; \\ np_i + w_i, & \text{якщо } e_n < np_i + w_i < 1; \\ 0, & \text{якщо } 0 < np_i + w_i < e_n, \end{cases}$$

де np_i – нормована корисність *i-го* агента; w_i – шум у корисності. Шум рівномірно розподілений у межах (–*w*; +*w*). Значення *w*

обирається експериментально (пропонується w = 0,1), e_n – мінімальний поріг корисності. Мінімальний поріг обирається експериментально (пропонується e_n = 0,1).

Крок 5.3. Визначення переваги танцю кожного агента:

$$L_i = \max\{np_i - \eta \cdot \overline{np}, 0\},$$

де L_i – перевага танцю i-го агента; η – коефіцієнт, керуючий впливом величини $\overline{np}$ на L_i; $\overline{np}$ – середнє значення нормованої корисності всіх агентів, які допущені до моделювання виконання танцю:

$$\overline{np} = \frac{1}{B_c}\sum_{i=1}^{B_c} np_i ,$$

де B_c – кількість агентів, допущених до виконання танцю.

Крок 5.4. Вибір тих агентів, які за рахунок виконання танцю виконують вербування інших агентів для дослідження знайденого ними рішення. Вважається, що агент виконав вербування, якщо

$$\frac{L_i}{\beta} > \gamma \cdot \overline{np}, \forall i = \overline{1, B_c} ,$$

де $\beta > 0$ – коефіцієнт, що знижує вплив переваги танцю на ймовірність виконання вербування; $\gamma \in [0; 1]$ – граничний коефіцієнт, що визначає, наскільки вище повинна бути перевага танцю цього i-го агента щодо середньої корисності рішень всіх B_c агентів.

Крок 6. Якщо потрібно проводити локальну оптимізацію для знайдених кращих рішень, тоді виконується локальна оптимізація за допомогою зазначеного методу локальної оптимізації (наприклад, метод Нелдера-Мида, Хука-Дживса, Пауелла та ін.). Залежно від установлених параметрів локальна оптимізація виконується або для рішень тих агентів, які зробили вербування, або тільки для рішення кращого агента.

Крок 7. Обирається агент із кращим рішенням *best*.

Крок 8. Перезапуск агентів. Створюються агенти, які будуть розглядатися як агенти-розвідники для наступної ітерації.

До нових агентів-розвідників будуть відноситися:

– агенти, що виконали вербування, та кращий агент;

– агенти, які стали зайнятими фуражирами внаслідок вербування. Оскільки такі агенти повинні виконувати поліпшене ви-

вчення вже існуючого джерела з нектаром, то при створенні рішень для даних агентів повинні враховуватися рішення агентів, що їх завербували. У зв'язку із цим для завербованих агентів рішення створюється в такий спосіб:

$$x_j = dancedBee.x_j + range \cdot rand - \frac{range}{2}, \forall j = \overline{1, argCnt},$$

де *range* – це межа, у якій величина змінної може відхилятися від значення даної *j*-ої змінної в рішенні агента, що моделював танець, *dancedBee*.

– агенти, рішення яких створюється випадковим чином:

$$x_j = rand \cdot (Range_j^{\max} - Range_j^{\min}) + Range_j^{\min}, j = \overline{1, argCnt}.$$

Також для всіх створених агентів розраховується корисність обраного рішення.

Крок 9. Відновлення динамічних параметрів методу:

– збільшується кількість ітерацій: *iter*=*iter*+1;

– змінюється поточна температура: $T=\alpha \cdot T$;

– змінюється межа *range*:

$$range = range \cdot \frac{iter_{\max} - iter}{iter_{\max}}.$$

Крок 10. Перевірка на зупинення. Перевірка на зупинення вважається успішною, якщо виконується хоча б одна з двох умов:

– досягнуто максимальну кількість ітерацій, тобто $iter = iter_{max}$;

– поточна температура дорівнює кінцевій температурі ($T = T_{final}$).

Якщо перевірка на зупин дала успішний результат, тоді виконується перехід до кроку 11, у протилежному випадку – до кроку 3.

Крок 11. Зупинення.

Виходячи з різних застосувань методу бджолиної колонії можна виділити такі його **переваги**:

– метод є несхильним до зациклення в локальних оптимумах, оскільки заснований на випадковому пошуку;

– мультиагентність реалізації;

– пошук кращого рішення ґрунтується на рішеннях агентів всієї колонії бджіл;

– може використовуватися в динамічних застосуваннях, оскільки здатен адаптуватися до змін навколишнього середовища;

– може використовуватися для вирішення як дискретних, так і безперервних задач оптимізації.

До **недоліків** методу бджолиної колонії можна віднести:

– досить високу ітеративність;

– труднощі теоретичного аналізу процесу одержання рішень, обумовлені тим, що пошук рішення має стохастичну природу;

– апріорну невизначеність часу збіжності, хоча збіжність гарантується;

– залежність методу від настроювання параметрів, що підбирають експериментально.

Метод бджолиної колонії має такі **особливості**:

1) Всі агенти поділяються на різні типи у відповідності з діями, які вони виконують у процесі вирішення задачі.

1.1) Зайняті фуражири забезпечують використання вже знайдених джерел нектару, тобто незначно змінюють уже знайдені раніше рішення.

1.2) Незайняті фуражири забезпечують продовження пошуку нових джерел нектару, тобто агенти такого типу виконують пошук нових припустимих рішень. Незайняті фуражири у свою чергу бувають двох типів.

1.2.1) Спостерігачі – очікують у вулику інших агентів. Вони не виконують ніяких дій, вони фактично очікують моменту, коли їм потрібно буде також почати пошук рішень;

1.2.2) Розвідники – забезпечують пошук нових джерел нектару. При цьому пошук здійснюється випадковим чином, тобто вони випадково обирають у просторі пошуку можливе рішення.

2) Зв’язок між рішеннями агентів здійснюється шляхом моделювання виконання бджолами виляючого танцю. При цьому виконання виляючого танцю забезпечує утворення двох типів зворотного зв’язку: позитивний зворотний зв’язок і негативний зворотний зв’язок.

Позитивний зворотний зв’язок полягає в тому, що агенти, ґрунтуючись на інформації про рішення інших агентів, можуть почати досліджувати рішення, отримане іншим агентом.

Негативний зворотний зв'язок полягає в тому, що агенти, одержавши інформацію про знайдені рішення іншими агентами, можуть прийняти рішення щодо припиненні дослідження свого рішення у зв'язку з гіршими характеристиками в порівнянні з іншими отриманими рішеннями.

3) Процес пошуку рішення забезпечується двома процедурами.

3.1) Пошук нових джерел нектару у всьому просторі пошуку, що досягається за допомогою агентів-розвідників. Таким чином, забезпечується дослідження всього простору пошуку.

3.2) Поглиблене використання областей, у яких розміщено уже знайдені джерела нектару (досягається за допомогою зайнятих фуражирів). Тобто знаходяться рішення, що перебувають у просторі пошуку поблизу від розглянутого рішення.

3.4 Програмні засоби розв'язання оптимізаційних завдань для побудови інтелектуальних моделей

Багатовимірний градієнтний безумовний пошук мінімуму функції $f(w)$, де w – вектор керуючих змінних, а $f(w)$ – функція, що повертає скаляр, здійснюється за допомогою функції fminu:

```
w = fminu('f',w0),
```

яка починаючи з точки `w0` шукає мінімум функції f.

Оптимізацію на основі **еволюційного пошуку** дозволяє виконувати бібліотека Genetic Algorithm and Direct Search Toolbox пакету Matlab, що містить такі компоненти для генетичної оптимізації:

– функції, що реалізують процес виконання генетичного пошуку та основні генетичні оператори;

– візуальний інтерфейсний модуль для роботи з генетичними методами;

– демонстраційні приклади.

Для використання генетичних методів, у середовищі Matlab передбачена функція ga, формат виклику якої наведено нижче:

```
[x, Fmin] = ga(@fitnessfun, n, options).
```

Функція `ga` знаходить мінімум `Fmin` функції `fitnessfun`, що має `n` параметрів, а також вектор `x`, при якому досягається мінімум цільової функції.

Параметри роботи функції `ga` задаються в змінної `options`, що являє собою структуру, у якій можуть бути задані: спосіб подання інформації в хромосомі, використовувані генетичні оператори відбору, схрещування й мутації, критерії зупинення й інші параметри генетичного пошуку. За допомогою структури `options` також можна вибрати графіки, які будуть відображати основні результати генетичного пошуку в процесі оптимізації цільової функції.

Для зміни значень параметрів функції `ga` передбачена функція `gaoptimset`, а для одержання поточних параметрів – функція `gaoptimget`.

Приклад 1. За допомогою пакету Matlab знайти мінімум функції $f(x) = x_1^2 + x_2^2 + 5$, використовуючи методи генетичного пошуку. Вивчити використання різних еволюційних операторів. Застосувати острівну модель еволюційного пошуку, що використовує 20 особин на 1 острові та 30 – на другому

Розв'язок: функція $f(x)$ оформляється в окремому файлі F.m:

```
function mean1 = F(x)
mean1=x(1)*x(1)+x(2)*x(2)+5;
```

Відповідно до завдання для пошуку мінімуму функції $f(x) = x_1^2 + x_2^2 + 5$ за допомогою острівної моделі еволюційного пошуку, що використовує 20 особин на 1 острові та 30 – на другому, у командному рядку пакета Matlab необхідно прописати наступні команди:

```
options = gaoptimset('PopulationSize', [20 30])
[x Fmin] = ga(@F, 2, options)
```

Результат: `x = 0.0004  0.0020`

`Fmin = 5.0000`

Приклад 2. Написати функцію, що налагоджує ваги двошарової нейронної мережі, що містить $k1$ нейронів на першому шарі й $k2$ нейронів – на другому й навчається за вибіркою X (матриця

незалежних змінних) і *Y* (вектор значень цільової функції). Точність апроксимації 0,01. Підрахувати кількість викликів функції помилки.

У випадку, якщо не задані параметри *k*1, *k*2, *X* и *Y*, настроїти нейронну мережу, що містить 4 нейрони на першому шарі й 1 нейрон на другому шарі.

Розв'язок оформимо у вигляді двох файлів NNweights.m і ErrorPodbor.m.

Файл NNweights.m

```
function mean1 = NNweights(k1_, k2_, x1, y1)
global Q k1 k2 net x y %net - двошарова нейрон-
                    на мережа
          %k1 - кількість нейронів на 1-ому шарі
          %k2 - кількість нейронів на 2-ому шарі
if (nargin~=4) % параметри не задані
    k1=4;
    k2=1;
x=[елементи матриця X]';
y=[елементи.масиву Y];
end;
if (nargin==4)       % всі параметри задані
    k1=k1_;
    k2=k2_;
    x=x1;
    y=y1;
end;
%нормализация X и Y
x_min=min(x',[],1);
x_max=max(x',[],1);
for i=1:1:size(x,1)
   for j=1:1:size(x,2)
     x(i,j)=(x(i,j)-x_min(i))./abs(x_max(i)-
x_min(i));
   end;
end;
y_max=max(y);
y_min=min(y);
y=(y-y_min)/abs(y_max-y_min);
```

```
net=newff(repmat([0 1], size(x,1),1),
[k1,k2],{'logsig','logsig'},'trainlm');
Q=0;
[W error] = ga(@ErrorPodbor,
k1*(size(x,1)+1)+k2*(k1+1),gaoptimset('FitnessLimit',
0.01));
Q
error
net.IW{1,1}
net.b{1,1}
net.b{2,1}
net.LW{2,1}
mean1 = W';
```

Файл ErrorPodbor.m

```
function mean1 = ErrorPodbor(w)
global k1 k2 net x y Q  %Q - номер поточної
ітерації
net_IW=[];
for i=1:1:k1
    for j=1:1:size(x,1)
        net_IW(i,j)=w(j+size(x,1)*(i-1));
    end;
end;
net.IW{1,1}=net_IW;
net.IW{1,1};
net_b11=[];
for i=1:1:k1
    net_b11(i)=w(i+size(x,1)*k1);
end;
net.b{1,1} = net_b11';
net_b21=[];
for i=1:1:k2
    net_b21(i)=w(i+size(x,1)*k1+k1);
end;
net.b{2,1} = net_b21;
net_LW=[];
for i=1:1:k2
    for j=1:1:k1
```

```
        net_LW(i,j)=w(j+size(x,1)*(i-
1)+size(x,1)*k1+k1+k2);
    end;
end;
net.LW{2,1} = net_LW;
a=sim(net,x);
for i=1:1:size(y,1)
    E(i)=abs(y(i)-a(i))/(abs(y(i)));
end;
Q=Q+1;
A=[Q mean(E)];
mean1=mean(E); % результат функції - середня
помилка
```

Виклик функції `NNweights` у середовищі MatLab і результати її роботи:

```
    1.  NNweights
    Q = 180         %кількість ітерацій
    error = 0.0019 %похибка
    net.IW{1,1}=
      Columns 1 through 9
-1.3283    2.5324   -1.3129   -0.2808    0.4730
-0.0987   0.3018   1.1025   2.2214
 0.4990   -1.1696    1.4755    0.1822   -0.8203
1.6955  -1.9219  -1.1233   2.8977
-0.4352    1.2547    3.0918   -0.1316    1.6743
-0.6672   2.0966  -2.4275   1.2555
 2.1228    1.0495    3.6832    1.3034   -0.1187
1.1635   0.2265  -0.6895  -2.3788
     Columns 10 through 11
      -3.3060    0.2481
       3.8343   -0.9004
       1.9228   -0.2934
       0.9605    3.0611

   net.b{1,1}=
       0.8495
      -2.8727
```

```
       -2.7475
        1.7031

    net.b{2,1} = -0.6223
    net.LW{2,1}= -1.7539    2.4341    0.9476
-2.3492
    2. NNweights(5, 1, x, y)
    Q = 160
    error = 0.0026
    net.IW{1,1}=
      Columns 1 through 9
-0.9647    1.7090   -0.0529    2.1717    0.3155
1.0761   0.5945  -0.9221   0.0985
 1.4284   -0.6901    0.8076    1.5180    2.9064
2.2507  -0.2081  -0.3386   0.7981
 0.3559  -0.6540    0.3723  -1.5776  -0.2134  -
0.0923  -0.4807   2.2941   0.3423
-0.9890   -0.1484    1.0102    1.1160   -0.9156
1.7254  -0.2030   0.9167   0.5958
-0.4758    0.3028   -0.8149   -0.1103   -3.0851
0.8314   2.3185   0.8609  -0.9077
      Columns 10 through 11
       -2.9822    2.0700
        0.6837   -1.0005
       -0.8365    0.6877
        2.7636    1.6551
        1.9377    0.0296

    net.b{1,1}=
       -0.5353
       -0.5897
        0.7673
        0.3227
       -1.9030
    net.b{2,1} = -1.6559
    net.LW{2,1}= -0.5934  -0.2919  -1.1430   0.5409
1.6481
```

3.5 Контрольні питання

Градієнтні методи

1. Проаналізуйте метод Коші.
2. Поясніть схему роботи методу Ньютона.
3. Наведіть особливості алгоритмів спряжених градієнтів.
4. Порівняйте алгоритми спряжених градієнтів.
5. До оптимізації яких цільових функцій може бути застосований Партан-метод?
6. Проаналізуйте метод Заутендайка
7. Особливості багатопараметричного пошуку.
8. Порівняйте квазіньютонівські методи.
9. Поясніть призначення методу Левенберга-Марквардта. Порівняйте його з іншими градієнтними методами.
10. Наведіть особливості, переваги та недоліки градієнтних методів оптимізації.

Еволюційні методи

11. Порівняйте методи еволюційного пошуку з іншими методами оптимізації. Які методи відносять до еволюційних?
12. В чому переваги еволюційних методів?
13. Проаналізуйте умови ефективного використання методів еволюційного пошуку.
14. Назвіть особливості еволюційних методів.
15. Які недоліки еволюційного пошуку та в чому вони полягають?
16. Проаналізуйте узагальнену схему роботи еволюційних методів.
17. Наведіть послідовність виконання узагальненого еволюційного пошуку.
18. Які параметри необхідно визначати для роботи еволюційних методів?
19. Які існують способи кодування параметрів, що оптимізуються, при використанні еволюційних методів?
20. Що таке фітнесс-функція?

21. Порівняйте стратегії створення початкової популяції.

22. Виконайте порівняльний аналіз операторів відбору (пропорційний відбір, відбір за допомогою ранжирування, турнірний відбір та відбір з використанням порогу).

23. Які способи формування батьківської пари використовуються в еволюційних методах?

24. Проаналізуйте оператори схрещування (точкове, рівномірне, порівняльне, арифметичне).

25. Для чого призначений оператор мутації? Які оператори мутації використовуються в еволюційних методах?

26. Порівняйте класичну із одноточечною мутацією обміну.

27. Яким чином відбувається формування нового покоління?

28. Які критерії зупину використовуються при еволюційному пошуку?

29. Які параметри можна використовувати в функції ga? Яким чином вони задаються? Як отримати поточні параметри функції ga?

30. Проаналізуйте внутрішню структуру функції ga пакету Matlab: основні змінні, параметри, методи та допоміжні функції, їх призначення та використання.

Мультиагентні методи

31. Які методи відносяться до інтелектуальних методів мультиагентної оптимізації, заснованих на моделюванні суспільного інтелекту?

32. Назвіть елементи, які використовуються при описі мультиагентних систем.

33. Яким чином забезпечується зв'язок Агентів з Середовищем?

34. Порівняйте агентно-орієнтоване та об'єктно-орієнтоване програмування.

35. Що таке самоорганізація агентів? На яких чотирьох складових вона базується? Якими властивостями характеризується самоорганізація агентів?

36. Проаналізуйте функції, що виконують агенти в процесі пошуку оптимального розв'язку: координація, кооперація, колективне прийняття рішень, спеціалізація.

37. Наведіть переваги та недоліки мультиагентних систем.

38. В чому полягає основна ідея мультиагентного методу з непрямим зв'язком між агентами (методу мурашиних колоній)?

39. Для розв'язання яких задач використовується метод мурашиних колоній?

40. В який умовах особливо ефективне використання мурашиних алгоритмів?

41. Порівняйте метод мурашиних колоній з іншими оптимізаційними методами.

42. Що являє собою навколишнє середовище в методі мурашиних колоній?

43. З якою метою використовується список табу?

44. Яким чином розташовуються вузли в списку «поточної подорожі»?

45. Що таке феромон? Яке його призначення в методі мурашиних колоній?

46. Проаналізуйте послідовність виконання методу мурашиних колоній.

47. Як розраховується кількість феромону, що було залишено на кожній грані шляху і-го агенту?

48. Проаналізуйте засоби пакету Matlab, які можуть бути використані при розробці програмного забезпечення, що реалізує метод мурашиних колоній.

49. В чому полягає основна ідея мультиагентного методу з прямим зв'язком між агентами (метод бджолиної колонії)?

50. Наведіть особливості, переваги та недоліки мультиагентного методу з прямим зв'язком між агентами.

3.6 Практичні завдання

 *Завдання 1.* Відповідно до номеру варіанту за допомогою бібліотеки Optimization Toolbox пакету Matlab знайти мінімум функції $y(x_1,x_2)$ в осередку точки $x^{(0)}$ на основі градієнтних методів пошуку (табл. 3.1).

Таблиця 3.1 – Індивідуальні завдання

№ варіанту	Цільова функція	Початкова точка пошуку
1	$y = (1 - x_1)^2 + (2 - x_2)^2$	$x^{(0)} = [0, 0]^T$
2	$y = (5 - 2x_1)^8 + (6 - 3x_2)^4$	$x^{(0)} = [3, 2]^T$
3	$y = (31 - 8x_1)^6 + (2 - 3x_2)^2$	$x^{(0)} = [1, -1]^T$
4	$y = 9 - 2(5x_1 + 2x_2) + x_1^2 + x_2^2$	$x^{(0)} = [0{,}2, 0{,}1]^T$
5	$y = (1 - x_1)^2 + (2 - x_2)^2 - 0{,}01x_1x_2$	$x^{(0)} = [-1, 0]^T$
6	$y = (10 - 2x_1)^2 + (12 - 5x_2)^4$	$x^{(0)} = [0{,}3, 0{,}5]^T$
7	$y = (8 - x_1)^2 - (7 - x_2)^2 + 3x_2^4$	$x^{(0)} = [-3, 5]^T$
8	$y = 19 - 15x_1 - 8x_2 + 3x_1^2 + x_2^2$	$x^{(0)} = [0, 0]^T$
9	$y = x_1^6 - (1 - x_1)^2 + (2 - x_2)^2$	$x^{(0)} = [10, 20]^T$
10	$y = 2x_1^2 + 4x_1x_2^3 - 10x_1x_2 + x_2^3$	$x^{(0)} = [0, 0]^T$
11	$y = 6x_1^4 + 8x_1x_2^6 - 13x_1x_2 + 4x_2^3$	$x^{(0)} = [3, 7]^T$
12	$y = 9 - 25x_1 + x_1^2 - 22x_2 + x_2^2$	$x^{(0)} = [0, 1]^T$
13	$y = 22\lvert x_1\rvert^7 + 24x_1^3x_2^6 - x_1x_2 + x_2^3$	$x^{(0)} = [-12, 17]^T$
14	$y = (1 - x_1)^2 + (2 - x_2)^2 - 3x_1x_2$	$x^{(0)} = [-6, 7]^T$
15	$y = x_1^2 - x_1^3x_2^2 - 9x_1x_2 + x_2^3$	$x^{(0)} = [20, -10]^T$
16	$y = 18 - 20x_1 - 8x_2 + 2x_1^2 + 2x_2^2$	$x^{(0)} = [0, 0]^T$
17	$y = (1 - x_1)^2 + (2 - x_2)^2 - x_1 - x_2$	$x^{(0)} = [-1, 6]^T$
18	$y = 3 - 3{,}3x_1 - 1{,}1x_2 + 3x_1^2 + 4x_2^2$	$x^{(0)} = [0, 0]^T$
19	$y = (1 - x_1)^2 + (2 - x_2)^2 - x_1^{-1}x_2^2$	$x^{(0)} = [-1, 0]^T$
20	$y = 4x_1^2 + 3x_2^2 - 4x_1x_2 + x_1$	$x^{(0)} = [5, 3]^T$
21	$y = 2x_1^2x_2^2 - 40x_1x_2 + x_1^3$	$x^{(0)} = [8, -7]^T$
22	$y = 5x_1^3 + 8x_2^6 - 4x_1^3x_2 + 5x_2$	$x^{(0)} = [2, 17]^T$
23	$y = 8x_1^6 + 33x_2^6 - 24x_1^3x_2^3 + x_1^9$	$x^{(0)} = [3, -6]^T$
24	$y = 6x_1^{-2} + 2x_2^8 - 6x_1x_2 + 8x_2^4$	$x^{(0)} = [-1, 2]^T$
25	$y = (3 - x_1)^2 + (4 - x_2)^2 - x_1 - x_2$	$x^{(0)} = [-3, 12]^T$

За допомогою функцій тривимірної графіки пакету Matlab побудувати графsк функції у осередку знайденої точки мінімуму. На побудованому графіку зобразити початкову точку $x^{(0)}$ та знайдену точку мінімуму x.

Порівняти різні градієнтні методи пошуку за такими критеріями, як час пошуку, кількість ітерацій, точність (помилка) визначення точки оптимуму.

Завдання 2. Розробити за допомогою пакету Matlab програму, що оптимізує функцію $f(x_1,x_2) = x_1 \sin(4x_1) + 1{,}1x_2 \sin(2x_2)$, використовуючи еволюційний пошук. В якості еволюційних операторів обрати: відбір – пропорційний, схрещування – однорідне, мутація – гауссовська; розмір популяції – 200 особин; кількість елітних особин – 3.

Як аргумент P_c програма приймає ймовірність схрещування. При цьому програма повинна перевіряти коректність введених даних, у випадку некоректності яких повинно виводити відповідне повідомлення. Передбачити ситуацію виклику розробленої програми без параметрів (встановити значення P_c за замовчанням 0,75).

Застосувати гібридну модель, що використовує результат еволюційного пошуку як початкову точку для оптимізації за допомогою функції patternsearch.

Інші параметри еволюційного пошуку обрати самостійно. Вибір параметрів обґрунтувати.

Завдання 3. За допомогою розробленої у попередньому завданні програми дослідити ефективність еволюційного пошуку для оптимізації заданої функції. Для цього провести серію зі 100 експериментів з фіксованими параметрами еволюційної оптимізації та отримати середнє, середньоквадратичне відхилення, мінімальне та максимальне значення отриманого оптимуму цільової функції та часу оптимізації.

Завдання 4. Дослідити еволюційні оператори схрещування та мутації. Отримати нові хромосоми, застосувавши еволюційні оператори схрещування (одноточкове, двоточкове, однорідне, рівномірне, порівняльне, арифметичне, лінійне арифметичне) та мутації (проста мутація, випадкова мутація) до заданих хромосом (табл. 3.2). Завдання виконати вручну. Описати хід розв’язання.

Таблиця 3.2 – Значення хромосом для дослідження еволюційних операторів

№ варіанту	Схрещування				Мутація
	одноточкове, двоточкове, однорідне, рівномірне, порівняльне		арифметичне, лінійне арифметичне		проста мутація, випадкова мутація
	H_1	H_2	H_1	H_2	H
1	00001000	01011011	57358612	12053461	67545338
2	11001001	10011100	86157584	51967592	62423846
3	00001010	01011101	62264572	15198476	65434724
4	11001011	10011110	14377256	72205273	56745121
5	00001100	01011111	23483445	38314635	24556343
6	11001101	10111111	72598642	24424175	97367252
7	00001110	01111110	51603457	32536371	46978363
8	11001111	10111101	27715721	13642673	85289361
9	00011000	01111100	83822574	75755243	34590314
10	11011001	10111011	55935813	37865312	65413452
11	00011010	01111010	40164735	82975372	48834136
12	11011011	10111001	75275932	83108371	95545461
13	00011100	01111000	68387905	18215167	47756352
14	11011101	10101111	85499053	94328362	42267125
15	00011110	01011011	32507452	82431636	37878432
16	11011111	10011100	15615832	52544326	51489264
17	01111111	00101000	64724831	78654714	48390573
18	10111110	11101001	53831357	37763715	46814878
19	01111101	00101010	11947532	18874732	52525987
20	10111100	11101011	35175625	18986836	72536085
21	01111011	00101100	64284271	15115278	58947874
22	10111010	11101101	75391378	18229254	24858674
23	01111001	00101110	85405216	18330860	57469573
24	10111000	11101111	56516424	38440978	47270584
25	01101111	00111000	76624624	42550978	18481573

Вважати, що гени числових хромосом приймають значення в інтервалі [0; 9]. Інші параметри, необхідні для виконання еволюційних операторів, обрати самостійно. Вибір параметрів обґрунтувати.

Завдання 5. Розробити програму, що за допомогою еволюційного пошуку реалізує один з етапів побудови нейромережевих моделей:

– відбір інформативних ознак;

– параметричний синтез;

– структурний синтез;

– оптимізація структури.

Виконати тестування розробленої програми, побудувавши нейромережеві моделі для різних вибірок даних.

Порівняти одержані результати синтезу нейронних мереж. Результати порівняльного аналізу звести до таблиці, попередньо розробивши відповідну систему критеріїв.

Завдання 6. Розробити за допомогою пакету Matlab програмне забезпечення, що реалізує мультиагентний метод з непрямим зв'язком між агентами (метод мурашиних колоній). При програмній реалізації методу дотримуватися вимог, наведених в п.1–4.

1. Для гнучкості програмного забезпечення всі параметри методу (наприклад, кількість агентів, коефіцієнт випаровування і т. ін.) зберігати в змінних або константах.

2. Для реалізації агентів, що моделюють поведінку мурах, використовувати структуру, що має наступні поля:

– поточна позиція, в якій знаходиться агент;

– наступна позиція;

– список табу (бітовий рядок), в якому зберігаються ті пункти, в яких вже побував агент;

– кількість пунктів, що вже відвідав агент;

– масив подорожі, де зберігається послідовність пунктів, в яких побував агент;

– довжина шляху, що пройшов агент. Розраховується після закінчення подорожі.

3. Виділити основні етапи методу в окремі функції, що реалізуються у відповідних *.m*-файлах:

– ініціалізація методу;

– моделювання переміщення агентів;

– вибір наступного пункту;

– оновлення феромонів на шляхах;

– перезавантаження агентів.

Також необхідно реалізувати окрему функцію розрахунку довжини шляху.

4. Передбачити можливість наглядного (графічного) відображення змін у поточних результатах роботи методу.

Виконати тестування розробленого програмного забезпечення за допомогою вирішення конкретних прикладів задачі комівояжера. Задачі (не менше трьох) для виконання тестування програми сформулювати самостійно. Вибір тестових задач обґрунтувати.

Порівняти одержані результати вирішення різних прикладів задачі комівояжера. Результати порівняльного аналізу звести до таблиці, попередньо розробивши систему критеріїв порівняння результатів вирішення задачі комівояжера.

3.7 Тестові завдання

Т *Завдання 1.* В основі градієнтних методів оптимізації інтегрального критерію лежить …

А паралельне наближення із використанням простих ітерацій.

Б використання інформації про кривизну цільової функції $F(x)$.

В використання градієнта цільової функції $F(x)$.

Г перебирання вершин опуклого багатогранника.

Д мінімізація функції за певною координатою на кожній ітерації.

Е Серед відповідей А–Д правильної немає.

Т *Завдання 2.* Що означає α_k в формулі ітераційної процедури оптимізації $w_{k+1} = w_k + \alpha_k s(w_k)$?

А Поточне приближення значень керованих змінних до оптимального рішення.

Б Крок збіжності.

В Нове приближення значень керованих змінних до оптимального рішення.

Г Напрямок пошуку в k-вимірному просторі керованих змінних.

Д Припустима помилка обчислення.

Е Серед відповідей А–Д правильної немає.

Т *Завдання 3.* Метод Коші полягає в реалізації правила…

А $w_{k+1} = -g_k \cdot w_k$.

Б $w_{k+1} = w_k - \gamma_k g_k$.

В $w_k = -\gamma_{k+1} w_{k+1}$.

Г $w_{k+1} = -g_k + w_k$.

Д $w_k = w_{k+1} + \gamma_k g_k$.

Е Серед відповідей А–Д правильної немає.

Т *Завдання 4.* Який вигляд має матриця других похідних мінімізуючого фукціоналу в методі Ньютона?

А $\sum_{m=1}^{k} \frac{\partial}{\partial w} \left(\frac{\partial (Q(m))}{\partial w} \right)$.

Б $\prod_{m=1}^{k} \frac{\partial}{\partial w} \left(\frac{\partial (Q(m))}{\partial w} \right)$.

В $\sum_{m=1}^{k} \left(\frac{\partial (Q(m))}{\partial w} \right)^T$.

Г $\sum_{m=1}^{k} \frac{\partial}{\partial w} \left(\frac{\partial (Q(m))}{\partial w} \right)^T$.

Д $\prod_{m=1}^{k} \frac{\partial}{\partial w} \left(\frac{\partial (Q(m))}{\partial w} \right)^T$.

Е Серед відповідей А–Д правильної немає.

Т *Завдання 5.* Яке з тверджень про алгоритм Левенберга-Марквардта є вірним?

А Алгоритм вимагає наявності інформації про треті похідні цільової функції.

Б В алгоритмі використовується метод Ньютона.

В Алгоритм вимагає наявності інформації про другі похідні цільової функції.

Г В алгоритмі використовується метод Коші.
Д Вірними є відповіді А та Б.
Е Вірними є відповіді В та Г.

Т *Завдання 6.* Яке з наступних тверджень є невірним?

А Еволюційні методи є методами багатовимірного пошуку.
Б Еволюційні методи основані на аналогії з природними процесами селекції та генетичними перетвореннями.
В Еволюційні методи є методами локального пошуку.
Г Еволюційні методи можуть використовуватися для оптимізації одновимірних функцій.
Д Еволюційні методи не використовують значення похідних цільової функції.
Е Серед відповідей А–Д усі вірні.

Т *Завдання 7.* Яке з тверджень є невірним?

А Суть генетичного пошуку полягає в циклічній заміні однієї популяції наступною, більш пристосованою.
Б Початкова популяція створюється на етапі ініціалізації генетичного пошуку.
В Робота генетичного пошуку представляє собою послідовний ітеративний процес виконання генетичних операторів схрещування, відбору та мутації.
Г Генетичні оператори необхідні для того, щоб застосовувати принципи спадковості й мінливості до популяції.
Д Генетичні оператори мають властивості імовірності.
Е Всі перелічені відповіді А–Д є вірними.

Т *Завдання 8.* Серед відповідей А–Д виберіть вірну.

Основний цикл роботи генетичного методу представляє собою ітеративний процес ...

А виконання операторів ініціалізації, відбору, схрещування та мутації.
Б виконання операторів відбору, схрещування та мутації.

В послідовного виконання операторів ініціалізації, схрещування та мутації.

Г послідовного виконання операторів схрещування, відбору, та мутації.

Д послідовного виконання операторів ініціалізації, відбору та мутації.

Е серед відповідей А–Д вірної не має.

Т *Завдання 9.* За методами представлення генів хромосоми можна розподілити на:

А бінарні, числові, векторні.

Б бінарні, дискретні, неперервні.

В числові, векторні, матричні.

Г гомологічні, негомологічні, векторні.

Д бінарні, векторні, матричні.

Е серед відповідей А–Д правильної не має.

Т *Завдання 10.* Виберіть невірне твердження.

А Занадто велика довжина кодування прискорює процес збіжності до кращого рішення.

Б У випадку передчасної збіжності досліджується більша частина простору пошуку.

В Передчасна збіжність є небажаним ефектом при еволюційному пошуку.

Г Передчасна збіжність може не привести до оптимального рішення.

Д Швидка збіжність до однієї області не гарантує виявлення декількох рівних екстремумів.

Е Всі відповіді А–Д є правильними.

Т *Завдання 11.* Визначте кількість генів хромосоми при розв’язанні задачі апроксимації з використанням функції вигляду

$f = a^{x_1 x_2} + \dfrac{x_3^2}{x_1 + b}$, де x_1, x_2, x_3 – незалежні змінні апроксимуючої функції; a, b – параметри апроксимуючої функції.

А 1. Б 2. В 3.
Г 4. Д 5. Е Інша відповідь.

Т *Завдання 12.* Визначте розрядність бінарної хромосоми, необхідну для кодування двох ознак, перша з яких приймає значення в інтервалі [0,177; 0,689), а друга – в інтервалі [0,582; 0,646). Похибка кодування ознак в хромосомі ε = 0,001.

А 12. Б 13. В 14.
Г 15. Д 16. Е Інша відповідь.

Т *Завдання 13.* У чому полягає принцип елітизму в еволюційному пошуку?

А Нова популяція формується з елітних хромосом.
Б До нової популяції гарантовано переходять особини, що мають найбільшу кількість нащадків.
В Особини з найбільшою пристосованістю гарантовано переходять в нову популяцію.
Г В схрещуванні гарантовано беруть участь хромосоми з найбільшою пристосованістю.
Д Найбільш пристосовані особини гарантовано беруть участь у схрещуванні і мутації.
Е Серед відповідей А–Д правильної не має.

Т *Завдання 14.* Якого оператору відбору не існує?

А Пропорційний.
Б Одноточковий.
В Турнірний.
Г Відбір ранжируванням.
Д Відбір з використанням порогу.
Е Існують усі види операторів відбору, наведені у відповідях А–Д.

Т *Завдання 15.* Яка з хромосом є результатом одноточкового схрещування хромосом $H_1 = (11010101)$ та $H_2 = (11000100)$. Точка схрещування – після 5-го гену.

А $H_3 = (01111111)$.
Б $H_4 = (10010000)$.
В $H_5 = (00000010)$.
Г $H_6 = (11010100)$.
Д $H_7 = (11110110)$.
Е Серед відповідей А–Д вірної не має.

Т *Завдання 16.* Які хромосоми серед $H_3 – H_9$ ($H_3 = (11001100)$, $H_4 = (01010010)$, $H_5 = (10110001)$, $H_6 = (11001011)$, $H_7 = (10111011)$, $H_8 = (01000100)$, $H_9 = (00101110)$) є результатом двоточкового схрещування хромосом $H_1 = (11000100)$ та $H_2 = (01001100)$. Точки схрещування – після 3-го та 6-го генів.

А H_4 та H_7. Б H_3 та H_8. В H_5 та H_6. Г H_4 та H_9.
Д H_4 та H_8. Е Серед відповідей А–Д вірної не має.

Т *Завдання 17.* Нехай випадковим чином згенеровано набір чисел (0,6 0,3 0,1 0,7 0,4 0,4 0,8 0,8). Яка з хромосом $H_2 – H_8$ ($H_2 = (10101010)$, $H_3 = (10111110)$, $H_4 = (00001101)$, $H_5 = (11110110)$, $H_6 = (00000011)$, $H_7 = (01001010)$, $H_8 = (11000010)$) є результатом однорідного схрещування хромосом $H_0 = (01011011)$ та $H_1 = (01000010)$.

А H_4. Б H_5. В H_6. Г H_7.
Д H_8. Е Серед відповідей А–Д вірної не має.

Т *Завдання 18.* До інтелектуальних методів мультиагентної оптимізації, заснованих на моделюванні суспільного інтелекту, не відноситься …

А метод мурашиних колоній (Ant Colony Optimization, ACO).
Б метод бджолиної колонії (Bee Colony Optimization, BCO).
В оптимізація за допомогою рою часток (Particle Swarm Optimization, PSO).

Г еволюційні методи (Evolutionary Methods, EA).
Д Вірними є відповіді В та Г.
Е Серед відповідей А–Г вірної не має.

Т *Завдання 19.* З яких елементів в загальному випадку складається мультиагентна система?
А Агенти, Середовище, Зв'язки між Середовищем та Агентами.
Б Хромосоми, Середовище, Зв'язки між Середовищем та Хромосомами.
В Агенти, Зв'язки, Середовище між Зв'язками та Агентами.
Г Агенти, Хромосоми, Зв'язки між Хромосомами та Агентами.
Д Агенти, Хромосоми, Зв'язки, Середовище.
Е Серед відповідей А–Д вірної не має.

Т *Завдання 20.* Як називається організація задач окремих індивідів у часі й просторі, що дозволяє вирішити виниклу перед колонією проблему (дана функція приводить до певного просторово-часового розподілу особин, їхніх дій та/або результатів їхніх дій для досягнення поставленої мети.)?
А Кооперація.
Б Координація.
В Спеціалізація.
Г Емерджентність.
Д Мультистійкість.
Е Серед відповідей А–Д вірної не має.

ПРАВИЛЬНІ ВІДПОВІДІ ДО ТЕСТОВИХ ЗАВДАНЬ

№ завдання	№ розділу		
	1	2	3
1	В	Б	В
2	Е	В	Б
3	Г	Г	Б
4	Б	Г	Г
5	В	В	Е
6	Г	Е	В
7	Е	Б	В
8	В	В	Б
9	Б	В	А
10	В	Е	Б
11	Г	Д	Б
12	А	Д	Г
13	Е	Б	В
14	В	Е	Б
15	В	В	Г
16	А	Г	Б
17	В	А	Г
18	Д	В	Г
19	Г	В	А
20	Е	В	Б

АЛФАВІТНО-ПРЕДМЕТНИЙ ПОКАЖЧИК

Х

СПИСОК ЛІТЕРАТУРИ

1. Encyclopedia of artificial intelligence / Eds.: J. R. Dopico, J. D. de la Calle, A. P. Sierra. – New York : Information Science Reference, 2009. – Vol. 1–3. – 1677 p.

2. Haupt R. Practical genetic algorithms / R. Haupt, S. Haupt. – New Jersey : John Wiley & Sons, 2004. – 261 p.

3. Барсегян А. А. Технологии анализа данных: Data Mining, Visual Mining, Text Mining, OLAP: Учебн. пос. / А. А. Барсегян. – С. Пб. : BHV, 2007. – 384 с.

4. Бодянский Е. В. Нейро-фаззи сети Петри в задачах моделирования сложных систем / Е. В. Бодянский, Е. И. Кучеренко, А. И. Михалев. – Днепропетровск : Системные технологии, 2005. – 311 с.

5. Брянцев И. Н. Data Mining. Теория и практика / И. Н. Брянцев. – М. : БДЦ-Пресс, 2006. – 208 с.

6. Гибридные нейро-фаззи модели и мультиагентные технологии в сложных системах : монография / В. А. Филатов, Е. В. Бодянский, В. Е. Кучеренко и др. ; под общ. ред. Е. В. Бодянского. – Дніпропетровськ : Системні технології, 2008. – 403 с.

7. Джонс М. Т. Программирование искусственного интеллекта в приложениях / М. Т. Джонс / Пер. с англ. А. И. Осипов. – М. : ДМК Пресс, 2004. – 312 с.

8. Круглов В. В. Искуственные нейронные сети. Теория и практика / В. В. Круглов, В. В. Борисов. – М. : Горячая линия – Телеком, 2001. – 382 с.

9. Люгер Дж. Ф. Искусственный интеллект: стратегии и методы решения сложных проблем / Дж. Ф. Люгер. – М. : Вильямс, 2005. – 864 с.

10. Олійник А. О. Еволюційні обчислення та програмування: Навчальний посібник / А. О. Олійник, С. О. Субботін, О. О. Олійник. – Запоріжжя : ЗНТУ, 2010. – 324 с.

11. Прогрессивные технологии моделирования, оптимизации и интеллектуальной автоматизации этапов жизненного цикла авиадвигателей : Монография / А. В. Богуслаев, Ал. А. Олейник, Ан. А. Олейник, Д. В. Павленко, С. А. Субботин ; под ред.

Д. В. Павленко, С. А. Субботина. – Запорожье : ОАО «Мотор Сич», 2009. – 468 с.

12. Рассел С. Искусственный интеллект: современный подход / С. Рассел, П. Норвиг. – М. : Вильямс, 2006. – 1408 с.

13. Ротштейн А. П. Интеллектуальные технологии идентификации: нечеткая логика, генетические алгоритмы, нейронные сети / А. П. Ротштейн. – Винница : УНИВЕРСУМ-Винница, 1999. – 320 с.

14. Руденко О. Г. Штучні нейронні мережі / О. Г. Руденко, Є. В. Бодянський. – Харків : Компанія СМІТ, 2006. – 404 с.

15. Скобцов Ю. А. Основы эволюционных вычислений / Ю. А. Скобцов. – Донецк : ДонНТУ, 2008. – 330 с.

16. Субботін С. О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей: монографія / С. О. Субботін, А. О. Олійник, О. О. Олійник ; під заг. ред. С.О. Субботіна. – Запоріжжя : ЗНТУ, 2009. – 375 с.

17. Субботін С. О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень: Навчальний посібник / С. О. Субботін. – Запоріжжя : ЗНТУ, 2008. – 341 с.

18. Черноруцкий И. Г. Методы принятия решений / И. Г. Черноруцкий. – СПб. : БХВ-Петербург, 2005. – 416 с.

19. Хайкин С. Нейронные сети: полный курс / С. Хайкин. – С. Пб. : Издательский дом «Вильямс», 2005. – 1104 с.

Навчальне видання

ОЛІЙНИК Андрій Олександрович
СУББОТІН Сергій Олександрович
ОЛІЙНИК Олексій Олександрович

ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ

Навчальний посібник

Комп'ютерний набір *Олійник А. О.*
Верстання *Зуб С. В.*

Оригінал-макет підготовлено
в редакційно-видавничому відділі ЗНТУ

Підписано до друку 24.10.2011. Формат 60×84/16. Ум. друк. арк. 16.
Тираж 300 прим. Зам. № 1570.

Запорізький національний технічний університет
Україна, 69063, м. Запоріжжя, вул. Жуковського, 64
Тел.: (061) 769–82–96, 220–12–14

Свідоцтво про внесення суб'єкта видавничої справи
до державного реєстру видавців, виготівників
і розповсюджувачів видавничої продукції
від 27.12.2005 р., серія ДК № 2394