МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

О. О. Сергеєв-Горчинський, Г. В. Іщенко

ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ
КОМП'ЮТЕРНИЙ ПРАКТИКУМ

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського як навчальний посібник для студентів, які навчаються за спеціальністю 122 «Комп'ютерні науки та інформаційні технології», спеціалізаціями «Інформаційні системи та технології проектування», «Системне проектування сервісів»

Київ
КПІ ім. Ігоря Сікорського
2018

Рецензент: *Рогоза Валерій Станіславович*,
доктор технічних наук, професор

Відповідальний редактор *Кисельов Геннадій Дмитрович*,
кандидат технічних наук, старший наук. співр.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 5 від 25.01.2018 р.)
за поданням Вченої ради ІПСА
(протокол № 11 від 19.12.2017 р.)

Сергеєв-Горчинський Олексій Олександрович,
кандидат технічних наук
Іщенко Ганна Валеріївна

ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ
КОМП'ЮТЕРНИЙ ПРАКТИКУМ

Інтелектуальний аналіз даних: Комп'ютерний практикум [Електронний ресурс] : навч. посіб. для студ. спеціальності 122 «Комп'ютерні науки та інформаційні технології», спеціалізацій «Інформаційні системи та технології проектування», «Системне проектування сервісів» / О. О. Сергеєв-Горчинський, Г. В. Іщенко ; КПІ ім. Ігоря Сікорського. – Електронні текстові данні (1 файл: 1,72 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2018. – 73 с.: Іл.

Зміст

Рисунки

Таблиці

Лістинги

Розділ 1. Застосування та задачі інтелектуального аналізу даних

1.1. Застосування інтелектуального аналізу даних

Люди шукають закономірності в даних з тих пір, як почалось їх осмислене життя. Мисливці шукають закономірності в поведінці тварин, фермери шукають закономірності у рості сільськогосподарських культур, політики шукають закономірності в думці виборців. Робота дослідника полягає в тому, щоб осмислити інформативність та повноту даних, які є в його розпорядженні, виявити зв'язки між ними і побудувати моделі з метою прийняття правильних рішень. Наприклад, метою продавця товарів є визначення закономірностей у поведінці різних категорій покупців з врахуванням їх купівельної спроможності, уподобань, віку, тощо.

В процесі накопичення даних збільшується необхідність застосування комп'ютерних методів інтелектуального аналізу даних як ефективного допоміжного засобу, який дозволяє досліднику отримати додаткові знання з предметної галузі, в якій він працює і має приймати зважені і обґрунтовані рішення. Економісти, статистики, прогнозисти і інженери давно усвідомили, що закономірності в даних можна шукати автоматично, ідентифікувати, перевіряти і застосовувати їх для прогнозування нових даних.

Класичним прикладом в галузі прогнозування товарообігу, який вимагає застосування методів інтелектуального аналізу даних, є проблема зміни уподобань клієнтів на конкурентному ринку. Ключем до розв'язання цієї проблеми є база даних профілів клієнтів і їх покупок. Моделі поведінки покупців можна проаналізувати, щоб визначити їх відмінні характеристики. Якщо такі характеристики вдалося знайти їх можна застосувати для ідентифікації непостійних клієнтів або тих, які в даний момент не користуються певними товарами, але можуть зацікавитися ними.

У Таблиці 1.1 представлений спрощений приклад даних про рекомендації контактних лінз, які описують питання, за якими окуліст може визначити необхідний тип лінз – м'які, жорсткі або ніякі. В рядках таблиці наведені дані про пацієнтів, тобто параметри, на підставі яких робляться висновки (в термінології інтелектуального аналізу даних, атрибути ознак). На підставі цих даних лікар може зробити висновки щодо призначення пацієнтам специфічних контактних лінз або щодо відсутності необхідності в них. Лікарські рекомендації представлені в останньому стовпчику кожного рядка. Тобто кожен рядок таблиці є прикладом певного набору атрибутів ознак стану пацієнта та лікарської рекомендації (атрибуту класу).

Таблиця 1.1. Дані про контактні лінзи

Вік	**Передумови**	**Астигматизм**	**Рівень вологості очей**	**Рекомендовані лінзи**
Підлітки	короткозорість	немає	знижений	ніякі
Підлітки	короткозорість	немає	нормальний	м’які
Підлітки	короткозорість	є	знижений	ніякі
Підлітки	далекозорість	є	нормальний	жорсткі
Дорослі	короткозорість	немає	знижений	ніякі
Дорослі	короткозорість	немає	нормальний	м’які
Дорослі	короткозорість	є	знижений	ніякі
Дорослі	короткозорість	є	нормальний	жорсткі
Старі	короткозорість	немає	знижений	ніякі
Старі	короткозорість	є	нормальний	жорсткі
Старі	далекозорість	немає	нормальний	м’які
Старі	далекозорість	є	знижений	ніякі

Таблиця 1.1 містить 12 рядків, що включають три можливих значення для атрибуту вікової групи, два значення для атрибуту схильності до далекозорості або короткозорості, два значення для атрибуту наявності астигматизму і два значення для атрибуту рівня виділення сліз. Кількість можливих комбінацій значень для чотирьох атрибутів дорівнює $3 \times 2 \times 2 \times 2 = 24$.

Вивчивши дані можна сформулювати асоціативні правила для лінз (див. Лістинг 1.1).

Лістинг 1.1. Приклад асоціативного правила

```
ЯКЩО Рівень вологості очей = = знижений ТОДІ Рекомендовані лінзи = ніякі
ЯКЩО Вік = = дорослі ТА Рівень вологості очей = = нормальний ТОДІ Рекомендовані лінзи = м’які
```

Крім асоціативних правил, поширеним способом наочного представлення закономірностей є дерева рішень, що містять в вузлах питання, а в листах – висновки.

У більшості ситуацій, пов'язаних з навчанням, набір прикладів, наведених як вхідні дані, є в найкращому випадку таким, який дозволяє робити узагальнюючі висновки, але не містить всіх можливих ситуацій, тобто в цьому сенсі є неповним, і тому аналіз процесу, який досліджується, полягає у створенні його моделі на підставі наявних прикладів (в теорії штучного інтелекту такий підхід називається навчанням з вчителем, тобто іншими словами, побудовою моделей на підставі екземплярів даних, які доступні для дослідника).

В реальних ситуаціях задача побудови моделей на підставі навчання ускладнюється тим, що в наборах даних можуть бути приклади, в яких

значення деяких атрибутів невідомі – наприклад, дані про вимірювання не були записані або були втрачені. Також часто трапляються некоректні класифікації через помилки або похибки в даних, необхідних для побудови моделі класифікації.

Нижче наведені типові приклади задач для пояснення головних цілей інтелектуального аналізу даних.

1.1.1. Ухвалення рішення про видачу кредиту

(задача класифікації)

У банківській сфері для видачі кредиту, позичальнику необхідно заповнити анкету, яка містить дані про клієнта. Ці дані приймаються до уваги кредитною компанією як обґрунтування для прийняття рішення про надання фінансових послуг. Такі рішення зазвичай приймаються в два етапи. Спершу застосовують статистичні методи для визначення однозначних рішень про «схвалення» або «відхилення» прохання клієнта про надання кредиту. Якщо прохання клієнта не відхилено на першому етапі, далі включається більш детальний аналіз, який вимагає більш складного аналізу ситуації і найчастіше вимагає залучення до процесу прийняття рішень експертів банку.

В очевидних ситуаціях кредитні компанії застосовують процедуру статистичного прийняття рішень і розрахунку певного числового параметра за інформацією, наданою в анкеті. Заявка вважається «схваленою», якщо статистичний параметр перевищує встановлений поріг і «відхиленою» в іншому випадку. У неочевидних випадках заявка направляється до співробітників кредитної компанії для досягнення компромісного рішення між точкою зору тих, хто схильний не приймати рішень з високим рівнем ризику (скажімо, бухгалтера), і тих, хто зацікавлений в утриманні клієнтів (наприклад, керівника відділу продажів).

1.1.2. Екологічний моніторинг

(задача класифікації)

В екологічному моніторингу важливим є питання виявлення нафтових плям на супутникових зображеннях, для раннього попередження екологічних катастроф і запобігання незаконному захороненню промислових відходів. Радіолокаційні супутники дозволяють спостерігати за прибережними водами вдень і вночі, незалежно від погодних умов. Нафтові плями виглядають як темні області на зображенні, площа і форма яких змінюються в залежності від погодних і морських умов. У системах виявлення застосовуються методи інтелектуального аналізу, призначені для навчання на прикладах розливів і

контролю компромісу між невиявленими розливами і помилковими тривогами.

Вхідними даними в цій задачі є набір необроблених цифрових зображень з радіолокаційного супутника, а вихідними даними є набагато менший набір зображень з класифікованими нафтовими плямами, позначеними кольоровим контуром. Перш за все, для нормалізації зображення застосовуються стандартні операції обробки зображень. Потім ідентифікуються підозрілі темні області. Для кожного регіону визначаються кілька десятків атрибутів, що характеризують форму, площу, інтенсивність, різкість і нерівність меж, близькість до інших регіонів і інформацію про передісторію в околиці регіону. До знайдених векторів атрибутів застосовуються стандартні методи інтелектуального аналізу для побудови моделей класифікації.

1.1.3. Прогнозування електроспоживання

(задача прогнозування)

Автоматизоване прогнозування навантаження дозволяє великим постачальникам комунальних послуг генерувати погодинні прогнози на два дні вперед. Стандартна модель навантаження включає три компоненти: базове навантаження за рік, періодичність завантаження протягом року і вплив свят. Електричне навантаження характеризується періодичністю: добовою – попит має ранній ранковий мінімум, екстремум опівдні і денний максимум; щотижневою – попит нижче по вихідних днях; сезонну – підвищений попит взимку і літом для опалення та охолодження, відповідно, яка створює річний цикл.

Частотні ефекти враховуються шляхом реконструкції річного навантаження у вигляді послідовності днів з урахуванням свят, денормалізації навантаження для обліку загальної тенденції. Модель прогнозу розраховують за допомогою лінійного регресійного аналізу значень температури, вологості, швидкості вітру і хмарного покриву, мінімізації функції відстані і знаходження оптимальних параметрів регресійної моделі.

1.1.4. Інтелектуальний аналіз веб-сторінок

(задача класифікації)

Пошукові Інтернет-провайдери вивчають зміст веб-сторінок, щоб розрахувати їх змістовну вагу і релевантність при відображенні в результатах пошуку. Одним з механізмів вимірювання змістовної ваги веб-сторінки серед інших сторінок є алгоритм PageRank, розроблений компанією Google, відповідно до якого чим більше сторінок посилаються на створену сторінку,

тим вище її змістовна вага, особливо якщо сторінки що посилаються, в свою чергу також мають високу змістовну вагу.

Іншим способом визначення ваги веб-сторінки є розрахунок значень набору атрибутів – наприклад, значення частоти появи терміна з пошукового запиту в URL-адресі сторінки, у заголовку та мета-даних веб-сторінки в текстах посилань. Для складних запитів функції ваги враховують кілька різних термінів, які відображаються поруч на сторінках. Типові алгоритми зважування веб-сторінок враховують сотні або тисячі атрибутів.

Крім аналізу інформації в мережі Інтернет, пошукові системи також аналізують інформацію про пошукові запити – терміни, які шукаються – для визначення рекламних оголошень, які можуть зацікавити користувача і забезпечити перехід на сайт рекламодавця. Переходи на сайти також враховуються для поліпшення подальших результатів пошуку.

1.1.5. Технічна діагностика

(задача пошуку асоціативних правил)

Діагностика – одна з основних областей застосування експертних систем. Хоча складені вручну правила, що застосовуються в експертних системах, часто є цілком інформативними, але інтелектуальний аналіз даних може стати в нагоді в ситуаціях, коли об’єднання (додавання) таких правил може стати занадто трудомісткою працею або коли важко передбачити заздалегідь всі можливі комбінації вхідних даних та можливі запити.

Профілактичне обслуговування електромеханічних пристроїв може запобігти збоям, які порушують промислові процеси. Несправності визначаються шляхом вимірювання вібрацій в різних місцях монтажу пристрою і застосуванням до значень аналізу Фур'є. Інформація, яка є спотвореною через обмеження в процедурі вимірювань, вивчається експертом при діагностиці. Вимірювання зазвичай характеризуються досить низькою точністю і вимагають доповнення проміжними значеннями, тобто значеннями модельних функцій, які визначаються в процесі консультацій з експертом. Правила діагностики можуть генеруватися за допомогою методу пошуку асоціативних правил. Хоча знайдені правила можуть бути досить складними для сприйняття, тільки експерти здатні перевірити їх з урахуванням довідкових даних.

1.1.6. Маркетинг та продажі

(задача пошуку асоціативних правил)

Деякі з найбільш активних застосувань інтелектуального аналізу даних

відносяться до сфери маркетингу і продажів, в якій компанії володіють величезними обсягами точно записаних даних. Інтелектуальний аналіз таких даних може визначати групи клієнтів, для яких будуть цікаві нові послуги. Аналіз споживчого кошика передбачає застосування методів пошуку асоціативних правил для визначення груп товарів, які записані у чеках та купуються разом. Така інформація може використовуватись для планування складських приміщень, організації спеціальних знижок тощо.

Прямий маркетинг – також популярний напрям для інтелектуального аналізу даних. На відміну від торгових центрів роздрібної торгівлі, компанії прямої поштової розсилки мають повну історію покупок для кожного окремого клієнта і можуть застосовувати інтелектуальний аналіз даних для визначення тих клієнтів, які зреагують на нові пропозиції. Цільові рекламні компанії істотно зменшують фінансові витрати, оскільки застосовують моделі відгуку, отримані у відповідь на первинну розсилку, коли покупці висилають анкети або телефонують для отримання додаткової інформації.

1.1.7. Інші застосування

Існує багато наукових напрямків застосування методів інтелектуального аналізу даних. У біології інтелектуальний аналіз застосовують, щоб допомогти ідентифікувати гени в геномах. У біомедицині інтелектуальний аналіз застосовується для прогнозування активності препарату та дозволяє. враховувати не тільки хімічні властивості ліків, але і їх тривимірну структуру, що прискорює розробку ліків і знижує їх вартість. В астрономії інтелектуальний аналіз даних застосовується для розробки повністю автоматичної системи каталогізації небесних об'єктів, які занадто малі і непомітні. У хімії аналіз даних застосовується для прогнозування структури деяких органічних сполук по магнітно-резонансному спектру. У всіх перерахованих додатках методи інтелектуального аналізу даних досягли рівня практичного застосування.

Методи інтелектуального аналізу даних можуть врятувати життя, наприклад, при виявленні змін характеристик фізіологічних процесів, які не можна пояснити добовим ритмом пацієнта, дією лікарських препаратів тощо. Також, оскільки сучасний світ спирається на вразливі мережеві комп'ютерні системи і все більше турбується про кібер-безпеку, інтелектуальний аналіз все частіше застосовується для виявлення атак та розпізнавання незвичайних моделей поведінки інформаційних систем.

1.1.8. Етика в задачах інтелектуального аналізу даних

Як вже було згадано вище, інтелектуальний аналіз даних часто застосовується при видачі кредиту, генеруванні пропозицій до конкретних покупців на товари тощо. В деяких випадках правильні висновки щодо індивідуальних особливостей людей та їх потреб можуть вимагати врахування расових та гендерних ознак або, скажімо, релігійної приналежності людей, тоді як в інших випадках намагання зібрати такі дані можуть розглядатися як неетичні та незаконні.

Наприклад, облік гендерної та расової інформації для медичного діагнозу, безумовно, є етичним, однак не є таким при аналізі поведінки платника при виплаті позики. Навіть коли конфіденційна інформація буде свідомо виключена, існує ймовірність того, що побудовані аналітичні моделі будуть залежати від змінних, які опосередковано будуть пов’язані з расовими або гендерними характеристиками. Наприклад, люди часто живуть в районах, пов'язаних з конкретними етнічними особливостями, і отже застосування поштового коду при аналізі даних пов'язане з ризиком створення моделей, заснованих на расовій ознаці, навіть якщо расова інформація була явно виключена з даних.

Перш ніж люди приймуть рішення надати особисту інформацію, їм необхідно знати, як і для чого вона буде задіяна, які дії будуть зроблені для захисту їх конфіденційності, які права на відшкодування вони можуть мати.

Потенціал методів інтелектуального аналізу, що застосовуються до баз даних, є ширший ніж тих, які застосовуються при простому накопиченні даних. Саме тому необхідно визначити умови, в яких були зібрані дані, і для яких цілей вони можуть застосовуватися. Важливим є також вирішення питання, чи наділяє володіння даними правом застосовувати їх в інших формах, аніж ті, які були спочатку зареєстровані? Очевидно, що у багатьох випадках така проблема не виникає, проте в цілому ситуація з етичною стороною процесу збирання даних може бути неоднозначною.

В результаті інтелектуального аналізу даних може виникнути надзвичайна ситуація, при якій побудована аналітична модель не буде очевидною для аналітика. У такій ситуації необхідно визначити, за якими даними побудована модель, за яких умов зібрані дані, чи етично їх застосовувати в конкретній задачі? Для відповіді на ці питання мало покладатися на статистичні показники, необхідно володіти знаннями про психологію людини, також знаннями в інших наукових областях.

При роботі з даними, необхідно дізнатися, кому дозволено мати доступ до них, з якою метою вони були зібрана і які висновки можуть бути застосовні. Таким чином, етичний аспект ставить складні питання тим, хто займається інтелектуальним аналізом даних. Необхідно враховувати правові норми і стандарти, які нажаль можуть бути невідомі фахівцям з інформаційних технологій. Наприклад, в бібліотечному співтоваристві прийнято вважати, що конфіденційність читачів є правом, яке має ретельно захищатися. Якщо

зателефонувати в університетську бібліотеку і запитати, у кого є певний підручник, бажаної відповіді скоріш за все не нададуть. Такі правила дозволяють студентам не піддаватися тиску з боку викладачів, які можуть бути зацікавлені в розповсюджені літератури або її резервування для власних потреб. Ті, хто створюють, скажімо, цифрові бібліотеки, можуть і не знати про ці нюанси і створювати системи, які надають доступ до персональних переваг користувачів або аргументувати таким чином рекомендації.

На додаток до різноманітних етичних норм, які існують в середовищі Інтернет-спільноти щодо застосування даних необхідно дотримуватися логічних і наукових стандартів при формуванні висновків. Якщо, наприклад, зроблено висновок, що, власники червоних автомобілів пов'язані з більш значними кредитними ризиками, необхідно визначити обмеження і підкріпити їх іншими аргументами, крім чисто статистичних.

Інтелектуальний аналіз даних формує ще одне питання, яке в багатьох ситуаціях може бути важливим для правильної інтерпретації даних. Яку користь можна отримати від інформації, зібраної при інтелектуальному аналізі споживчого кошика, в якому порівнюються записи з продажу в супермаркетах для виявлення зв'язків між придбаними товарами? Наприклад, чи повинен менеджер супермаркету розміщувати на полицях пиво і чіпси поруч, щоб полегшити їх покупку або навпаки, може краще їх розмістити в різних частинах магазину, щоб максимально збільшити час перебування покупця в магазині і, таким чином, збільшити ймовірність здійснення додаткових покупок?

Звичайно, кожний, хто застосовує результати інтелектуального аналізу даних, повинен керуватися здоровим глуздом. Якщо дані характеризуються як зафіксовані факти, то інформація є набором розміщених в основі даних закономірностей або очікувань, які необхідно перевіряти.

1.2. Етапи інтелектуального аналізу даних

Організація процесу інтелектуального аналізу даних визначається стандартами, які містять покрокові рекомендації, задачі та цілі для всіх етапів процесу аналізу даних. Консорціумом компаній NCR, SPSS та DaimlerChrysler розроблено поширений в світі стандарт The Cross Industry Standard Process for Data Mining (CRISP-DM), який є продовженням підходів Knowledge Discovery in Databases (KDD) та SEMMA (Sample – збір даних, Explore – дослідження зв’язків, Modify – модифікування, Model – моделювання, Assess – оцінювання).

На Рисунку 1.1 показано життєвий цикл проекту інтелектуального аналізу даних у вигляді структурної моделі CRISP-DM. На початковому етапі "Розуміння бізнес-контексту" виконується дослідження бізнес-цілей і вимог,

приймається рішення про те, чи може застосування інтелектуального аналізу задовольнити бізнес-цілі, і визначається те, які дані необхідно зібрати для побудови адекватної аналітичної моделі.

На наступному етапі «Підготовки даних» створюється і вивчається початковий набір даних, щоб визначити, чи придатний він для подальшої обробки. Якщо кількість даних невелика, може знадобитися збір нових даних на базі більш строгих критеріїв. Аналіз даних, отриманих на даному етапі, може також призвести до перегляду бізнес-контексту – можливо, знадобиться перегляд мети застосування інтелектуального аналізу даних.

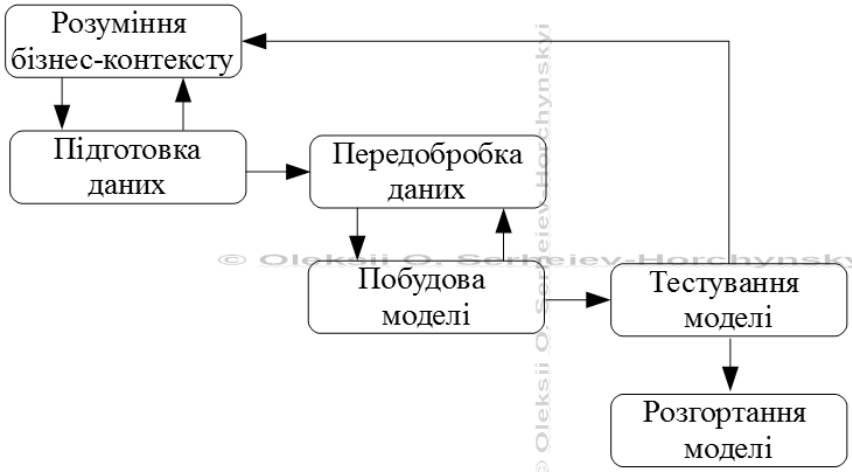

Рисунок 1.1. Структурна модель CRISP-DM

Етап «Передобробки даних» включає попередню обробку необроблених даних, щоб методи машинного навчання були здатні побудувати аналітичну модель, відображаючу структурний опис інформації, яка неявно міститься в навчальних даних.

На етапі «Побудова моделі» створюється робоча (внутрішня) модель даних для її подальшої перевірки. Етапи «Підготовка даних» і «Побудова моделі» зазвичай виконуються паралельно. У більшості випадків необхідно їх повторювати, оскільки результати, отримані в ході моделювання, забезпечують краще розуміння вимог до методів попередньої обробки.

Важливим є етап «Тестування моделі», тобто перевірки побудованої аналітичної моделі. Тестування дозволяє зробити висновки, чи може аналітична модель, побудована за навчальними даними, бути застосована для класифікації тестових даних або ж вона відображає помилкові закономірності.

Існує множина методів тестування побудованих аналітичних моделей. Якщо етап тестування показує, що модель є незадовільною, можливо, буде потрібно переглянути весь проект з аналізу даних і повернутися до етапу "Розуміння бізнес-контексту", щоб уточнити бізнес-цілі або способи збору

даних. Якщо точність моделі досить висока, далі виконується етап «Розгортання моделі» в програмній системі, для цього модель передається інженерам з програмного забезпечення.

Метою інтелектуального аналізу даних є побудова аналітичних моделей, оптимізованих для вирішення конкретних класів прикладних задач. Аналітичні моделі будуються для навчальних екземплярів даних з певної проблемної області. Як тільки модель побудована, вона перевіряється на тестових екземплярах даних. Після підтвердження адекватності та точності побудованої аналітичної моделі, її можна застосовувати до нових некласифікованих даних.

У відповідності зі стилем пошуку закономірностей в навчальних даних методи інтелектуального аналізу даних ділять на наступні категорії:

– навчання з вчителем: модель будується по заздалегідь класифікованим навчальним даними і описує закономірності між значеннями атрибутів ознак і значенням атрибуту класу;

– навчання без вчителя: в навчальних даних відсутній атрибут класу, закономірності шукаються між значеннями усіх атрибутів, які вважаються атрибутами ознак;

– часткове навчання з вчителем: кількість класифікованих навчальних даних набагато менша ніж кількість некласифікованих даних, тому спочатку для класифікованих даних вирішується задача навчання з вчителем і будується аналітична модель, після чого вирішується задача навчання без вчителя з підкріпленням за побудованою аналітичною моделлю;

– навчання з підкріпленням: при появі нових навчальних даних, попередня модель не будується заново, а успадковує знайдені раніше закономірності, котрі показали коректні результати класифікації на тестових наборах даних.

1.2.1. Навчання з вчителем

Навчання з вчителем – основна концепція, що лежить в основі таких прикладних задач, як розпізнавання голосу, категоризація спаму, розпізнавання осіб на фотографіях і виявлення шахрайських операцій з кредитними картами. Метою навчання з вчителем є пошук значень функції f, яка із заданою точністю прогнозує значення атрибуту класу Y за значеннями атрибутів ознак X, що містяться в множині навчальних екземплярів даних D. Функція f, що описує зв'язок між значеннями атрибутів ознак X і значенням атрибуту класу Y, називається аналітичної моделлю і записується як $f(X) \rightarrow Y$.

Стандартна структура методів навчання з вчителем визначається наступним алгоритмом:

1. Визначити прикладну задачу; визначити атрибути ознак і атрибут

класу, їх типи.

2. Сформулювати припущення щодо аналітичної моделі, що ставить у відповідність значенням атрибутів ознак значення атрибуту класу.

3. Вибрати метод інтелектуального аналізу для побудови аналітичної моделі.

4. Вибрати міру оптимальності аналітичної моделі за тестовими даних.

5. Визначити чисельний метод оптимізації аналітичної моделі.

6. Вирішити задачу оптимізації і знайти аналітичну модель, що описує зв'язок між значеннями атрибутів *X* і *Y*.

Під навчанням з вчителем розуміється робота з набором класифікованих навчальних даних. Припустимо, що є набір навчальних текстових даних (див. Таблицю 1.2). Для того, щоб побудувати аналітичну модель класифікації необхідно апріорно вказати класи навчальних повідомлень.

Таблиця 1.2. Набір навчальних текстових даних

№	Клас	Повідомлення
1	музика	Новий альбом гурту СКАЙ!
2	одяг	#мода Продається смокінг. #одяг
3	дані	Запустили кластер Hadoop для обробки даних. #веб

Очевидно, що для побудови точної моделі класифікації, знадобиться набагато більша кількість навчальних прикладів. Існують обмеження, які необхідно враховувати при навчання з вчителем, наприклад, дилема "зміщення-дисперсії" – як організувати побудову аналітичної моделі, яка навчившись на одних навчальних даних буде коректно працювати з новими навчальними даними, зібраними в інший період часу.

Для малої кількості навчальних наборів даних аналітичні моделі зазвичай характеризуються високим зміщенням середнього значення. Для спотворених даних властива висока дисперсія значень. Між двома обмеженнями для конкретних даних можна знайти компроміс вибравши адекватну аналітичну модель.

Процес класифікації з вчителем складається з двох етапів: побудови моделі класифікації і класифікації нових екземплярів даних. Обидва етапи включають такі дії: підготовку набору навчальних даних; попередню обробку даних; визначення інформативних ознак; побудова аналітичної моделі; збір нових даних; попередню обробку нових даних; визначення інформативних ознак у нових даних; класифікацію нових даних.

При навчанні з вчителем, кожен екземпляр даних є незалежним екземпляром прикладної області, яку необхідно вивчити. Звичайно, є багато задач, в яких необроблені дані не можуть бути виражені як індивідуальні, незалежні екземпляри і для формування екземплярів слід приймати до уваги довідкову інформацію про прикладну область.

Кожен екземпляр даних характеризується значеннями атрибутів, які виражають різні його характеристики. Хоча існує множина різних типів атрибутів, для більшості методів інтелектуального аналізу основними є числові (дійсні) та категоріальні (строкові) значення.

1.2.1.1. Задача класифікації

Задача класифікації передбачає навчання з вчителем і вимагає наявності попередньої інформації про класи навчальних екземплярів даних. Побудовану аналітичну модель перевіряють на незалежному наборі тестових даних, для яких справжні класи екземплярів даних є відомими, проте недоступними для методу при тестуванні точності класифікації.

Показник успішності тестування дає об'єктивну характеристику того, наскільки точно побудована модель. У багатьох практичних прикладних задачах успіх навчання вимірюється з точки зору того, наскільки зрозумілими є візуальне представлення і структура аналітичної моделі.

Для зображення моделей класифікації застосовують таблиці і дерева. Якщо дані є агрегованими з декількох предметних областей і можуть бути класифіковані по декільком атрибутам, задачу навчання слід розглядати як декілька задач побудови моделей класифікації, що припускають класифікацію всіх екземплярів, по кожному класу окремо.

1.2.1.2. Задача чисельного прогнозування

Чисельне прогнозування – різновид класифікації, в якій атрибут класу є чисельними значенням. Одним із прикладів є задача, яка відображає дані про погоду (див. Таблицю 1.3) для прогнозування тривалості гри в хвилинах.

Таблиця 1.3. Дані про погоду з числовим атрибутом класу Тривалість гри

Прогноз	**Температура**	**Вологість**	**Вітер**	**Тривалість гри**
сонячно	85	85	немає	5
сонячно	80	90	є	0
хмарно	83	86	немає	55
дощ	70	96	немає	40
дощ	68	80	немає	65
дощ	65	70	є	45
хмарно	64	65	є	60
сонячно	72	95	немає	0
сонячно	69	70	немає	70
дощ	75	80	немає	45
сонячно	75	70	є	50

хмарно	72	90	є	55
хмарно	81	75	немає	75
дощ	71	91	є	10

При розгляді задач чисельного прогнозування, розрахунок нових значень атрибуту класу є настільки ж важливим, як і розрахунок значень вагових коефіцієнтів атрибутів, які визначають внесок кожного атрибуту ознаки в значення атрибуту класу.

1.2.2. Навчання без вчителя

У порівнянні з методами навчання з вчителем метою навчання без вчителя є визначення прихованих зв'язків між некласифікованими даними. Навчання без вчителя можна застосовувати для виявлення груп однорідних даних в задачі кластеризації або для групування значень атрибутів в задачі пошуку асоціативних правил. Зазвичай всі дані можуть зберігатися в пам'яті і оброблятися за допомогою програмних бібліотек для інтелектуального аналізу даних, наприклад Weka, Java-ML.

1.2.2.1. Задача кластеризації

Коли у даних не визначений атрибут класу або невідомі його можливі значення, для групування екземплярів застосовують кластеризацію, яка групує екземпляри, які мають близькі за величиною значенням атрибутів ознак. Уявімо дані в задачі класифікації квітів ірисів, в яких не вказані їх різвиди (див. Таблицю 1.4).

Таблиця 1.4. Дані ірисів в задачі кластеризації

Довжина чашолистка	**Ширина чашолистка**	**Довжина пелюстки**	**Ширина пелюстки**
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
...	...	...	...
7.0	3.2	4.7	1.4
6.4	3.2	4.5	1.5
6.9	3.1	4.9	1.5
5.5	2.3	4.0	1.3
6.5	2.8	4.6	1.5

...	...	...	...
6.3	3.3	6.0	2.5
5.8	2.7	5.1	1.9
7.1	3.0	5.9	2.1
6.3	2.9	5.6	1.8
6.5	3.0	5.8	2.2
...	...	...	...

За атрибутом ознаки «ширина пелюстки» наявні екземпляри інтуїтивно можна згрупувати в три кластери. Задача кластеризації полягає в тому, щоб автоматично визначати аналітичні моделі кластерів для подальшої умовної класифікації нових екземплярів даних.

1.2.2.2. Задача пошуку асоціативних правил

Якщо в даних можливе лише одне значення атрибуту класу (див. Таблицю 1.5, атрибут класу «Придбано»), для даних можна застосувати пошук класифікаційних або асоціативних правил між значеннями атрибутів ознак.

Асоціативні правила відрізняються від класифікаційних правил наступними умовами:

- усі атрибути є атрибутами ознак;
- розрізняють атрибути умов та атрибути наслідків правил.

Таблиця 1.5. Дані покупок в супермаркеті

Випічка	**Молочні продукти**	**М'ясо**	**Хімія**	**Придбано**
хліб	молоко	грудки	паста	1
хліб	кефір	печінка	паста	1
макарони	йогурт	грудки	порошок	1
хліб	молоко	стегно	шампунь	1
булка	сметана	крильця	порошок	1
…	…	…	…	…

Через згадані вище умови, асоціативних правил можна знайти набагато більше, аніж класифікаційних правил. Пошук правил зазвичай виконується для нечислових атрибутів ознак.

Задача пошуку правил полягає в тому, щоб знайти значущі, незалежні та компактні правила. З цієї причини пошук асоціативних правил завершується вибором правил, які охоплюють задану мінімальну кількість екземплярів – наприклад, 80% від всього набору даних і мають значення рівня точності більше за мінімальний встановлений рівень – скажімо, 95%.

1.2.3. Програмна бібліотека Weka

Weka (є скороченням від Waikato Environment for Knowledge Analysis) – середовище для аналізу даних і виявлення знань, є найвідомішою програмної бібліотекою на мові Java для інтелектуального аналізу даних, розробленої в університеті Ваікато, Нова Зеландія. У програмній бібліотеці реалізовані методи класифікації, регресії, пошуку асоціативних правил і кластеризації, розроблений зручний графічним інтерфейс користувача, інтерфейс у вигляді командного рядка і фреймворк у вигляді Java-класів.

Адреса проекту в мережі Інтернет:

http://www.cs.waikato.ac.nz/ml/weka/.

Weka в цілому містить більше 267 алгоритмів: 82 для попередньої обробки даних, 33 для вибору атрибутів, 133 для класифікації і регресії, 12 для кластеризації і 7 для генерування асоціативних правил. Графічний інтерфейс добре підходить для вивчення даних, в той час як API дозволяє розробляти нові системи інтелектуального аналізу даних і застосовувати наявні програмні реалізації методів. Weka поширюється по ліцензії GNU GPL, що означає, що можна копіювати, поширювати і змінювати її програмний код до тих пір, поки дотримуєтесь умов ліцензії GNU GPL.

Крім кількох підтримуваних форматів опису даних, Weka пропонує власний формат – ARFF для опису даних у вигляді пар атрибутів. Зміст ARFF складається з двох частин. Перша частина містить заголовок, який визначає всі атрибути і їх типи, наприклад, категоріальний, числовий, строковий і у вигляді часової позначки. Для задачі класифікації останній атрибут в заголовку неявно вважається атрибутом класу. Друга частина містить дані, де кожен рядок відповідає значенням атрибутів екземплярів даних, перерахованих через кому, відсутні значення позначаються знаком питання. Наприклад, профілі користувачів веб-сайту можна представити у вигляді ARFF файлу, як зображено на Лістингу 1.2:

Лістинг 1.2. ARFF- файл з профілями користувачів

```
@relation person_dataset % показники людини
@attribute ‘Name’ string % ім’я
@attribute ‘Height’ numeric % ріст
@attribute ‘Eye color’{blue, brown, green} % колір очей
@attribute ‘Hobbies’ string % хобі
@data
‘Bob’, 185.0, blue, ‘climbing, sky diving’
‘Anna’, 163.0, brown, ‘reading’
‘Jane’, 168.0, ?, ?
```

Опис атрибутів в заголовку складається з трьох частин. Перша частина починається з ключового слова @relation і назви набору даних. Наступний розділ починається з ключового слова @attribute, за яким слідує назва і тип атрибута. Останній атрибут неявно вважається таким, який позначає клас екземпляра даних. Остання частина починається з ключового слова @data, за яким слідує перелік рядків із значеннями атрибутів для окремих екземплярів даних. Значення екземплярів розділені комами і повинні слідувати по порядку, визначеному у другій частині заголовку ARFF-файлу.

Java Weka API організовано в наступних пакетах:

– weka.associations: структури даних і алгоритми для пошуку асоціативних правил, включаючи методи Apriori, прогнозуючий Apriori, FilteredAssociator, FP-Growth, узагальнення послідовних закономірностей (Generalized Sequential Patterns, GSP), Hotspot і Tertius.

– weka.classifiers: алгоритми навчання з вчителем, діляться на наступні компоненти:

  – weka.classifiers.bayes: реалізації методів Байеса, включаючи наївний класифікатор Байєса, мережу Байєса, логістичну регресію Байеса;

  – weka.classifiers.evaluation: алгоритми тестування для категоріальних і числових класів, статистичні критерії, матриця достовірності, ROC-крива;

  – weka.classifiers.functions: регресійні алгоритми, лінійна регресія, isotonic регресія, процеси Гаусса, метод опорних векторів, багатошаровий персептрон, голосуючий персептрон;

  – weka.classifiers.lazy: алгоритми засновані на порівнянні екземплярів даних, k-найближчих сусідів, правила Байеса;

  – weka.classifiers.meta: мета-алгоритми навчання з вчителем, включаючи AdaBoost, аддитивна регресія;

  – weka.classifiers.mi: алгоритми навчання з вчителем по декількох екземплярах, MI AdaBoost;

  – weka.classifiers.rules: методи побудови таблиць рішень і правил прийняття рішень за принципом послідовного поділу – Ripper, Part, Prism;

  – weka.classifiers.trees: алгоритми побудови дерев рішень, включаючи ID3, C4.5, M5, функціональне дерево, логістичне дерево, випадкові ліси;

– weka.clusterers: алгоритми кластеризації, методи k-середніх, Clope, Cobweb, ієрархічна кластеризація;

– weka.core: структури представлення даних, файли конфігурації;

– weka.datagenerators: генератори даних для класифікації, регресії, кластеризації;

– weka.estimators: алгоритми розрахунку статистичного розподілу

ймовірностей для дискретних / неперервних значень, умовні ймовірності;

– weka.experiment: набори даних, налаштування моделей і статистики для виконання експериментів;

– weka.filters: алгоритми фільтрації даних на базі атрибутів і екземплярів для попередньої обробки даних з вчителем і без вчителя;

– weka.gui: графічний інтерфейс, додаток Explorer, Experimenter і KnowledgeFlow. Explorer дозволяє досліджувати набір даних, алгоритми, а також їх параметри і візуалізувати набори даних за допомогою графіків і інших візуальних діаграм. Experimenter необхідний для виконання серій експериментів. KnowledgeFlow надає графічний інтерфейс користувача для моделювання потоків даних за допомогою візуального комбінування окремих блоків по обробці даних, включаючи завантаження даних, застосування фільтрів, ручну побудову моделі класифікації і визначення її точності.

1.2.4. Програмна бібліотека Java-ML

Бібліотека Java-ML являє набір алгоритмів для інтелектуального аналізу даних із загальним Java API. Бібліотека призначена насамперед для інженерів і програмістів. Java-ML містить алгоритми попередньої обробки даних, визначення інформативних атрибутів в даних, класифікацію і кластеризацію. У бібліотеці реалізовано кілька інтерфейсів для доступу до алгоритмів Weka безпосередньо через Java-ML API.

Адреса проекту в мережі Інтернет:

http://java-ml.sourceforge.net.

Java-ML є програмною бібліотекою загального призначення для інтелектуального аналізу даних і поширюється за ліцензією GNU GPL. Java-ML підтримує будь-які типи файлів з наборами даних, за умови що кожен екземпляр даних записаний в окремому рядку, а значення атрибутів розділені комою, крапкою з комою або табуляцією.

Бібліотека організована у вигляді набору пакетів:

– net.sf.javaml.classification: алгоритми класифікації, включаючи наївний класифікатор Байєса, випадкові ліси, bagging, самоорганізучі карти, k-найближчих сусідів;

– net.sf.javaml.clustering: алгоритми кластеризації, як k-середніх, самоорганізуючі карти, просторову кластеризацію, Cobweb, AQBC;

– net.sf.javaml.core: класи, що представляють екземпляри і набори даних;

– net.sf.javaml.distance: алгоритми, які розраховують міри відмінності і подібності екземплярів даних, такі як, відстань Чебишева, косинусна міра,

відстань Евкліда, відстань Жаккарда, відстань Махаланобіса, Манхеттенська відстань, відстань Міньківського, коефіцієнт кореляції Пірсона;

– net.sf.javaml.featureselection: алгоритми розрахунку інформативності атрибутів, розрахунок міри приросту інформації при побудові дерев прийняття рішень, ReliefF, розбіжність Кульбака-Лейблера;

– net.sf.javaml.filter: методи маніпулювання екземплярами даних шляхом їх фільтрації, видалення неінформативних атрибутів, встановлення значень класів або значень атрибутів;

– net.sf.javaml.matrix: робота з масивами;

– net.sf.javaml.sampling: алгоритми формування підмножин з наборів даних;

– net.sf.javaml.tools: службові методи для наборів даних, маніпулювання екземплярами даних, серіализації даних;

– net.sf.javaml.utils: службові методи.

1.2.5. Великі дані

Задача обробки великих даних існувала задовго до того, як дане поняття було введено в застосування, наприклад, банки і фондові біржі щодня виконують мільярди фінансових операцій, авіакомпанії об'єднані в міжнародну інфраструктуру для управління резервуванням квитків і т.д.

В 2001 р. META Group визначила модель великих даних 3V (від трьох слів «volume» – великий обсяг даних, «velocity» – велика швидкість зміни даних, «variety» – велика різноманітність даних. Але пізніше в 2012 р. ця ж організація (тільки змінилася її назва – тепер вона називається *Gartner*) додала до цієї тріади слово «value» – оцінка (верифікація) даних і тепер модель великих масивів даних називається моделлю 4V

Задачі великих даних прийнято визначати за відповідями на чотири питання:

1. Чи є обсяг даних великим?
2. Чи швидкість обробки даних може бути недостатньою для їх обробки?
3. Чи значення атрибутів даних є різноманітними, як і самі атрибути?
4. Чи вимагає обробка даних верифікації?

Позитивні відповіді на усі питання характеризують задачу аналізу даних як таку, що відноситься до категорії великих даних. Якщо на окремі питання дані негативні відповіді, тоді задача не пов’язана з великими даними, і рішення можна істотно спростити.

Для роботи з великими даними поширення набули програмні бібліотеки з відкритим вихідним кодом, такі як Apache Mahout або Spark. Проект Apache Mahout заснований на концепції масштабованих, розподілених архітектур, таких як Hadoop, що реалізують парадигму MapReduce по обробці великих

наборів даних за допомогою паралельних, розподілених алгоритмів з застосуванням серверів, об'єднаних в кластери.

Бібліотека Mahout надає консольний інтерфейс і Java API для масштабованих методів кластеризації, класифікації та спільної фільтрації. Mahout поширюється по ліцензії Apache, тобто бібліотеку можна застосовувати, коли згадка про ліцензію Apache додана у розроблювану програмну систему.

Як приклад можна назвати три задачі, які можна вирішити за допомогою програмної бібліотеки Mahout: генерування рекомендацій (пошук асоціативних правил); групування близьких за змістом документів (задача кластеризації); визначення жанрів музичних творів (задача класифікації).

1.3. Категорії атрибутів

Набір вхідних даних можна представити у вигляді таблиці, де екземпляри записані у рядках, значення атрибутів у стовпцях. Деякі специфічні атрибути можуть взаємно виключати наявність інших атрибутів, Якщо вхідні дані об'єднані з суміжних прикладних областей.

Так, якщо екземпляри є транспортними засобами, то кількість коліс є атрибутом, який можна застосувати до багатьох наземних транспортних засобів, в той час як кількість щогл – тільки для надводного транспорту. Під час побудови моделі класифікації для того, щоб навчити класифікатор відрізняти імовірні значення атрибутів від тих, які в принципі не можуть описувати об’єкти даного класу, свідомо використовують аномальні значення деяких атрибутів.

Вище було сказано, що існує широке розмаїття типів атрибутів, проте найбільш поширеними є числові і категоріальні. Числові атрибути іноді іменуються неперервними, дійсними або цілочисельними. Слід пам'ятати, що в математичному сенсі, наприклад, цілочисельні значення не є неперервними.

1.3.1. Категоріальний атрибут

Категоріальні атрибути характеризуються скінченою множиною значень (див. Таблицю 1.6), представлених у вигляді міток або назв, які іноді називаються номінальними значеннями (nominal від латинського слова "назва"). Крім числових і категоріальних атрибутів, також розрізняють порядкові і інтервальні.

Таблиця 1.6. Можливість проведення гри за погодних умов

Прогноз	**Температура**	**Вологість**	**Вітер**	**Гра**
сонячно	жарко	висока	немає	ні
сонячно	жарко	висока	є	ні

хмарно	жарко	висока	немає	так
дощ	тепло	висока	немає	так
дощ	холодно	нормальна	немає	так
дощ	холодно	нормальна	є	ні
хмарно	холодно	нормальна	є	так
сонячно	тепло	висока	немає	ні
сонячно	холодно	нормальна	немає	так
дощ	тепло	нормальна	немає	так
сонячно	тепло	нормальна	є	так
хмарно	тепло	висока	є	так
хмарно	жарко	нормальна	немає	так
дощ	Тепло	висока	є	ні

Наприклад, в даних про погоду атрибут «Прогноз» має категоріальний тип і значення – дощ, хмарно, сонячно. Можливі питання для атрибуту «Прогноз» зображені на Лістингу 1.3.

Лістинг 1.3. Питання і висновки для категоріального атрибуту

```
Питання для атрибуту Прогноз ({дощ, хмарно, сонячно}):
ЯКЩО { Прогноз = хмарно } = = сонячно → FALSE
ЯКЩО { Прогноз = хмарно } = = хмарно → TRUE
```

Зрозуміло, додавати, множити або порівнювати кількості букв немає сенсу. Правила, що містять такий атрибут, можна перевіряти тільки на рівність чи нерівність. Категоріальні атрибути іноді називають перераховуваними або дискретними.

1.3.2. Порядковий атрибут

Порядкові атрибути передбачають впорядкованість (ранжування) значень, до яких застосовується поняття «відстань». Наприклад, в даних про погоду атрибут «температура» має значення «холодно», «тепло» і «жарко», які розташовуються послідовно за змістом (див. Лістинг 1.4).

Важливо те, що значення «тепло» знаходиться між двома іншими. Є сенс порівнювати послідовності двох значень, при цьому додавання або віднімання позбавлене сенсу, оскільки різниця між жарким і теплим, така ж як і різниця між теплим і холодним. На Лістингу 1.4 зображені можливі питання для значень порядкового атрибута.

Лістинг 1.4. Питання і висновки для порядкового атрибута

```
Питання для атрибуту Температура (холодно < тепло < жарко):
ЯКЩО { Температура = холодно } = = жарко → FALSE
ЯКЩО { Температура = холодно } < жарко → TRUE
```

1.3.3. Інтервальний атрибут

Інтервальні атрибути мають чисельні значення, які не тільки впорядковані, а й вимірюються в фіксованих і рівних одиницях. Наприклад атрибут «температура» частіше записується в градусах (наприклад, за Цельсієм), а не у вигляді категорій – {холодно, тепло, жарко}. Для інтервальних атрибутів, можна розрахувати різницю між двома значеннями температури – 8˚С і 10˚С градусів, і порівняти з результатом різниці між іншими значеннями температури – 23˚С і 25˚С. Віднімання має сенс для порівнювання відмінностей в значеннях атрибуту «температура», при цьому знаходження суми градусів (18˚С або 48˚С) позбавлене сенсу.

1.3.4. Числовий атрибут

Наприклад, вимірювання відстані від одного об'єкта до іншого передбачає порівняння чисельних значень атрибута «координати». Для числових атрибутів дозволені будь-які математичні операції, в тому числі додавання і множення.

1.3.5. Комбінований атрибут

Деякі атрибути можуть складатися з групи атрибутів, які при загальному розгляді можуть відноситься до одного і того ж об'єкту або поняттю, однак при детальному розгляді послідовність їх значень може формувати різні значення для комбінованого атрибута. Дану особливість необхідно враховувати при застосуванні операцій сортування або циклічного зсуву по відношенню до запису послідовності атрибутів в описі кожного екземпляра даних, оскільки тим самим можна поміняти значення комбінованого атрибута. Наприклад, в часовому контексті атрибут «день» може включати значення – «наступний день», «попередній день», «наступний день тижня» або «один і той же день на наступному тижні», все з перерахованих атрибути описують день тижня, проте в контексті часу відносяться до різних понять. Відомості, які описують комбінований атрибут часто називаються метаданими – даними про дані.

1.3.6. Формат ARFF

Підготовка вхідних даних для інтелектуального аналізу даних зазвичай займає більшу частину часу, витраченого на весь процес інтелектуального аналізу даних. На початку інтелектуального аналізу даних необхідно об'єднати всі дані в набір екземплярів. У реальних додатках дані можуть бути об'єднані з різних прикладних областей. Наприклад, в маркетинговому

дослідженні часто необхідні дані відділу продажів, відділу виставлення рахунків і відділу обслуговування клієнтів.

Інтеграція даних з різних джерел, як правило, пов'язана з низкою проблем, пов'язаних з особливостями реєстрації даних і їх представлення, так в компаніях різні підрозділи застосовують різні стилі обліку даних, періоди часу реєстрації, типи даних, типи неточностей, стилі зберігання, ступеня стискання даних. Ідея інтеграції баз даних в масштабах всієї компанії реалізується в створенні «сховищем даних» (data warehousing), яке забезпечує єдину узгоджену точку доступу до корпоративних даних.

Сховища даних можуть містити не всі необхідні дані, і тоді, знадобляться дані, зібрані за межами компанії. Наприклад, демографічні дані можуть бути зібрані відділом маркетингу і продажів, а дані про погоду – веб-сервісом прогнозування погоди. Вибір рівня агрегування даних має вирішальне значення для побудови точних аналітичних моделей.

Розглянемо стандартний спосіб представлення наборів даних, у форматі Attribute-Relation File Format (ARFF). Існує також версія під назвою XRFF, яка успадковує ARFF і представляє екземпляри даних на мові розмітки XML.

1.3.6.1. Структура ARFF

На Лістингу 1.5 показано ARFF-файл з даними про погоду для прогнозування значення атрибуту класу «можливість проведення гри», за значеннями атрибутів ознак («прогноз», «температура», «вологість», «вітер»). Рядки, що починаються зі знака % є коментарями. Після коментарів на початку файлу задається назва набору атрибутів (@relation weather).

За назвою групи атрибутів слідує блок назв і типів атрибутів. За категоріальними атрибутами слідують перелік значень у фігурних дужках ({...}). Якщо значення містить пробіл, його необхідно записувати в лапках, наприклад, "День міста".

Лістинг 1.5. Структура ARFF з даними про погоду

```
% ARFF файл з даними про погоду та деякими числовими атрибутами
@relation weather % погода

@attribute outlook { sunny, overcast, rainy } % прогноз: сонячно, хмарно, дощ
@attribute temperature numeric % температура
@attribute humidity numeric % вологість
@attribute windy { true, false } % вітер: є, немає
@attribute play? { yes, no } % гра: так, ні

@data
% 14 instances
%
```

```
sunny, 85, 85, false, no
sunny, 80, 90, true, no
overcast, 83, 86, false, yes
rainy, 70, 96, false, yes
rainy, 68, 80, false, yes
rainy, 65, 70, true, no
overcast, 64, 65, true, yes
sunny, 72, 95, false, no
sunny, 69, 70, false, yes
rainy, 75, 80, false, yes
sunny, 75, 70, true, yes
overcast, 72, 90, true, yes
overcast, 81, 75. false, yes
rainy, 71, 91, true, no
```

Після ініціалізації типів атрибутів слідує блок @data, який вказує на початок перерахування значень атрибутів. Даний формат дозволяє шукати зв'язки між атрибутами ознак без вибору атрибуту класу в задачах навчання без вчителя, таких як пошук асоціативних правил і кластеризація.

Ініціалізація типів атрибутів в ARFF- файлі дозволяє автоматично перевіряти допустимі значення атрибутів. Екземпляри записуються по рядках, значення для кожного атрибута через кому. Якщо значення атрибутів відсутні, вони записуються в вигляді знаків питання – «значення, ?, значення».

1.3.6.2. Типи атрибутів в ARFF-файлі

Формат ARFF враховує два основних типи даних: категоріальний і числовий. Строковий і часовий типи є, відповідно, категоріальними і числовими, хоча перед тим, як їх застосовувати, рядки часто перетворюються в числову форму, наприклад, вектор. Реляційний тип об'єднує декілька атрибутів в один атрибут.

Числові атрибути часто розглядаються як порядкові і використовуються тільки для порівняння значень. Деякі методи навчання розглядають числові атрибути як порядкові і додають знаки нерівності для порівняння їх значень. Також числові атрибути можуть розглядатися як інтервальні для розрахунку відстані між значеннями.

Якщо необхідно обробляти числові атрибути так, якби вони вимірювалися інтервально (за шкалою), тоді виникає необхідність їх нормалізації. Нормалізація необхідна для приведення різних значень до фіксованого діапазону – зазвичай від нуля до одиниці – шляхом ділення всіх значень на максимальне значення або віднімання мінімального значення і ділення на діапазон між максимальним і мінімальним значеннями.

Інший метод нормалізації передбачає обчислення статистичного

середнього і стандартного відхилення значень атрибутів, віднімання середнього значення з кожного значення і розподіл результату на стандартне відхилення. Даний процес називається стандартизацією статистичної змінної і призводить до набору значень, для яких середнє значення дорівнює нулю, стандартне відхилення дорівнює одиниці.

Деякі методи інтелектуального аналізу, наприклад, методи регресії застосовуються тільки до інтервальних атрибутів, оскільки розраховують відстань між двома екземплярами по заданим значенням атрибутів. Будь-яку категоріальну (номінальну) кількість можна розглядати як чисельну при розрахунку міри відмінності.

Деякі методи інтелектуального аналізу можуть бути застосовані лише до категоріальних атрибутів. Наприклад, у випадку з контактним лінзами атрибут «вік» приймає категоріальні (номінальні) значення, в результаті можливі правила зображені на Лістингу 1.6.

Лістинг 1.6. Правила рекомендування лінз для категоріального атрибута

```
ЯКЩО Вік = = підлітки ТА Астигматизм = = немає ТА Рівень вологості очей = = нормальний ТОДІ Лінзи = м’які
ЯКЩО Вік = = Дорослі ТА Астигматизм = = немає ТА Рівень вологості очей = = нормальний ТОДІ Лінзи = м’які
```

На відміну від категоріального атрибута, якщо вік розглядати як порядковий атрибут (див. Лістинг 1.7), тоді два правила можна об’єднати в одне правило (див. Лістинг 1.8).

Лістинг 1.7. Умови для порядкового атрибута «вікова категорія»

```
підлітки < дорослі < старі
```

Лістинг 1.8. Правило рекомендування лінз для порядкового атрибута

```
ЯКЩО Вік ≤ Дорослий ТА Астигматизм = = немає ТА Рівень вологості очей = = нормальний ТОДІ Лінзи = м’які
```

Зображене на Лістингу 1.8 правило являється більш компактним і, отже, більш задовільним представленням кількох правил.

1.3.6.3. Строковий атрибут

Крім категоріальних і числових типів атрибутів формат ARFF допускає ще три типи атрибутів: строковий, часовий і реляційний (у вигляді посилання). Строковий тип (@attribute ___ string) відповідає текстовим даними (див. Лістинг 1.9).

Лістинг 1.9. Ініціалізація строкового типу

```
@attribute назва_строки string
```

У розділі даних (@data) серед інших значень, строкове значення записується в лапках. Перед лапками, додають зворотну косу риску \, наприклад – \ "вітряно \". Атрибут string може мати значення, яке є текстом, в тому числі документом. Атрибут string можна перетворити в множину числових атрибутів, по одному для кожного слова в рядку, значення якого дорівнює кількості повторень в тексті.

1.3.6.4. Часовий атрибут

Часовий тип ініціалізується наступним чином (див. Лістинг 1.10).

Лістинг 1.10. Ініціалізація часового типу

```
@attribute назва_дати date
```

Weka передбачає комбінований формат запису дати і часу – yyyy-MM-dd'T'HH: mm:ss, визначений стандартом ISO-8601, з чотирма цифрами для року, двома для місяця і двома для дня, потім буквою 'T' з подальшим зазначенням часу з двома цифрами для години, двом для хвилин і двома для секунд. У розділі даних @data, часові мітки вказуються у вигляді відповідного строкового представлення дати і часу, наприклад, 2004-04-03'T'12:00:00.

Хоча часові значення записуються в розділі @data як рядки, при програмному читанні ARFF-файлу вони перетворюються в числову форму. Часові атрибути можуть бути перетворені в різні формати, в тому числі в абсолютні часові мітки, такі як час дня чи день тижня, для виявлення періодичності в значеннях екземплярів даних.

1.3.6.5. Реляційний атрибут

Довідкові типи дозволяють описувати комбіновані атрибути, складені з групи вкладених атрибутів. Реляційний тип представляє опис в описі. Атрибут задається директивою @relational, за яким слідує блок вкладених атрибутів, який відображає структуру пов'язаних екземплярів. Наприклад, комбінований атрибут "упаковка" можна представити у вигляді набору вкладених атрибутів зі структурою, схожою з описом атрибутів ознак для моделі можливості проведення гри, проте без атрибуту класу Гра (див. Лістинг 1.11).

Лістинг 1.11. Ініціалізація реляційного типу

```
@attribute bag relational

@attribute outlook { sunny, overcast, rainy } % прогноз (сонячно, хмарно, дощ)
@attribute temperature numeric % температура
@attribute humidity numeric % вологість
@attribute windy { true, false } % вітер (є, немає)

@end bag
```

Рядок @end bag вказує на кінець перерахування вкладених атрибутів.

1.4. Набори даних

На Лістингу 1.12 зображено зміст ARFF-файлу з даними про погоду. Кожен екземпляр складається із значення ідентифікатора, двох послідовних екземплярів з вихідних даних про погоду і мітки класу. Кожне значення атрибута є рядком, який комбінує два екземпляра погодних умов, розділених символом "\n" для гри тривалістю в 2 дні.

Подібний набір даних може застосовуватися в іграх, які проводяться протягом тривалого інтервалу часу (наприклад, крикет може тривати 3-5 днів). Зверніть увагу, що при поєднанні декількох екземплярів у вигляді довідкового атрибута, порядок перерахування екземплярів не має значення. Вивчивши дані, можна прийти до висновку, що гру можна провести, якщо жоден із днів не є дощовим і принаймні один є сонячним.

Лістинг 1.12. ARFF-файл з реляційним атрибутом

```
% Multiple instance ARFF file for the weather data % Файл ARFF для даних про погоду
@relation weather % погода

@attributo bag_ID { 1, 2, 3, 4, 5, 6, 7 }
@attribute bag relational
  @attribute outlook { sunny, overcast, rainy } % прогноз (сонячно, хмарно, дощ)
  @attribute temperature numeric % температура
  @attribute humidity numeric % вологість
  @attribute windy { true, false } % вітер (є, немає)
@end bag
@attribute play? { yes, no } % гра (так, ні)

@data
% seven “multiple instance' instances
%
1, "sunny, 85, 85, false\nsunny, 80, 90, true", no
2, "overcast, 83, 86, false\nrainy, 70, 96, false", yes
3, "rainy, 68, 80, false\nrainy, 65, 70, true", yes
4, "overcast, 64, 65, true\nsunny, 72, 95, false", yes
5, "sunny, 69, 70, false\nrainy, 75, 80, false", yes
```

```
6, "sunny, 75, 70, true\novercast, 72, 90, true", yes
7, "overcast, 81, 75, false\nrainy, 71, 91, true", yes
```

1.4.1. Розріджені дані

Іноді більшість атрибутів мають значення 0 для більшості екземплярів. Наприклад, дані споживчого кошика містять покупки, вироблені клієнтами супермаркету. Незалежно від того, наскільки детальним є дослідження, клієнти ніколи не купують більше, ніж малу частину товарів, пропонованих магазином, тому в споживчому кошику майже для всіх товарів з асортименту магазину буде вказано 0.

Файл даних можна розглядати як матрицю, чиї рядки представляють клієнтів і стовпці – одиниці товару, при цьому матриця вважається розрідженою, якщо більша частина елементів матриці рівні 0. Інший приклад зустрічається в інтелектуальному аналізі тексту, де екземплярами є документи. Тут стовпці і рядки представляють документи і слова, а числа вказують, скільки разів в певному документі зустрічається певне слово. Більшість документів мають досить невеликий словник, тому більшість записів дорівнюють 0 (див. Лістинг 1.13).

Лістинг 1.13. Запис всіх нульових значень

```
0, X, 0, 0, 0, 0, Y, 0, 0, 0, “class A”
0, 0, 0, W, 0, 0, 0, 0, 0, 0, “class B”
```

Запис екземплярів даних у розрідженій матриці може виявитися громіздким. Замість того, щоб записувати всі екземпляри зі значеннями 0, ARFF підтримує запис ненульових значень атрибутів у вигляді комбінацій порядкових номерів і через пробіл значень атрибутів (див. Лістинг 1.14).

Лістинг 1.14. Запис тільки ненульових значень

```
{1 X, 6 Y, 10 “class A”}
{3 W, 10 “class B”}
```

Кожен екземпляр, розміщений у фігурних дужках {n1 v1, n2 v2, ...}, включає у себе номер індексу кожного ненульового атрибута (індекси починаються з 0) і його значення. Опис розріджених даних включає розділи @relation і @attribute, за якими слідує рядок @data, але розділ даних відрізняється і містить специфікації в фігурних дужках, як показано на Лістингу 1.14. Зверніть увагу, що на Лістингу 2.10 пропущені значення записані як 0 і не вважаються «відсутніми» значеннями! Якщо значення невідоме, воно повинно бути явно представлено знаком "?".

1.4.2. Неповні дані

Більшість наборів даних, що зустрічаються на практиці, наприклад дані про трудові контракти представлені в Таблиці 1.7, містять відсутні значення. Відсутні значення в записах часто опиняються поза діапазоном, зазначеним в розділі ініціалізації типів (наприклад, від'ємні числа для додатніх значень атрибута або нульові числа для ненульових значень атрибута). Для категоріальних атрибутів відсутні значення можуть бути позначені пропусками або дефісами. Також є різні типи відсутніх значень, наприклад, невідомі, незаписані, нерелевантні), які можуть бути представлені різними від'ємними цілими числами (-1, -2, і т.д.).

Таблиця 1.7. Дані по трудових контрактах

Атрибут	**Тип**	1	2	3	…	40
Тривалість	(кількість років)	1	2	3		2
Збільшення заробітної плати 1-й рік	процент	2%	4%	4.3%		4.5
Збільшення заробітної плати 2-й рік	процент	?	5%	4.4%		4.0
Збільшення заробітної плати 3-й рік	процент	?	?	?		?
Час роботи на тиждень	(кількість годин)	28	35	38		40
Допомога на освіту	{так, ні}	так	?	?		?
Нормативні свята	(кількість днів)	11	15	12		12
Довгострокова допомога по інвалідності	{так, ні}	ні	?	?		так
Внесок стоматологічного плану	{ніяка, часткова, повна}	ніяка	?	повна		повна
Допомога по збитку	{так, ні}	ні	?	?		так

Необхідно ретельно подумати про те, яким чином позначати відсутні значення. Вони можуть виникати з ряду причин, таких як несправність вимірювального обладнання, або, скажімо, коли респонденти під час опитування можуть відмовитися відповідати на певні питання, такі як вік або дохід. В археологічному дослідженні зразок черепа може виявитися пошкоджений, у результаті чого деякі атрибути параметрів будуть не

доступні для вимірювання.

У біологічному середовищі рослини або тварини можуть загинути до того, як будуть виміряні всі атрибути. Часто виникає питання, як оцінити важливість тих атрибутів, які відсутні в даних? Наприклад, чи може пошкодження черепа мати якесь значення для узагальнюючих висновків або це просто наслідок якоїсь випадкової події? Або чи може рання смерть рослини мати якийсь стосунок до справи дослідження закономірностей змін в рослинному світу чи ні?

Більшість методів інтелектуального аналізу роблять неявне припущення про відсутність значення через невідомість. Однак, можливо, є вагомі причини, за якими значення атрибута відсутнє, або, можливо, було прийнято рішення з певних причин не проводити заміри конкретного атрибута, і цей факт може мати значення для отримання додаткової інформації про екземпляр. В таких випадках більш коректним вважається такий спосіб представлення величин атрибутів, в якому явно вказуються певні атрибути, величини яких не несуть додаткової інформації.

1.4.3. Неточні дані

Інколи значення категоріального атрибуту записані неповністю або помилково, це дозволяє припустити існування додаткових значень атрибуту. Може виявитися, що на справді справа в різних назвах того самого об'єкта, наприклад, York, New York, New York city. Щоб передбачити такі ситуації, дослідник має продумати питання, яким чином уникнути таких помилок в переліку можливих величин тих чи інших атрибутів.

Числові атрибути можуть містити аномальні викиди, які найбільш просто можуть бути виявлені лише шляхом візуального аналізу експертом. Інколи похибки та неточності в даних обумовлені свідомими діями людей або обмеженнями комп’ютерних систем обробки даних.

Якщо клієнт магазину забув дисконтну картку, продавець магазину може застосувати свою картку, або щоб клієнт міг отримати знижку, або просто щоб накопичити кредитні бали на власному рахунку. Тільки семантична класифікація предметної області, яка аналізується, може пояснити неоднозначності в даних, подібні описаним вище.

Варто пам'ятати, що дані мають властивість змінюватися з часом. Наприклад, часто змінюються значення атрибутів у екземплярів зі списку розсилки, такі як адреси, номери телефонів тощо. Тому на етапі підготовки даних необхідно перевіряти, чи всі дані є актуальними.

1.4.4. Незбалансовані дані

При застосуванні методів класифікації часто трапляється, що певне

значення атрибуту класу набагато більш поширене, ніж інші. Припустимо, що, якщо одне і те ж значення атрибуту класу прогнозується для більшості екземплярів, тоді точність прогнозу може складати навіть ~ 99%. Важко уявити, що пошук більш складного правила поліпшить результат. Ймовірно, можна поліпшити результат прогнозування на 1% знайшовши правила для меншої кількості екземплярів, однак ці правила спричинять помилки для неврахованих екземплярів. Перевага кращих прогнозів для 1% екземплярів напевно буде нівельована точністю правил, знайдених для 99% випадків.

Проблема полягає в тому, що точність моделі, вимірювана часткою правильних прогнозів, не обов'язково є найкращим критерієм успіху. Важливість прогнозування ядерної катастрофи, яка може не відбутися, може виявитися чималою, проте несумісної за важливістю з відсутністю прогнозу для ядерної катастрофи, в разі якщо вона відбудеться. Отже поняття коректності моделі прогнозування є відносним для різних прикладних задач.

1.4.5. Аномальні дані

В практиці застування методів інтелектуального аналізу часто є корисними прості інструменти візуалізації даних, які відображаються гістограмами розподілу значень категоріальних атрибутів і графіками значень числових атрибутів. Візуалізація даних дозволяє ідентифікувати викиди, наприклад відсутній рік як 9999 або відсутню вагу як -1 кг, які цілком можуть представляти помилки у файлі даних або невідомі закономірності для незвичайних ситуацій.

Для пояснення аномальних або відсутніх значень категоріальних атрибутів, представлених у вигляді цілих чисел, необхідно консультуватися з профільними експертами. При помилках в числових атрибутах, відхилення можна ідентифікувати, побудувавши попарні графіки для значень двох атрибутів ознак або атрибуту ознаки по відношенню до значення атрибуту класу.

Очищення даних – трудомістка процедура, але є необхідною для успішного аналізу даних. При наявності великих наборів даних, можна спочатку вибрати кілька екземплярів і ретельно вивчити значення їх атрибутів, а вже потім приступати до аналізу всіх доступних даних.

Розділ 2. Завдання до експериментальних досліджень з інтелектуального аналізу даних з використанням програмного пакету Weka

2.1. Знайомство з методами класифікації даних

Мета роботи: Ознайомитися та отримати навички побудови моделей класифікації за допомогою Data Mining GUI бібліотеки Weka.

Завдання: Побудувати модель класифікації за допомогою методу класифікації.

2.1.1. Сутність класифікації

Метод класифікації – це метод аналізу даних, який дозволяє оцінити ймовірність приналежності екземплярів даних до деякого класу залежно від значень їх атрибутів. Як модель класифікації рекомендується використовувати структуру даних «дерево» (див. Лістинг 2.1), в якій кожен вузол являє собою точку прийняття рішення на підставі значень атрибутів даних, що класифікуються.

Лістинг 2.1. Приклад дерева класифікації

На Лістинг 2.1 наведено дерево класифікації, яке дає відповідь на питання «Ви купите книгу?». У кожному вузлі ставиться уточнююче питання (до атрибуту) з відповідями (значення атрибуту) у гілках, відповідаючи ви переходите до наступного вузла (питання, атрибуту) до тих пір, поки не дійдете до листа зі значенням класу, у прикладі це відповіді «Купите» чи «Не купите» книгу. Перевага класифікаційних дерев полягає у тому, що вони не вимагають надмірної кількості інформації для побудови досить точного та інформативного дерева рішень.

Метод класифікації використовує відомі значення атрибутів екземплярів

даних та зв'язки між їх значеннями при побудові моделі класифікації. При наявності *нових* екземплярів даних невідомого класу, до даних застосовується раніше побудована модель класифікації і визначається відповідний клас.

2.1.2. Підготовка даних

Набір даних, який буде застосований для прикладу класифікаційного аналізу, містить інформацію (bmw-training.arff, bmw-test.arff), зібрану центром продажу компанії BMW. Центр починає рекламну компанію, пропонуючи розширену дворічну гарантію своїм постійним клієнтам. Подібні компанії вже проводилися, так що центр продажу має у розпорядженні 4500 екземплярів даних щодо попередніх продажів з розширеною гарантією. Цей набір даних охоплює наступні атрибути:

– Розподіл за доходами [0=$0-$30k, 1=$31k-$40k, 2=$41k-$60k, 3=$61k-$75k, 4=$76k-$100k, 5=$101k-$150k, 6=$151k-$500k, 7=$501k+];
– Рік / місяць покупки першого автомобіля BMW;
– Рік / місяць покупки останнього автомобіля BMW;
– Чи скористався клієнт розширеною гарантією?

Файл даних у форматі Attribute-Relation File Format (ARFF) буде виглядати наступним чином, див. Лістинг 2.2.

Лістинг 2.2. Файл даних для класифікаційного аналізу у Weka

```
@attribute IncomeBracket {0,1,2,3,4,5,6,7} % Групи за доходом
@attribute FirstPurchase numeric % перша покупка
@attribute LastPurchase numeric % остання покупка
@attribute responded {1,0} % відгук

@data

4,200210,200601,0
5,200301,200601,1
...
```

Тестові дані у форматі ARFF можна знайти за адресою:

http://repository.seasr.org/Datasets/UCI/arff/ .

При побудові моделі класифікації набір даних зазвичай ділять так, щоб частина даних використовувалася для побудови моделі (навчання), частина – для перевірки її коректності (тестування), щоб переконатися, що модель не є

навченою тільки під конкретний набір даних.

Розділіть вибраний набір даних на два файли *.arff в співвідношенні «2/3» (навчальні дані bmw-training.arff) та «1/3» (тестові дані bmw-test.arff) від загальної кількості даних. Завантажте файл «2/3» в програмний пакет Weka.

2.1.3. Завантаження даних у Weka Explorer

Коли файл з навчальними даними готовий, його потрібно завантажити у Weka. Запустіть Weka і виберіть опцію Explorer. У результаті відкриється закладка Preprocess у вікні Explorer. Натисніть кнопку Open File і виберіть створений вами ARFF-файл. Вікно Weka Explorer з завантаженими даними показано на Рисунку 2.1.

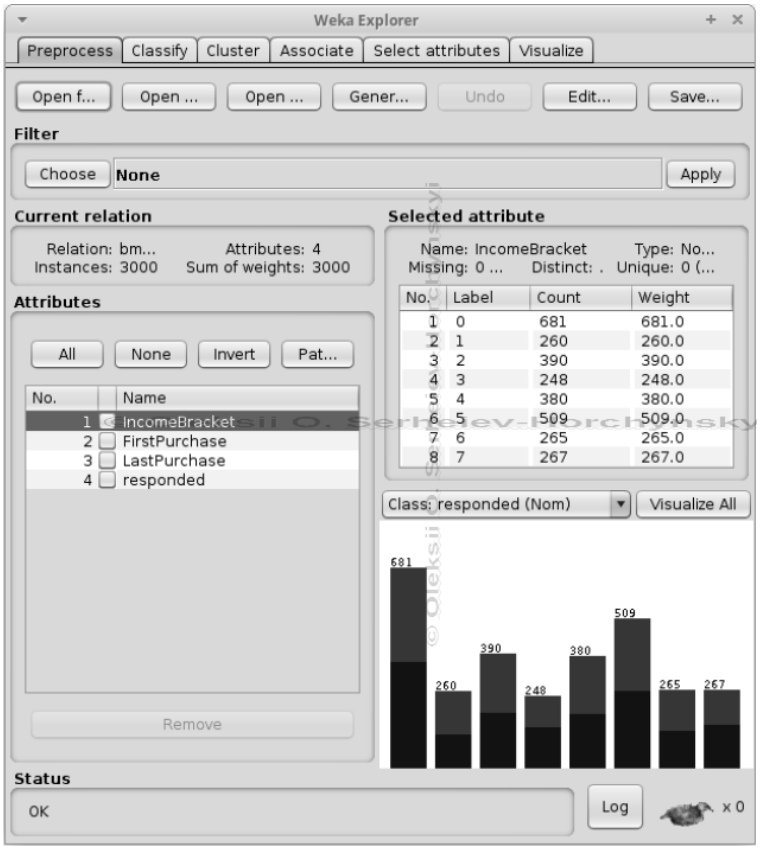

Рисунок 2.1. Дані дилерського центру авто

У цьому вікні ви можете перевірити дані, на підставі яких ви збираєтесь будувати модель. У лівій частині вікна Explorer показані атрибути даних (Attributes), які відповідають заголовкам стовпців таблиці, також вказано кількість екземплярів даних (Instances), тобто рядків таблиці. Якщо виділити мишкою однин з заголовків стовпців, тоді в правій частині вікна з'являться значення відповідного атрибуту для різних екземплярів даних. Також існує

можливість візуального аналізу даних за допомогою кнопки Visualize All.

2.1.4. Побудова моделі класифікації у Weka Explorer

Виберіть метод класифікації (див. Рисунок 2.2): відкрийте закладку Classify, виберіть опцію trees, потім опцію з відповідним номером Вашої залікової книжки (див. Таблицю 2.1).

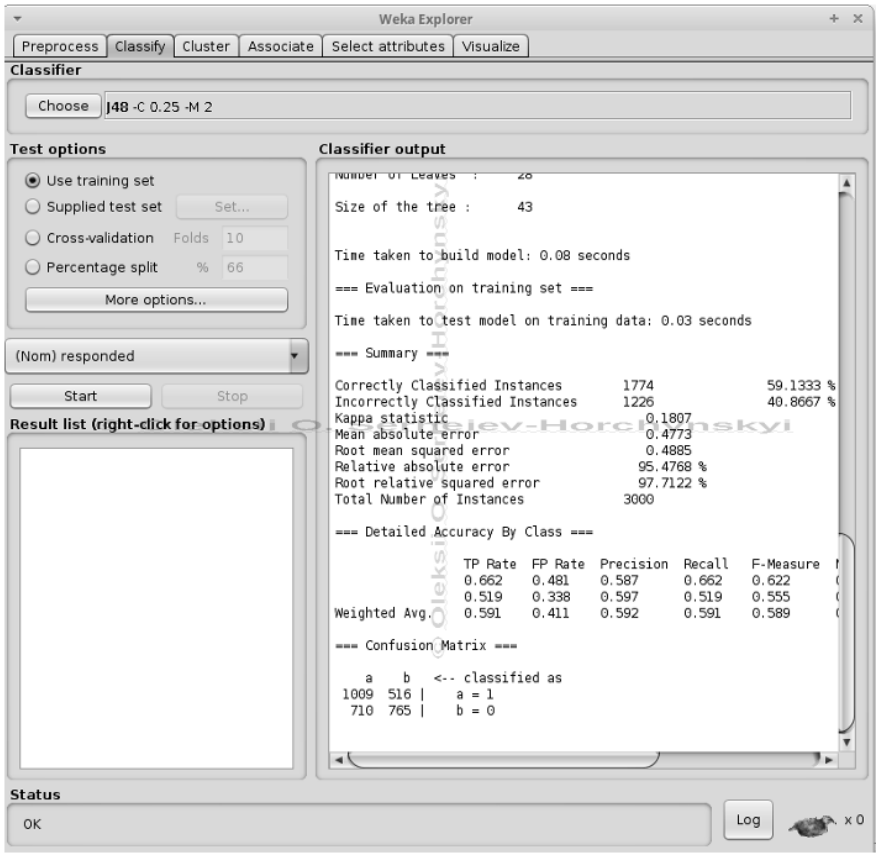

Рисунок 2.2. Вибір методу класифікації даних

Тепер можна розпочати побудову моделі класифікації засобами пакету Weka. Переконайтеся, що обрана опція Use training set, для того щоб пакет Weka при побудові моделі використовував саме ті дані, які ви тільки що завантажили з файлу. Натисніть Start. Результуюча модель повинна виглядати так, як показано на Лістинг 2.3.

Лістинг 2.3. Результат роботи класифікаційної моделі Weka

```
Number of Leaves  :     28
Size of the tree :      43
Time taken to build model: 0.18 seconds
=== Evaluation on training set ===
```

```
=== Summary ===
Correctly Classified Instances        1774               59.1333 %
Incorrectly Classified Instances      1226               40.8667 %
Kappa statistic                          0.1807
Mean absolute error                      0.4773
Root mean squared error                  0.4885
Relative absolute error                 95.4768 %
Root relative squared error             97.7122 %
Total Number of Instances             3000
=== Detailed Accuracy By Class ===
               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.662    0.481      0.587     0.662     0.622      0.616    1
                 0.519    0.338      0.597     0.519     0.555      0.616    0
Weighted Avg.    0.591    0.411      0.592     0.591     0.589      0.616
=== Confusion Matrix ===
   a     b   <-- classified as
 1009   516 |   a = 1
 710    765 |   b = 0
```

На Рисунку 2.3 представлена побудована модель класифікації.

Модель класифікації (див. Рисунок 2.3) побудована за навчальним набором даних (training set). Цей підхід використовує відомі дані для аналізу зв'язків значень атрибутів навчальних екземплярів даних. Таким чином, коли з'являється новий екземпляр даних (набір значень атрибутів) невідомого класу, потрібно лише провести аналіз значень атрибутів за допомогою побудованої моделі класифікації та визначити відповідний клас.

При побудові моделі класифікації зазвичай використовують набір даних, який ділиться на дві частини. Перша частина (60-80% або 2/3 даних) використовується як навчальний набір для побудови моделі. Після цього тестові дані класифікуються за допомогою побудованої моделі та порівнюються з їх дійсними класами. Такий підхід дозволяє оцінити точність побудованої моделі класифікації.

2.1.5. Тестування моделі класифікації

Тестова перевірка моделі класифікації дозволяє уникнути зайвого перенавчання моделі. Оскільки модель класифікації будується для класифікації некласифікованих екземплярів, при перевірці її оптимальності використовується тестовий набір даних. Таким чином, гарантується, що побудована модель класифікації зможе з досить високою ймовірністю визначити клас ще некласифікованого екземпляру.

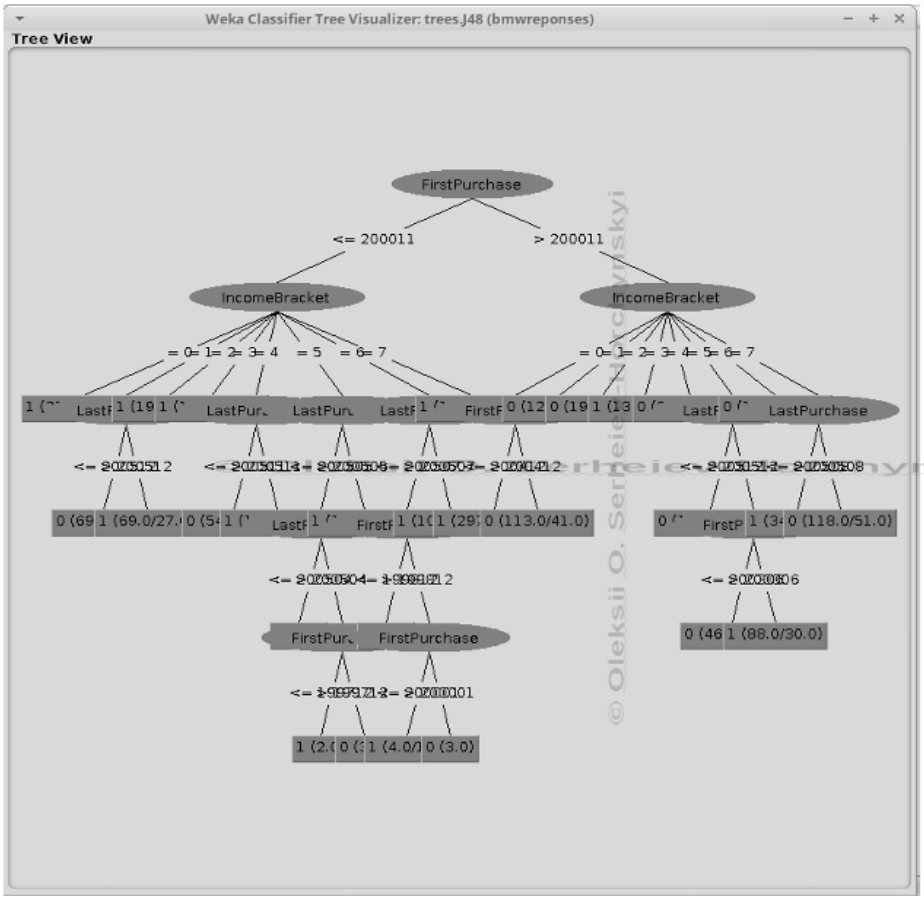

Рисунок 2.3. Графічне відображення моделі класифікації

Перевірку моделі треба провести на тестовому наборі даних «1/3» і оцінити, наскільки результати класифікації відрізняються від тестових класів. Для цього в секції Test options виберіть опцію Supplied test set і натисніть Set. Вкажіть файл з даними, що містить тестові «1/3» дані, які не були включені в навчальний набір. При натисканні на кнопку Start, Weka проведе класифікацію тестових даних і виведе інформацію про оптимальність побудованої моделі (див. Рисунок 2.4).

Порівнюючи показник Correctly Classified Instances для тестового набору (55,7%) з цим же показником для навчального набору (59,1%), видно, що точність моделі для двох різних наборів даних приблизно однакова. Це означає, що нові дані, які будуть класифікуватися за допомогою цієї моделі в майбутньому, не зменшать точність її роботи. Для підвищення точності класифікації рекомендується збільшити кількість навчальних і тестових даних та число атрибутів класів.

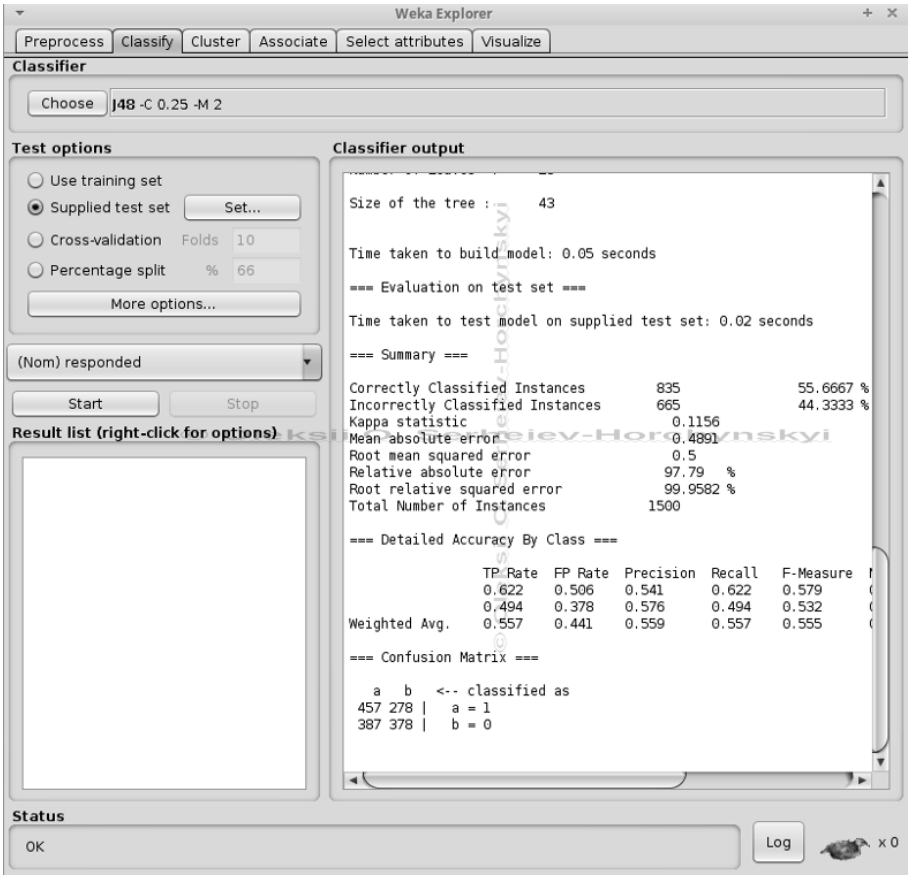

Рисунок 2.4. Перевірка моделі класифікації

2.1.6. Зміст протоколу виконаної роботи

Таблиця 2.1. Варіанти завдання

Номер	Метод	Номер	Метод
1	Id3	11	NBTree
2	J48	12	REPTree
3	J48graft	13	Id3
4	LADTree	14	J48
5	NBTree	15	J48graft
6	REPTree	16	LADTree
7	J48graft	17	REPTree
8	J48	18	NBTree
9	Id3	19	J48graft
10	LADTree	20	Id3

1. Титульний аркуш.
2. Мета роботи.
3. Інформація про дані та атрибути екземплярів даних.

4. Скриншот статистики за даними.
5. Скриншот опису побудованої моделі класифікації, скриншот графічного відображення моделі.
6. Скриншот перевірки побудованої моделі.
7. Висновки по роботі.

2.2. Програмна розробка методу класифікації даних

Мета роботи: Здобути навички програмної розробки методу класифікації.

Завдання: Розробити програму для класифікації даних.

В першому завданні Ви здобули навички застосування методів класифікації, які доступні в пакеті Weka. В даному завданні Вам потрібно розробити програмну реалізацію одного з методів класифікації – «наївний» класифікатор Байєса.

2.2.1. Метод класифікації «наївний» класифікатор Байєса

Вирішується задача класифікації листів з електронної скриньки на наявність спаму. Деякі електронні листи вже мають відмітку «Спам», інші вважаються – «Не спам». Необхідно розробити програму, яка буде аналізувати назви нових листів і повідомляти про можливість спаму з певною ймовірністю.

Назви електронних листів з навчальної вибірки:

Лістинг 2.4. Навчальні дані

```
Назва 1:          New meeting tomorrow (file)
Назва 2:          Corporate party tomorrow
Назва 3 (SPAM): Free sales party
Назва 4 (SPAM): Free file for you
Назва 5:          New greeting text
Назва 6 (SPAM): Free file upload
```

Аналіз відношення повідомлення до певного класу можна виконати за різними показниками, в тому числі за кількістю букв в слові, відношенням слова до певної частини мови, порядковим номером в реченні тощо.

При аналізі зустрічальності слів складається словник з переліку власне слів та кількості їх повторень, визначаються класи, в нашому прикладі їх буде два – «Спам» та «Не спам».

Таблиця 2.2. Словник слів та кількість їх повторень

Слово	**Кількість**
tomorrow	2
free	3
file	3
new	2
party	2
інші	1

2.2.2. Ймовірність класу слова

Розрахована кількість повторень слів розділяється на повторення кожного з перерахованих класів.

Таблиця 2.3. Словник слів та кількість їх повторень за класами

Слово	**Кількість екземплярів класу “Не спам”**	**Кількість екземплярів класу “Спам”**
tomorrow	2	0
free	0	3
file	1	2
new	2	0
party	1	1

Використовуючи кількість повторень слів можна розрахувати ймовірність їх відношення до певного класу.

Таблиця 2.4. Словник слів з ймовірностями класів

Слово	**Ймовірність класу “Не спам”, 0.5**	**Ймовірність класу “Спам”, 0.5**
tomorrow	1	0
free	0	1
file	0.33	0.66
new	1	0
party	0.5	0.5

У нашому прикладі побудова моделі класифікації «наївний» класифікатор Байєса ґрунтується на статистиці слів у назвах електронних листів.

2.2.3. Ймовірність класу повідомлення

Для того, щоб класифікувати назву листа (повідомлення) необхідно мати словник слів та ймовірності їх відношення до певних класів. Якщо класів

повідомлень є два, як у даному прикладі з однаковою кількістю речень для обох класів, ймовірність класу буде дорівнювати $p_{класу\_спам} = 0.5$, в випадку 3-х значень атрибуту класу, ймовірність кожного класу дорівнює $p_{значення\_класу} = 1/3 = 0.33$. Для кожного класу потрібно розрахувати загальну ймовірність відношення повідомлення до класу за наступною формулою:

$$p_{повідомлення\_до\_класу} = p_{класу} \times \prod_{i=1}^{n} p_{слова\_до\_класу}[i], \tag{2.1}$$

де $p_{повідомлення\_до\_класу}$ – ймовірність відношення тестового повідомлення до класу, $p_{класу}$ – ймовірність класу, $p_{слова\_до\_класу}$ – ймовірність відношення слова у тестовому реченні до певного класу.

2.2.4. Нормована ймовірність класу слова

Якщо ймовірність хоча б одного слова у назві рівна нулю загальна ймовірність також буде рівна нулю. Щоб уникнути цього, усі ймовірності у словнику можна нормувати за наступною формулою:

$$p_{слова\_нормована} = \frac{n_{повторень\_слова} \cdot p_{слова\_ненормована} + p_{класу}}{n_{повторень\_слова} + 1}, \tag{2.2}$$

де $p_{слова\_нормована}$ – нормована ймовірність відношення слова до класу, $p_{слова\_ненормована}$ – ненормована ймовірність відношення слова до класу, $p_{класу}$ – ймовірність класу (для прикладу з листами $p_{класу\_спам} = 0.5$), $n_{повторень\_слова}$ – кількість повторень слова у навчальному наборі.

Таблиця 2.5. Словник слів з нормованими ймовірностями класів

Слово	**Нормована ймовірність класу “Не спам”, 0.5**	**Нормована ймовірність класу “Спам”, 0.5**
tomorrow	0.83	0.16
free	0.13	0.87
file	0.37	0.62
new	0.83	0.16
party	0.5	0.5

2.2.5. Класифікація тестового повідомлення

Нехай модель класифікації вже побудована і необхідно визначити, чи відноситься назва нового листа до класу «Спам»?

Назва листа:

«Free file tomorrow»

Розрахуємо загальну ймовірність класу «Не спам»:

$$p_{не\ спам} = 0.5 \cdot (0.13 \cdot 0.37 \cdot 0.83) = 0.019$$

Розрахуємо загальну ймовірність класу «Спам»:

$$p_{спам} = 0.5 \cdot (0.87 \cdot 0.62 \cdot 0.16) = 0.043$$

Оскільки ймовірність класу «Спам» більша, ніж «Не спам», можна зробити висновок, що електронних лист відноситься до класу «Спам».

2.2.6. Зміст протоколу виконаної роботи

Варіанти завдання

№	Тематики	№	Тематики	№	Тематики
1	Транспорт, здоров'я, шоу-бізнес	11	Музика, кіно, театр	21	Автоспорт, кіно, шоу-бізнес
2	Нерухомість, фінанси, енергетика	12	Енергетика, нерухомість, кіно	22	Інтернет, космос, наука
3	Автоспорт, хокей, баскетбол	13	Транспорт, здоров'я, шоу-бізнес	23	Нерухомість, фінанси, енергетика
4	Музика, кіно, театр	14	Нерухомість, фінанси, енергетика	24	Музика, наука, баскетбол
5	Інтернет, космос, наука	15	Хокей, фінанси, кіно	25	Хокей, фінанси, кіно
6	Автоспорт, кіно, шоу-бізнес	16	Автоспорт, хокей, баскетбол	26	Автоспорт, кіно, шоу-бізнес
7	Хокей, фінанси, кіно	17	Інтернет, космос, наука	27	Музика, наука, баскетбол
8	Енергетика, нерухомість, кіно	18	Музика, наука, баскетбол	28	Транспорт, здоров'я, шоу-бізнес
9	Музика, наука, баскетбол	19	Енергетика, нерухомість, кіно	29	Здоров'я, театр, космос
10	Здоров'я, театр, космос	20	Транспорт, здоров'я, шоу-бізнес	30	Транспорт, енергетика, баскетбол

1. Розробіть програму для класифікації даних.
2. За варіантом підготуйте вибірку повідомлень, наприклад, статті, анотації.
3. Підготуйте навчальний набір повідомлень (2/3 від об'єму вибірки N) для побудови моделі класифікації.
4. Підготуйте тестовий набір повідомлень (1/3 від об'єму вибірки N).
5. Побудуйте графік залежності проценту коректних класів, визначених за допомогою моделі класифікації, від об'єму вибірки N.
6. Наведіть код програми.
7. Сформулюйте висновок щодо адекватності моделі класифікації.

2.3. Знайомство з методами кластеризації даних

Мета роботи: Ознайомитися та отримати навички кластеризації даних за допомогою Data Mining GUI бібліотеки Weka.

Завдання: Виконати кластеризацію тестових даних за допомогою методу кластеризації.

2.3.1. Сутність кластеризації

Метод кластеризації – це метод аналізу даних, який дозволяє розділити екземпляри даних за значеннями їх атрибутів на класи, кожен з яких має певні ознаки. Кластерний аналіз використовується в тих випадках, коли необхідно автоматично виділити деякі правила, взаємозв'язки або тенденції у сукупності даних. Як модель кластеризації можна використовувати двовимірний простір Евкліда, в якому відносне скупчення екземплярів даних являє собою певний клас (див. Рисунок 2.5).

Просторовий підхід використовує *некласифіковані* екземпляри для аналізу зв'язків між значеннями їх атрибутів. Побудовану модель кластеризації можна використовувати для оцінювання приналежності нових екземплярів даних до вже відомих кластерів.

2.3.2. Підготовка даних

Набір даних, який буде застосований для виконання кластерного аналізу, містить інформацію про усіх відвідувачів демонстраційного залу, про машини, які їх зацікавили, про те, наскільки часто відвідувачі купували цікаві їм автомобілі. Тепер дилерському центру треба проаналізувати ці дані для

того, щоб виділити різні групи відвідувачів і зрозуміти, чи можна визначити деякі тенденції в їх поведінці.

У демонстраційному прикладі використовується 100 екземплярів, і кожен стовпець описує певний крок, який, як правило, проходить покупець у процесі вибору та купівлі автомобіля. Відповідно, значення 1 та 0 показують чи відвідувач виконав конкретну дію, чи ні. Частина опису тестових даних у форматі ARFF зображена на Лістинг 2.5.

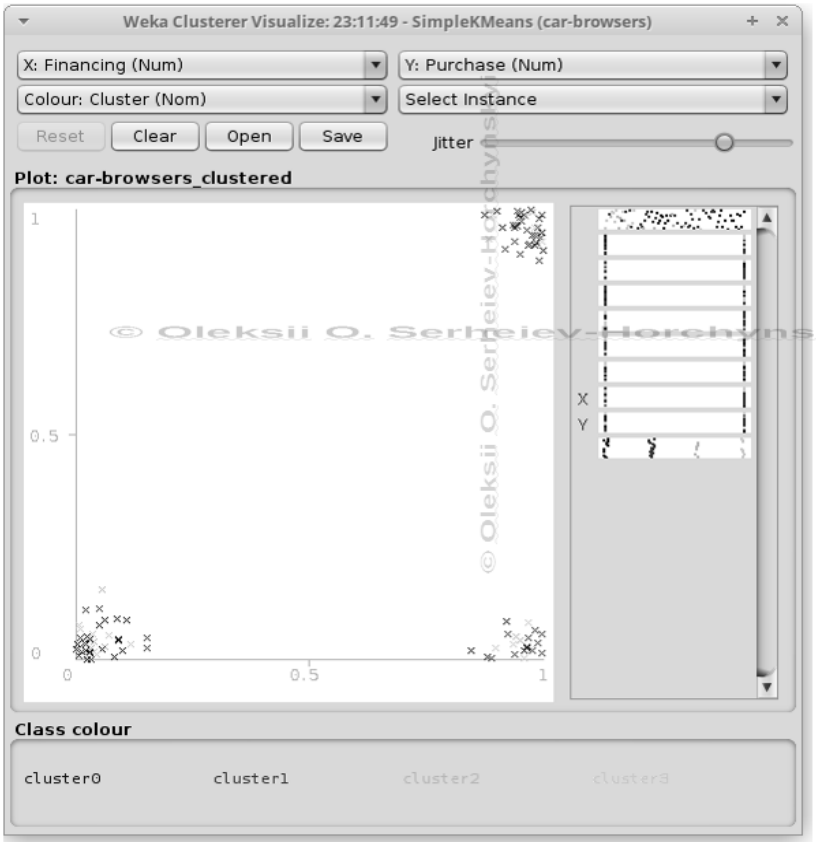

Рисунок 2.5. Приклад результату кластеризації екземплярів даних

Лістинг 2.5. Файл даних для кластеризації у Weka

```
@attribute Dealership numeric % автосалон
@attribute Showroom numeric % стенд
@attribute ComputerSearch numeric % попередній пошук
@attribute M5 numeric % модель
@attribute 3Series numeric % серія
@attribute Z4 numeric % модифікація
@attribute Financing numeric % кредит
@attribute Purchase numeric % покупка

@data
```

```
1,0,0,0,0,0,0,0
1,1,1,0,0,0,1,0
...
```

Тестові дані у форматі ARFF можна знайти за адресою:

http://repository.seasr.org/Datasets/UCI/arff/ .

При виконанні кластеризації кожен атрибут має бути приведений до нормального вигляду. Для цього кожен показник ділиться на різницю між найбільшим і найменшим значенням, які приймає розглянутий атрибут для конкретного набору даних. Наприклад, якщо розглянутий атрибут – кількість років, відповідно при найбільшому значені – 72, найменшому – 16. Нормалізоване значення атрибута зі значенням 32 дорівнює – 32 / (72-16) = 0.5714.

2.3.3. Завантаження даних у Weka Explorer

Коли файл з навчальними даними готовий, його потрібно завантажити у Weka. Запустіть Weka і виберіть опцію «Explorer». В результаті відкриється закладка «Preprocess» вікна «Explorer». Натисніть кнопку Open File і виберіть створений вами ARFF-файл. Вікно Weka Explorer з завантаженими даними показано на Рисунку 2.6.

У цьому вікні ви можете перевірити дані, на підставі яких ви збираєтеся будувати модель. У лівій частині вікна Explorer показані параметри об'єктів (Attributes), які відповідають заголовкам стовпців вихідної таблиці, також вказано кількість об'єктів (Instances), тобто рядків таблиці. Якщо виділити мишкою один з заголовків стовпців, тоді в правій панелі буде виведена повна інформація про набір даних в даному стовпці. Також є можливість візуального аналізу даних (Visualize All).

2.3.4. Побудова моделі кластеризації у Weka Explorer

Виберіть метод кластеризації: відкрийте закладку Cluster, в меню виберіть опцію відповідно до варіанту (див. Таблицю 2.6). В результаті вікно Weka Explorer буде виглядати так, як зображено на Рисунку 2.7.

Тепер можна розпочати виконання кластерного аналізу засобами пакету Weka. Переконайтеся, що обрана опція «Use training set», для того щоб пакет Weka при створенні моделі використовував саме ті дані, які ми тільки що

завантажили у вигляді файлу.

Тепер потрібно вибрати необхідні параметри методу кластеризації. Натисніть на назві методу та вкажіть значення в полі numClusters, яке визначає кількість кластерів для розділення (нагадуємо, що це значення потрібно вибрати ще до створення моделі). Натисніть кнопку OK, щоб зберегти вибрані параметри.

Натисніть кнопку Start. Результуюча модель повинна виглядати так, як показано на Лістинг 2.6.

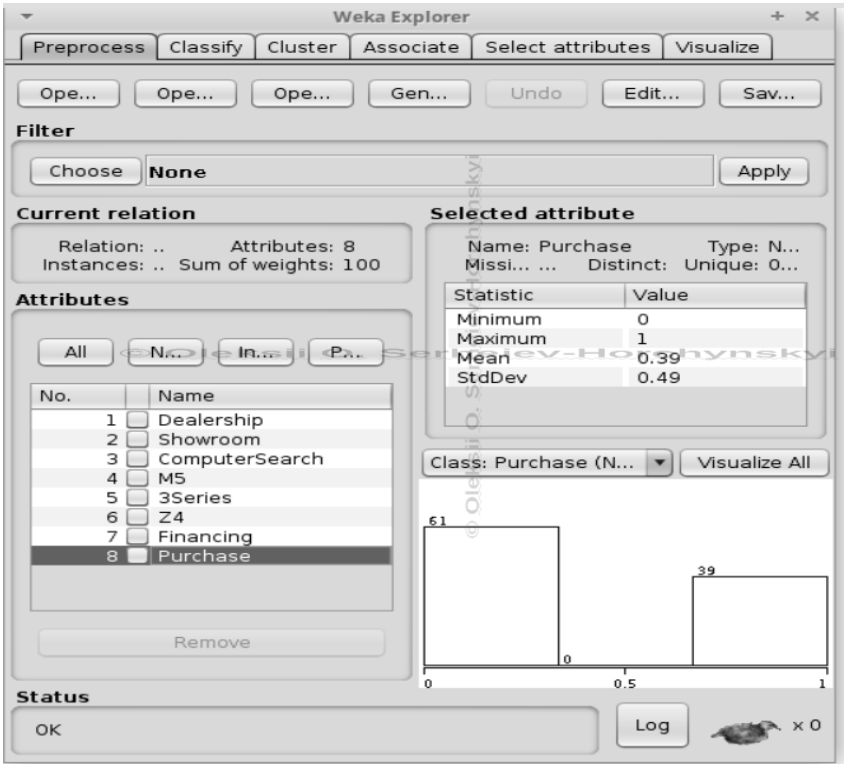

Рисунок 2.6. Дані дилерського центру авто для кластеризації

Лістинг 2.6. Результат виконання кластеризації у Weka

Attribute	Full Data (100)	Cluster# 0 (26)	1 (27)	2 (5)	3 (14)	4 (28)
Dealership	0.6	0.9615	0.6667	1	0.8571	0
Showroom	0.72	0.6923	0.6667	0	0.5714	1
ComputerSearch	0.43	0.6538	0	1	0.8571	0.3214
M5	0.53	0.4615	0.963	1	0.7143	0
3Series	0.55	0.3846	0.4444	0.8	0.0714	1
Z4	0.45	0.5385	0	0.8	0.5714	0.6786
Financing	0.61	0.4615	0.6296	0.8	1	0.5
Purchase	0.39	0	0.5185	0.4	1	0.3214

```
Clustered Instances
0       26 ( 26%)
1       27 ( 27%)
2        5 (  5%)
3       14 ( 14%)
4       28 ( 28%)
```

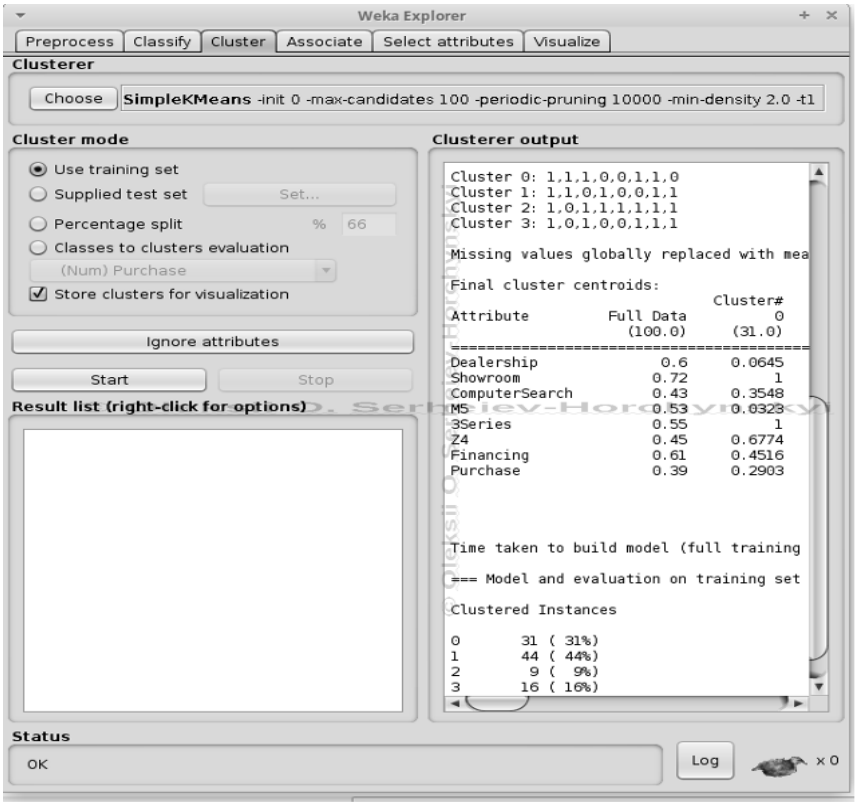

Рисунок 2.7. Вікно вибору методу кластеризації даних

Один зі способів аналізу результатів кластеризації – це візуальне подання даних (див. Рисунок 2.8). Натисніть правою клавішею миші в секції Result List закладки Cluster. У контекстному меню виберіть опцію Visualize Cluster Assignments.

В результаті відкриється вікно з графічним представленням результатів кластеризації, регулюючи налаштування можна вибрати зручне відображення кластерів. Для нашого прикладу, встановіть за віссю *X* – атрибут для кількості автомобілів *M5* (*Num*), а за віссю *Y* – атрибут для кількості придбаних автомобілів *Purchase* (*Num*), вкажіть виділення кожного кластеру окремим кольором, для цього встановіть значення поля *Color* в *Cluster* (*Nom*).

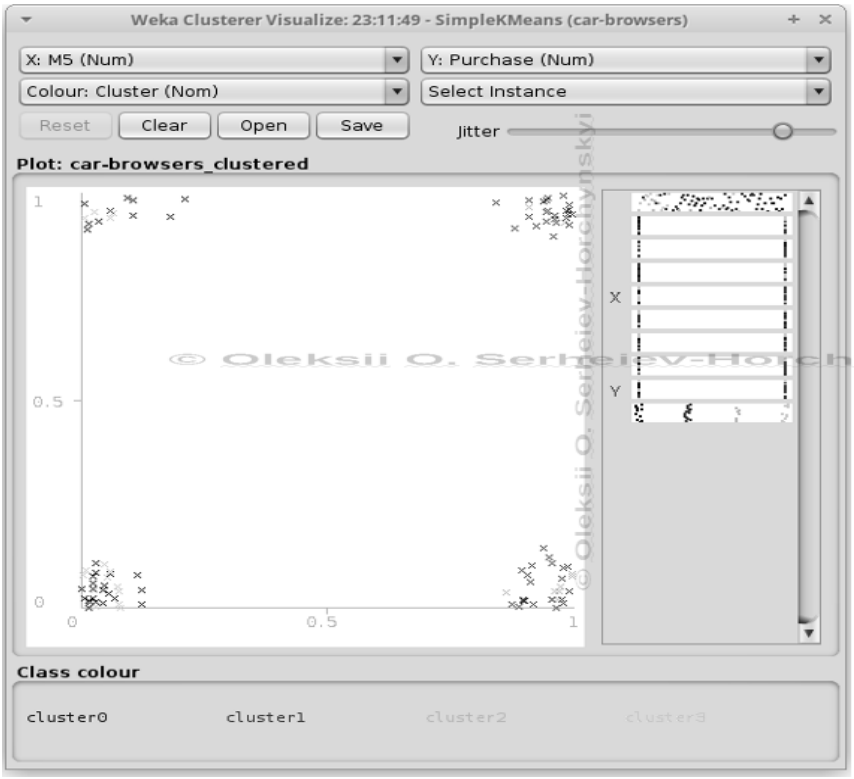

Рисунок 2.8. Графічне відображення результату кластеризації

Така послідовність дій допоможе оцінити розподіл по кластерах залежно від того, скільки людей цікавилося BMW M5, і скільки купило цю модель. Посуньте риску Jitter приблизно на три чверті в бік максимуму, це штучним чином збільшить розкид між групами точок, щоб було зручніше їх переглядати.

2.3.5. Тестування моделі класифікації

Метод кластеризації використовує відомі дані для аналізу зв'язків значень атрибутів. Коли з'являється новий екземпляр даних, потрібно лише оцінити дані за допомогою побудованої раніше моделі кластеризації та визначити приналежність даних до певного кластеру.

2.3.6. Зміст протоколу виконаної роботи

Таблиця 2.6. Варіанти завдання

Номер	Метод
1	SimpleKMeans
2	FarthestFirst

3	Hierarchical
4	FarthestFirst
5	SimpleKMeans
6	Hierarchical
7	SimpleKMeans
8	J48Hierarchical
9	FarthestFirst
10	SimpleKMeans

1. Титульний аркуш.
2. Мета роботи.
3. Інформація про дані та атрибути екземплярів даних.
4. Скриншот статистики за даними.
5. Скриншот опису побудованої моделі кластеризації.
6. Скриншот графічного відображення моделі.
7. Висновки по роботі.

2.4. Програмна розробка методу кластеризації даних

Мета роботи: Здобути навички програмної розробки методу кластеризації даних.

Завдання: Розробити програму для ієрархічної кластеризації даних.

У даному завданні Вам потрібно розробити програмну реалізацію методу ієрархічної кластеризації.

2.4.1. Метод ієрархічної кластеризації

Нехай вирішується задача кластеризації листів з електронної скриньки на наявність спаму. Усі електронні листи ще не позначені як «Спам» чи «Не спам». Необхідно розробити програму, яка проаналізує назви листів і спробує визначити відношення екземплярів слів до певних груп.

З попереднього завдання Вам відомо, що для класифікації екземплярів необхідно спершу позначити відношення навчальних даних до класів, далі побудувати модель класифікації та перевірити модель на тестових даних. Якщо відношення навчальних даних до класів невідомі, тоді застосовують метод кластеризації даних для пошуку спільних груп (кластерів).

Як навчальні дані розглянемо назви електронних листів (див. Лістинг 2.7).

Лістинг 2.7. Навчальні дані

```
Назва 1: New meeting tomorrow (file)
Назва 2: Corporate party tomorrow
Назва 3: Free sales party
Назва 4: Free file for you
Назва 5: New greeting text
Назва 6: Free file upload
```

Виконувати кластерний аналіз текстових даних можна за різними метриками, у тому числі: за кількістю букв у слові, відношенням слова до певної частини мови, за змістом. Складемо словник з переліку слів та кількості їх повторень та залишимо слова з кількістю повторень більше 1.

Таблиця 2.7. Словник слів та кількість їх повторень

Слово	**Кількість**
tomorrow	2
free	3
file	3
new	2
party	2
Інші	1

2.4.2. Визначення атрибутів екземплярів даних

Нехай кожне слово з навчального набору речень можна описати двома атрибутами: кількість літер в реченні перед словом (не враховуючи пробіли, “(“, “)”) (атрибут_X) та кількість літер в слові (атрибут_Y).

Таблиця 2.8. Перелік слів та їх атрибутів

Слово	**атрибут_X**	**атрибут_Y**
tomorrow	10	8
tomorrow	14	8
free	0	4
free	0	4
free	0	4
file	18	4
file	4	4
file	4	4
new	0	3
new	0	3
party	9	5
party	9	5

Тепер розрахуємо медіанні значення атрибутів для кожного слова.

Таблиця 2.9. Перелік слів та їх атрибутів

Слово	**атрибут_X**	**атрибут_Y**
tomorrow	median {10, 14} = 12	8
free	0	4
file	median {4, 4, 18} = 4	4
new	0	3
party	9	5

При кластеризації схожість між екземплярами визначається метриками їх відносного розташування у просторі. Екземпляри, які описуються двома атрибутами можна зобразити у двомірному просторі Евкліда. Для розрахунку відстані між даними у просторі Евкліда прийнято застосовувати метрику відстані Евкліда, яку можна визначити наступним виразом:

$$D(A,B)=\sqrt{\left(B_x-A_x\right)^2+\left(B_y-A_y\right)^2}, \qquad (2.3)$$

де D – відстань між двома екземплярами у двомірному просторі, A та B – екземпляри даних, x та y – атрибути екземплярів.

2.4.3. Визначення у просторі двох найближчих екземплярів

Наступним кроком після розташування у просторі визначають два найближчі екземпляри, для цього будують таблицю відстаней (див. Таблицю 2.10).

Таблиця 2.10. Відстані між словами, ітерація №1

Слово	**tomorrow**	**free**	**file**	**new**	**party**
tomorrow	0				
free	sqrt(pow(12-0,2)+ pow(8-4,2))	0			
file	sqrt(pow(12-4,2)+ pow(8-4,2))	sqrt(pow(0-4,2)+ pow(4-4,2))	0		
new	sqrt(pow(12-0,2)+ pow(8-3,2))	sqrt(pow(0-0,2)+ pow(4-3,2))	sqrt(pow(4-0,2)+pow(4-3,2))	0	
party	sqrt(pow(12-9,2)+ pow(8-5,2))	sqrt(pow(0-9,2)+ pow(4-5,2))	sqrt(pow(4-9,2)+pow(4-5,2))	sqrt(pow(0-9,2)+pow(3-5,2))	0

Таблиця 2.11. Відстані між словами, ітерація №1 (продовження)

Слово	**tomorrow**	**free**	**file**	**new**	**party**
tomorrow	0				
free	12.65	0			
file	8.94	4	0		
new	13	***1***	4.12	0	
party	4.24	9.06	5.1	9.22	0

2.4.4. Розрахунок найменшої відстані та об'єднання екземплярів

На першій ітерації слова free та new об'єднуємо у групу (див. Таблицю 2.12).

Таблиця 2.12. Групи слів, ітерація №1

Слово	**Група**
tomorrow	0
free	0 → 1
file	0
new	0 → 1
party	0

Для об'єднаних слів розраховуємо центроїду (див. Таблицю 2.13).

Таблиця 2.13. Перелік слів та їх атрибутів, ітерація №1

Слово	**атрибут_X**	**атрибут_Y**
tomorrow	12	8
free + new	(0 + 0)/2 = 0	(4 + 3)/2 = 3.5
file	4	4
party	9	5

Наступник кроком після об'єднання слів визначають два найближчі екземпляри, для цього будують таблицю відстаней (див. Таблицю 2.14).

Таблиця 2.14. Відстані між словами, ітерація №2

Слово	**tomorrow**	**free + new**	**file**	**party**
tomorrow	0			
free + new	sqrt(pow(12-0,2)+ pow(8-3.5,2))	0		
file	sqrt(pow(12-4,2)+ pow(8-4,2))	sqrt(pow(0-4,2)+pow(3.5-4,2))	0	
party	sqrt(pow(12-	sqrt(pow(0-	sqrt(pow(4-	0

	9,2)+ pow(8-5,2))	9,2)+pow(3.5-5,2))	9,2)+pow(4-5,2))	

Таблиця 2.15. Відстані між словами, ітерація №2 (продовження)

Слово	**tomorrow**	**free + new**	**file**	**party**
tomorrow	0			
free + new	12.82	0		
file	8.94	***4.03***	0	
party	4.24	9.12	5.1	0

Знаходимо найменшу відстань. На другій ітерації об'єднуємо у групу слова free, file та new (див. Таблицю 2.16).

Таблиця 2.16. Групи слів, ітерація №2

Слово	**Група**
tomorrow	0
free	0 → 1 → 2
file	0 →→→ 2
new	0 → 1 → 2
party	0

Для об'єднаних слів розраховуємо центроїду (див. Таблицю 2.17).

Таблиця 2.17. Перелік слів та їх атрибутів, ітерація №2

Слово	**атрибут_X**	**атрибут_Y**
tomorrow	12	8
free + new + file	(0+4)/2 = 2	(3.5+4)/2 = 3.75
party	9	5

Наступник кроком після об'єднання слів визначають два найближчі екземпляри, для цього будують таблицю відстаней (див. Таблицю 2.18).

Таблиця 2.18. Відстані між словами, ітерація №3

Слово	**tomorrow + file**	**free + new**	**party**
tomorrow	0		
free + new + file	sqrt(pow(12-2,2)+pow(8-3.75,2))	0	
party	sqrt(pow(12-9,2)+pow(8-5,2))	sqrt(pow(2-9,2)+pow(3.75-5,2))	0

Таблиця 2.19. Відстані між словами, ітерація №3 (продовження)

Слово	**tomorrow + file**	**free + new**	**party**
tomorrow	0		

free + new + file	10.87	0	
party	***4.24***	7.11	0

Знаходимо найменшу відстань. На третій ітерації об'єднуємо у групу слова tomorrow та party (див. Таблицю 2.20).

Таблиця 2.20. Групи слів, ітерація №3

Слово	**Група**
tomorrow	0 → → 2 → 3
free	0 → 1
file	0 → → 2 → 3
new	0 → 1
party	0 → → → → 3

Для об'єднаних слів розраховуємо центроїду (див. Таблицю 2.21).

Таблиця 2.21. Перелік слів та їх атрибутів, ітерація №3

Слово	**атрибут_X**	**атрибут_Y**
tomorrow + party	(12+9)/2 = 10.5	(8+5)/2 = 6.5
free + new + file	2	3.75

2.4.5. Модель ієрархічної кластеризації

На Рисунку 2.1 зображено результат роботи методу ієрархічної кластеризації у вигляді моделі ієрархічної кластеризації (дендограми). Результат кластеризації можна розділити на необхідну кількість кластерів починаючи з кореня. Виберемо два кластери та розділимо тестові слова за моделлю кластеризації на дві групи (кластери): C1 = {free, new, file} ; C2 = {tomorrow, party}.

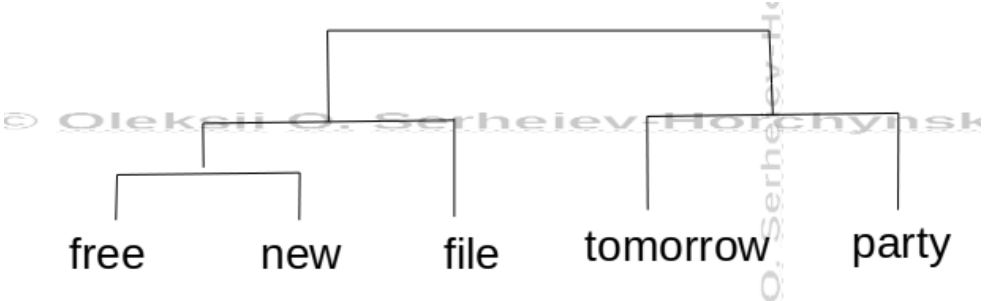

Рисунок 2.9. Модель ієрархічної кластеризації

Кластеризуємо тестове повідомлення «Free file tomorrow». В реченні два

слова «free» та «file», що відноситься до кластеру С1 та одне слово «tomorrow», що відносяться до кластеру С2. Якщо кількість тематик дорівнює 3, тоді можна сформувати три кластери, наприклад С1 ({free, new}), С2 ({file}), С3 ({tomorrow, party}).
За відношеннями слів до окремих кластерів можна визначити найбільш суттєвий кластер для усього речення. Для речення зі словами free (С1), file (С1), tomorrow (С2) таким кластером є С1.

2.4.6. Зміст протоколу виконаної роботи

Таблиця 2.22. Варіанти завдання

№	**Тематики**	№	**Тематики**	№	**Тематики**
1	Транспорт, здоров'я, шоу-бізнес	11	Музика, кіно, театр	21	Автоспорт, кіно, шоу-бізнес
2	Нерухомість, фінанси, енергетика	12	Енергетика, нерухомість, кіно	22	Інтернет, космос, наука
3	Автоспорт, хокей, баскетбол	13	Транспорт, здоров'я, шоу-бізнес	23	Нерухомість, фінанси, енергетика
4	Музика, кіно, театр	14	Нерухомість, фінанси, енергетика	24	Музика, наука, баскетбол
5	Інтернет, космос, наука	15	Хокей, фінанси, кіно	25	Хокей, фінанси, кіно
6	Автоспорт, кіно, шоу-бізнес	16	Автоспорт, хокей, баскетбол	26	Автоспорт, кіно, шоу-бізнес
7	Хокей, фінанси, кіно	17	Інтернет, космос, наука	27	Музика, наука, баскетбол
8	Енергетика, нерухомість, кіно	18	Музика, наука, баскетбол	28	Транспорт, здоров'я, шоу-бізнес
9	Музика, наука, баскетбол	19	Енергетика, нерухомість, кіно	29	Здоров'я, театр, космос
10	Здоров'я, театр, космос	20	Транспорт, здоров'я, шоу-бізнес	30	Транспорт, енергетика, баскетбол

1. Розробити програму.
2. Вибрати за варіантом набір даних для побудови моделі кластеризації.
3. Побудувати модель кластеризації.
4. Сформувати за моделлю кластеризації навчальні та тестові набори даних, побудувати модель класифікації за методом «наївний» класифікатор Байєса, розрахувати процент коректно класифікованих тестових даних.
5. Навести код програми побудови модель кластеризації.
6. Сформулювати висновок щодо можливості застосування моделі кластеризації для задачі побудови моделі класифікації, навести графік залежності точності класифікації тестових даних від кількості навчальних даних кластеризованих за допомогою побудованої моделі кластеризації.

2.5. Знайомство з Weka API для регресійного аналізу

Мета роботи: Ознайомитися та отримати навички практичного застосування методів бібліотеки Weka для регресійної класифікації даних.

Завдання: Розробити програму для побудови регресійної моделі даних.

2.5.1. Сутність регресії

Модель регресійного аналізу використовується для прогнозування значення однієї залежної змінної, виходячи з відомих значень незалежних змінних. Наприклад ціна на будинок (залежна змінна) визначається наступними незалежними змінними (значеннями атрибутів екземплярів даних): площею будинку, площа ділянки, оформленням кімнат, сантехнікою і т.д. Для побудови регресійної моделі оцінюються параметри схожих будинків та їх ціна, за якою ці будинки були продані (тобто формується навчальна вибірка). Після побудови моделі вказуються значення атрибутів будинку та розраховується вартість.

У Таблиці 2.23 зазначено перелік параметрів будинків, виставлених на продаж.

2.5.2. Підготовка даних

Таблиця 2.23. Учбовий набір даних

Площа будинку (houseSize)	**Площа ділянки (lotSize)**	**Кількість спалень (bedrooms)**	**Граніт у кухні**	**Сучасне сантехнічне обладнання**	**Ціна**
3529	9191	6	0	0	$205,000
3247	10061	5	1	1	$224,900
4032	10150	5	0	1	$197,900
2397	14156	4	1	0	$189,900
2200	9600	4	0	1	$195,000
3536	19994	6	1	1	$325,000
2983	9365	5	0	1	$230,000

На Лістинг 2.8 зображений приклад опису навчальних даних для побудови регресійної моделі.

Лістинг 2.8. Дані для регресійного аналізу

```
@relation house % будинок
@attribute houseSize numeric % площа будинку
@attribute lotSize numeric % площа ділянки
@attribute bedrooms numeric % кількість спалень
@attribute granite numeric % граніт у кухні
@attribute bathroom numeric сучасна сантехніка
@attribute sellingPrice numeric % ціна

@DATA
3529,9191,6,0,0,205000
3247,10061,5,1,1,224900
4032,10150,5,0,1,197900
2397,14156,4,1,0,189900
2200,9600,4,0,1,195000
3536,19994,6,1,1,325000
2983,9365,5,0,1,230000
```

2.5.3. Застосування Weka API

Для того щоб використовувати інтерфейс прикладного програмування (application programming interface, API) Weka в коді своєї програми, в середовище розробки потрібно додати шлях до JAR-файлу бібліотеки Weka. Лістинг 2.9 демонструє завантаження файлу з даними у середовище аналізу даних Weka.

Лістинг 2.9. Завантаження даних у Weka

```
import java.io.BufferedReader;
import java.io.FileReader;

import weka.classifiers.functions.LinearRegression;
import weka.core.Instance;
import weka.core.Instances;

public class Regression {
public static void main(String args[]) throws Exception {
Instances data = new Instances(new BufferedReader(new FileReader("house.arff")));
data.setClassIndex(data.numAttributes() - 1);
```

2.5.4. Побудова регресійної моделі

Після завантаження даних, їх необхідно проаналізувати за допомогою методу регресійного аналізу і переконатися, що результат збігається з результатом, який можна отримати за допомогою Weka Explorer.

На Лістинг 2.10 показано як побудувати регресійну модель у своїй програмі.

Лістинг 2.10. Створення моделі регресійного аналізу в Weka

```
LinearRegression model = new LinearRegression();
model.buildClassifier(data);
System.out.println(model);
```

Опис побудованої моделі зображено на Лістинг 2.11.

Лістинг 2.11. Опис побудованої моделі

```
sellingPrice =
 -26.6882 × houseSize
 +7.0551 × lotSize
 +43166.0767 × bedrooms
 +42292.0901 × bathroom
 -21661.1208
```

2.5.5. Перевірка побудованої моделі

Побудовану модель можна застосувати для оцінювання нових будинків. Приклад оцінювання останнього будинку зображено на Лістинг 2.12.

Лістинг 2.12. Опис побудованої моделі

```
Instance myHouse = data.lastInstance();
double price = model.classifyInstance(myHouse);
System.out.println("My house ("+myHouse+"): "+price);
```

Результат оцінювання будинку:

My house (2983,9365,5,0,1,230000): 222921.57101904938

Якщо з’явиться повідомлення «*Exception in thread "main" java.lang.NoClassDefFoundError: no/uib/cipr/matrix/Matrix*», тоді у проект треба додати бібліотеки mtj.jar, arpack_combined_all.jar та core.jar.

2.5.6. Зміст протоколу виконаної роботи

Наведіть у протоколі скриншот опису моделі побудованої у Weka Explorer та порівняйте з моделлю побудованою за допомогою Weka API (див. Лістинг 2.12).

Таблиця 2.24. Варіанти завдання

1	LinearRegression	11	GaussianProcesses	21	GaussianProcesses
2	SimpleLinear Regression	12	SimpleLinear Regression	22	SimpleLinear Regression
3	GaussianProcesses	13	LinearRegression	23	LinearRegression
4	LinearRegression	14	GaussianProcesses	24	SMOreg
5	SMOreg	15	SMOreg	25	LinearRegression
6	GaussianProcesses	16	LinearRegression	26	SimpleLinear Regression
7	SimpleLinear Regression	17	SimpleLinear Regression	27	GaussianProcesses
8	LinearRegression	18	GaussianProcesses	28	LinearRegression
9	GaussianProcesses	19	LinearRegression	29	SMOreg
10	SMOreg	20	SMOreg	30	LinearRegression

Тестові дані знаходяться за адресою:

https://archive.ics.uci.edu/ml/datasets.html .

Наприклад:

1. https://archive.ics.uci.edu/ml/machine-learning-databases/housing/ .
2. https://archive.ics.uci.edu/ml/machine-learning-databases/cmc/ .
3. Тощо.

1. Титульний аркуш.
2. Мета роботи.
3. Інформація про дані (вибрати відмінні від прикладу).
4. Код програми.
5. Результат роботи програми на тестовому екземплярі.
6. Скриншот моделі у Weka Explorer.
7. Висновки.

2.6. Знайомство з методами побудови асоціативних правил

Мета роботи: Ознайомитися та набути навички побудови асоціативних правил за допомогою Weka.

Завдання: Побудувати асоціативні правила для тестових екземплярів даних за допомогою зазначеного методу.

2.6.1. Сутність асоціації

На відміну від методів побудови моделей класифікації, методи пошуку асоціативних правил не потребують вибору атрибуту класу, у*сі атрибути вважаються атрибутами ознак,* класом є комбінація значень окремих атрибутів ознак. *Методи побудови асоціативних правил призначені для пошуку комбінацій значень атрибутів, на базі яких за комбінацією значень атрибутів першої множини («умова») можна спрогнозувати значення атрибутів другої множини («наслідок»).*

Пошук асоціативних правил часто виконують супермаркети під час аналізу споживчого кошику для визначення продуктів, які покупці часто купують разом, після чого знайдені комбінації товарів розміщують поруч, щоб збільшити ймовірність їх купівлі. Для ілюстрації ідеї пошуку асоціативних правил виконаємо аналіз тестового файлу contact-lenses.arff (у папці data з каталогу Weka) з записами рекомендацій контактних лінз за фізіологічними характеристиками пацієнтів.

Для пошуку виберемо метод Apriori – класичний метод пошуку закономірностей у значеннях атрибутів у вигляді асоціативних правил. Метод Apriori перебирає усі можливі комбінації значень заданої кількості атрибутів, наприклад, *комбінація*$_1$={*атрибут А* значення A_1, *атрибут Б значення* $Б_1$}, та вибирає ті комбінації, які перевищують мінімальне значення критерію підтримки (support criterion, задається у параметрах методу).

Наприклад, у папці data є файл contact-lenses.arff, що містить 24 екземпляри даних, тому можна послідовно шукати закономірності, які містять від 1 до 24 атрибутів. Для кожної комбінації задається мінімальне значення критерію підтримки, нехай буде 20% від загальної вибірки з 24-х екземплярів, тоді мінімальна допустима кількість екземплярів, у яких буде знайдено конкретну закономірність має перевищувати чи дорівнювати $24 \times 0.1 = 4.8 \geq 5$ екземплярів.

2.6.2. Пошук асоціативних правил у Weka Explorer

Виконаємо у пакеті Weka пошук асоціативних правил за допомогою методу Apriori з заданими параметрами (див. Рисунок 2.4.1). Для відображення закономірностей вкажіть значення параметру outputItemSets=True.

Метод Apriori починає пошук з закономірностей, які містять лише одне значення атрибуту. Далі, в процесі пошуку, виходячи з того факту, що менші комбінації атрибутів зустрічаються частіше, кількість атрибутів у шуканих закономірностях поступово збільшується. Пошук зупиняється за умови відсутності знайдених екземплярів для поточної кількості атрибутів.

Результат пошуку для різних кількостей атрибутів зображено на Лістингу 2.13.

Лістинг 2.13. Закономірності для різних кількостей атрибутів

Apriori	
Minimum support: 0.2 (5 instances) Minimum metric <confidence>: 0.9 Number of cycles performed: 16 Generated sets of large itemsets: Size of set of large itemsets L(1): 11 Large Itemsets L(1): age=young 8 age=pre-presbyopic 8 age=presbyopic 8 spectacle-prescrip=myope 12 spectacle-prescrip=hypermetrope 12 astigmatism=no 12 astigmatism=yes 12 tear-prod-rate=reduced 12 tear-prod-rate=normal 12 contact-lenses=soft 5 contact-lenses=none 15 Size of set of large itemsets L(2): 21 Large Itemsets L(2): age=pre-presbyopic contact-lenses=none 5 age=presbyopic contact-lenses=none 6 spectacle-prescrip=myope astigmatism=no 6 spectacle-prescrip=myope astigmatism=yes 6	*знайдено 21 закономірностей* *правило знайдено для 5 екземплярів з 24-х можливих*

spectacle-prescrip=myope tear-prod-rate=reduced 6 spectacle-prescrip=myope tear-prod-rate=normal 6 spectacle-prescrip=myope contact-lenses=none 7 spectacle-prescrip=hypermetrope astigmatism=no 6 spectacle-prescrip=hypermetrope astigmatism=yes 6 spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6 spectacle-prescrip=hypermetrope tear-prod-rate=normal 6 spectacle-prescrip=hypermetrope contact-lenses=none 8 astigmatism=no tear-prod-rate=reduced 6 astigmatism=no tear-prod-rate=normal 6 astigmatism=no contact-lenses=soft 5 astigmatism=no contact-lenses=none 7 astigmatism=yes tear-prod-rate=reduced 6 astigmatism=yes tear-prod-rate=normal 6 astigmatism=yes contact-lenses=none 8 tear-prod-rate=reduced contact-lenses=none 12 tear-prod-rate=normal contact-lenses=soft 5 Size of set of large itemsets L(3): 6 Large Itemsets L(3): spectacle-prescrip=myope tear-prod-rate=reduced contact-lenses=none 6 spectacle-prescrip=hypermetrope astigmatism=yes contact-lenses=none 5 spectacle-prescrip=hypermetrope tear-prod-rate=reduced contact-lenses=none 6 astigmatism=no tear-prod-rate=reduced contact-lenses=none 6 astigmatism=no tear-prod-rate=normal contact-lenses=soft 5 astigmatism=yes tear-prod-rate=reduced contact-lenses=none 6	

2.6.3. Зміст параметрів методу Apriori

Асоціативне правило складається з двох множин *X* («умова») і *Y* («наслідок») у вигляді конструкції IF-THEN: $X \rightarrow Y$, тобто, якщо знайдені значення атрибутів множини *X*, тоді з ним ймовірно будуть знайдені значення атрибутів множини *Y*.

Критерій підтримки (support criterion) – частка кількості екземплярів, яка припадає на кожне асоціативне правило (від 0.0 до 1.0). Для 24 екземплярів підтримка у 20% (minSupport=0.2) дорівнює 5, тобто кожне асоціативне правило має виконуватися щонайменше для 5 екземплярів з усього набору даних.

Критерій достовірності (confidence criterion) – відношення кількості екземплярів множини (*X U Y) до кількості екземплярів множини X у знайденному правилі. Відношення* характеризує рівень зв'язку атрибутів

множин *X* та *Y*.

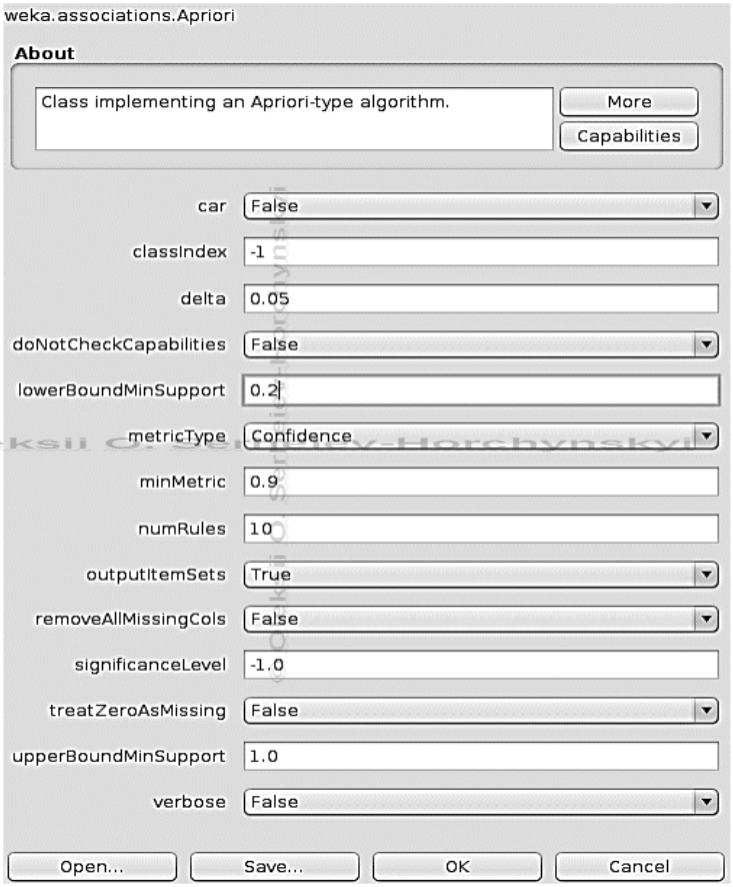

Рисунок 2.10. Параметри методу Apriori у Weka

2.6.4. Застосування Weka API для пошуку асоціативних правил

На Лістингу 2.14 зображено код програми пошуку асоціативних правил з застосуванням методу Apriori (weka.jar).

Лістинг 2.14. Програма пошуку асоціативних правил

```
import java.io.BufferedReader;
import java.io.FileReader;
import weka.core.Instances;
import weka.associations.Apriori;

public class AssociationRules{
 public static void main(String args[]) throws Exception{

  //load data
  Instances data = new Instances(new BufferedReader(new FileReader("contact-lenses.arff")));
```

```
  //build model
  Apriori model = new Apriori();
  model.buildAssociations(data);
  System.out.println(model);
  }
}
```

Результат роботи програми зображено на Лістингу 2.15.

Лістинг 2.15. Результат роботи програми

```
Apriori
=======
Minimum support: 0.2 (5 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 16
Generated sets of large itemsets:
Size of set of large itemsets L(1): 11
Size of set of large itemsets L(2): 21
Size of set of large itemsets L(3): 6
Best rules found:
 1. tear-prod-rate=reduced 12 ==> contact-lenses=none 12  <conf:(1)> lift:(1.6) lev:(0.19) [4]
conv:(4.5)
 2. spectacle-prescrip=myope tear-prod-rate=reduced 6 ==> contact-lenses=none 6  <conf:(1)>
lift:(1.6) lev:(0.09) [2] conv:(2.25)
 3. spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6 ==> contact-lenses=none 6
<conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)
 4. astigmatism=no tear-prod-rate=reduced 6 ==> contact-lenses=none 6
 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)
 5. astigmatism=yes tear-prod-rate=reduced 6 ==> contact-lenses=none 6  <conf:(1)> lift:(1.6)
lev:(0.09) [2] conv:(2.25)
 6. contact-lenses=soft 5 ==> astigmatism=no 5
<conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
 7. contact-lenses=soft 5 ==> tear-prod-rate=normal 5
<conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
 8. tear-prod-rate=normal contact-lenses=soft 5 ==> astigmatism=no 5
<conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
 9. astigmatism=no contact-lenses=soft 5 ==> tear-prod-rate=normal 5
<conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
10. contact-lenses=soft 5 ==> astigmatism=no tear-prod-rate=normal 5
<conf:(1)> lift:(4) lev:(0.16) [3] conv:(3.75)
```

На Лістингу 2.15 зображено 10 асоціативних правил (numRules=10), виокремлених з 38 закономірностей за критерієм minMetric=0.9 (metricType= Confidence). Метод Apriori поступово зменшує мінімальну підтримку (minSupport) з 1.0 до заданих 0.2 (1.0, 0.9, 0.8, 0.7, ..., 0.2), поки не буде знайдена необхідна кількість правил (numRules=10) з заданим мінімальним

значенням критерію достовірності (minMetric=0.9). Якщо 10 правил будуть знайдені з minSupport=0.3, тоді метод зупиниться та залишить закономірності, значення атрибутів яких містяться щонайменше у $24 \times 0.3 = 7.2 \geq 7$ екземплярів.

2.6.5. Приклад асоціативного правила

Розглянемо правило № 5 зображене на Лістинг 2.16.

Лістинг 2.16. Приклад знайденого асоціативного правила

```
5. astigmatism=yes tear-prod-rate=reduced 6 ==> contact-lenses=none 6
<conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)
```

Значення атрибутів множини *X* («умова») (astigmatism=yes tear-prod-rate=reduced) були знайдені у 6 екземплярів так само, як і значення атрибутів множини *Y* («наслідок») (contact-lenses=none), відповідно достовірність асоціативного правила дорівнює conf = 6/6 = 1.0, тобто виконується умова minMetric=0.9. Розробіть програму та побудуйте перелік асоціативних правил за допомогою метода, зазначеного у варіанті.

2.6.6. Зміст протоколу виконаної роботи

Таблиця 2.25. Варіанти завдання

1	Apriori	11	Generalized Sequential	21	Apriori
2	FPGrowth	12	FPGrowth	22	PredictiveApriori
3	Generalized Sequential	13	Apriori	23	Tertius
4	PredictiveApriori	14	PredictiveApriori	24	FPGrowth
5	Tertius	15	Tertius	25	Generalized equential
6	Generalized Sequential	16	Apriori	26	PredictiveApriori
7	FPGrowth	17	FPGrowth	27	Generalized equential
8	Apriori	18	Generalized Sequential	28	FPGrowth
9	PredictiveApriori	19	PredictiveApriori	29	Apriori
10	Tertius	20	Tertius	30	Tertius

1. Титульний аркуш.
2. Мета роботи.
3. Інформація про атрибути даних, відмінних від прикладу.
4. Код програми, пояснити вибір значень параметрів.
5. Перелік знайдених асоціативних правил.
6. Висновки.

Список використаних джерел

1. Christopher Pal, Mark Hall, Eibe Frank, Ian Witten. Data Mining: Practical Machine Learning Tools and Techniques, 4rd ed. / Morgan Kaufmann, 2016.
2. Jennifer Reese, Richard Reese. Java for Data Science / Packt Publishing, 2017.
3. Bostjan Kaluza. Machine Learning in Java / Packt Publishing, 2016.
4. David Hand, Heikki Manilla, Padhraic Smyth. Principles of data mining / MIT Press, 2001.
5. Jason Bell. Machine Learning: Hands-On for Developers and Technical Professionals / John Wiley & Sons, 2014.
6. Michael Abernethy. Data mining with WEKA / IBM developerWorks, 2010.

Підписано до друку 23.04.2018 р. Формат 60х90 1/16.
Папір офсетний. Умовн. др. арк. 1,9
Друк різограф. Тираж 100 прим. Зам. № 2304/01.

Підприємство Підприємство «УВОІ «Допомога» УСІ»
Свідоцтво про державну реєстрацію №31245580
03056, м. Київ, вул. Борщагівська, 46/2, к. 1
Тел.: 277-80-08.