ЛАБОРАТОРНА РОБОТА 4

Тема: Багатовимірна візуалізація даних та теплова карта

Теплова карта — це графічне відображення числових даних, де для позначення значень використовується колір. У контексті аналізу даних теплові карти особливо корисні для двох цілей: для візуалізації таблиць кореляції та для візуалізації відсутніх значень у даних. В обох випадках інформація передається у двовимірній таблиці. Таблиця кореляції для *n* змінних має *n* рядків і *n* стовпців. Таблиця даних містить *n* стовпців (змінних) і *n* рядків (спостереження). Якщо кількість рядків величезна, то можна використовувати підмножину. В обох випадках сканування кольорового кодування, а не значень набагато простіше і швидше. Теплові карти є корисними при дослідженні великої кількості значень, але вони не замінюють більш точне графічне відображення, наприклад стовпчасті діаграми, оскільки відмінності кольорів не можуть бути сприйняті точно.

Реалізуємо теплову карту на прикладі файлу Boston.csv (його опис наведено у методичних вказівках до лабораторної роботи 3). Для її побудови використаємо як Pandas, так і Matplotlib:

```
import os
import calendar
import numpy as np
import pandas as pd
from pandas.plotting import scatter_matrix, parallel_coordinates
import seaborn as sns
from sklearn import preprocessing
import matplotlib.pylab as plt
```

Підключаємо Boston.csv :

```
housing_df = pd.read_csv('BostonHousing.csv')
housing_df
```

	ID	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
3	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
4	7	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.9
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
328	500	0.17783	0.0	9.69	0	0.585	5.569	73.5	2.3999	6	391	19.2	395.77	15.10	17.5
329	502	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.4
330	503	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.6
331	504	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.9
332	506	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88	11.9

333 rows × 15 columns

Додамо до даних нову категорію і відповідний стовпчик “cat_medv”, яка вказує на вартість житла, що перевищує 30 тис. (значення 1) або менше неї (значення 0). Використання подібних категорій є зручними у бізнес аналізі для виявлення певних властивостей речей або процесів, або віднесення їх до певних категорій, наприклад, до класу VIP або до класу “небезпечно”, тощо.

Для додавання нової категорії і відповідного стовпчика використовуємо наступний код:

```
housing_df["cat_medv"] = [1 if medv>=30.0 else 0 for medv in housing_df.medv]
```

Як видно із наведеного коду, значення нового стовпчика “cet_medv” обчислюються за значеннями стовпчика “medv” і, якщо вони перевищують значення 30.0 (у тисячах), то у новий стовпчик заноситься значення 1, інакше до нього потрапляє значення 0:

```
housing_df
```

	ID	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv	cat_medv
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0	0
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6	0
2	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4	1
3	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2	1
4	7	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.9	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
328	500	0.17783	0.0	9.69	0	0.585	5.569	73.5	2.3999	6	391	19.2	395.77	15.10	17.5	0
329	502	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.4	0
330	503	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.6	0
331	504	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.9	0
332	506	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88	11.9	0

333 rows × 16 columns

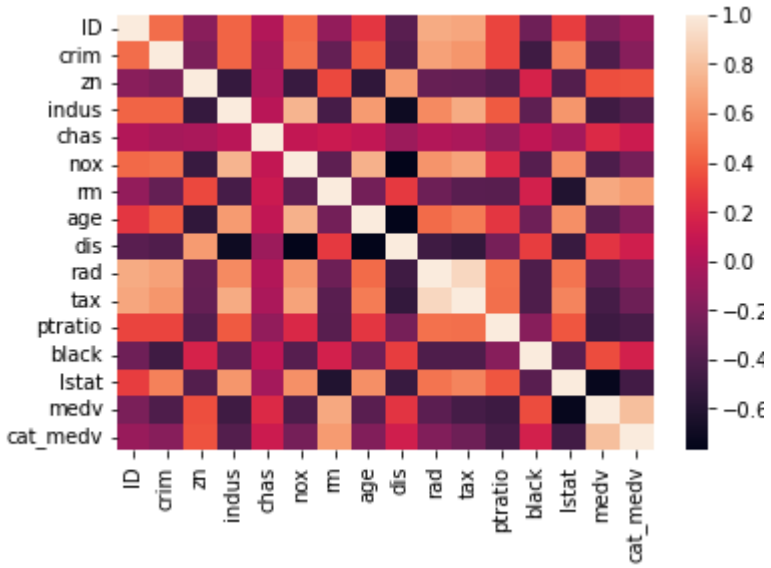
Тепер побудуємо теплову карту, для чого використовується бібліотека візуалізації seaborn та її метод heatmap(). По-перше знайдемо кореляцію між значеннями у різних стовпчиках:

```
corr = housing_df.corr()
corr
```

	ID	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat
ID	1.000000	0.456312	-0.155639	0.421978	0.007958	0.440185	-0.112790	0.257300	-0.356461	0.707526	0.686246	0.309838	-0.271619	0.281953
crim	0.456312	1.000000	-0.210913	0.422228	-0.041195	0.463001	-0.310180	0.379034	-0.397067	0.666636	0.617081	0.313409	-0.475796	0.532077
zn	-0.155639	-0.210913	1.000000	-0.518679	-0.024442	-0.501990	0.328197	-0.544513	0.637142	-0.303663	-0.311180	-0.380449	0.168130	-0.388112
indus	0.421978	0.422228	-0.518679	1.000000	0.037496	0.750087	-0.440365	0.638378	-0.702327	0.569779	0.708313	0.391087	-0.335049	0.614155
chas	0.007958	-0.041195	-0.024442	0.037496	1.000000	0.080275	0.112251	0.068286	-0.081834	0.007714	-0.021826	-0.125067	0.062029	-0.050055
nox	0.440185	0.463001	-0.501990	0.750087	0.080275	1.000000	-0.338515	0.736000	-0.769364	0.612180	0.670722	0.192513	-0.369416	0.598874
rm	-0.112790	-0.310180	0.328197	-0.440365	0.112251	-0.338515	1.000000	-0.248573	0.269191	-0.272783	-0.356987	-0.366927	0.155202	-0.615747
age	0.257300	0.379034	-0.544513	0.638378	0.068286	0.736000	-0.248573	1.000000	-0.764208	0.447380	0.511893	0.259293	-0.268054	0.588834
dis	-0.356461	-0.397067	0.637142	-0.702327	-0.081834	-0.769364	0.269191	-0.764208	1.000000	-0.477610	-0.529539	-0.231101	0.284374	-0.505939
rad	0.707526	0.666636	-0.303663	0.569779	0.007714	0.612180	-0.272783	0.447380	-0.477610	1.000000	0.903562	0.470849	-0.406405	0.484568
tax	0.686246	0.617081	-0.311180	0.708313	-0.021826	0.670722	-0.356987	0.511893	-0.529539	0.903562	1.000000	0.467437	-0.406477	0.544485
ptratio	0.309838	0.313409	-0.380449	0.391087	-0.125067	0.192513	-0.366927	0.259293	-0.231101	0.470849	0.467437	1.000000	-0.164614	0.374802
black	-0.271619	-0.475796	0.168130	-0.335049	0.062029	-0.369416	0.155202	-0.268054	0.284374	-0.406405	-0.406477	-0.164614	1.000000	-0.356693
lstat	0.281953	0.532077	-0.388112	0.614155	-0.050055	0.598874	-0.615747	0.588834	-0.505939	0.484568	0.544485	0.374802	-0.356693	1.000000
medv	-0.221694	-0.407454	0.344842	-0.473932	0.204390	-0.413054	0.689598	-0.358888	0.249422	-0.352251	-0.448078	-0.481376	0.336660	-0.738600
cat_medv	-0.094243	-0.162457	0.363551	-0.385186	0.117190	-0.242158	0.637396	-0.190188	0.137147	-0.189357	-0.276562	-0.431125	0.152365	-0.463445

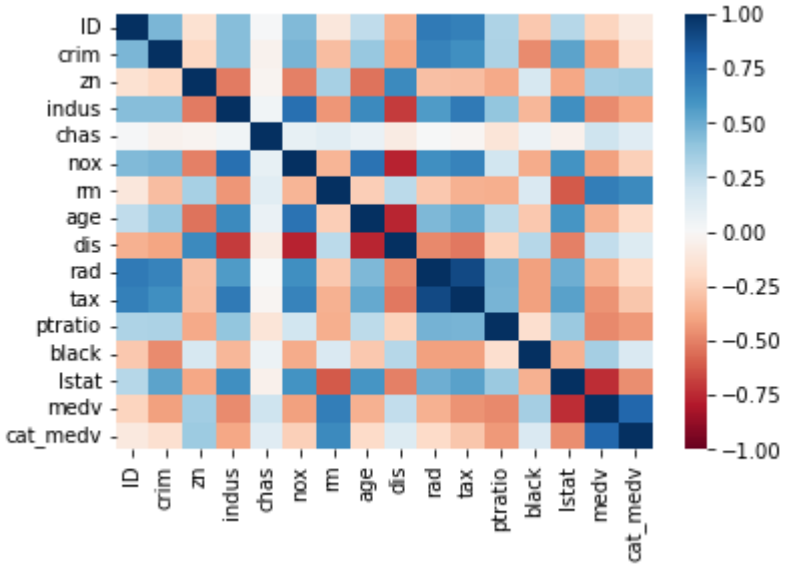
Найпростішу візуалізацію отриманих значень можна побудувати за допомогою:

```
sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns)
```



Більш світлі тони вказують на більшу кореляцію, тобто зв’язок, між різними параметрами в таблиці, темні — на їх відсутність. Для покращення сприйняття можна змінити гаму використовуваних кольорів:

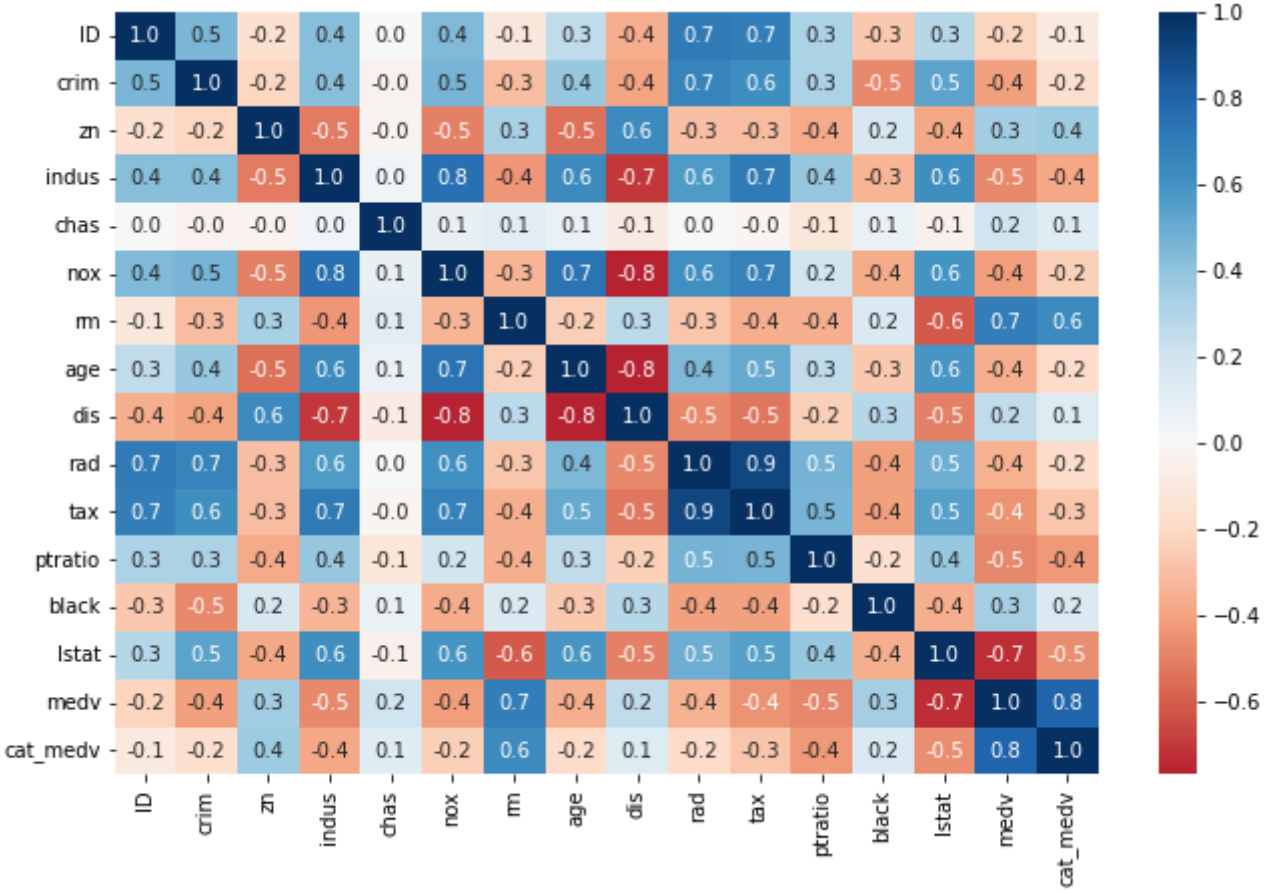
```
sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns, vmin=-1, vmax=1, cmap="RdBu")
```



Тепер глибина синього кольору вказує на силу зв’язку між параметрами, а червоні — на її відсутність.

Інколи такого роду графіки необхідно доповнити значеннями у кожній комірці теплової карти. Для реалізації цього використаємо наступний код:

```
fig, ax = plt.subplots()
fig.set_size_inches(11, 7)
sns.heatmap(corr, annot=True, fmt=".1f", cmap="RdBu", center=0, ax=ax)
```



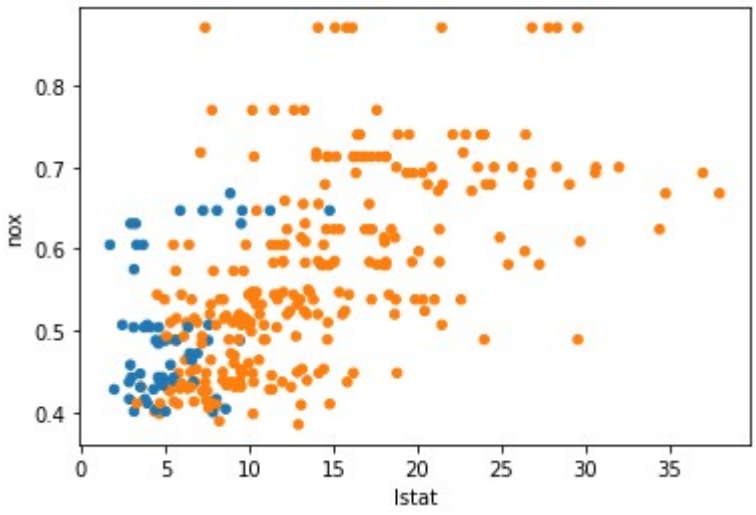
Базові графіки можуть передавати багатшу інформацію за допомогою таких функцій, як колір, розмір та декілька панелей, а також дозволяючи такі операції, як масштабування, агрегування та інтерактивність. Ці доповнення дозволяють переглядати більше однієї або двох змінних одночасно. Принадність цих доповнень полягає в їх ефективності у відображенні складної інформації у легкий зрозумілий спосіб. Ефективні функції засновані на розумінні того, як працює візуальне сприйняття. Мета полягає в тому, щоб зробити інформацію більш зрозумілою, а не просто представити дані у більш високих вимірах. Наприклад, тривимірні графіки є більш складними, але, зазвичай, малоефективними візуалізаціями.

Для того, щоб включити більше змінних у графік, необхідно розглянути тип змінної для включення. Для представлення додаткової категорійної інформації найкраще використовувати відтінок, форму або кілька панелей. Для отримання додаткової числової інформації можна використовувати інтенсивність кольору або розмір. Тимчасова інформація може бути додана за допомогою анімації.

Включення додаткових категорійних та/або числових змінних до графіків означає, що їх можна використовувати у задачах прогнозування та класифікації. Наприклад, базова діаграма розсіювання не може бути використана для вивчення зв’язку між категорійним результатом і певним параметром у задачі класифікації. Але діаграма розсіювання для двох числових параметрів, позначених кольором певної категоріальної змінної є вже ефективним графіком для класифікації. Для побудови такого типу діаграми використаємо наступний код:

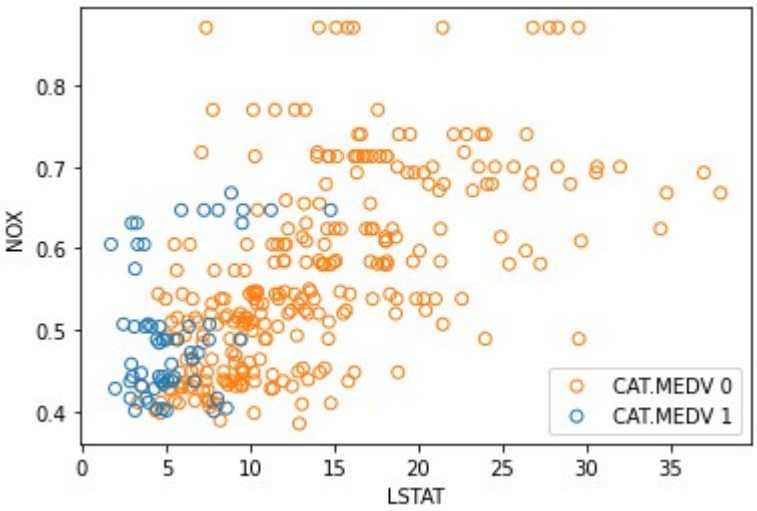
```
housing_df.plot.scatter(x='lstat', y='nox', c=['C0' if c == 1 else 'C1' for c in housing_df.cat_medv])
```

де положення точки на графіку залежить від двох параметрів lstat — відсоток населення нижчого статусу та nox - концентрація оксиду азоту, а колір задається у відповідності до значень у стовпчику “cat_medv”, які було обчислено вище за умовою перевищення вартості у 30 тис.



У разі необхідності використання такого графіку у презентаціях або звітах його вигляд можна покращити за допомогою використання інших маркерів на графіку та їх підпису:

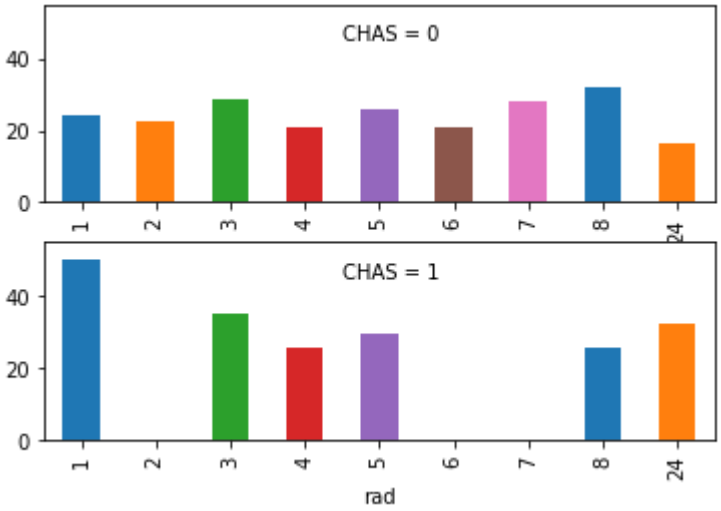
```
_, ax = plt.subplots()
for catValue, color in (0, 'C1'), (1, 'C0'):
  subset_df = housing_df[housing_df.cat_medv == catValue]
  ax.scatter(subset_df.lstat, subset_df.nox, color='none', edgecolor=color)
ax.set_xlabel('LSTAT')
ax.set_ylabel('NOX')
ax.legend(["CAT.MEDV 0", "CAT.MEDV 1"])
plt.show()
```



Для побудови діаграми було використано *subset_df* , яка спочатку у циклу заповнювалась значеннями рядків, у яких у стовпчику “cat_medv” знаходиться 0, а потім — значеннями рядків з відповідним значенням у стовпчику “cat_medv” 1.

У контексті передбачення кольорове кодування підтримує дослідження умовного зв’язку між числовим результатом (на осі y) і числовим значенням параметру (на осі x). Також кольорові діаграми розсіювання допомагають оцінити необхідність створення умов взаємодії. Наприклад, для даних, що тут використовуються, можна визначити вплив наявності поблизу будинків річки на співвідношення між medv (вартості житла) і lstat (відсотку населення нижчого статусу).

Колір також можна використовувати для включення додаткових категоріальних змінних до стовпчастої діаграми, якщо кількість категорій невелика. Якщо кількість категорій велика, кращою альтернативою є використання кількох панелей. Створення кількох панелей, яке також зветься «решіткою», здійснюється шляхом поділу спостережень відповідно до категорійної змінної та створення окремого графіка (одного типу) для кожної категорії.



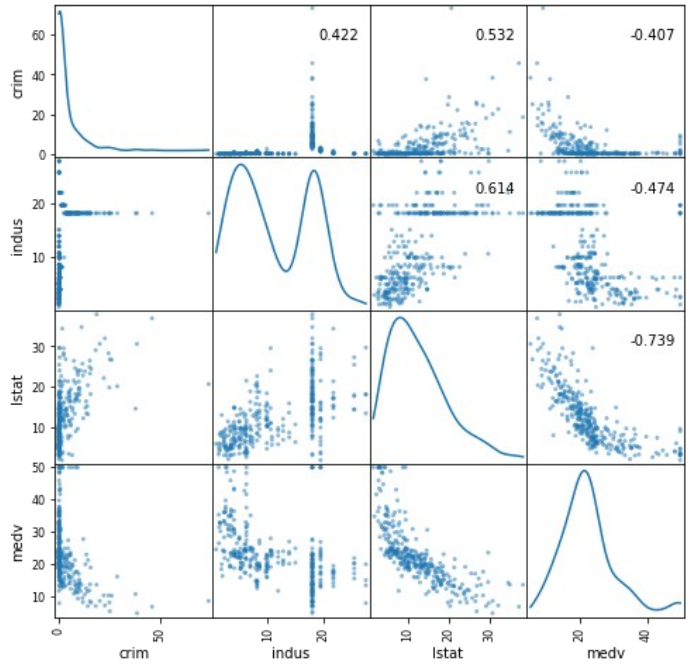
На прикладі наведено стовпчасту діаграму середнього за стовпчиком medv від rad, яку розбито на дві панелі за значеннями chas. Виходячи з даних, що аналізуються, це вказує, що середнє значення ціни (medv) для різних рівнів доступності шосе (rad) поводиться по-різному для будинків поблизу річки (chas=1, нижня панель) порівняно з будинками далеко від річки (chas=0, верхня панель). Такі дослідження корисні для прогнозування та класифікації. Наведені діаграми було побудовано за допомогою наступного коду:

```
dataForPlot_df = housing_df.groupby(['chas','rad']).mean()['medv']
ticks = set(housing_df.rad)
for i in range(2):
    for t in ticks.difference(dataForPlot_df[i].index):
        dataForPlot_df.loc[(i, t)] = 0
dataForPlot_df = dataForPlot_df[sorted(dataForPlot_df.index)]
yRange = [0, max(dataForPlot_df) * 1.1]
fig, axes = plt.subplots(nrows=2, ncols=1)
dataForPlot_df[0].plot.bar(x='rad', ax=axes[0], ylim=yRange, color=['C0','C1','C2','C3','C4','C5','C6'])
dataForPlot_df[1].plot.bar(x='rad', ax=axes[1], ylim=yRange, color=['C0','C1','C2','C3','C4','C5','C6'])
axes[0].annotate('CHAS = 0', xy=(3.5, 45))
axes[1].annotate('CHAS = 1', xy=(3.5, 45))
plt.show()
```

Спеціальний графік, який використовує діаграми розсіювання із кількома панелями — це матриця діаграми розсіювання. У ньому всі попарні діаграми розсіювання відображаються на одному дисплеї. Панелі в матричній діаграмі розсіювання організовані по особливому так, що кожен стовпець і кожен рядок відповідають змінній, тому перетини створюють усі можливі попарні діаграми розсіювання. Матриця діаграм розсіювання корисна у подальшому застосуванні у машинному навчанні для вивчення зв’язків між числовими змінними, виявлення викидів та визначення кластерів, а також для прогнозування.

Нижче наведено приклад матриці діаграми розсіювання з medv і трьома предикторами. Нижче діагоналі розташовані діаграми розсіювання. Назва змінної вказує на змінну вісі Y. Наприклад, усі графіки в нижньому рядку мають medv на вісі Y, що дозволяє вивчати індивідуальні відносини результат-провісник. Діаграма має різні типи зв’язків із різних форм. Наприклад, експоненційний зв’язок між medv та lstat. Уздовж діагоналі, де задіяно лише одну

змінну, відображається частотний розподіл для цієї змінної. Діаграми розсіювання над діагоналлю містять коефіцієнти кореляції, що відповідають двом змінним.



Для побудови діаграми було використано наступний код:

```
df = housing_df[['crim', 'indus', 'lstat', 'medv']]
axes = scatter_matrix(df, alpha=0.5, figsize=(8, 8), diagonal='kde')
coor = df.corr().values
for i, j in zip(*plt.np.triu_indices_from(axes, k=1)):
    axes[i,j].annotate('%.3f' %coor[i,j],(0.8,0.8),
            xycoords='axes fraction', ha='center', va='center')
plt.show()
```

Слід обережно додавати змінні, оскільки дисплей може стати перевантаженим, і тоді візуальне сприйняття буде втрачено.

Практичні завдання

1. У наданих файлах (див. лаб. роб. 2) розташовано статистичні дані Футбольної Прем’єр Ліги (FPL): успіхи кожного гравця за певний сезон у Лізі. Назва файлу відповідає певному сезону. Вміст файлу поділяється на колонки:
   - first_name — ім’я гравця
   - second_name — прізвище гравця
   - goals_scored - загальна кількість забитих голів за цей сезон
   - assists - загальна кількість гольових передач - присуджується гравцю з команди забиття воріт, який робить остаточний пас до того, як забити гол, включаючи автоголи.
   - total_points - загальна сума балів, зароблених у цьому сезоні
   - minutes - загальна кількість зіграних хвилин цього сезону
   - goals_conceded - загальна кількість голів, пропущених командою, поки гравець був на полі
   - creativity - творчість, оцінює ефективність гравців з точки зору створення можливостей для оцінки голів для інших гравців. Частина індексу ICT

- influence - вплив, оцінює вплив гравця на матч, враховуючи дії, які можуть прямо чи побічно вплинути на результат матчу. Частина індексу ICT.
- threat - загроза, вимірює гравців, які, швидше за все, заб'ють голи. Частина індексу ICT
- bonus - троє найкращих гравців у кожному матчі відповідно до BPS отримають додаткові бонусні бали - 3 бали будуть нараховані гравцеві з найвищою оцінкою, 2 - другому кращому та 1 - третьому.
- bps - система бонусних балів (BPS) використовує ряд статистичних даних для створення оцінки BPS для кожного гравця. Троє найкращих гравців у кожному матчі отримають бонусні бали.
- ict_index - статистичний індекс, розроблений спеціально для оцінки гравця як активу FPL, що поєднує показники впливу, творчості та загроз.
- clean_sheets - загальна кількість чистих аркушів - присуджується гравцям, які не пропустили гол і зіграли принаймні 60 хвилин.
- red_cards - кількість отриманих за сезон червоних карток.
- yellow_cards - кількість отриманих за сезон жовтих карток.

2. Оберіть категорію даних, на основі якої додайте нову категорію, що буде вказувати на перевищення першої певного значення. Додайте відповідний стовпчик із значеннями 0, 1.
3. На основі наявних даних побудуйте графіки, типи яких наведено в теоретичній частині лабораторної роботи.
4. Підготуйте звіт з виконання практичних завдань.