Лабораторна робота №7

ДОСЛІДЖЕННЯ МЕТОДІВ НЕКОНТРОЛЬОВАНОГО НАВЧАННЯ

Мета: використовуючи спеціалізовані бібліотеки та мову програмування Python дослідити методи неконтрольованої класифікації даних у машинному навчанні.

Хід роботи:

Завдання 7.1. Кластеризація даних за допомогою методу k-середніх.

Лістинг програми:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score, calinski_harabasz_score,
davies_bouldin_score

# Завантаження вхідних даних
X = np.loadtxt('data_clustering.txt', delimiter=',')

# Задання кількості кластерів
num_clusters = 5

# Візуалізація вхідних даних
plt.figure()
plt.scatter(X[:, 0], X[:, 1], marker='o', facecolors='none',
            edgecolors='black', s=80)
plt.title('Вхідні дані')
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

```
# Створення об'єкту KMeans
kmeans = KMeans(init='k-means++', n_clusters=num_clusters, n_init=10)

# Навчання моделі кластеризації
kmeans.fit(X)

# Створення сітки точок для візуалізації меж
step_size = 0.01
x_vals, y_vals = np.meshgrid(np.arange(x_min, x_max, step_size),
                             np.arange(y_min, y_max, step_size))

# Передбачення вихідних міток для всіх точок сітки
output = kmeans.predict(np.c_[x_vals.ravel(), y_vals.ravel()])
output = output.reshape(x_vals.shape)

# Графічне відображення областей та виділення їх кольором
plt.figure()
plt.clf()
plt.imshow(output, interpolation='nearest',
           extent=(x_vals.min(), x_vals.max(),
                   y_vals.min(), y_vals.max()),
           cmap=plt.cm.Paired,
           aspect='auto',
           origin='lower')

plt.scatter(X[:, 0], X[:, 1], marker='o', facecolors='none',
            edgecolors='black', s=80)

cluster_centers = kmeans.cluster_centers_
plt.scatter(cluster_centers[:, 0], cluster_centers[:, 1], marker='o', s=210,
            linewidths=4, color='black', zorder=12, facecolors='black')

plt.title('Границі кластерів')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()

print("Оцінка якості кластеризації:")
print("Inertia:", round(kmeans.inertia_, 2))
print("Silhouette Score:", round(silhouette_score(X, kmeans.labels_), 3))
print("Calinski-Harabasz Index:", round(calinski_harabasz_score(X,
kmeans.labels_), 2))
print("Davies-Bouldin Index:", round(davies_bouldin_score(X, kmeans.labels_),
2))
```

Результат виконання програми:

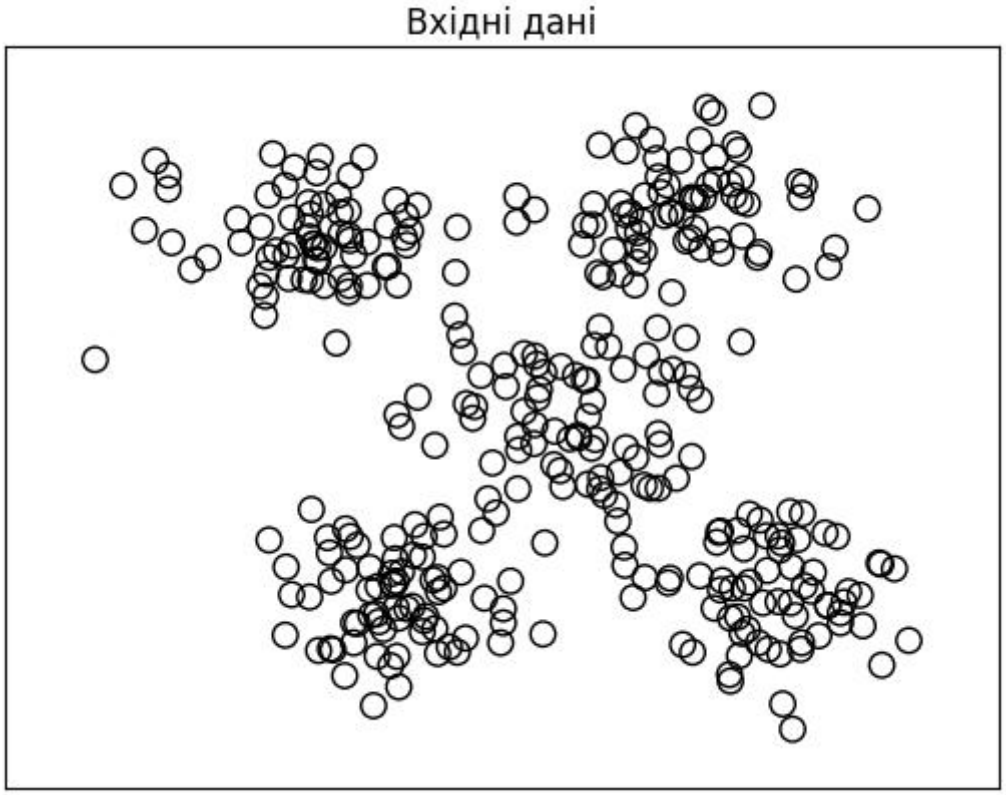

Рис. 1

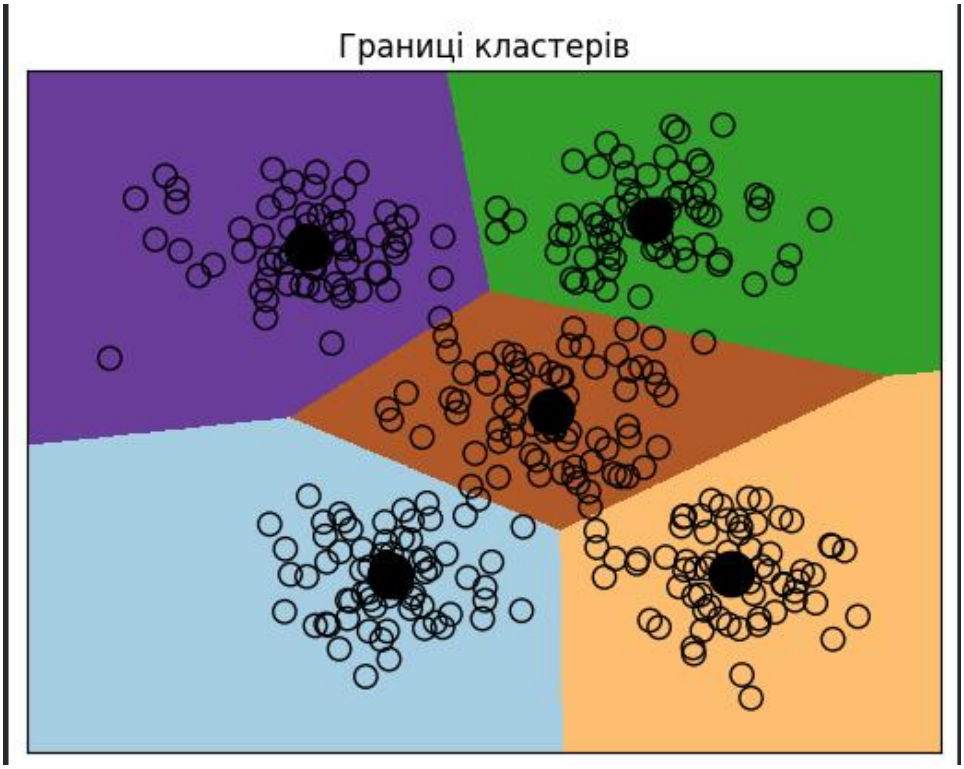

Рис. 2

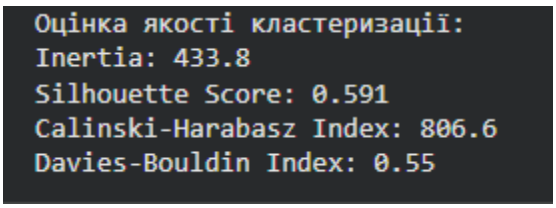

Рис. 3

У завданні було кластеризовано дані за допомогою методу k-середніх. Ми задали кількість кластерів 5 і навчили модель кластеризації KMeans. На графіку (рис. 2) точки даних розташовані у областях різних кольорів, які відповідають окремим кластерам. У центрі груп точок показано центроїди кластерів.

Для оцінки якості кластеризації було розраховано наступні метрики (рис. 3):

- Inertia (сумарна похибка) = 433.8 – це сума квадратів відстаней між точками та їх центроїдами; цей показник потрібно оцінювати відносно кількості кластерів;

- Silhouette Score (коефіцієнт силуету) = 0.591, цей результат можна вважати гарним, оскільки він близький до одиниці, точки добре відокремлені від інших кластерів;

- Calinski–Harabasz Index = 806.6 має велике значення, тобто кластеризація добре відокремлює групи точок;

- Davies–Bouldin Index = 0.55 – це ще один показник, який вказує на чіткість та якість кластеризації, він повинен бути якомога меншим. У нашому випадку кластери добре розділені, мають малу схожість один з одним.

Завдання 7.2. Кластеризація K-середніх для набору даних Iris.

Лістинг програми:

```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.cluster import KMeans
```

```
from sklearn.metrics import silhouette_score, calinski_harabasz_score,
davies_bouldin_score

# Завантажуємо набір даних Iris
iris = load_iris()
X = iris['data']
y = iris['target']

# Ініціалізуємо модель KMeans
# n_clusters=3 — очікуємо 3 класи ірисів
kmeans = KMeans(n_clusters=3, init='k-means++', n_init=10, max_iter=300,
                tol=0.0001, random_state=0)

# Навчаємо модель на даних
kmeans.fit(X)

# Отримуємо передбачені мітки кластерів
y_kmeans = kmeans.predict(X)

# Отримуємо координати центрів кластерів
centers = kmeans.cluster_centers_

# Візуалізація
plt.figure(figsize=(12, 5))

# Справжні класи
plt.subplot(1, 2, 1)
plt.scatter(X[:, 0], X[:, 1], c=y, cmap='viridis', s=50)
plt.title("Справжні класи Iris")
plt.xlabel("Довжина чашолистка")
plt.ylabel("Ширина чашолистка")
for i, name in enumerate(iris['target_names']):
    plt.scatter([], [], color=plt.cm.viridis(i / 2), label=name)
plt.legend()

# Кластери, знайдені методом K-Means
plt.subplot(1, 2, 2)
plt.scatter(X[:, 0], X[:, 1], c=y_kmeans, cmap='viridis', s=50)
plt.scatter(centers[:, 0], centers[:, 1], c='red', s=200, alpha=0.6,
label='Центри кластерів')
plt.title("Результати кластеризації K-Means")
plt.xlabel("Довжина чашолистка")
plt.ylabel("Ширина чашолистка")
for i, name in enumerate(iris['target_names']):
    plt.scatter([], [], color=plt.cm.viridis(i / 2), label=name)
plt.legend()
plt.tight_layout()
plt.show()

# Порівнюємо кластеризацію з реальними класами
```

```
print("Справжні мітки класів (y):")
print(y)
print("Прогнозовані мітки кластерів (y_kmeans):")
print(y_kmeans)

# Оцінюємо якість кластеризації
print("\nОцінка якості кластеризації:")
print("Inertia:", round(kmeans.inertia_, 2))
print("Silhouette Score:", round(silhouette_score(X, y_kmeans), 3))
print("Calinski-Harabasz Index:", round(calinski_harabasz_score(X, y_kmeans),
2))
print("Davies-Bouldin Index:", round(davies_bouldin_score(X, y_kmeans), 2))
```

Результат виконання програми:

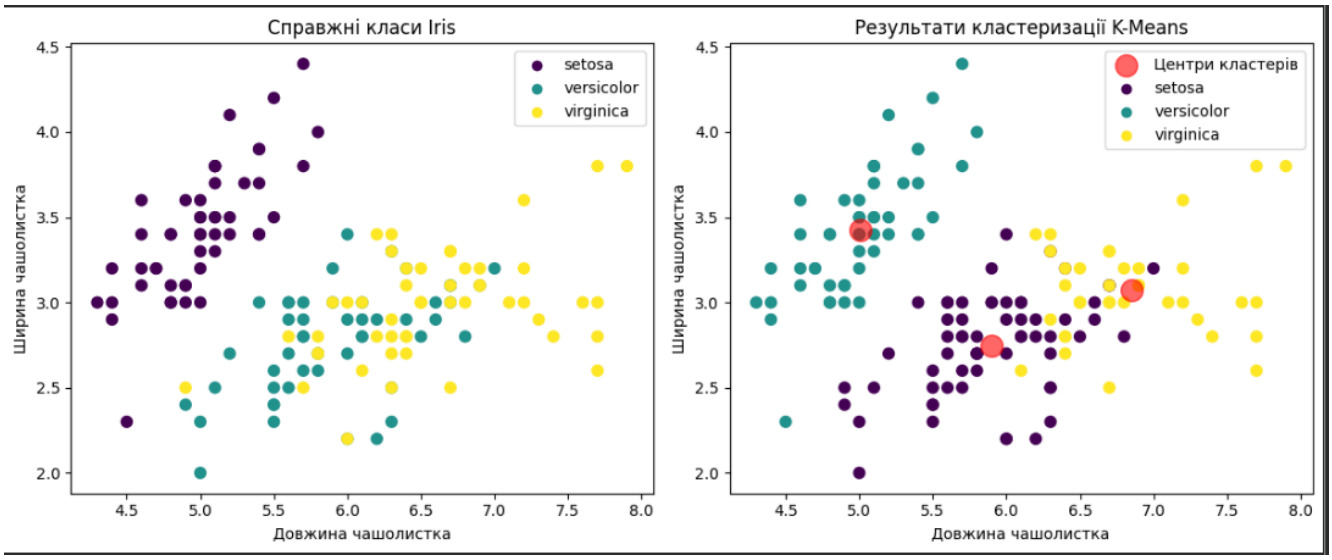

Рис. 4

```
Справжні мітки класів (y):
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
Прогнозовані мітки кластерів (y_kmeans):
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 2 2 2 0 2 2 2 2
 2 2 0 0 2 2 2 2 0 2 0 2 0 2 2 0 0 2 2 2 2 2 0 2 2 2 2 0 2 2 2 0 2 2 2 0 2
 2 0]

Оцінка якості кластеризації:
Inertia: 78.85
Silhouette Score: 0.553
Calinski-Harabasz Index: 561.63
Davies-Bouldin Index: 0.66
```

Рис. 5

У завданні було кластеризовано дані з набору Iris методом K-середніх. Результатом є виділення трьох кластерів (відповідають видам квітів Setosa, Versicolour та Virginica), зображених на графіку (рис. 4).

Алгоритм успішно розділив дані на 3 кластери, проте деякі кластерні мітки не повністю збігаються з реальними класами, що підтверджується виведенням масивів справжніх та прогнозованих міток у консоль (рис. 5). Втім, це властиво для даного методу, тому що він є ненаглядовим. На графіку видно, що клас Setosa відокремлений ідеально, а класи Versicolour і Virginica лише частково перетинаються через схожість їхніх ознак.

Значення вимірюваних метрик також підтверджують високу якість та точність кластеризації.

Завдання 7.3. Оцінка кількості кластерів з використанням методу зсуву середнього.

Лістинг програми:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import MeanShift, estimate_bandwidth
from sklearn.metrics import silhouette_score
from itertools import cycle

# Завантаження даних
X = np.loadtxt('data_clustering.txt', delimiter=',')

# Оцінка ширини вікна для X
bandwidth_X = estimate_bandwidth(X, quantile=0.1, n_samples=len(X))

# Кластеризація даних методом зсуву середнього
meanshift_model = MeanShift(bandwidth=bandwidth_X, bin_seeding=True)
meanshift_model.fit(X)

# Отримання центрів та кількості кластерів
cluster_centers = meanshift_model.cluster_centers_
print(f"Координати центрів кластерів:\n{cluster_centers}")

labels = meanshift_model.labels_
num_clusters = len(np.unique(labels))
```

```
print(f"Кількість кластерів = {num_clusters}")

# Візуалізація результатів
plt.figure(figsize=(10, 6))
markers = 'o*dvs'
colors = cycle('bgrcmyk')
for i, marker, col in zip(range(num_clusters), markers, colors):
    cluster_data = X[labels == i]
    plt.scatter(cluster_data[:, 0], cluster_data[:, 1],
                marker=marker, color=col, edgecolor='black', s=70,
label=f'Кластер {i+1}')

plt.scatter(cluster_centers[:, 0], cluster_centers[:, 1], marker='o',
            color='black', s=200, label='Центри кластерів')

plt.title('Результат кластеризації')
plt.xlabel('X')
plt.ylabel('Y')
plt.legend()
plt.grid(True)
plt.show()

# Оцінюємо якість кластеризації
print("\nОцінка якості кластеризації:")
print("Silhouette Score:", round(silhouette_score(X, labels), 3))
```

Результат виконання програми:

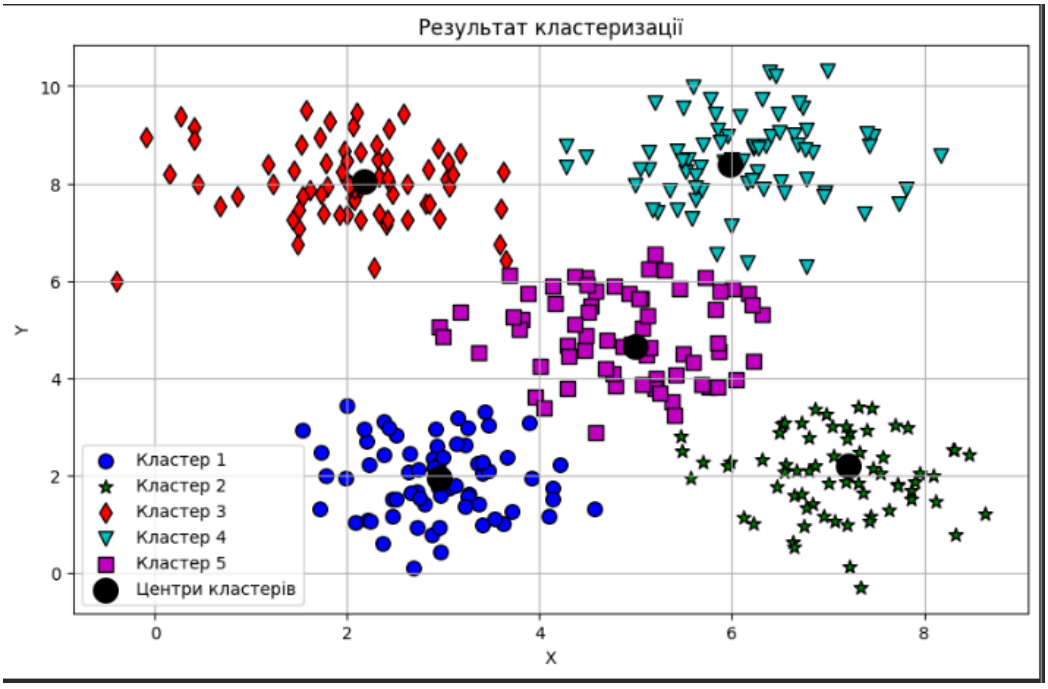

Рис. 6

```
Координати центрів кластерів:
[[2.95568966 1.95775862]
 [7.20690909 2.20836364]
 [2.17603774 8.03283019]
 [5.97960784 8.39078431]
 [4.99466667 4.65844444]]
Кількість кластерів = 5
Оцінка якості кластеризації:
Silhouette Score: 0.587
```

Рис. 7

У завданні було використано метод зсуву середнього (Mean Shift) для кластеризації даних. Даний алгоритм самостійно визначає кількість кластерів (на відміну від K-Means) після оцінки ширини вікна для кожної точки навчального набору та, власне, навчання. Було знайдено 5 кластерів та виведено координати їхніх центрів у консоль (рис. 7).

На графіку (рис. 6) зображено точки даних різного кольору, які відповідають окремим кластерам, а також їхні центроїди. Візуально кластери добре розділені, що підтверджується значенням метрики Silhouette Score = 0.587.

Завдання 7.4. Знаходження підгруп на фондовому ринку з використанням моделі поширення подібності.

Лістинг програми:

```
import datetime
import json
import numpy as np
from sklearn import covariance, cluster
import yfinance as yf
from sklearn.preprocessing import StandardScaler

# Вхідний файл із символічними позначеннями компаній
input_file = 'company_symbol_mapping.json'

# Завантаження прив'язок символів компаній до їх повних назв
with open(input_file, 'r') as f:
    company_symbols_map = json.loads(f.read())

symbols, names = np.array(list(company_symbols_map.items())).T
```

```
# Завантаження архівних даних котирувань
start_date = datetime.datetime(2003, 7, 3)
end_date = datetime.datetime(2007, 5, 4)
quotes = yf.download(list(symbols), start=start_date, end=end_date)

# Вилучення котирувань, що відповідають відкриттю та
# закриттю біржі, та обчислення різниці між ними
opening_quotes = quotes['Open']
closing_quotes = quotes['Close']
quotes_diff = opening_quotes - closing_quotes

# Видалення компаній з некоректними даними
quotes_diff.dropna(axis='columns', how='all', inplace=True)
quotes_diff.dropna(axis='rows', how='any', inplace=True)

# Нормалізація даних
X = quotes_diff.copy()
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Створення моделі графа та її навчання
edge_model = covariance.GraphicalLassoCV(assume_centered=True)
edge_model.fit(X)

# Створення моделі кластеризації на основі поширення подібності
affinity_model =
cluster.AffinityPropagation(preference=np.median(edge_model.covariance_),
random_state=42)
affinity_model.fit(edge_model.covariance_)

labels = affinity_model.labels_
num_labels = len(labels)
valid_symbols = []
for symbol in quotes_diff.columns.tolist():
    valid_symbols.append(company_symbols_map[symbol])

valid_symbols = np.array(valid_symbols)

# Виведення результатів
print(f"Кількість компаній: {len(valid_symbols)}")
print(f"Кількість знайдених кластерів: {num_labels}")
print("\nРезультати кластеризації:")
for i in range(num_labels):
    cluster_companies = valid_symbols[labels == i]
    print(f"Кластер {i+1} ==> {', '.join(cluster_companies)}")
```

Результат виконання програми:

```
[***                   7%                       ]  4 of 60 completedЗавантаження даних з Yahoo Finance...
[**********************100%**********************]  60 of 60 completed
ERROR:yfinance:
11 Failed downloads:
ERROR:yfinance:['RTN', 'CAJ', 'YHOO', 'WBA', 'SNE', 'NAV', 'UN']: YFTzMissingError('possibly delisted; no timezone found')
ERROR:yfinance:['CVC', 'MTU']: YFPricesMissingError('possibly delisted; no price data found  (1d 2003-07-03 00:00:00 -> 2007-05-04 00:00:00)')
ERROR:yfinance:['TOT', 'DELL']: YFPricesMissingError('possibly delisted; no price data found  (1d 2003-07-03 00:00:00 -> 2007-05-04 00:00:00) (Yahc
Початкова кількість компаній: 60
Кількість компаній після очищення: 49
Побудова моделі зв'язків (GraphicalLasso)...
Виконується кластеризація...
```

Рис. 8

```
==================================================
Кількість компаній: 49
Кількість кластерів: 49
==================================================
Кластер 1 ==> Apple
Кластер 2 ==> AIG
Кластер 3 ==> Amazon
Кластер 4 ==> American express
Кластер 5 ==> Boeing
Кластер 6 ==> Bank of America
Кластер 7 ==> Caterpillar
Кластер 8 ==> Colgate-Palmolive
Кластер 9 ==> Comcast
Кластер 10 ==> ConocoPhillips
Кластер 11 ==> Cisco
Кластер 12 ==> CVS
Кластер 13 ==> Chevron
Кластер 14 ==> DuPont de Nemours
Кластер 15 ==> Ford
Кластер 16 ==> General Dynamics
Кластер 17 ==> General Electrics
Кластер 18 ==> Goldman Sachs
Кластер 19 ==> GlaxoSmithKline
Кластер 20 ==> Home Depot
Кластер 21 ==> Honda
Кластер 22 ==> HP
Кластер 23 ==> IBM
Кластер 24 ==> JPMorgan Chase
Кластер 25 ==> Kellogg
Кластер 26 ==> Kimberly-Clark
Кластер 27 ==> Coca Cola
Кластер 28 ==> Lookheed Martin
Кластер 29 ==> Marriott
Кластер 30 ==> Mc Donalds
Кластер 31 ==> Kraft Foods
Кластер 32 ==> 3M
Кластер 33 ==> Microsoft
Кластер 34 ==> Northrop Grumman
Кластер 35 ==> Novartis
Кластер 36 ==> Pepsi
Кластер 37 ==> Pfizer
Кластер 38 ==> Procter Gamble
Кластер 39 ==> Ryder
Кластер 40 ==> SAP
Кластер 41 ==> Sanofi-Aventis
Кластер 42 ==> Toyota
Кластер 43 ==> Time Warner
Кластер 44 ==> Texas instruments
Кластер 45 ==> Valero Energy
Кластер 46 ==> Wells Fargo
Кластер 47 ==> Wal-Mart
Кластер 48 ==> Exxon
Кластер 49 ==> Xerox
```

Рис. 9

У завданні було завантажено дані компаній, символи яких містяться у файлі. Список компаній було модифіковано, для навчання моделі зі списку були видалені компанії, які взагалі не мають даних або вони некоректні. Після цього було нормалізовано дані. Створено крайову модель, за допомогою якої навчено модель кластеризації. У результаті компанії у списку було розділено між кластерами.

Отримано наступні результати (рис. 8 – 9):

- Кількість компаній: 49.
- Кількість знайдених кластерів: 49.

Для кожної компанії утворено окремий кластер, так як алгоритм не виявив необхідної подібності між поведінкою котирувань різних компаній для їх об'єднання у спільний кластер. Для моделі ми задали параметр preference=np.median(edge_model.covariance_), який визначає, наскільки кожна точка схильна бути центром кластера, враховуючи медіанне значення для матриці коваріацій, тобто зв'язків між компаніями. Для отримання меншої кількості кластерів необхідно зменшити preference, задавши йому числове значення менше 0 або середнє арифметичне np.mean(edge_model.covariance_).

Висновок: в ході виконання лабораторної роботи ми дослідили методи неконтрольованої класифікації даних у машинному навчанні, використовуючи спеціалізовані бібліотеки та мову програмування Python.

Репозиторій: https://github.com/Vanchik21/AI