Лабораторна робота №7

Тема: ДОСЛІДЖЕННЯ МЕТОДІВ НЕКОНТРОЬОВАНОГО НАВЧАННЯ

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

Хід роботи:

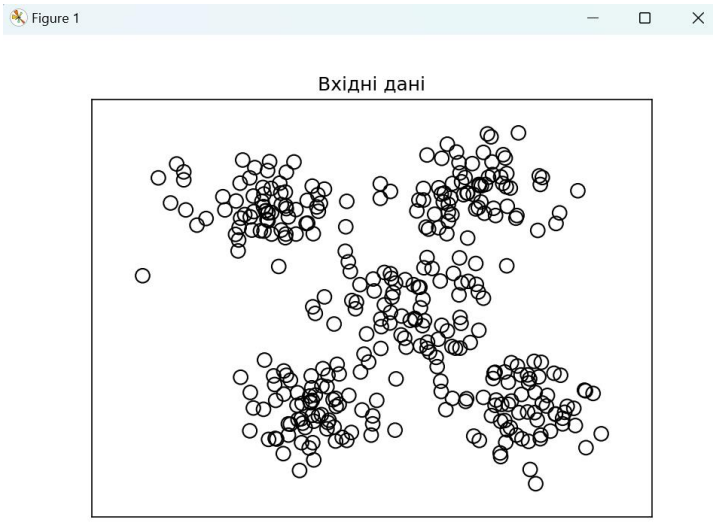
Завдання 7.1. Кластеризація даних за допомогою методу k-середніх.

Лістинг програми:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn import metrics

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

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()

num_clusters = 5
kmeans = KMeans(init='k-means++', n_clusters=num_clusters, n_init=10)
kmeans.fit(X)
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)
)
output = kmeans.predict(np.c_[x_vals.ravel(), y_vals.ravel()])
output = output.reshape(x_vals.shape)

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("="*40)
print("РЕЗУЛЬТАТИ ОЦІНКИ ЯКОСТІ КЛАСТЕРИЗАЦІЇ")
print("="*40)

silhouette_avg = metrics.silhouette_score(X, kmeans.labels_)
print(f"Коефіцієнт силуету (Silhouette Score): {silhouette_avg:.4f}")

ch_score = metrics.calinski_harabasz_score(X, kmeans.labels_)
print(f"Індекс Калінскі-Харабаса: {ch_score:.4f}")

db_score = metrics.davies_bouldin_score(X, kmeans.labels_)
print(f"Індекс Девіса-Болдіна: {db_score:.4f}")

print("="*40)
```

Результат роботи програми:



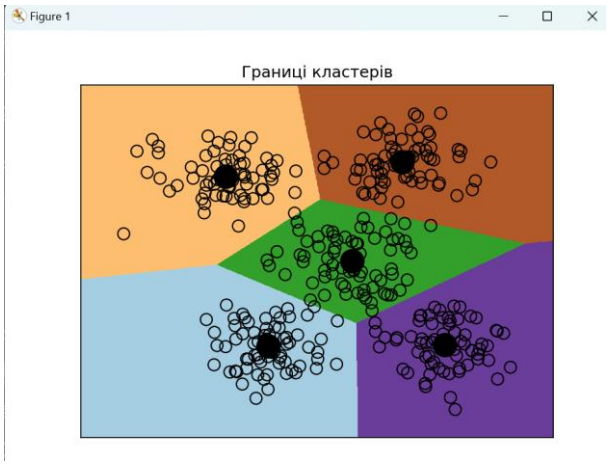


Рис. 7.1. Границі кластерів

```
C:\Users\Admin\AppData\Local\Programs\Python\Python39
========================================
РЕЗУЛЬТАТИ ОЦІНКИ ЯКОСТІ КЛАСТЕРИЗАЦІЇ
========================================
Коефіцієнт силуету (Silhouette Score): 0.5907
Індекс Калінскі-Харабаса: 806.6048
Індекс Девіса-Болдіна: 0.5513
========================================

Process finished with exit code 0
```

Рис. 7.2. Вивід у консоль

Висновок до завдання: Силуетний коефіцієнт = 0.5907. Значення близько до 0.6 свідчить про достатню структуру кластерів: об’єкти переважно добре відокремлені один від одного, але ще є невелике перекриття між деякими кластерами.

Індекс Калінскі-Харабаса = 806.60. Високе значення цього індексу підтверджує хорошу компактність і відокремленість кластерів. Чим вище цей показник, тим чіткіше виділяються кластери.

Індекс Девіса-Болдіна = 0.5513. Низьке значення вказує на менше перекриття між кластерами та високу відокремленість. Значення близько 0.55 свідчить про гарну якість кластеризації.

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

Лістинг програми:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.metrics import pairwise_distances_argmin
from sklearn.datasets import load_iris

iris = load_iris()
X = iris['data']
y = iris['target']

kmeans = KMeans(n_clusters=5, init='k-means++', n_init=10, max_iter=300,
random_state=0)
kmeans.fit(X)
y_kmeans = kmeans.predict(X)

plt.figure(figsize=(6, 5))
plt.scatter(X[:, 0], X[:, 1], c=y_kmeans, s=50, cmap='viridis')
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)
plt.title("KMeans (sklearn) на Iris")
plt.show()

def find_clusters(X, n_clusters, rseed=2):
    rng = np.random.RandomState(rseed)
    i = rng.permutation(X.shape[0])[:n_clusters]
    centers = X[i]
    while True:
        labels = pairwise_distances_argmin(X, centers)
        new_centers = np.array([X[labels == j].mean(0) for j in
range(n_clusters)])
        if np.allclose(centers, new_centers):
            break
        centers = new_centers
    return centers, labels

centers, labels = find_clusters(X, 3)
plt.figure(figsize=(6, 5))
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)
plt.title("Власна реалізація KMeans (3 кластери, rseed=2)")
plt.show()

centers, labels = find_clusters(X, 3, rseed=0)
plt.figure(figsize=(6, 5))
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)
plt.title("Власна реалізація KMeans (3 кластери, rseed=0)")
plt.show()

labels = KMeans(n_clusters=3, random_state=0).fit_predict(X)
plt.figure(figsize=(6, 5))
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
plt.title("KMeans (sklearn, 3 кластери)")
plt.show()
```

Результат роботи програми:

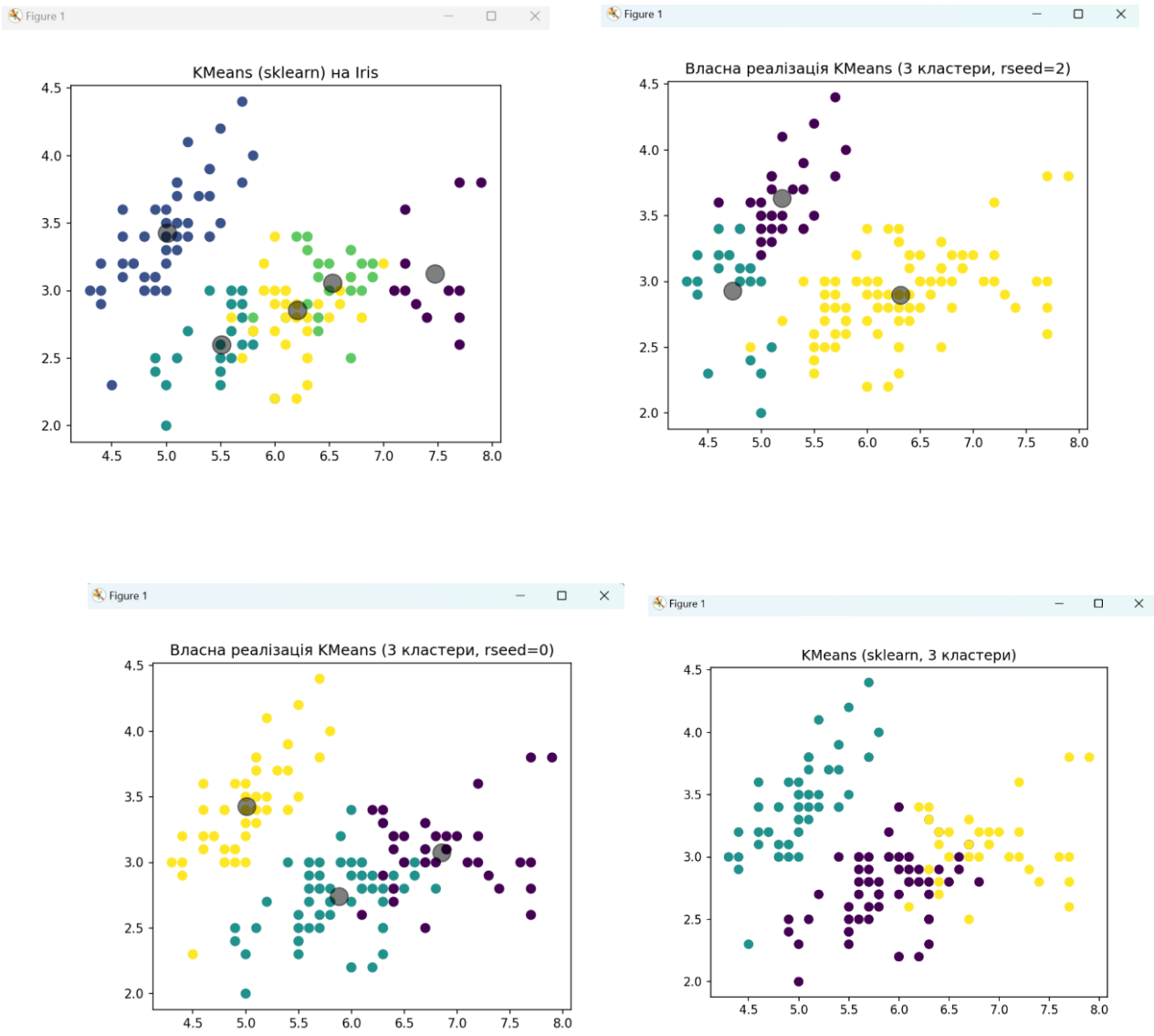


Рис. 7.3. Порівняння результатів

Висновки до завдання: під час виконання завдання з кластеризації даних Iris за допомогою алгоритму KMeans було виділено окремі групи квіток за схожістю ознак. Алгоритм коректно розділив дані на кластери, близькі до реальних видів Iris, хоча результати залежать від початкової ініціалізації центрів. Власна реалізація KMeans допомогла зрозуміти механізм роботи алгоритму та вплив параметрів на результат. Отримані кластери демонструють хорошу відокремленість та узгоджуються з очікуваною структурою даних.

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 itertools import cycle

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

bandwidth_X = estimate_bandwidth(X, quantile=0.1, n_samples=len(X))

meanshift_model = MeanShift(bandwidth=bandwidth_X, bin_seeding=True)
meanshift_model.fit(X)

cluster_centers = meanshift_model.cluster_centers_
labels = meanshift_model.labels_
num_clusters = len(np.unique(labels))

print(f"Координати центрів кластерів:\n{cluster_centers}")
print(f"Оцінена кількість кластерів = {num_clusters}")

plt.figure()
markers = 'o*xvsD'
marker_cycle = cycle(markers)

for i in range(num_clusters):
    cluster_points = X[labels == i]
    marker = next(marker_cycle)
    plt.scatter(
        cluster_points[:, 0],
        cluster_points[:, 1],
        marker=marker,
        color='black',
        s=50,
        label=f'Кластер {i}'
    )

plt.scatter(
    cluster_centers[:, 0],
    cluster_centers[:, 1],
    marker='P',
    color='red',
    s=200,
    zorder=10,
    label='Центри кластерів'
)

plt.title('Кластеризація методом зсуву середнього (Mean Shift)')
plt.xlabel('Ознака 1')
plt.ylabel('Ознака 2')
plt.legend()
plt.grid(True, linestyle='--', alpha=0.6)
plt.show()
```

Результат роботи програми:

```
C:\Users\Admin\AppData\Local\Programs\
Координати центрів кластерів:
[[2.95568966 1.95775862]
 [7.20690909 2.20836364]
 [2.17603774 8.03283019]
 [5.97960784 8.39078431]
 [4.99466667 4.65844444]]
Оцінена кількість кластерів = 5

Process finished with exit code 0
```

Рис. 7.4. Вивід у консоль

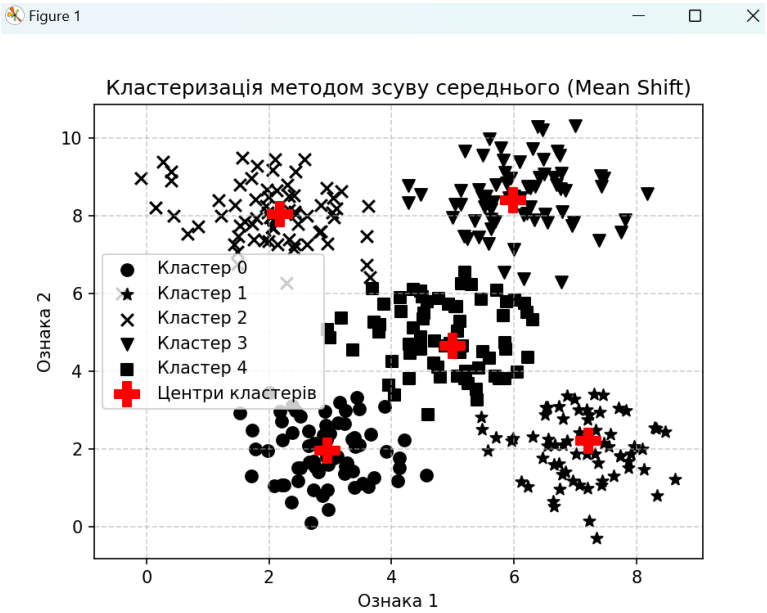


Рис. 7.5. Кластерізація

Висновки до завдання: результаті виконання кластеризації методом зсуву середнього (Mean Shift) на наборі даних data_clustering.txt було виявлено 5 кластерів. Координати центрів кластерів відображають локальні групи даних у двовимірному просторі ознак:

1. [2.956, 1.958]
2. [7.207, 2.208]
3. [2.176, 8.033]
4. [5.980, 8.391]
5. [4.995, 4.658]

Це свідчить, що дані мають структуру, де спостереження природньо групуються у п’ять різних категорій. Візуалізація показала чітке розділення груп і дозволила ідентифікувати центри кластерів.

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

Лістинг програми:

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

input_file = 'company_symbol_mapping.json'
with open(input_file, 'r') as f:
    company_symbols_map = json.loads(f.read())

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

start_date = datetime.datetime(2003, 7, 3)
end_date = datetime.datetime(2007, 5, 4)

print(f"Завантаження даних для {len(symbols)} компаній...")
data = yf.download(list(symbols), start=start_date, end=end_date)
print("Завантаження завершене.")

opening_quotes = data['Open']
closing_quotes = data['Close']
quotes_diff = opening_quotes - closing_quotes

print(f"\nПочаткова кількість компаній: {quotes_diff.shape[1]}")

quotes_diff.dropna(axis='columns', how='all', inplace=True)
print(f"Компаній після видалення незавантажених: {quotes_diff.shape[1]}")

quotes_diff.dropna(axis='rows', how='any', inplace=True)
print(f"Кількість днів з повними даними: {quotes_diff.shape[0]}")

remaining_symbols = quotes_diff.columns.tolist()
names = np.array([company_symbols_map[s] for s in remaining_symbols])

X = quotes_diff.copy()
scaler = RobustScaler()
X = scaler.fit_transform(X)

edge_model = covariance.GraphicalLassoCV(assume_centered=True)
edge_model.fit(X)

median_val = np.median(edge_model.covariance_)
af_model = cluster.AffinityPropagation(preference=median_val, random_state=42)
af_model.fit(edge_model.covariance_)

labels = af_model.labels_
num_labels = labels.max()
print("\n--- Результати кластеризації компаній ---")
for i in range(num_labels + 1):
    cluster_members = names[labels == i]
    print(f"Кластер {i+1} ==> {', '.join(cluster_members)}")
```

Результат роботи програми:

```
Початкова кількість компаній: 60
Компаній після видалення незавантажених: 49
Кількість днів з повними даними: 965
```

```
--- Результати кластеризації компаній ---
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
```

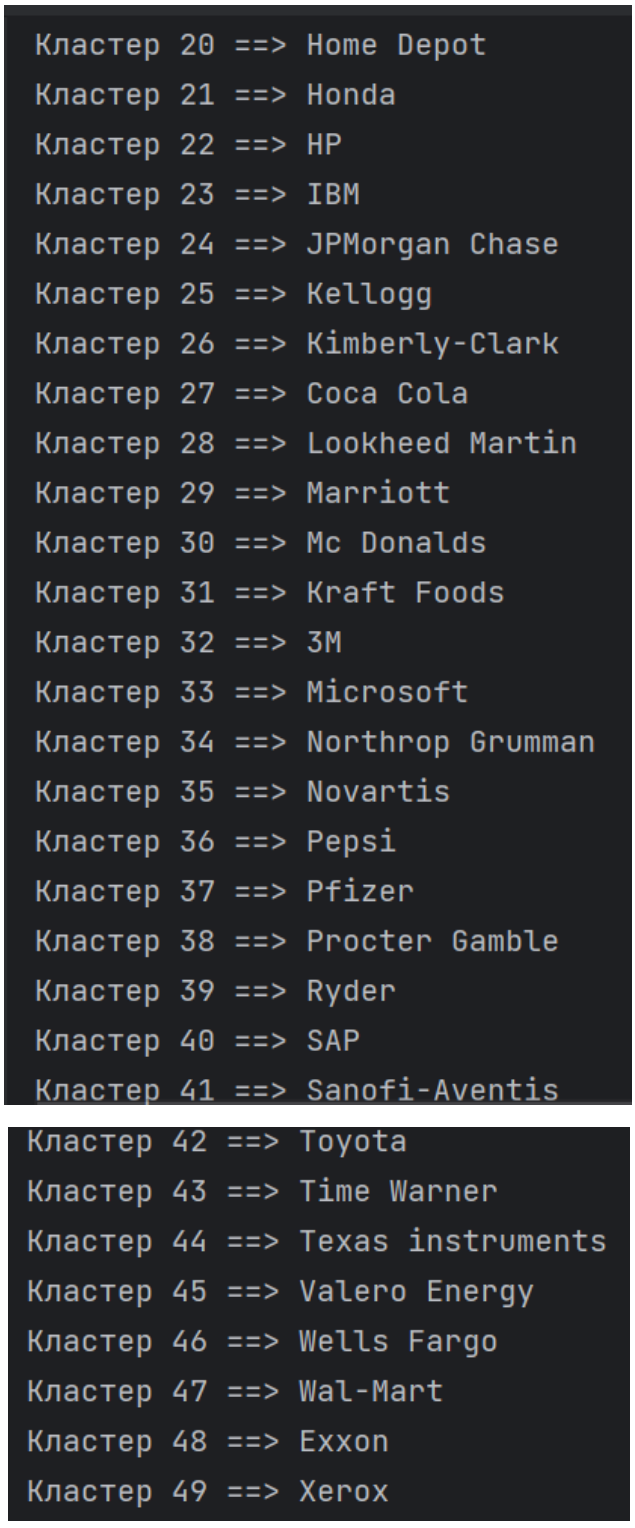


Рис. 7.6. Вивід у консоль

Висновок: під час виконання лабораторної роботи використовуючи спеціалізовані бібліотеки та мову програмування Python досліджено методи неконтрольованої класифікації даних у машинному навчанні.

Посилання на git: https://github.com/KokhanTetiana/AI-Systems