ЛАБОРАТОРНА РОБОТА № 4

ДОСЛІДЖЕННЯ МЕТОДІВ НЕКОНТРОЬОВАНОГО НАВЧАННЯ

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

Хід роботи

Завдання 2.1. Кластеризація даних за допомогою методу k-середніх.

```
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=',')
num_clusters = 5


plt.figure()
plt.scatter(X[:,0], X[:,1], marker='o', facecolors='none',edgecolors='black', s=80)
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.title('Input data')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())

kmeans = KMeans(init='k-means++', n_clusters=num_clusters, n_init=10)

kmeans.fit(X)

step_size = 0.01

x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
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',
```

					Державний університет "Житомирська політехніка"		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Дроботун Д. Я.			Звіт з практичної роботи	Літ.	Арк. 1 / Аркушів
Перевір.							
Керівник						Гр. ІПЗК-19-1	
Н. контр.							
Зав. каф.							

```
        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')

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.title('Boundaries of clusters')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

Рис. 1.1 Код програми

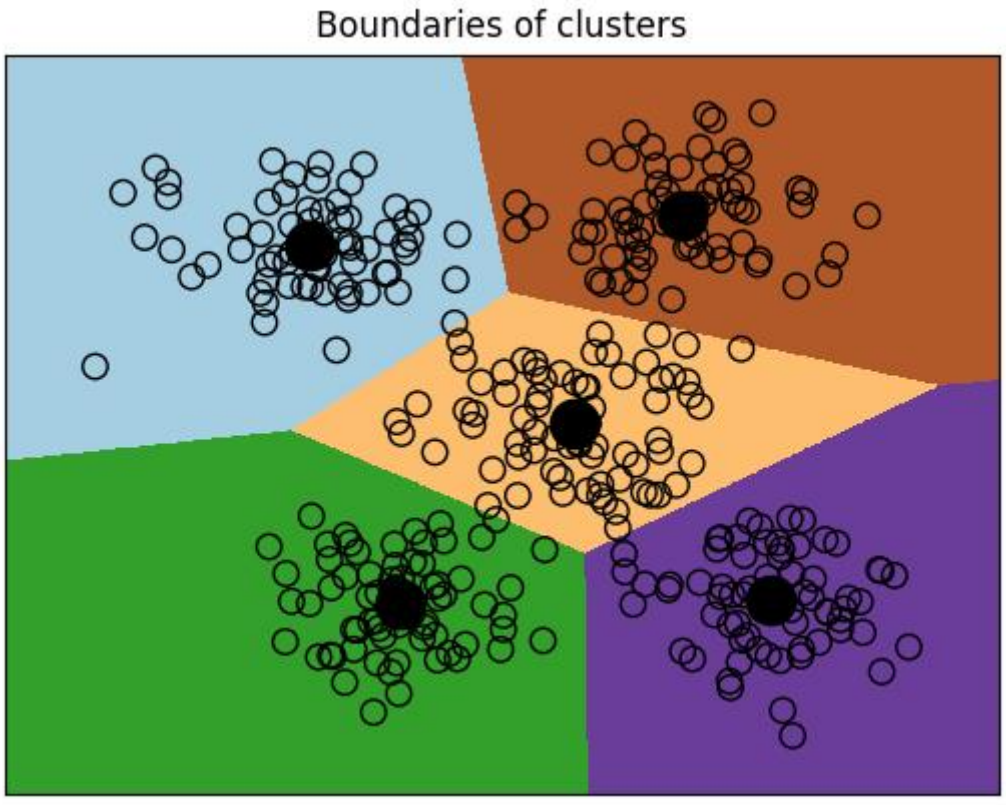

Рис. 1.2 Результат виконання програми

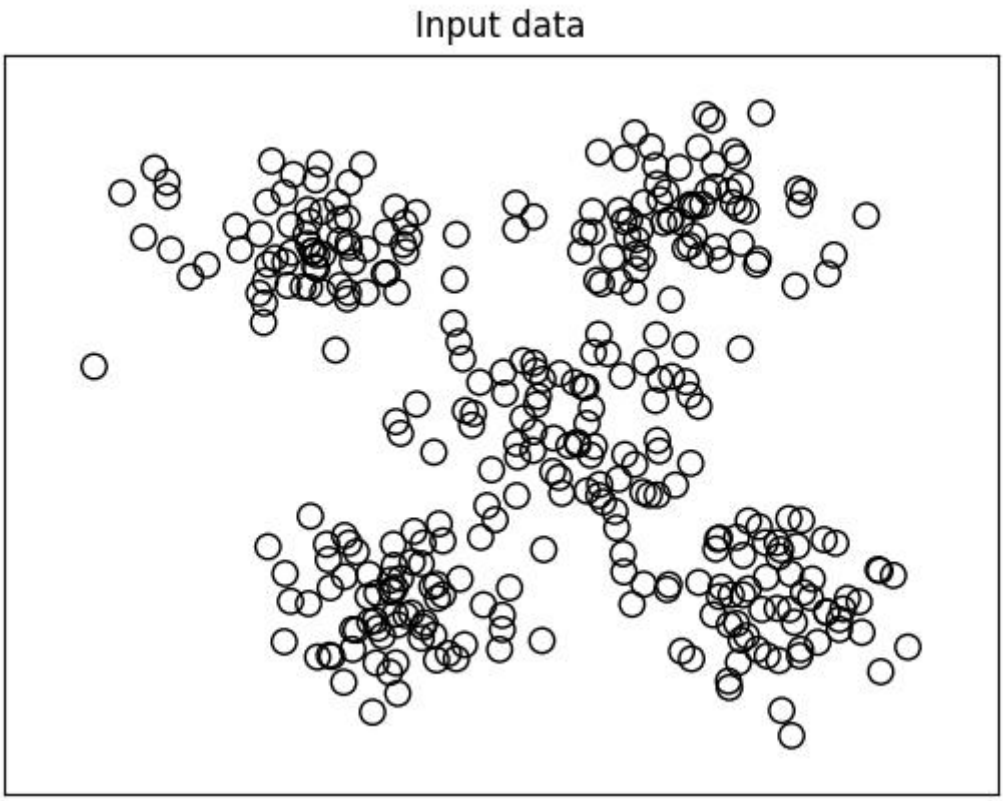

Рис. 1.3 Результат виконання програми

Завдання 2.2. Кластеризація K-середніх для набору даних Iris

```
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
import numpy as np
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans

X, y_true = make_blobs(n_samples=300, centers=5, cluster_std=0.60, random_state=0)

plt.scatter(X[:, 0], X[:, 1], s=50)

# Створення об'єкту KMeans
kmeans = KMeans(n_clusters=5)

# Навчання моделі кластеризації KMeans
kmeans.fit(X)
y_kmeans = kmeans.predict(X)

# Відображення вхідних точок
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)

from sklearn.metrics import pairwise_distances_argmin
```

```
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 == i].mean(0)
                                for i in range(n_clusters)])

        if np.all(centers == new_centers):
            break
        centers = new_centers

    return centers, labels


# Відображення центрів кластерів
centers, labels = find_clusters(X, 5)
plt.scatter(X[:, 0], X[:, 1], c=labels,
            s=50, cmap='viridis')
plt.show()
centers, labels = find_clusters(X, 5, rseed=0)
plt.scatter(X[:, 0], X[:, 1], c=labels,
            s=50, cmap='viridis')
plt.show()
labels = KMeans(5, random_state=0).fit_predict(X)
plt.scatter(X[:, 0], X[:, 1], c=labels,
            s=50, cmap='viridis')
plt.show()
```

Рис. 1.4 Код програми

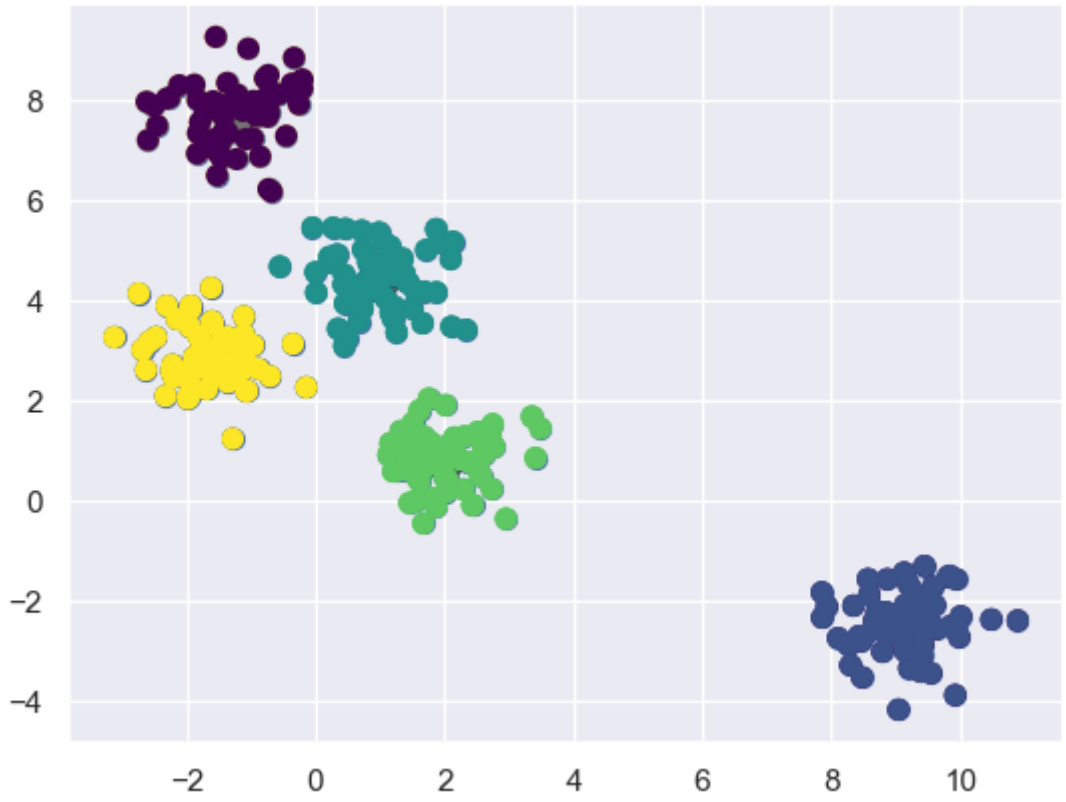

Рис. 1.5 Результат виконання програми

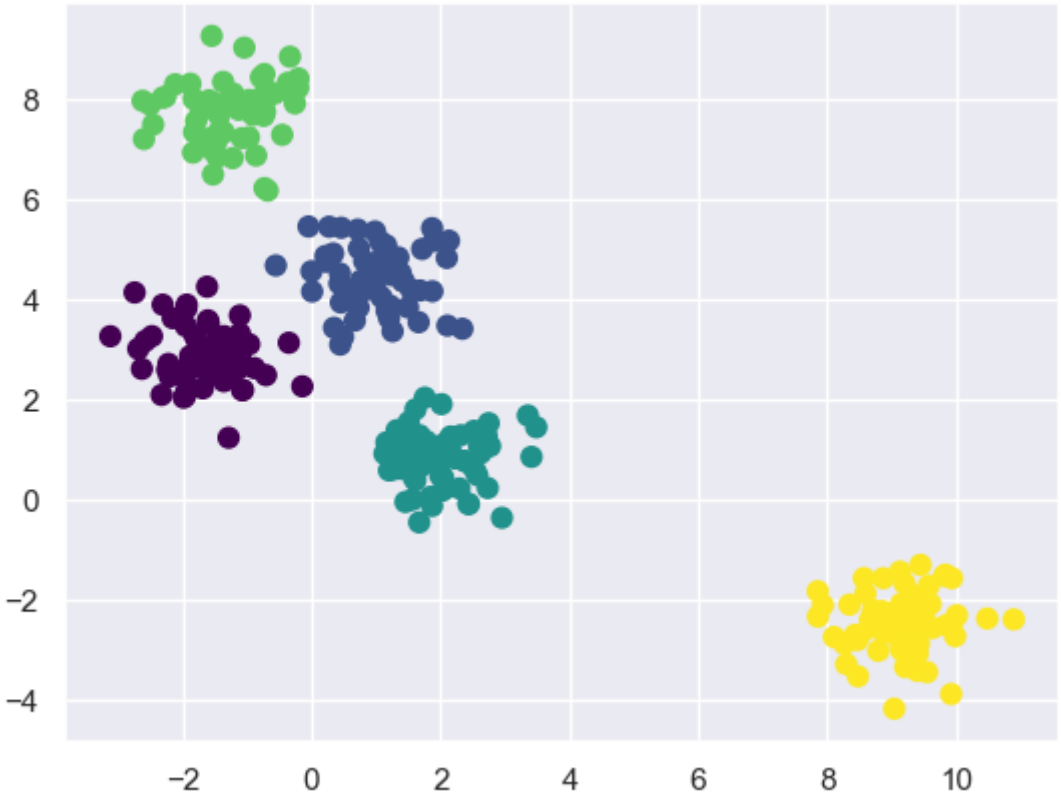

Рис. 1.6 Результат виконання програми

Рис. 1.7 Результат виконання програми

Завдання 2.3. Оцінка кількості кластерів з використанням методу зсуву

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import MeanShift, estimate_bandwidth

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_
print('\nCenters of clusters:\n', cluster_centers)

labels = meanshift_model.labels_
num_clusters = len(np.unique(labels))
print("\nNumber of clusters in input data =", num_clusters)

plt.figure()
markers = 'o*xvs'
for i, marker in zip(range(num_clusters), markers):

    plt.scatter(X[labels==i, 0], X[labels==i, 1], marker=marker, color='black')
```

```
    cluster_center = cluster_centers[i]
    plt.plot(cluster_center[0], cluster_center[1], marker='o',
            markerfacecolor='black', markeredgecolor='black',
            markersize=15)

plt.title('Clusters')
plt.show()
```

Рис. 1.8 Код програми

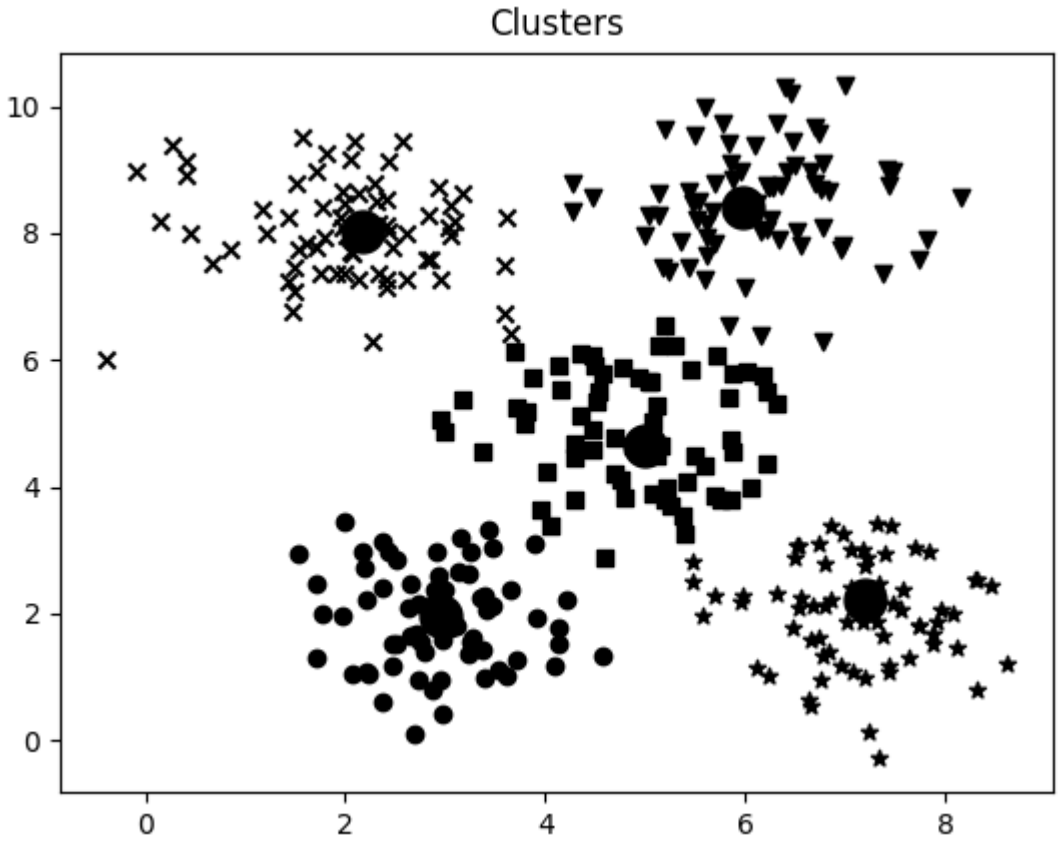

Рис. 1.9 Результат виконання програми

```
Centers of clusters:
 [[2.95568966 1.95775862]
 [7.20690909 2.20836364]
 [2.17603774 8.03283019]
 [5.97960784 8.39078431]
 [4.99466667 4.65844444]]

Number of clusters in input data = 5
```

Рис. 2.1 Результат виконання програми

Завдання 2.4. Знаходження підгруп на фондовому ринку з використанням моделі поширення подібності

```
import datetime
import json

import numpy as np
from sklearn import covariance, cluster

from matplotlib.finance import quotes_historical_yahoo_ochl as quotes_yahoo

input_file = 'company_symbol_mapping.json'


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

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

start_date = datetime.datetime(2003, 7, 3)
end_date = datetime.datetime(2007, 5, 4)
quotes = [quotes_yahoo(symbol, start_date, end_date, asobject=True)
            for symbol in symbols]

opening_quotes = np.array([quote.open for quote in quotes]).astype(np.float)
closing_quotes = np.array([quote.close for quote in quotes]).astype(np.float)

quotes_diff = closing_quotes - opening_quotes

X = quotes_diff.copy().T
X /= X.std(axis=0)

edge_model = covariance.GraphLassoCV()

with np.errstate(invalid='ignore'):
    edge_model.fit(X)

_, labels = cluster.affinity_propagation(edge_model.covariance_)
num_labels = labels.max()

for i in range(num_labels + 1):
    print("Cluster", i+1, "==>", ', '.join(names[labels == i]))
```

Рис. 2.2 Код програми

```
{
    "TOT": "Total",
    "XOM": "Exxon",
    "CVX": "Chevron",
    "COP": "ConocoPhillips",
    "VLO": "Valero Energy",
    "MSFT": "Microsoft",
    "IBM": "IBM",
    "TWX": "Time Warner",
    "CMCSA": "Comcast",
    "CVC": "Cablevision",
    "YHOO": "Yahoo",
```

```
  "DELL": "Dell",
  "HPQ": "HP",
  "AMZN": "Amazon",
  "TM": "Toyota",
  "CAJ": "Canon",
  "MTU": "Mitsubishi",
  "SNE": "Sony",
  "F": "Ford",
  "HMC": "Honda",
  "NAV": "Navistar",
  "NOC": "Northrop Grumman",
  "BA": "Boeing",
  "KO": "Coca Cola",
  "MMM": "3M",
  "MCD": "Mc Donalds",
  "PEP": "Pepsi",
  "MDLZ": "Kraft Foods",
  "K": "Kellogg",
  "UN": "Unilever",
  "MAR": "Marriott",
  "PG": "Procter Gamble",
  "CL": "Colgate-Palmolive",
  "GE": "General Electrics",
  "WFC": "Wells Fargo",
  "JPM": "JPMorgan Chase",
  "AIG": "AIG",
  "AXP": "American express",
  "BAC": "Bank of America",
  "GS": "Goldman Sachs",
  "AAPL": "Apple",
  "SAP": "SAP",
  "CSCO": "Cisco",
  "TXN": "Texas instruments",
  "XRX": "Xerox",
  "LMT": "Lookheed Martin",
  "WMT": "Wal-Mart",
  "WBA": "Walgreen",
  "HD": "Home Depot",
  "GSK": "GlaxoSmithKline",
  "PFE": "Pfizer",
  "SNY": "Sanofi-Aventis",
  "NVS": "Novartis",
  "KMB": "Kimberly-Clark",
  "R": "Ryder",
  "GD": "General Dynamics",
  "RTN": "Raytheon",
  "CVS": "CVS",
  "CAT": "Caterpillar",
  "DD": "DuPont de Nemours"
}
```


Рис. 2.3 Результат виконання програми

Висновок: я використовав спеціалізовані бібліотеки та мову програмування Python дослідивши методи регресії даних у машинному навчанні.

Метод k-середніх (k-means) - це добре відомий алгоритм кластеризації. Його використання передбачає, що кількість кластерів заздалегідь відома. Далі ми сегментуємо дані до підгруп, застосовуючи різні атрибути даних. Ми починаємо з того, що фіксуємо кількість кластерів та, виходячи з цього, класифікуємо дані.

Основна ідея полягає в оновленні положень центроїдів (центрів тяжіння кластеру, або головні точки) на кожній ітерації. Ітеративний процес продовжується до тих пір, поки всі центроїди не займуть оптимального положення.Як неважко здогадатися, у цьому алгоритмі вибір початкового розташування центроїдів відіграє дуже важливу роль, оскільки це безпосередньо впливає на кінцеві результати.

Одна із стратегій полягає в тому, щоб центроїди розташовувалися на якомога більшій відстані один від одного. Базовому методу k-середніх відповідає випадкове розташування центроїдів, тоді як у вдосконаленому варіанті методу (k-means++) ці точки вибираються алгоритмічно з списку вхідних точок даних.