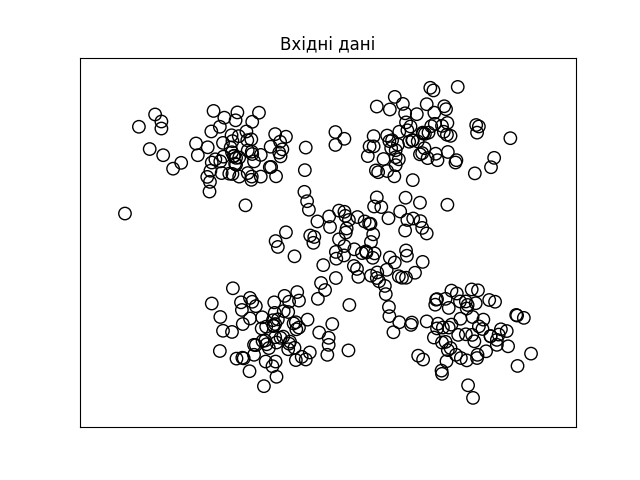
**Лабораторна робота № 7**

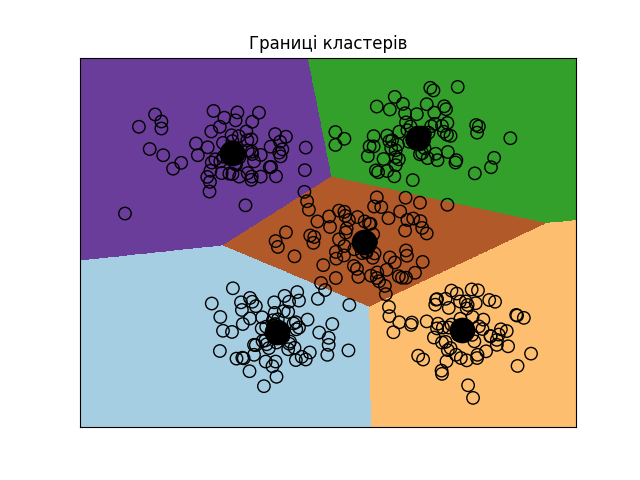
**ДОСЛІДЖЕННЯ МЕТОДІВ НЕКОНТРОЬОВАНОГО НАВЧАННЯ**

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**





Лістинг:

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("Вхідні дані")

plt.xlim(x\_min, x\_max)

plt.ylim(y\_min, y\_max)

plt.xticks(())

plt.yticks(())

#plt.show()

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', *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, *linewidth*=4, *color*='black', *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("Границі кластерів")

plt.xlim(x\_min, x\_max)

plt.ylim(y\_min, y\_max)

plt.xticks(())

plt.yticks(())

plt.show()

**Завдання 2.2. Кластеризація K-середніх для набору даних Iris**

Лістинг:

from sklearn.svm import SVC

from sklearn.metrics import pairwise\_distances\_argmin

import numpy as np

from sklearn.datasets import load\_iris

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

iris = load\_iris()

X = iris['data']  # Завантаження даних із набору "iris"

y = iris['target']  # Завантаження цільових міток із набору "iris"

# Створення об'єкта KMeans з неправильними аргументами (потребує виправлення)

kmeans = KMeans(*n\_clusters*=8, *init*='k-means++', *n\_init*=10, *max\_iter*=300, *tol*=0.0001,

*verbose*=0, *random\_state*=None, *algorithm*='auto')

kmeans = KMeans(*n\_clusters*=5)  # Ініціалізація KMeans з 5 кластерами

kmeans.fit(X)  # Навчання моделі KMeans на даних X

y\_kmeans = kmeans.predict(X)  # Прогнозування кластерів для X

# Візуалізація даних із кольоровими мітками кластерів

plt.scatter(X[:, 0], X[:, 1], *c*=y\_kmeans, *s*=50, *cmap*='viridis')

centers = kmeans.cluster\_centers\_  # Отримання центрів кластерів

plt.scatter(centers[:, 0], centers[:, 1], *c*='black', *s*=200, *alpha*=0.5)  # Візуалізація центрів кластерів

# Визначення функції для пошуку кластерів

*def* find\_clusters(*X*, *n\_clusters*, *rseed*=2):

    rng = np.random.RandomState(*rseed*)  # Створення об'єкта генератора випадкових чисел

    i = rng.permutation(*X*.shape[0])[:*n\_clusters*]  # Випадковий вибір початкових центрів

    centers = *X*[i]  # Ініціалізація центрів кластерів

    while True:

        # Присвоєння кожній точці найближчого центру кластера

        labels = pairwise\_distances\_argmin(*X*, centers)

        # Обчислення нових центрів як середнє значення точок у кожному кластері

        new\_centers = np.array([*X*[labels == i].mean(0) for i in range(*n\_clusters*)])

        # Перевірка, чи змінилися центри; якщо ні, припинити цикл

        if np.all(centers == new\_centers):

            break

        centers = new\_centers  # Оновлення центрів

    return centers, labels  # Повернення центрів і міток

centers, labels = find\_clusters(X, 3)  # Виклик функції для пошуку 3 кластерів

plt.scatter(X[:, 0], X[:, 1], *c*=labels, *s*=50, *cmap*='viridis')  # Візуалізація з новими мітками

plt.show()

centers, labels = find\_clusters(X, 3, *rseed*=0)  # Виклик функції з іншим значенням rseed

plt.scatter(X[:, 0], X[:, 1], *c*=labels, *s*=50, *cmap*='viridis')  # Візуалізація результатів

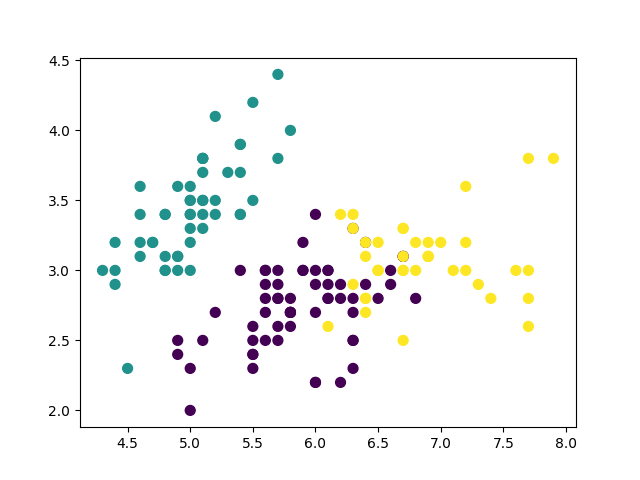
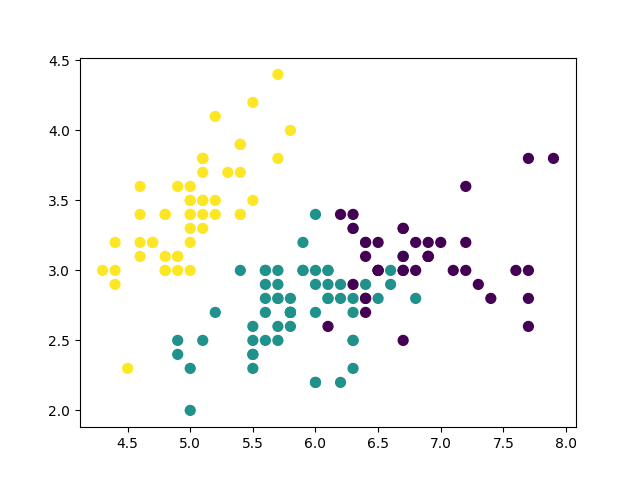
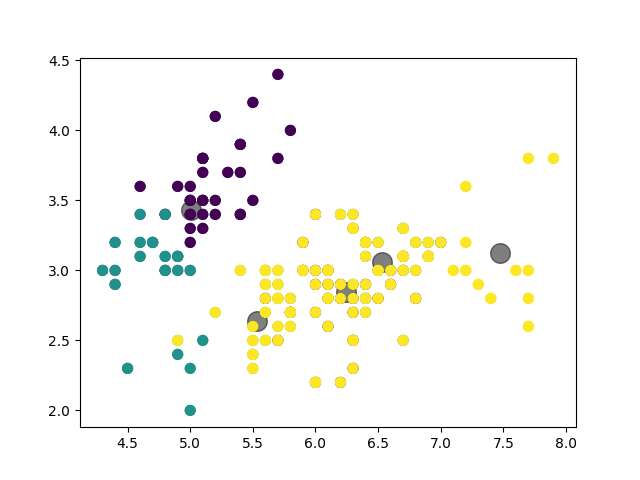
plt.show()

# Прогнозування кластерів за допомогою KMeans з 3 кластерами

labels = KMeans(3, *random\_state*=0).fit\_predict(X)

plt.scatter(X[:, 0], X[:, 1], *c*=labels, *s*=50, *cmap*='viridis')  # Візуалізація кластерів

plt.show()



**Завдання 2.3. Оцінка кількості кластерів з використанням методу зсуву середнього**

Лістинг:

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\_

print("clusters centers: \n", cluster\_centers)

labels = meanshift\_model.labels\_

num\_clusters = len(np.unique(labels))

print("\n Number 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("Кластери")

    plt.show()

Виконання програми:

[Running] python -u "d:\ztu\KURS4\ai\Lab7\LR\_4\_task\_3.py"

clusters centers:

 [[2.95568966 1.95775862]

 [7.20690909 2.20836364]

 [2.17603774 8.03283019]

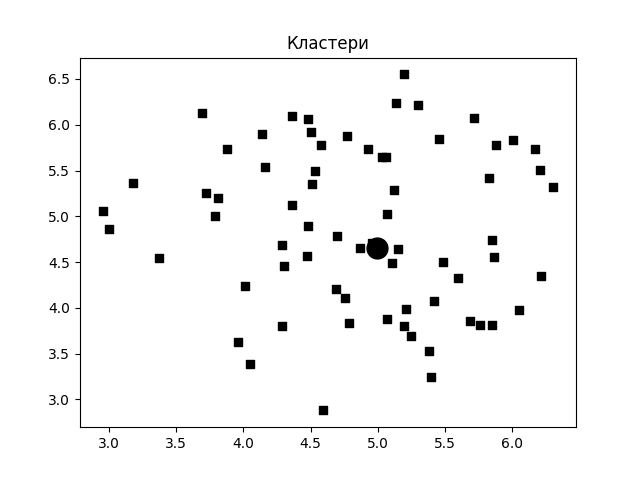
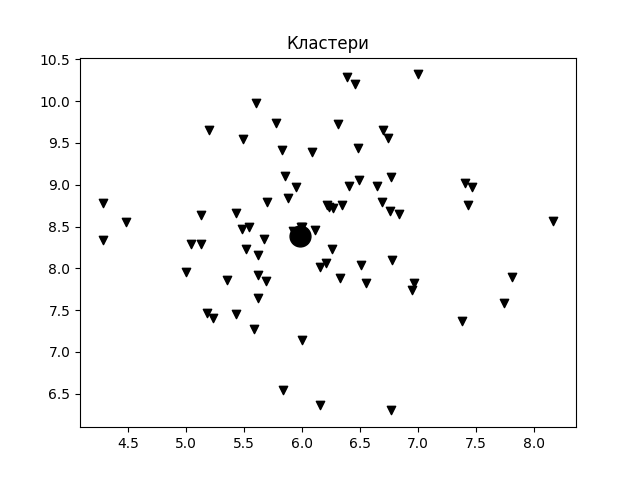
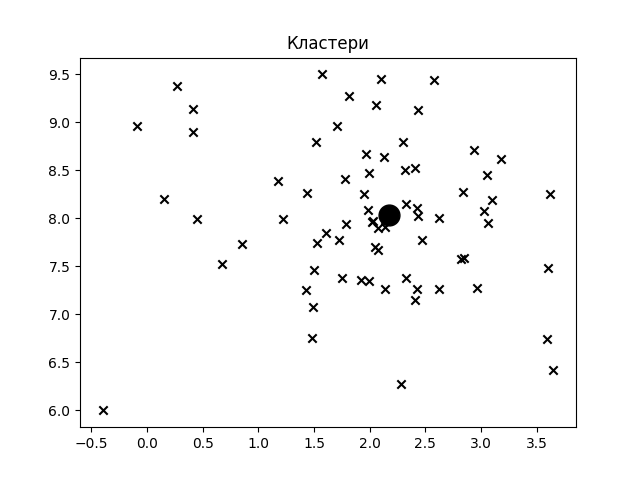
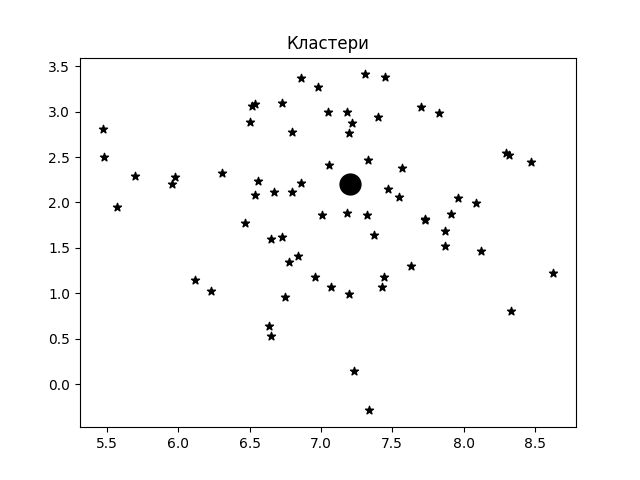
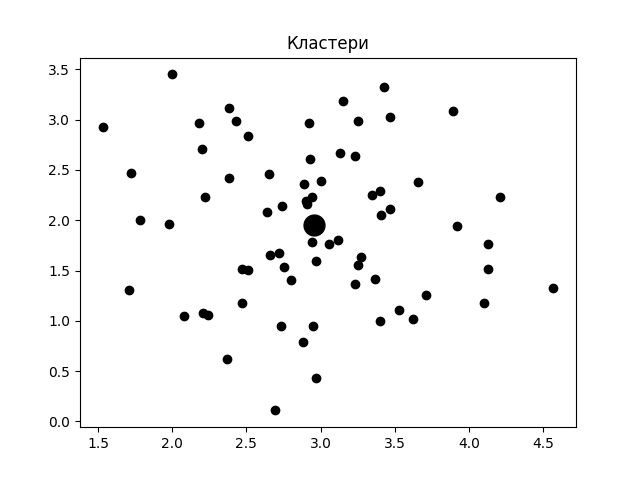
 [5.97960784 8.39078431]

 [4.99466667 4.65844444]]

 Number of clusters in input data =  5

[Done] exited with code=0 in 51.719 seconds

Графіки:



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

Файл **company\_symbol\_mapping.json** було завантажено з відкритого репо на гіті тому що його чомусь не було в додатках до лаби на моменті її виконання, дані з файлу було перенесено в код.

Також крім цього бібліотеки from matplotlib.finance import quotes\_historical\_yahoo\_ochl as quotes\_yahoo більше не існує, і її прямі аналогі на разі також не дієздані, тому було обрано трохи інший підхід, можливо він не побачив всіх даних, але це ледь не єдиний метод який вдалось знайти.

Лістинг:

import datetime

import numpy as np

import matplotlib.pyplot as plt

import yfinance as yf

from sklearn import covariance, cluster

company\_symbols\_map = {

    "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"

}

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 = []

valid\_symbols = []

for symbol in symbols:

    try:

        stock\_data = yf.Ticker(symbol).history(*start*=start\_date, *end*=end\_date)

        if stock\_data.empty:

            print(*f*"{symbol}: No data found")

        else:

            quotes.append(stock\_data)

            valid\_symbols.append(symbol)

    except Exception as e:

        print(*f*"{symbol}: Error fetching data - {e}")

min\_length = min(len(quote) for quote in quotes)

quotes = [quote.iloc[:min\_length] for quote in quotes]

opening\_quotes = np.array([quote['Open'].values for quote in quotes], *dtype*=np.float64)

closing\_quotes = np.array([quote['Close'].values for quote in quotes], *dtype*=np.float64)

quotes\_diff = closing\_quotes - opening\_quotes

X = quotes\_diff.copy().T

X /= X.std(*axis*=0)

edge\_model = covariance.GraphicalLassoCV()

with np.errstate(*invalid*='ignore'):

    edge\_model.fit(X)

\_, labels = cluster.affinity\_propagation(edge\_model.covariance\_)

num\_labels = labels.max()

print('\nClustering of stocks based on difference in opening and closing quotes:\n')

for i in range(num\_labels + 1):

    cluster\_symbols = np.array(valid\_symbols)[labels == i]

    cluster\_names = np.array(names)[np.isin(symbols, cluster\_symbols)]

    print("Cluster", i+1, "==>", ', '.join(cluster\_names))

Виконання:

[Running] python -u "d:\ztu\KURS4\ai\Lab7\LR\_4\_task\_4.py"

$TOT: possibly delisted; no timezone found

TOT: No data found

$CVC: possibly delisted; no price data found  (1d 2003-07-03 00:00:00 -> 2007-05-04 00:00:00)

CVC: No data found

$YHOO: possibly delisted; no timezone found

YHOO: No data found

$DELL: possibly delisted; no price data found  (1d 2003-07-03 00:00:00 -> 2007-05-04 00:00:00) (Yahoo error = "Data doesn't exist for startDate = 1057204800, endDate = 1178251200")

DELL: No data found

$CAJ: possibly delisted; no timezone found

CAJ: No data found

$MTU: possibly delisted; no price data found  (1d 2003-07-03 00:00:00 -> 2007-05-04 00:00:00)

MTU: No data found

$SNE: possibly delisted; no timezone found

SNE: No data found

$NAV: possibly delisted; no timezone found

NAV: No data found

$UN: possibly delisted; no timezone found

UN: No data found

$RTN: possibly delisted; no timezone found

RTN: No data found

Clustering of stocks based on difference in opening and closing quotes:

Cluster 1 ==> Exxon, Chevron, ConocoPhillips, Valero Energy

Cluster 2 ==> Toyota, Ford, Honda, Boeing, Mc Donalds, Apple, SAP, Caterpillar

Cluster 3 ==> Kraft Foods

Cluster 4 ==> Coca Cola, Pepsi, Kellogg, Procter Gamble, Colgate-Palmolive, Kimberly-Clark

Cluster 5 ==> Time Warner, Comcast, Marriott, Wells Fargo, JPMorgan Chase, AIG, American express, Bank of America, Goldman Sachs, Xerox, Wal-Mart, Home Depot, Ryder, DuPont de Nemours

Cluster 6 ==> Microsoft, IBM, HP, Amazon, 3M, General Electrics, Cisco, Texas instruments

Cluster 7 ==> Northrop Grumman, Lookheed Martin, General Dynamics

Cluster 8 ==> Walgreen, CVS

Cluster 9 ==> GlaxoSmithKline, Pfizer, Sanofi-Aventis, Novartis

[Done] exited with code=0 in 19.161 seconds

GIT: <https://github.com/Alhim616/AI_Labs_Yanushevych>