Лабораторна робота 7

ДОСЛІДЖЕННЯ МЕТОДІВ НЕКОНТРОЬОВАНОГО НАВЧАННЯ

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

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

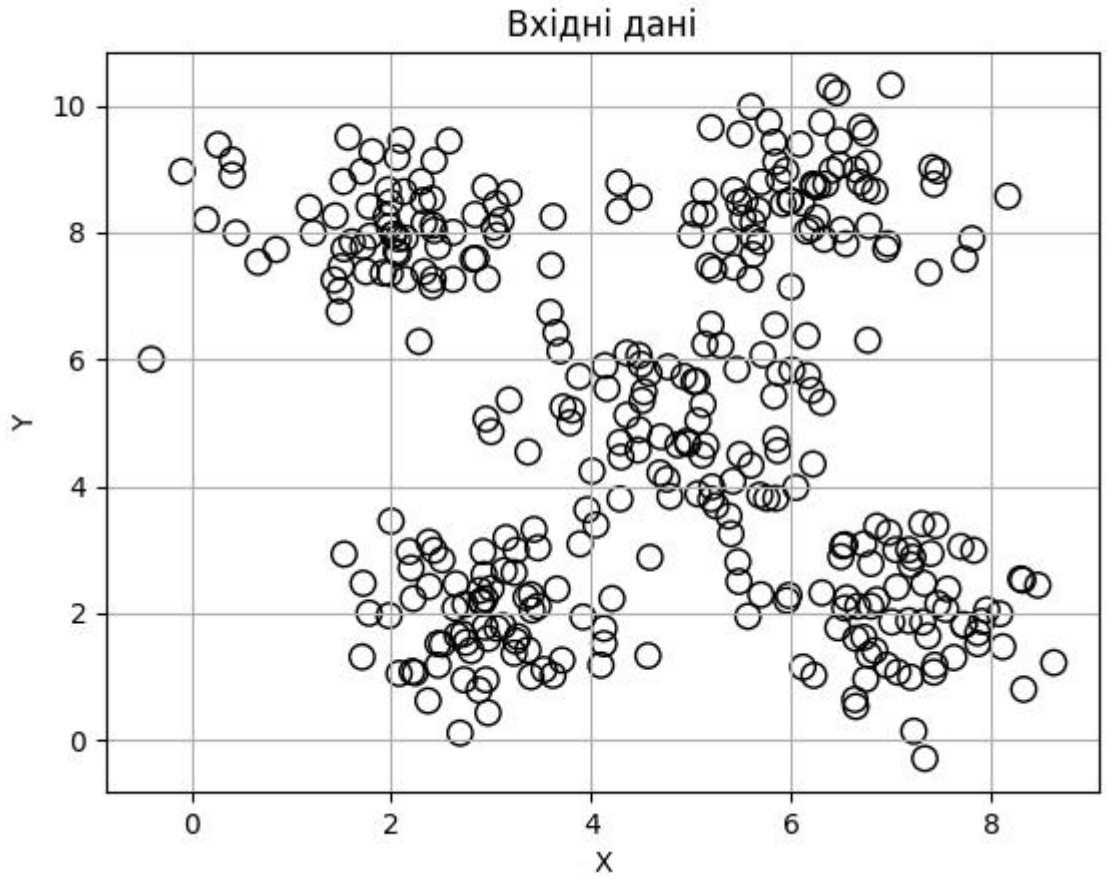

Рис. 1 Початкові дані для кластеризації

					ДУ «Житомирська політехніка».25.121.14.003 – Лр7		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Кольцова Н.О.			Звіт з лабораторної роботи	Літ.	Арк. 1 / Аркушів 11
Перевір.		Маєвський О.В.					
Керівник						ФІКТ Гр. ІПЗ-22-4[1]	
Н. контр.							
Зав. каф.							

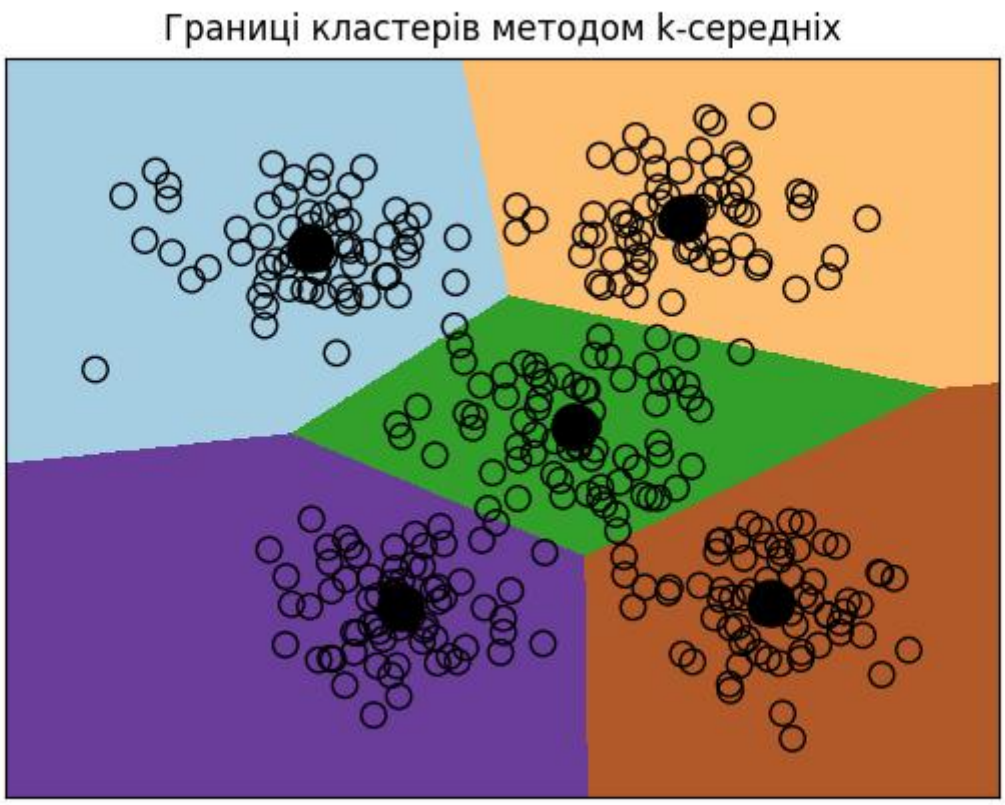

Рис. 2 Результат кластеризації методом k-means

Висновок: Виконана кластеризація методом k-середніх дозволила розбити вхідні дані на п’ять груп, що чітко простежується на графіку, де кожна область виділена окремим кольором, а центри кластерів позначені чорними точками. Вхідні дані, відображені окремо, показують початковий розподіл точок, що виглядає досить розсіяним, проте алгоритм зміг знайти певні групи, до яких точки найбільш схожі між собою. Значення сумарної квадратичної помилки (Inertia) становить 433.8, що відображає середнє відхилення точок від центрів своїх кластерів, а Silhouette score рівний 0.59 свідчить про помірну щільність і відокремленість кластерів. Отримані результати демонструють, що k-means адекватно поділив дані на п’ять підгруп, однак існує певна кількість точок, які знаходяться ближче до меж різних кластерів, що впливає на середню якість кластеризації. Загалом, метод успішно виявив приховану структуру даних і наочно відобразив її на графіку.

Завдання 2.2. Кластеризація K-середніх для набору даних Iris

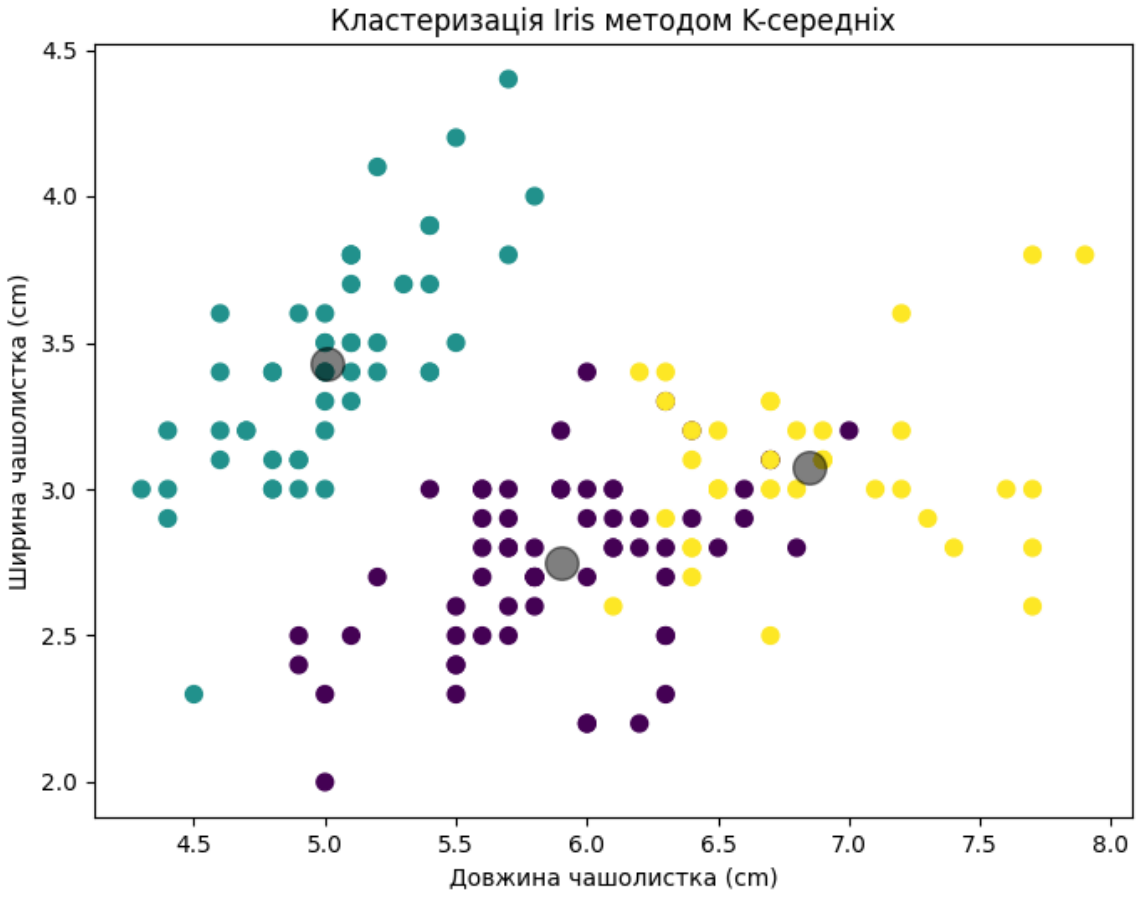

Рис. 3 Кластеризація Sris методом K-середніх

Висновок: Проведена кластеризація набору даних Iris методом K-середніх дозволила розбити дані на три кластери, що відповідає фактичній кількості класів квітів у наборі (Setosa, Versicolour та Virginica). На графіку видно, що точки даних розподілені по різних областях кольорів, а центри кластерів позначені чорними великими точками. Найбільш компактно виділяється кластер, що відповідає Setosa, тоді як Versicolour та Virginica частково перекриваються, що свідчить про близькість їх ознак і деякі труднощі алгоритму у чіткому розділенні цих двох класів. Координати центрів кластерів відображають середні значення ознак для кожного кластера і підтверджують, що метод адекватно визначив середні характеристики

груп. Кластерні мітки для перших десяти зразків показують, що всі вони були віднесені до одного кластеру, що відповідає класу Setosa. Загалом, K-середніх ефективно виявив приховану структуру даних, дозволивши наочно оцінити поділ квітів за ознаками чашолистка та пелюстки, хоча для класів з близькими значеннями ознак можливі часткові накладення кластерів.

Завдання 2.3. Оцінка кількості кластерів з використанням методу зсуву середнього

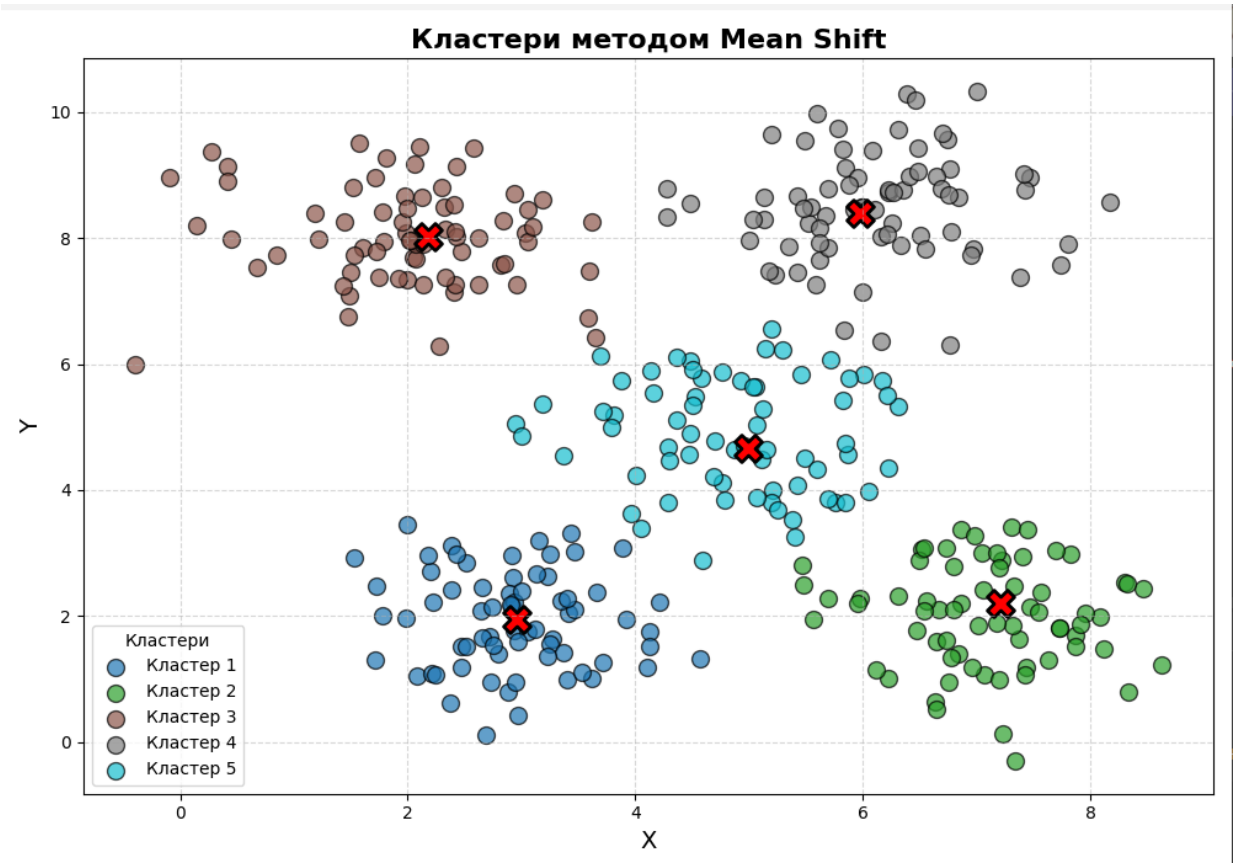

Рис. 4 Кластери даних методом Mean Shift з автоматичним визначенням кількості кластерів

```
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. 5 Вивід (результати кластеризації):

Висновки: На основі виконаної кластеризації методом Mean Shift можна зробити висновок, що набір даних складається з п’яти чітко виражених груп. Алгоритм автоматично визначив кількість кластерів, не потребуючи попереднього завдання K, і розташував центри кожного кластера в області максимальної концентрації точок. Графік демонструє, що дані добре сегментовані: різнокольорові точки чітко розділяють кластери, а великі червоні хрестики позначають центри кластерів, що знаходяться в середині кожної групи. Координати центрів і кількість кластерів підтверджують наочну оцінку з графіка. Таким чином, метод Mean Shift ефективно виділяє приховані структури даних, дозволяючи виявити природні групи без попереднього знання їхньої кількості.

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

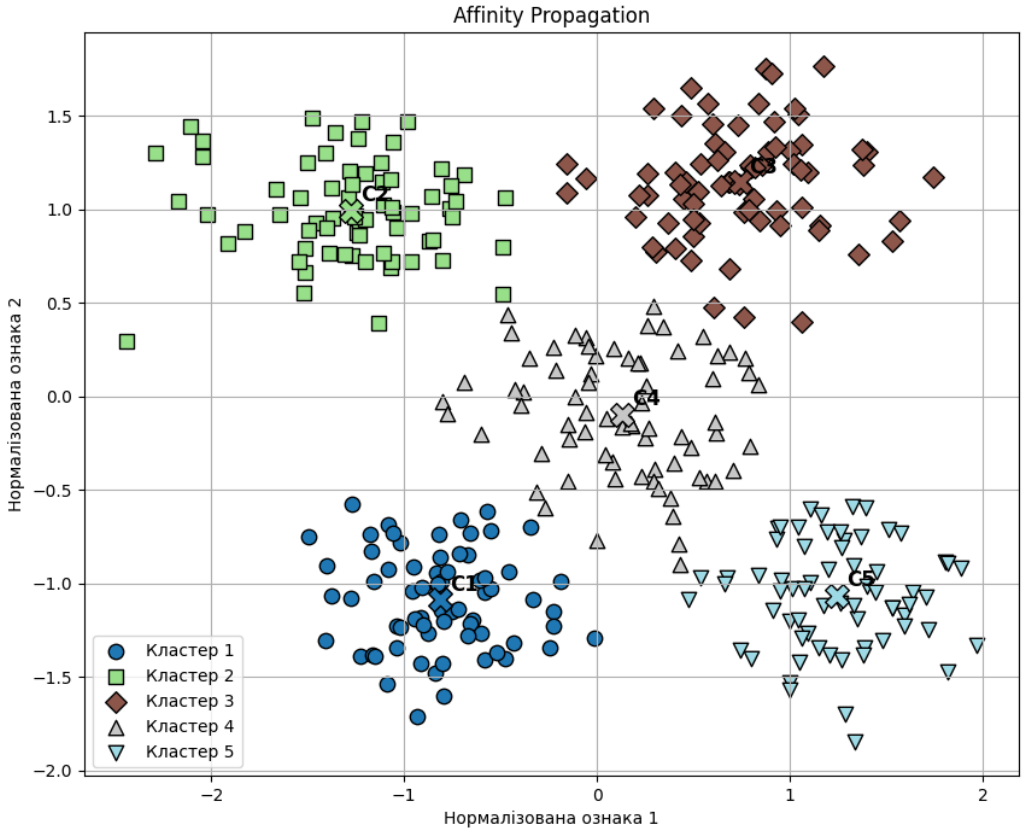

Рис. 6 Кластеризація учасників фондового ринку методом поширення подібності

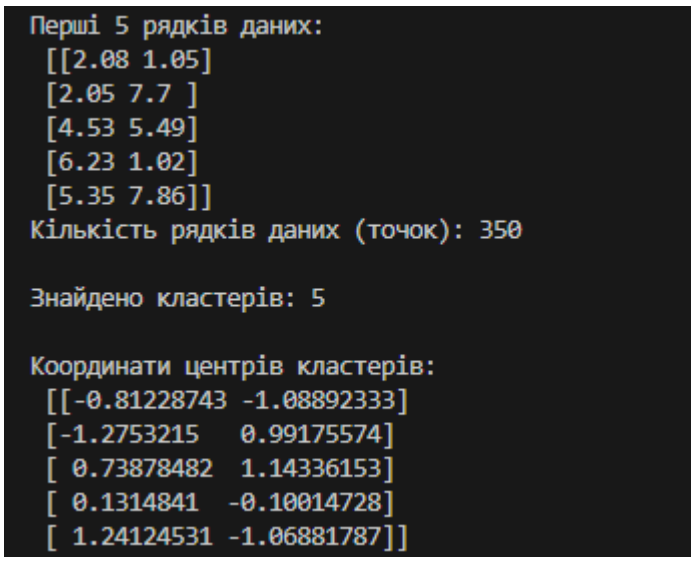

Рис. 7 Перші 5 рядків даних та характеристика кластерів

Висновки: Аналізуючи результати кластеризації методом поширення подібності для даних фондового ринку, можна зробити висновок, що набір з 350 точок поділяється на п’ять чітко виражених підгруп. Кластеризація дозволяє виявити схожі патерни коливань між відкриттям та закриттям біржі, адже кожен кластер характеризується певними середніми координатами, що відображають середню поведінку учасників у межах групи. Графічне відображення демонструє, що точки всередині кластерів щільно розташовані навколо своїх центрів, що свідчить про високий рівень подібності між елементами кластера. Такий аналіз допомагає структурувати великий масив даних фондового ринку, виділяючи основні підгрупи учасників, що можуть мати спільні ринкові стратегії або схожі реакції на зміни ринку. Отримані кластери наочно демонструють, що модель поширення подібності ефективно розділяє ринкових учасників на осмислені групи, що спрощує подальший аналіз та прийняття стратегічних рішень.

Висновки: в ході лабораторної роботи було набуто навичи працювання з даними і опонувати роботу у Python з використанням теореми Байэса.

Посилання на github: https://github.com/KoltcovaNadiia/Artificial-intelligence-systems-2025

Лістинг програми:

LR_7_task_1.py

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

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

#  Візуалізація вхідних даних
plt.figure()
plt.scatter(X[:, 0], X[:, 1], edgecolors='black', facecolors='none', s=80)
plt.title('Вхідні дані')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True)
plt.show()

#  Налаштування моделі KMeans
num_clusters = 5
kmeans = KMeans(init='k-means++', n_clusters=num_clusters, n_init=10)

#  Навчання моделі
kmeans.fit(X)


# Побудова сітки для візуалізації кордонів кластерів
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))

#  Прогноз кластерів на всій сітці
output = kmeans.predict(np.c_[x_vals.ravel(), y_vals.ravel()])
output = output.reshape(x_vals.shape)

#  Відображення областей кластерів, даних та центрів
plt.figure()
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')
```

```
# Вхідні точки
plt.scatter(X[:, 0], X[:, 1], edgecolors='black',
            facecolors='none', s=80)

# Центри кластерів
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], marker='o',
            s=210, linewidths=3, color='black', zorder=12)

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

#  Оцінка якості кластеризації
inertia = kmeans.inertia_
silhouette = metrics.silhouette_score(X, kmeans.labels_)

print("Сумарна квадратична помилка (Inertia):", inertia)
print("Silhouette score:", silhouette)
```

LR_7_task_2.py

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

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

kmeans = KMeans(n_clusters=3, init='k-means++', n_init=10, max_iter=300,
random_state=0)
kmeans.fit(X)
y_kmeans = kmeans.predict(X)

plt.figure(figsize=(8, 6))
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.xlabel('Довжина чашолистка (cm)')
plt.ylabel('Ширина чашолистка (cm)')
plt.title('Кластеризація Iris методом K-середніх')
plt.show()

print("Координати центрів кластерів:\n", kmeans.cluster_centers_)
print("\nКластерні мітки для перших 10 зразків:\n", y_kmeans[:10])
```

LR_7_task_3.py

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

# Завантаження вхідних даних
try:
    X = np.loadtxt('data_clustering.txt', delimiter=',')  # Завантаження даних з
файлу
except OSError:
    print("Файл 'data_clustering.txt' не знайдено. Будь ласка, перевірте шлях до
файлу або згенеруйте дані.")
    X = np.zeros((10, 2))  # Пустий масив для запобігання помилок

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

# Навчання моделі кластеризації методом зсуву середнього
meanshift_model = MeanShift(bandwidth=bandwidth_X, bin_seeding=True)
meanshift_model.fit(X)

# Витяг центру кластерів
cluster_centers = meanshift_model.cluster_centers_
print('\nCenters of clusters:\n', cluster_centers)

# Оцінка кількості кластерів
labels = meanshift_model.labels_
num_clusters = len(np.unique(labels))
print("\nNumber of clusters in input data =", num_clusters)
plt.figure(figsize=(10, 7))

# Використовуємо сучасний метод для отримання кольорової палітри
colors = matplotlib.colormaps['tab10'].resampled(num_clusters)

for i in range(num_clusters):
    # Відображення точок поточного кластера кольором та з прозорістю
    plt.scatter(X[labels == i, 0], X[labels == i, 1],
                marker='o',
                color=colors(i),
                s=100,           # розмір точок
                alpha=0.7,       # прозорість
                label=f'Кластер {i+1}',
                edgecolor='k')   # чорна обводка для кращої видимості

    # Відображення центру поточного кластера
    cluster_center = cluster_centers[i]
    plt.scatter(cluster_center[0], cluster_center[1],
```

```
                marker='X',
                color='red',
                s=250,          # великий розмір
                edgecolor='black',
                linewidth=2)

# Налаштування оформлення графіка
plt.title('Кластери методом Mean Shift', fontsize=16, fontweight='bold')
plt.xlabel('X', fontsize=14)
plt.ylabel('Y', fontsize=14)
plt.grid(True, linestyle='--', alpha=0.5)
plt.legend(title='Кластери')
plt.tight_layout()
plt.show()
```

LR_7_task_4.py

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import AffinityPropagation
from sklearn.preprocessing import StandardScaler

# === Завантаження даних ===
data = np.loadtxt('data_clustering.txt', delimiter=',')
print("Перші 5 рядків даних:\n", data[:5])
print("Кількість рядків даних (точок):", data.shape[0])

# === Нормалізація даних ===
scaler = StandardScaler()
X_scaled = scaler.fit_transform(data)

# === Кластеризація методом AffinityPropagation ===
# Параметр preference можна підібрати для отримання 5 кластерів
ap = AffinityPropagation(damping=0.9, preference=-10, random_state=42)
ap.fit(X_scaled)
labels = ap.labels_
num_labels = len(np.unique(labels))
print("\nЗнайдено кластерів:", num_labels)

# === Центри кластерів ===
cluster_centers = np.array([X_scaled[labels == i].mean(axis=0) for i in
range(num_labels)])
print("\nКоординати центрів кластерів:\n", cluster_centers)

# === Візуалізація кластерів ===
plt.figure(figsize=(10, 8))
colors = plt.cm.tab20(np.linspace(0, 1, num_labels))
markers = ['o', 's', 'D', '^', 'v', 'P', '*', 'X', 'h', '8']
```

```
for k, col in zip(range(num_labels), colors):
    class_members = labels == k
    plt.scatter(X_scaled[class_members, 0], X_scaled[class_members, 1],
                marker=markers[k % len(markers)], s=80, c=[col], edgecolor='k',
label=f'Кластер {k+1}')
    plt.scatter(cluster_centers[k, 0], cluster_centers[k, 1],
                marker='X', s=200, c=[col], edgecolor='k')
    plt.text(cluster_centers[k, 0]+0.05, cluster_centers[k, 1]+0.05, f'C{k+1}',
fontsize=12, fontweight='bold')

plt.title("Affinity Propagation")
plt.xlabel("Нормалізована ознака 1")
plt.ylabel("Нормалізована ознака 2")
plt.grid(True)
plt.legend(loc='best', fontsize=10)
plt.show()
```