Лабораторна робота №7

ДОСЛІДЖЕННЯ МЕТОДІВ НЕКОНТРОЬОВАНОГО НАВЧАННЯ

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

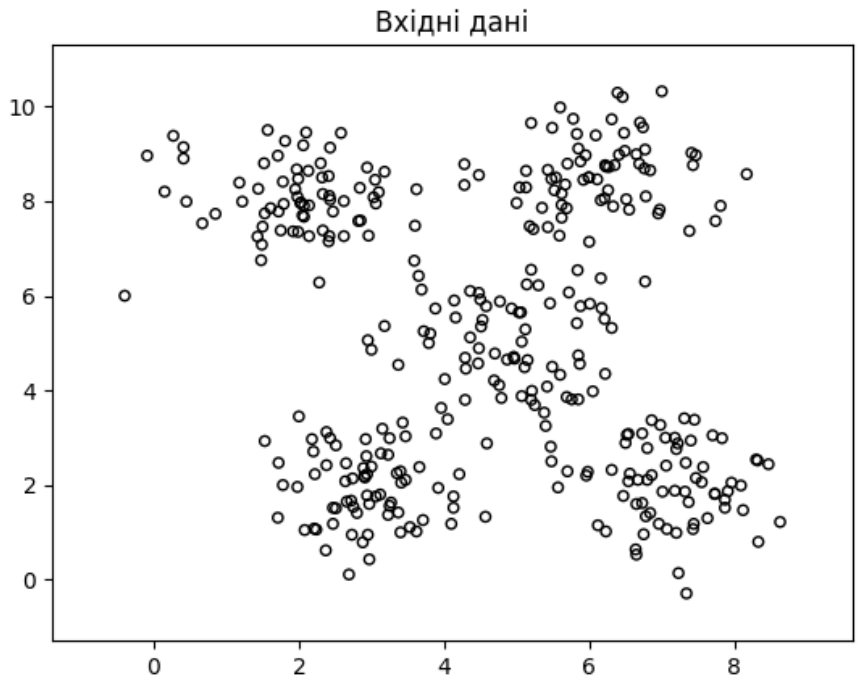
Git: https://github.com/PanchukPetro/SShILabsPanchuk/tree/main/Lab7

Завдання 2.1. Кластеризація даних за допомогою методу k-середніх Провести кластеризацію даних методом k-середніх. Використовувати файл вхідних даних: data_clustering.txt.

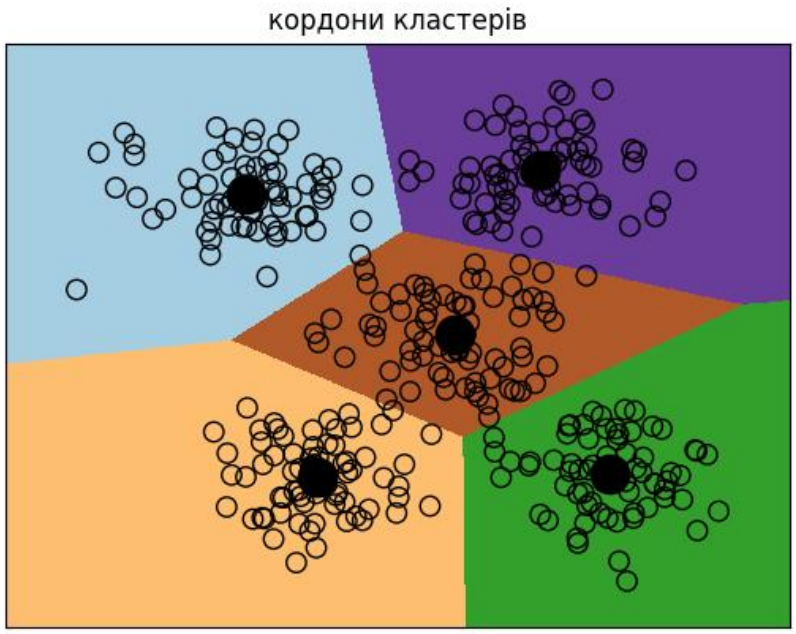
Код:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn import  metrics

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

num_clusters = 5

# Візуалізація вхідних даних
plt.scatter(X[:, 0], X[:, 1], marker='.', 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()

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

# Навчання моделі кластеризації KMeans
kmeans.fit(X)

# Щоб візуалізувати межі, ми маємо створити сітку точок та обчислити модель на
всіх вузлах сітки. Визначимо крок сітки.
step_size = 0.01

# Визначення меж сітки
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))

# Спрогнозуйте результати всіх точок сітки, використовуючи навчену модель k-
середніх
output = kmeans.predict(np.c_[x_vals.ravel(), y_vals.ravel()])

# Графічне відображення областей та виділення їх кольором
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')

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

# Відображення центрів кластерів
cluster_centers = kmeans.cluster_centers_
plt.scatter(cluster_centers[:,0], cluster_centers[:,1], marker = "o", s=210,
linewidth=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("кордони кластерів")
plt.xlim(x_min,x_max)
plt.ylim(y_min,y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

Результати роботи коду:



Графік 1



Графік 2

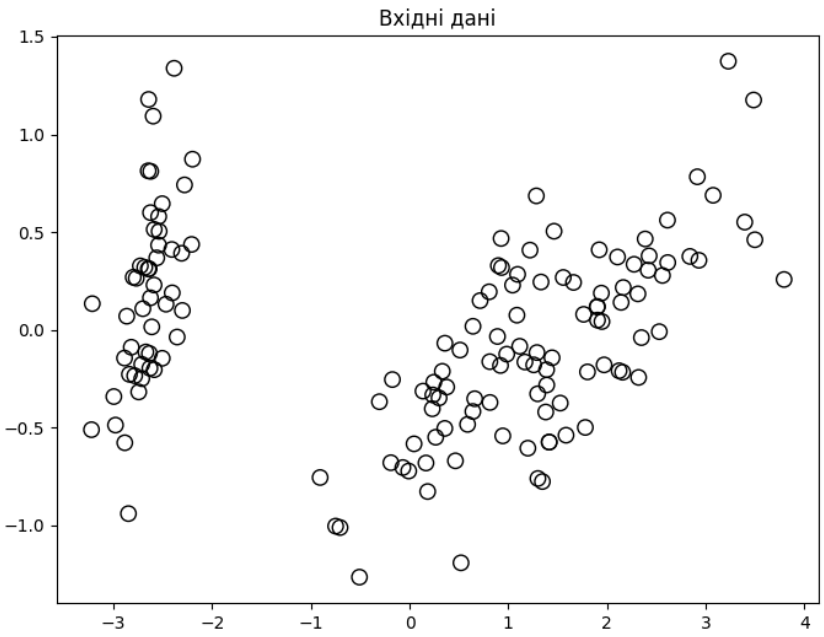
Завдання 2.2. Кластеризація K-середніх для набору даних Iris Виконайте кластеризацію K-середніх для набору даних Iris, який включає три типи (класи) квітів ірису (Setosa, Versicolour і Virginica) з чотирма атрибутами: довжина чашолистка, ширина чашолистка, довжина пелюстки та ширина пелюстки. У цьому завданні використовуйте sklearn.cluster.KMeans для пошуку кластерів набору даних Iris.

Код:

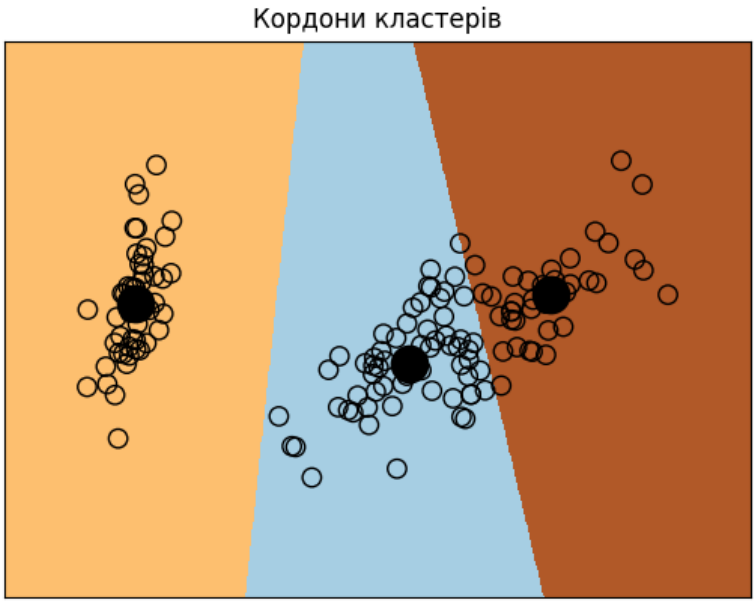
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

# Завантаження даних Iris
iris = load_iris()
X = iris.data

target = iris.target  # Істинні мітки класів для відображення
num_clusters = 3

kmeans = KMeans(init='k-means++', n_clusters=num_clusters, n_init=10,
random_state=42)
kmeans.fit(X)

# Візуалізація вхідних даних (зменшення до 2D для графічного відображення)
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X)
cluster_centers_reduced = pca.transform(kmeans.cluster_centers_)
```

```
# Візуалізація вхідних даних за справжніми класами
plt.figure(figsize=(8, 6))
plt.scatter(X_reduced[:, 0], X_reduced[:, 1], marker='o', facecolors = 'none',
edgecolors='black', s=80)
plt.title("Вхідні дані")
plt.show()

step_size = 0.01

x_min, x_max = X_reduced[:, 0].min() - 1, X_reduced[:, 0].max() + 1
y_min, y_max = X_reduced[:, 1].min() - 1, X_reduced[:, 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(pca.inverse_transform(np.c_[x_vals.ravel(),
y_vals.ravel()]))

# Графічне відображення областей та виділення їх кольором
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')

# Відображення вхідних точок
plt.scatter(X_reduced[:, 0], X_reduced[:, 1], marker='o', facecolors='none',
            edgecolors='black', s=80)

# Відображення центрів кластерів
plt.scatter(cluster_centers_reduced[:, 0], cluster_centers_reduced[:, 1],
marker="o", s=210, linewidth=4, color='black', zorder=12, facecolors='black')

plt.title("Кордони кластерів")
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

Результати виконання коду:



Графік 1



Графік 2

Завдання 2.3. Оцінка кількості кластерів з використанням методу зсуву середнього

Відповідно до рекомендацій, напишіть програму та оцініть максимальну кількість кластерів у заданому наборі даних за допомогою алгоритму зсуву середньою. Для аналізу використовуйте дані, які містяться у файлі data_clustering.txt.

Код:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import bandwidth
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("centers of clusters: \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()
```

Результат виконання коду: