Міністерство освіти і науки України

Національний технічний університет "ХПІ"

кафедра "Інформатика та інтелектуальна власність"

**ЗВІТ**

до лабораторної роботи № 8

Тема : «**Навчання без вчителя**»

Варіант номер 18

з дисципліни "Основи штучного інтелекту"

Виконав:

студент групи КН-321В

Хома Д.М.

Перевірив:

Паржин Ю. В.

Харків 2023

***Завдання 1***: Проаналізуйте, як розподіляються види ірисів при інших комбінаціях параметрів (ознак).

Для цього завдання написано такий код на рис.1.1.

import seaborn as sns  
from sklearn import datasets  
import pandas as pd  
  
  
# Загружаємо дані ірисів  
iris\_data = datasets.load\_iris()  
iris\_df = pd.DataFrame(data=iris\_data.data, columns=iris\_data.feature\_names)  
iris\_df['species'] = iris\_data.target\_names[iris\_data.target]  
  
# Застосовано власні маркери  
sns.pairplot(iris\_df, hue="species", palette="husl", markers=["o", "o", "o"])  
  
# Видаляємо верхні та праві краї графіку  
sns.despine()  
  
# Показуємо графік  
import matplotlib.pyplot as plt  
plt.show()

Рисунок 1.1 – Код програми

Результати програми, а саме класифікацію ірисів за різними ознаками я помістив на один малюнок, а також додав гістограми до кожної ознаки на рис. 1.2.

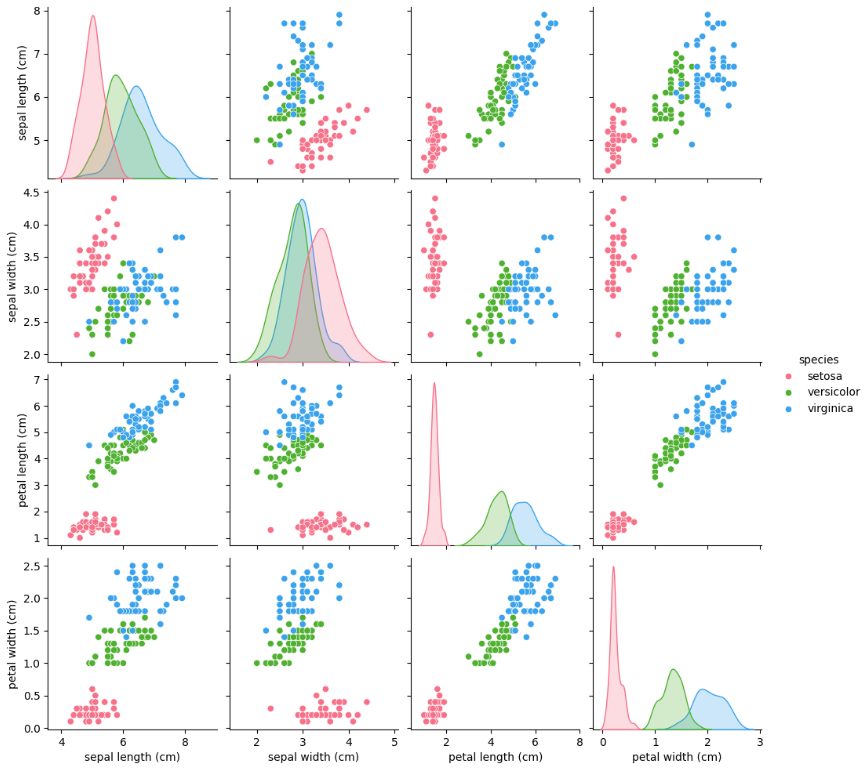


Рисунок 1.2 – Результати класифікації за різними ознаками

З графіків видно, що «petal width (cm)» та «petal length (cm)» мають високу взаємозалежність - точки розташовані вздовж однієї лінії. За цими ознаками можна проводити ефективну класифікацію, оскільки точки за кольором компактно згруповані.

Якщо вибрати класифікацію за ознаками «sepal width (cm)» и «sepal length (cm)» якісну класифікацію побудувати не вдасться, оскільки точки сортів Versicolor та Virginica дуже сильно змішані.

***Завдання 2:*** Візуалізуйте отримані результати кластеризації.

Було написано код на рис.2.1. для візуалізації

import matplotlib.pyplot as plt  
import seaborn as sns  
from sklearn.decomposition import PCA  
from sklearn import datasets  
from sklearn.cluster import KMeans  
  
# Завантажуємо набір даних  
iris\_df = datasets.load\_iris()  
  
# Описуємо модель  
model = KMeans(n\_clusters=3)  
  
# Проводимо моделювання  
model.fit(iris\_df.data)  
  
# Передбачення на одиничному прикладі  
predicted\_label = model.predict([[7.2, 3.5, 0.8, 1.6]])  
  
# Передбачення на всьому наборі даних  
all\_predictions = model.predict(iris\_df.data)  
  
# Виводимо результати передбачення  
print(predicted\_label)  
print(all\_predictions)  
  
# Використовуємо PCA для зменшення розмірності даних до 2D  
pca = PCA(n\_components=2)  
iris\_2d = pca.fit\_transform(iris\_df.data)  
  
# Додаємо прогнозовані мітки до даних  
iris\_df['predicted\_label'] = all\_predictions  
  
# Створюємо scatter plot для візуалізації  
plt.figure(figsize=(10, 6))  
sns.scatterplot(x=iris\_2d[:, 0], y=iris\_2d[:, 1], hue=iris\_df['predicted\_label'], palette='Set1', s=100)  
  
# Виводимо центроїди  
centers\_2d = pca.transform(model.cluster\_centers\_)  
plt.scatter(centers\_2d[:, 0], centers\_2d[:, 1], marker='X', s=200, c='red', label='Centroids')  
  
plt.title('KMeans Clustering of Iris Dataset')  
plt.xlabel('Principal Component 1')  
plt.ylabel('Principal Component 2')  
plt.legend(title='Cluster')  
plt.show()

Рисунок 2.1 – Код програми

Результати програми наведено на рис. 2.2.

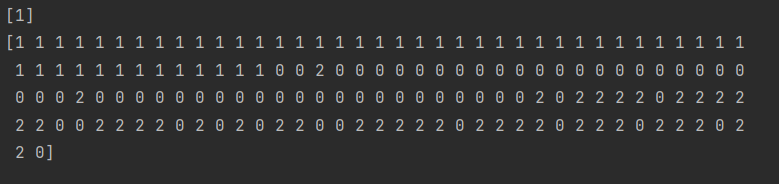


Рисунок 2.2 – Результати виведення програми

Результати візуалізації класифікованих даних наведено на рис. 2.3.

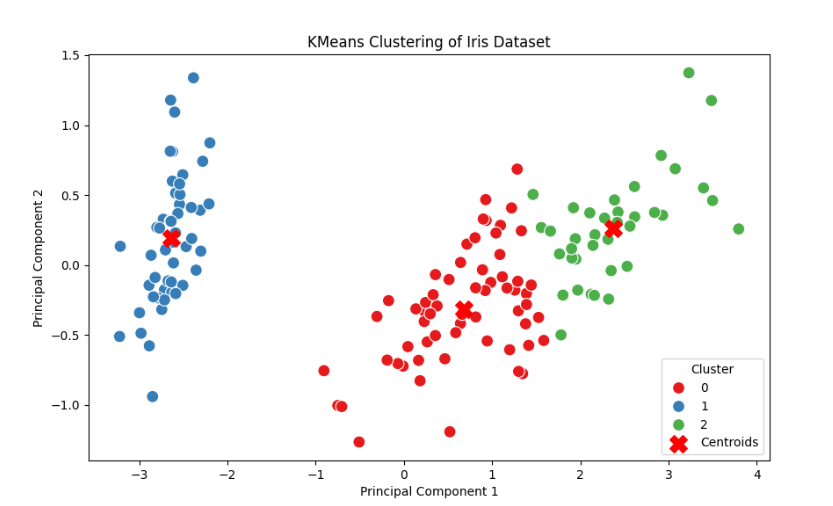


Рисунок 2.3 – Результати візуалізації

Наибільш популярним алгоритмом кластеризації даних є k-means. Це ітеративний алгоритм кластеризації, що ґрунтується на мінімізації сумарних квадратичних відхилень точок кластерів від центроїдів (середніх координат) цих кластерів.

Центроїди, які є на графіку, є центральними точками кожного кластера, і вони є ключовим елементом алгоритму k-means. Основна ідея методу k-means полягає в тому, що він групує дані в кластери, так що сума квадратів відстаней між точками даних і центроїдами кластерів є мінімальною. Центроїди представляють середні значення атрибутів всіх точок у кластері.

***Завдання 3:*** Здійсніть ієрархічну кластеризацію для датасету "Іриси Фішера". Побудуйте дендрограму. Порівняйте результати кластеризації двох методів для одного набору даних. Зробіть висновки: який метод краще підходить для великих наборів даних з точки зору часових та обчислювальних витрат і чому; який метод точніший; який метод більш чутливий до "зашумлення" даних.

Було створено код для побудови дендрограми на рис. 3.1.

from scipy.cluster.hierarchy import linkage, dendrogram  
import matplotlib.pyplot as plt  
import pandas as pd  
from sklearn.datasets import load\_iris  
  
# Завантажуємо дані ірисів  
iris = load\_iris()  
iris\_df = pd.DataFrame(data=iris.data, columns=iris.feature\_names)  
  
# Додаємо інформацію про вид ірису  
iris\_df['iris\_variety'] = iris.target  
  
# Вилучаємо виміри як масив NumPy  
iris\_samples = iris\_df.iloc[:, :-1].values  
  
# Реалізація ієрархічної кластеризації за допомогою функції linkage  
mergings = linkage(iris\_samples, method='complete')  
  
# Будуємо дендрограму, вказавши параметри, що зручні для відображення  
dendrogram(mergings,  
 labels=iris.target\_names[iris.target],  
 leaf\_rotation=90,  
 leaf\_font\_size=6,  
 color\_threshold=4 )  
  
plt.show()

Рисунок 3.1 – Код програми

Створена дендрограма на рис. 3.2.

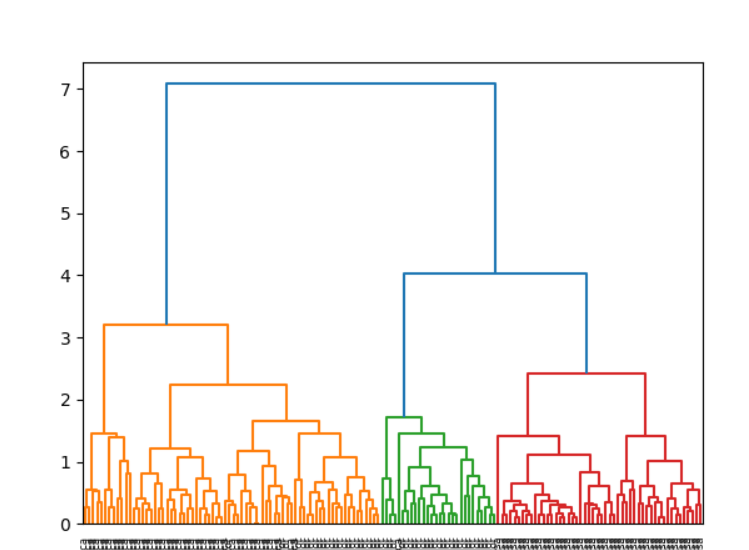


Рисунок 3.2 – Дендрограма

Метод k-means краще підходить підходить для великих наборів даних.

Тому, що часова складність алгоритму є лінійною для методу k-means і квадратичною для методу ієрархічної кластеризації.

У методі k-means алгоритм починає побудову з випадкового вибору початкових точок, тому результати можуть варіюватися при кількох повторних запусках алгоритму. З іншого боку, у випадку ієрархічної кластеризації результати відтворюються.

Оскільки метод k-means базується на центроїдальній геометрії, він ефективно працює, коли форма кластерів є гіперсферичною. З іншого боку, метод k-means є більш чутливим до шумових даних порівняно з ієрархічним методом.

***Завдання 4:*** Представте результат у тривимірному просторі. Порівняйте результати кластеризації датасету "Іриси Фішера" методами k-means та t-SNE, зробіть висновки.

До цього завдання створено код на рис. 4.1.

import matplotlib.pyplot as plt  
from sklearn import datasets  
from sklearn.manifold import TSNE  
  
  
# Завантаження даних ірисів Фішера  
iris = datasets.load\_iris()  
X = iris.data  
y = iris.target  
  
# Використання t-SNE для зменшення розмірності до 3D  
tsne = TSNE(n\_components=3, random\_state=42)  
X\_tsne = tsne.fit\_transform(X)  
  
# Створення 3D графіку  
fig = plt.figure(figsize=(8, 6))  
ax = fig.add\_subplot(111, projection='3d')  
  
# Розфарбовуємо точки відповідно до класів  
scatter = ax.scatter(X\_tsne[:, 0], X\_tsne[:, 1], X\_tsne[:, 2], c=y, cmap=plt.cm.get\_cmap("viridis"), marker='o')  
  
# Додаємо легенду  
legend = ax.legend(\*scatter.legend\_elements(), title="Classes")  
ax.add\_artist(legend)  
  
# Додаємо мітки до вісей  
ax.set\_xlabel('x')  
ax.set\_ylabel('y')  
ax.set\_zlabel('z')  
ax.set\_title('3D Visualization')  
  
# Показуємо графік  
plt.show()

Рисунок 4.1 – Код програми

Результат у тривимірному просторі на рис. 4.2.

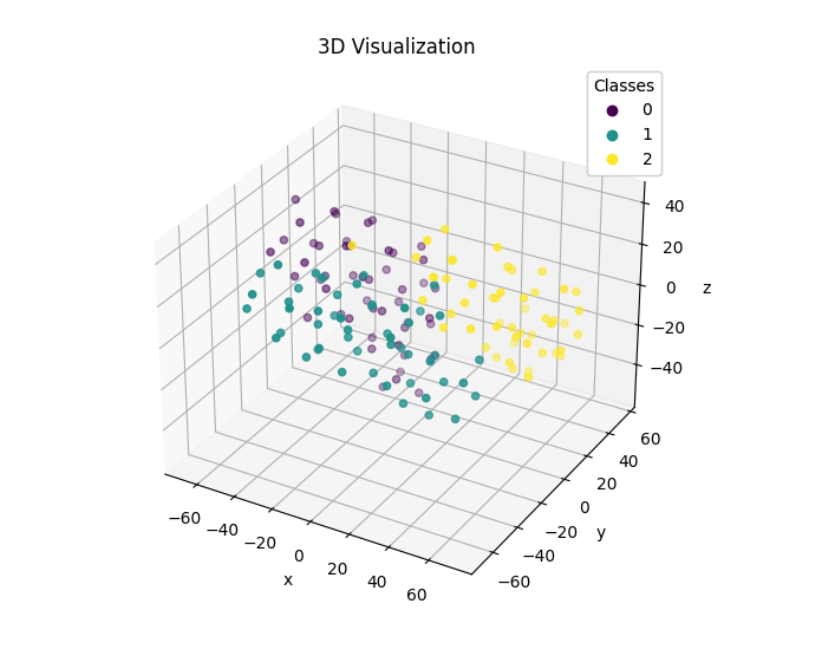


Рисунок 4.2 - Результат у тривимірному просторі

t-SNE використовується для зменшення розмірності та візуалізації даних, але не завжди гарантує однозначне визначення кількості кластерів. Цей метод слід використовувати переважно для візуалізації та розуміння взаємин між точками в просторі, а не для явного визначення кластерів.

В методі k-means кластери формуються за допомогою центрів, інтерпретація кластерів може бути дещо простішою. Результат класифікації може варіюватися в залежності від початкового вибору центрів.

t-SNE в свою чергу чутливий до параметрів та початкового розташування, але зазвичай застосовується для зменшення розмірності та візуалізації збереження відносин між точками.

І на кінець складність, k-means зазвичай швидше виконується, особливо на великих наборах даних. t-SNE може бути витратним з точки зору обчислень, особливо для великих наборів даних.

Висновок:

Метод k-середніх і t-SNE мають різні призначення та застосування.

***Завдання 5:*** Вивчіть докладніше цей метод і метод головних компонентів PCA змініть параметри **eps** та **min\_samples,** досягніть більш точної кластеризації розглянутого датасета. Зробіть висновки про роботу цього методу у порівнянні з методами, розглянутими раніше.

Створено код, його було оптимізовано та скорочено на рис. 5.1. Обрано оптимальні параметри *m* = 5, . Коли правильно відкидаються виброси.

from sklearn.datasets import load\_iris  
from sklearn.preprocessing import StandardScaler  
from sklearn.cluster import DBSCAN  
import matplotlib.pyplot as plt  
  
  
iris = load\_iris()  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(iris.data)  
  
dbscan = DBSCAN(eps=1.2, min\_samples=5)  
dbscan.fit(X\_scaled)  
labels = dbscan.labels\_  
  
plt.scatter(X\_scaled[:,0], X\_scaled[:,1], c=labels)  
plt.xlabel('Sepal Length')  
plt.ylabel('Sepal Width')  
plt.title('DBSCAN clusters')  
plt.show()

Рисунок 5.1 – Код програми

Результати класифікації наведено на рис.5.2.

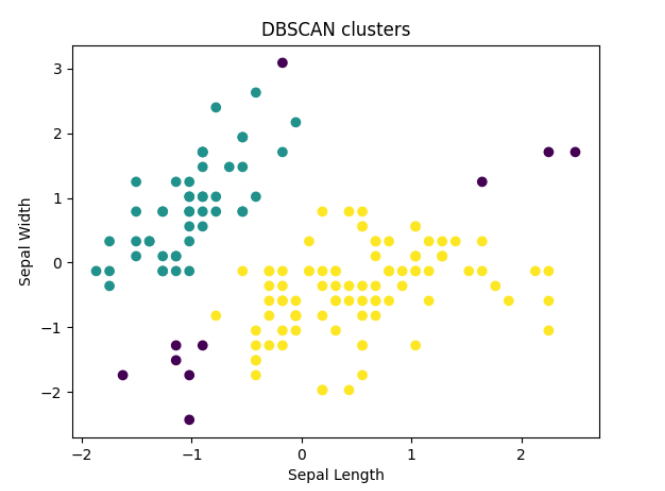


Рисунок 5.2 – Результати класифікації

В ідеальному випадку DBSCAN може досягти О(N) складності, але не варто особливо на це розраховувати. Якщо не перераховувати щоразу крапок, то очікувана складність — . O(N log N) Найгірший випадок (погані дані або брутфорс-реалізація) - .O(N^2) Є реалізації DBSCAN які люблять від'їдати пам'яті під матрицю відстаней - це явно надмірно.

DBSCAN не вираховує самостійні центри кластерів як k-means, однак вряд чи це проблема, особливо враховуючи вільну форму кластерів. Тому DBSCAN автоматично визначає викиди, що досить здорово.

Використовувати DBSCAN найдоцільніше коли:

* Великий набір даних більше 1 млн., або 10 млн.
* Ви очікуєте побачити сгустки даних екзотичної форми: вкладені та аномальні кластери, складки малої розмірності.
* Найкраще, якщо кластери все відділені друг від друга.
* Складність елементів набору даних не має Значення. Однак їх повинно бути достатньо, щоб не виникло сильних розривів у щільності
* Кількість елементів у кластері може змінюватися скільки завгодно.
* Кількість викидів значень не має, якщо вони розсіяні по більшому об'єму.
* Кількість кластерів значень не має.