**Bài 8: Unsupervised Learning: Clustering phần 1 - Lập trình AI bằng Python**

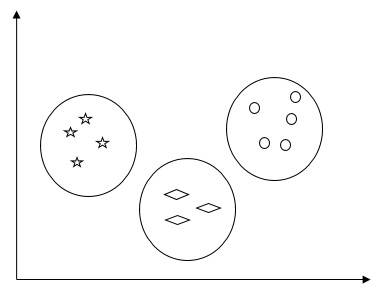
Các thuật toán học máy không giám sát không có bất kỳ người giám sát nào và cung cấp bất kỳ loại hướng dẫn nào. Đó là lý do tại sao chúng liên kết chặt chẽ với cái mà một số người gọi là trí tuệ nhân tạo thực sự.

Trong học tập không giám sát, sẽ không có câu trả lời chính xác và không có giáo viên hướng dẫn. Các thuật toán cần khám phá mô hình thú vị trong dữ liệu để học.

**1. Clustering là gì?**

Về cơ bản, nó là một loại phương pháp học không giám sát và là kỹ thuật phổ biến để phân tích dữ liệu thống kê được sử dụng trong nhiều lĩnh vực. Phân cụm chủ yếu là nhiệm vụ chia tập hợp các quan sát thành các tập con, được gọi là các cụm, theo cách sao cho các quan sát trong cùng một cụm giống nhau theo một nghĩa và chúng khác với các quan sát trong các cụm khác. Nói một cách dễ hiểu, chúng ta có thể nói rằng mục tiêu chính của phân cụm là nhóm dữ liệu trên cơ sở tương đồng và khác biệt.

Ví dụ dưới đây sẽ cho ta hình dung rõ hơn :



**3. Thuật toán Clustering :**

Cùng tìm hiểu qua một vài thuật toán trong clustering data

**a. K-Means**

Thuật toán phân cụm K-mean là một trong những thuật toán nổi tiếng để phân nhóm dữ liệu. Chúng ta cần giả định rằng số lượng các cụm đã được biết trước. Đây còn được gọi là phân cụm phẳng. Nó là một thuật toán phân cụm lặp đi lặp lại. Các bước thực hiện cho thuật toán này :

1. Xác định số lượng nhóm con K mong muốn
2. Cố định số lượng các cụm và chỉ định ngẫu nhiên từng điểm dữ liệu

Nói cách khác, chúng ta cần phân loại dữ liệu của mình dựa trên số lượng các cụm.

Trong bước này, các trọng tâm cụm sẽ được tính toán.

Vì đây là một thuật toán lặp lại, nên chúng ta cần cập nhật vị trí của K centroid với mỗi lần lặp lại cho đến khi chúng ta tìm thấy optima toàn cục hay nói cách khác là các centroid đạt đến vị trí tối ưu của chúng.

Đoạn code sau sẽ giúp thực hiện thuật toán phân cụm K-mean trong Python. Tôi sẽ sử dụng mô-đun Scikit-learning.

Import thư viện hỗ trợ :

**import** matplotlib.pyplot as plt

**import** seaborn as sns; sns.set()

**import** numpy as np

from sklearn.cluster **import** KMeans

Dòng code sau sẽ giúp tạo tập dữ liệu hai chiều, chứa bốn đốm màu, bằng cách sử dụng make\_blob từ package sklearn.dataset.

from sklearn.datasets.samples\_generator import make\_blobs

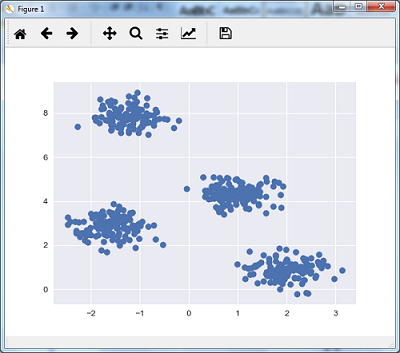
X, y\_true = make\_blobs(n\_samples = 500, centers = 4,

cluster\_std = 0.40, random\_state = 0)

Ta tiến hành visualize dataset như sau :

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

plt.show()



Ở đây, tôi đang khởi tạo kmeans thành thuật toán KMeans, với tham số bắt buộc là  n\_clusters (số lượng cluster).

kmeans = KMeans(n\_clusters = 4)

Ta tiến hành train model K-means với dữ liệu đầu vào

kmeans.fit(X)

y\_kmeans = kmeans.predict(X)

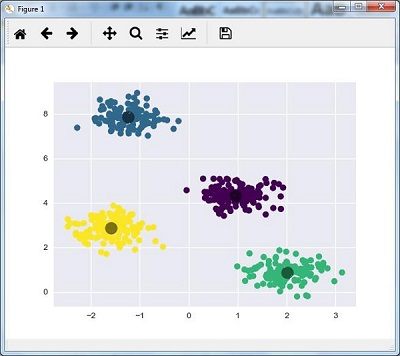
plt.scatter(X[:, 0], X[:, 1], c = y\_kmeans, s = 50, cmap = 'viridis')

centers = kmeans.cluster\_centers\_

Đoạn code được cung cấp bên dưới sẽ giúp ta vẽ và hình dung các phát hiện của máy dựa trên dữ liệu của mình và cách lắp ráp theo số lượng cụm sẽ được tìm thấy.

plt.scatter(centers[:, 0], centers[:, 1], c = 'black', s = 200, alpha = 0.5);

plt.show()



**b. Thuật toán Mean Shift :**

Nó là một thuật toán phân cụm phổ biến và mạnh mẽ khác được sử dụng trong học không giám sát. Nó không đưa ra bất kỳ giả định nào do đó nó là một thuật toán phi tham số. Nó còn được gọi là phân cụm phân cấp hoặc phân tích cụm dịch chuyển trung bình.Bao gồm các bước cơ bản sau :

* Trước hết, chúng ta cần bắt đầu với các điểm dữ liệu được gán cho một cụm riêng của chúng.
* Bây giờ, mô hình sẽ tính toán các trung tâm và cập nhật vị trí của các trung tâm mới.
* Bằng cách lặp lại quá trình này, chúng tôi di chuyển gần đỉnh của cụm, tức là về phía vùng có mật độ cao hơn.
* Thuật toán này dừng lại ở giai đoạn mà các centroid không di chuyển nữa.

Với sự trợ giúp của đoạn code sau, mình đang triển khai thuật toán phân cụm Mean Shift trong Python. Tôi sẽ sử dụng mô-đun Scikit-learning.

**import** numpy as np

from sklearn.cluster **import** MeanShift

**import** matplotlib.pyplot as plt

from matplotlib **import** style

style.use("ggplot")

Đoạn code sau sẽ giúp tạo tập dữ liệu hai chiều, chứa bốn đốm màu, bằng cách sử dụng make\_blob từ package sklearn.dataset.

from sklearn.datasets.samples\_generator **import** make\_blobs

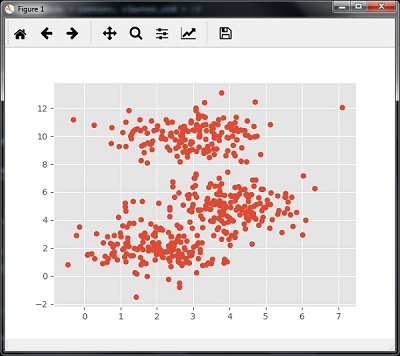
Chúng ta có thể hình dung tập dữ liệu bằng đoạn code sau

centers = [[2,2],[4,5],[3,10]]

X, \_ = make\_blobs(n\_samples = 500, centers = centers, cluster\_std = 1)

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

plt.show()



Bây giờ, chúng ta cần đào tạo mô hình cụm Mean Shift với dữ liệu đầu vào.

ms = MeanShift()

ms.fit(X)

labels = ms.labels\_

cluster\_centers = ms.cluster\_centers\_

Đoạn code sau sẽ in các trung tâm cụm và số lượng cụm dự kiến theo dữ liệu đầu vào:

print(cluster\_centers)

n\_clusters\_ = len(np.unique(labels))

print("Estimated clusters:", n\_clusters\_)

[[ 3.23005036 3.84771893]

[ 3.02057451 9.88928991]]

Estimated clusters: 2

Đoạn code được đưa ra dưới đây sẽ giúp vẽ biểu đồ và trực quan hóa các phát hiện của máy dựa trên dữ liệu đầu vào, và trang bị theo số lượng cụm sẽ được tìm thấy.

colors = 10\*['r.','g.','b.','c.','k.','y.','m.']

for i in range(len(X)):

plt.plot(X[i][0], X[i][1], colors[labels[i]], markersize = 10)

plt.scatter(cluster*\_centers[:,0],cluster\_*centers[:,1],

marker = "x",color = 'k', s = 150, linewidths = 5, zorder = 10)

plt.show()

