# Artificial Intelligence: Lab 2

## Unsupervised Learning (2) : DBSCAN & HAG

In the previous lab, we discussed a simple clustering algorithm : k-means. We have seen that it can be used as a method to create clusters or as a method to compress data.

In this lab we will study two additional clustering algorithms : Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Hierarchical Clustering (HC).

## 1 DBSCAN

1. Import necessary `python` modules to use `numpy` and `DBSCAN`.
2. Create a dataset X, a $6 \times 2$ numpy matrix with the following rows : (1,2), (2,2), (7,6), (8,7), (2,3) and (25,8), and plot the data.
3. Instantiate the DBSCAN model. The parameters will be epsilon equal to 3 and the minimum number of points required to form a cluster is 2.
4. Train the DBSCAN model with X as input data.
5. Print the predicted labels for each data point and draw the clusters.
6. Execute the following code and observe the result :

```
from sklearn.datasets import make_moons
from matplotlib import pyplot
from pandas import DataFrame
from sklearn.preprocessing import StandardScaler

# Generate test data using using make_moons()
X, y = make_moons(n_samples=750, shuffle=True, noise=0.11,
random_state=42)
X = StandardScaler().fit_transform(X)

# Split and organize the data
df = DataFrame(dict(x1=X[:,0], x2=X[:,1], label=y))
grouped = df.groupby('label')

# And plot it
colors = ["red", "blue"]
labels = ["x1", "x2"]
fig, ax = pyplot.subplots()
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x1', y='x2',
    label=labels[key], color=colors[key])
pyplot.show()
```

7. Use k-means algorithm to form the clusters. What do you observe ?
8. Try DBSCAN method with different values for `eps` and `min_samples`. What do you observe ?

# 2 Hierarchical Clustering

## 2.1 Example 1.

1. Execute the following code :

```
import numpy as np
X = np.array([[5,3],[10,15],[15,12],[24,10],[30,30],
    [85,70],[71,80],[60,78],[70,55],[80,91],])
```

2. Plot the above data points. To do so, execute the following code :

```
import matplotlib.pyplot as plt
%matplotlib inline
labels = range(1, 11)
plt.figure(figsize=(10, 7))
plt.subplots_adjust(bottom=0.1)
plt.scatter(X[:,0],X[:,1], label='True Position')
for label, x, y in zip(labels, X[:, 0], X[:, 1]):
    plt.annotate(label,xy=(x, y), xytext=(-3, 3),
    textcoords='offset points', ha='right', va='bottom')
plt.show()
```

## 2.2 Dendograms

A dendogram is diagram with a tree structure. It is used to divide into multiple clusters depending upon the problem. First execute the following code so we can observe how this tool can be used to get an idea on how our data can be clustered.

```
from scipy.cluster.hierarchy import dendrogram, linkage
from matplotlib import pyplot as plt
linked = linkage(X, 'single')
labelList = range(1, 11)
plt.figure(figsize=(10, 7))
dendrogram(linked,orientation='top',labels=labelList,
    distance_sort='descending',show_leaf_counts=True)
plt.show()
```

1. Add the following code in the suitable place to obtain Figure 1.

```
hline = np.array([[x,35] for x in range(10,80)])
plt.scatter(hline[:,0], hline[:,1], color='red')
```

2. Modify the code to obtain Figure 2. How many clusters can we expect ? Give the list of such clusters.

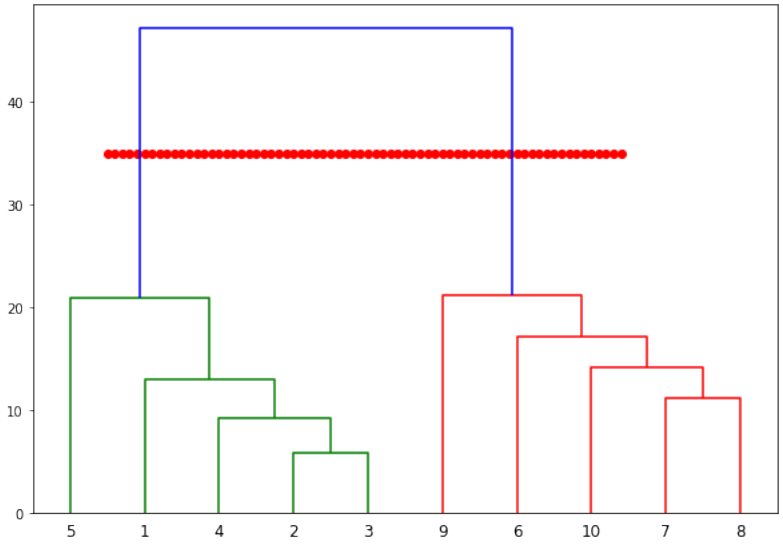


FIGURE 1 – Dendogram (1) of the first example.

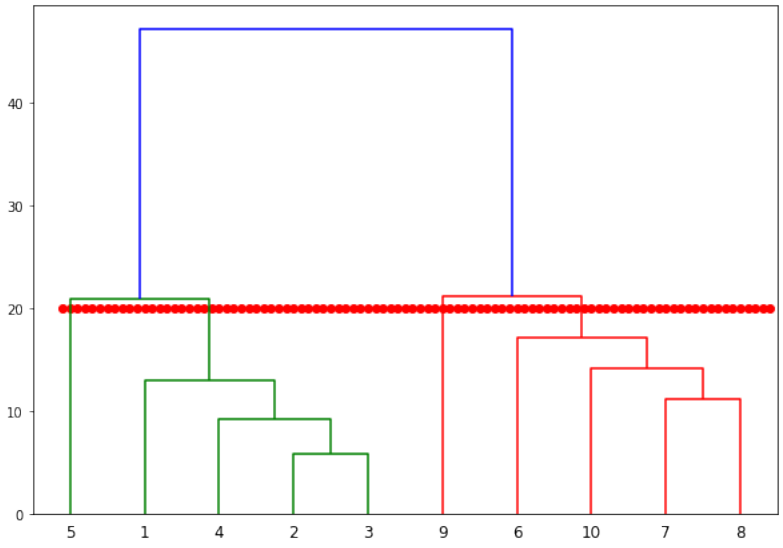


FIGURE 2 – Dendogram (2) of the first example.

### 2.3 Aglomerative Clustering

Now, we will use `AgglomerativeClustering` class from the `sklearn.cluster` module.

1. Execute the following code :

```
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=4,
    affinity='euclidean', linkage='ward')
cluster.fit_predict(X)
```

2. Print the labels of the clusters. Plot these clusters.

### 2.4 Hierarchical Clustering on Real Data

In the previous example, we performed hierarchical clustering on dummy data. In this example, we will perform hierarchical clustering on real-world data and see how it can be used to solve an actual problem. The problem that we are going to solve in this example is to segment customers into different groups based on their shopping trends.

The dataset is available at the following address :
`https://www.labri.fr/~zemmari/datasets/shopping-data.csv`

1. Write `python` instructions to open the dataset and visualise its content.
2. Execute the following script to filter the first three columns from our dataset :

```
data = custom_data[['Annual Income (k\$)','Spending Score (1-100)']]
```

3. Execute the following code to obtain the dendogram of this dataset.

```
import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10, 7))
plt.title("Customer Dendograms")
dend = shc.dendrogram(shc.linkage(data, method='ward'))
```

4. Add instructions to draw a horizontal line that passes through longest distance without a horizontal line. How many clusters do you expect ?
5. Use aglomeration clustering to create the clusters and plot them.
6. Give an interpretation of the result.