SIA 3611
Année universitaire 2022-2023

TP Machine Learning

TP 3 : Clustering

Vassilis.Christophides@ensea.fr
guillaume.renton@ensea.fr

Table des matières

Introduction

In machine learning, clustering is related to unsupervised learning approaches in which the algorithm fits from the distribution of given data. The main advantage of such methods is detection without a priori of patterns, sorting data and detecting outliers.

The given dataset is a combination of a dataset produced by the World Health Organization and the location of countries. It pooled the evolution of 23 features for 15 years among numerous countries. One of the goals of this TP3 is to detect continents and subcontinents.

Objectives :

- Visualize spatial datasets
- Apply K-Means and GMM on spatial datasets
- Normalize the datasets
- Propose a methodology to detect specific pools of data

1 Spatial datasets and first clusters

The first step is visualizing the mercator projection

1.1 To do :

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors

df = pd.read_csv("data/Life_Expectancy_Data.csv")
df = df.dropna()
df.info()

df1 = df[(df.Year == 2013)]
```

```
df_X = df1[['Longitude', 'Latitude']]
df_Status = df1[['Continent']]

df_Y = df_Status.replace(['Africa', 'Asia', 'Europe', 'North America', 'South America', 'Seven
    seas', 'Oceania'], [0, 1, 2, 3, 4, 5, 6])
np_Y = df_Y.to_numpy()
np_Y = np_Y.reshape((np_Y.shape[0],))

np1 = df_X.to_numpy()
plt.scatter(np1[:,0], np1[:,1], c=np_Y, cmap=matplotlib.colors.ListedColormap(['red', 'green',
    'blue', 'purple', 'yellow', 'magenta', 'cyan']))
plt.show()
```

Question 1

Identify each class with the plot colors.

What do the coordinates correspond to ?

1.2 To do :

Execute the following cell

```
from itertools import product
from sklearn import cluster

x_min, x_max = np1[:, 0].min() - 10, np1[:, 0].max() + 10
y_min, y_max = np1[:, 1].min() - 10, np1[:, 1].max() + 10
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),
                     np.arange(y_min, y_max, 0.1))

km2 = cluster.KMeans(n_clusters=2).fit(np1)
km3 = cluster.KMeans(n_clusters=3).fit(np1)
km4 = cluster.KMeans(n_clusters=4).fit(np1)
km6 = cluster.KMeans(n_clusters=6).fit(np1)

f, axarr = plt.subplots(2, 2, sharex='col', sharey='row', figsize=(10, 8))

for idx, km, tt in zip(product([0, 1], [0, 1]),
                       [km2, km3, km4, km6],
                       ["K=2, J=%.2f" % km2.inertia_,
                        "K=3, J=%.2f" % km3.inertia_,
                        "K=4, J=%.2f" % km4.inertia_,
                        "K=6, J=%.2f" % km6.inertia_]):

    Z = km.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    axarr[idx[0], idx[1]].contourf(xx, yy, Z, alpha=0.4)
    axarr[idx[0], idx[1]].scatter(np1[:, 0], np1[:, 1], c=np_Y,
                                  s=20, cmap=matplotlib.colors.ListedColormap(['red', 'green',
     'blue', 'purple', 'yellow', 'magenta', 'cyan']))
    axarr[idx[0], idx[1]].set_title(tt)

plt.show()
```

Question 2

How can inertia be used to compare cluster ? Can you propose a better metric for spatial dataset ?

Which is the principal problem with the mercantor representation for continent detection ?

2 Spatial dataset normalization

2.1 To code :

Displace the origin of longitude $30°$ east.

Question 3

What are the advantages of this normalization ?

2.2 To code :

Apply KMeans on the new normalized dataset

Gaussian Mixture Model is a clustering method allowing soft boundaries.

This method can be used through a sklearn function.

2.3 To code :

Apply GMM to the normalized dataset.

You have to test 3 conditions :

- GMM with default parameters for 2, 3, 4 and 6 components
- GMM with diagonal covariance matrix for 2, 3, 4 and 6 components
- GMM with random initialization for 2, 3, 4 and 6 components

Question 4

What is the default initialization for GMM ?

Which is the best method to detect continents ?

For this method, what would be the probability to find a country in the coordinate [-50,-40] ? What methodology can you think of to detect unpopulated oceans ?

2.4 Bonus

Rather than applying the previous longitudinal normalization in the mercator projected data and applying k-means, we could directly apply k-means on the unit-sphere representing the earth. The cosine similarity thus becomes a more suitable similarity measure than the euclidean distance.

Apply a KMeans with cosine similarity on the sphere instead of the previously tested euclidean distance in the plane.

3 Detection of carribean island

3.1 To code

Isolate the North and South American continents

3.2 To code

Propose a methodology to create a cluster including carribean independant nations.

Question 5

Which is the outlier of this clustering problem ?

4 Evaluating a cluster

In this section, we want to evaluate the quality of the different clusters computed.

4.1 To do

Execute the following cells. For the first cell, you can add code to once again displace the longitude by $30°$ east.

```
df_X = df1[['Longitude', 'Latitude']]
df_Status = df1[['Continent']]

df_Y = df_Status.replace(['Africa', 'Asia', 'Europe', 'North America', 'South America', 'Seven
    seas', 'Oceania'], [0, 1, 2, 3, 4, 5, 6])
np_Y = df_Y.to_numpy()
np_Y = np_Y.reshape((np_Y.shape[0],))

np1 = df_X.to_numpy()
plt.scatter(np1[:,0], np1[:,1], c=np_Y, cmap=matplotlib.colors.ListedColormap(['red', 'green',
    'blue', 'purple', 'yellow', 'magenta', 'cyan']))
plt.show()
```

```
x_min, x_max = np1[:, 0].min() - 10, np1[:, 0].max() + 10
y_min, y_max = np1[:, 1].min() - 10, np1[:, 1].max() + 10
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),
                     np.arange(y_min, y_max, 0.1))
```

```
from sklearn.metrics import silhouette_samples, silhouette_score

K = 3

fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.set_xlim([-0.1, 1])
ax1.set_ylim([0, len(np1) + (K + 1) * 10])

km = cluster.KMeans(n_clusters=K, random_state=10)
cluster_labels = km.fit_predict(np1)

silhouette_avg = silhouette_score(np1, cluster_labels)
print("For n_clusters =", K, "The average silhouette_score is :", silhouette_avg)
sample_silhouette_values = silhouette_samples(np1, cluster_labels)

y_lower = 10
for i in range(K):
    ith_cluster_silhouette_values = sample_silhouette_values[cluster_labels == i]
    ith_cluster_silhouette_values.sort()

    size_cluster_i = ith_cluster_silhouette_values.shape[0]
    y_upper = y_lower + size_cluster_i

```

```
    ax1.fill_betweenx(
        np.arange(y_lower, y_upper),
        0,
        ith_cluster_silhouette_values,
        alpha=0.7,
    )

    ax1.text(-0.05, y_lower + 0.5 * size_cluster_i, str(i))

    y_lower = y_upper + 10  # 10 for the 0 samples

    ax1.set_title("The silhouette plot for the various clusters.")
    ax1.set_xlabel("The silhouette coefficient values")
    ax1.set_ylabel("Cluster label")

    # The vertical line for average silhouette score of all the values
    ax1.axvline(x=silhouette_avg, color="red", linestyle="--")

    ax1.set_yticks([])  # Clear the yaxis labels / ticks
    ax1.set_xticks([-0.1, 0, 0.2, 0.4, 0.6, 0.8, 1])

    # 2nd Plot showing the actual clusters formed
    Z = km.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    ax2.contourf(xx, yy, Z, alpha=0.4)
    ax2.scatter(np1[:, 0], np1[:, 1], c=np_Y,
                s=20, cmap=matplotlib.colors.ListedColormap(['red', 'green', 'blue', 'purple',
 'yellow', 'magenta', 'cyan']))

    # Labeling the clusters
    centers = km.cluster_centers_
    # Draw white circles at cluster centers
    ax2.scatter(
        centers[:, 0],
        centers[:, 1],
        marker="o",
        c="white",
        alpha=1,
        s=200,
        edgecolor="k",
    )

    for i, c in enumerate(centers):
        ax2.scatter(c[0], c[1], marker="$%d$" % i, alpha=1, s=50, edgecolor="k")

    ax2.set_title("The visualization of the clustered data.")
    ax2.set_xlabel("Feature space for the 1st feature")
    ax2.set_ylabel("Feature space for the 2nd feature")

    plt.suptitle(
        "Silhouette analysis for KMeans clustering on sample data with n_clusters = %d"
        % K,
        fontsize=14,
        fontweight="bold",
    )

plt.show()
```

4.2 To do

Apply the silhouette computation for different values of K (at least 2,3,4 and 6).

Question 6

According to the silhouette score and the silhouette analysis, which is the most relevant value of K? Justify your response.

Bonus

Compute the silhouette score and analyze it for different parameters of Gaussian mixture.

4.3 To code

Another way to evaluate the quality of the clustering is through the homogeneity. This metric require a ground truth, so it can't be computed for every clustering problem. Luckily, we do have a ground truth here. Compute the homogeneity score for different values of K.

Question 7

According to the homogeneity score, which is the most relevant value of K ?

4.4 Bonus

Compute the homogeneity score and analyze it for different number of components and parameters of Gaussian Mixture.

Bonus : Alcoholism

This step is entirely optional and combines all the methods you used for this 3 TPs course.

The main goal is to develop a complete methodology to answer general questions.

All questions have to be justified by your homemade methodology and your methodology has also to be justified.

Question 1

In the year 2000, which countries are heavily concerned by an Alcohol issue ?

Question 2

In these countries and in 2000, which are the parameters linked with Alcoholism ? How do you explain these links ?

Question 3

Which is the evolution trend in these countries between 2000 and 2015 ? Try to separate these different trends.

Question 4

By selecting a specific country, can you explain a decrease or an increase through specific policies ?