VDD TP

Nom / Prenom	Numéro Etudiant
Sheikh Rakib	11502605
Daudin Louise	11606555

TP1 Python VDD

Sur le terminal taper les commandes suivant pour installer scikit

```
pip install -U scikit-learn scipy matplotlib
-- Pour python 3
pip3 install -U scikit-learn scipy matplotlib
```

Ensuite faire dans le terminal python :

```
from sklearn import *
import numpy
import matplotlib.pyplot

iris = datasets.load_iris()

len(iris.data)

print(iris.target)
print(iris.feature_names)
print(iris.target_names)
```

Print the number of data, name of variables, name of classes :

```
len(iris.data)
print(iris.feature_names)
print(iris.target_names)
```

TP-1 version fr

Importez les librairies numpy et preprocessing

```
import numpy
import sklearn.preprocessing
```

Creer la matrice X suivante :

Référence : https://www.programiz.com/python-programming/matrix

```
X = numpy.matrix('1 -1 2; 2 0 0; 0, 1, -1')
X.mean()
X.var()
```

```
0.4444444444444444
1.1358024691358024
```

```
x_scaled = preprocessing.scale(X)
print(x_scaled)
```

```
[[ 0.         -1.22474487  1.33630621]
 [ 1.22474487  0.         -0.26726124]
 [-1.22474487  1.22474487 -1.06904497]]
```

On constate que la fonction scale ajoute des valeurs décimale au hasard.

4. Calcule de la moyenne et de la variance de la matrice X Normalisé :

```
X_scaled.mean()
X_scaled.var()
```

```
4.9343245538895844e-17
1.0
```

Lorque on a normalisé la matrice, ca permet de remettre la variance à 1.

C Normalisation et reduction de dimensions

1. Créez la matrice de données X2 suivante :

```
X2 = numpy.matrix('1 -1 2; 2 0 0; 0 1 -1)
```

2. Visualisez la matrice et calculez la moyenne sur les variables

```
print(X2)
X2.mean()
```

```
print(X2)
[[ 1 -1  2]
 [ 2  0  0]
 [ 0  1 -1]]

X2.mean()
0.4444444444444444
```

3. Normalisez les données dans l'intervalle [0,1], Visualiser les données normalisées et calcule de moyenne sur les variables.

```
min_max_scaler = preprocessing.MinMaxScaler()
X2_min_max = min_max_scaler.fit_transform(X2)
```

```
  X2_min_max
  array([[0.5       , 0.        , 1.        ],
         [1.        , 0.5       , 0.33333333],
         [0.        , 1.        , 0.        ]])
```

4. Charger les données IRIS

```
iris = datasets.load_iris()
```

5. Afficher les données, les noms des variables et le nom des classes

```
len(iris.data)

print(iris.target)
print(iris.feature_names)
print(iris.target_names)
```

6. Visualisez les nuages de points en 2D avec des couleurs corrspondant aux classes en utilisant toutes les combinaisons de variables

```
plt.figure()
plt.scatter(iris.data[:,0],iris.data[:,1], c = iris.target)
plt.scatter(iris.data[:,0],iris.data[:,2], c = iris.target)
```

Il y a 6 manière de faire le plot (0-1 | 0-2 | 0-3 | 1-2 | 1-3 | 2-3).

8. Analysez le manuel d'aide pour ces deux fonctions (pcaet lda) et appliquez les sur la base Iris. Il faudra utiliser pca.fit(Iris).transform(Iris)et sauvegardez les résultats dans IrisPCA pour la PCA et IrisLDA pour la LDA

```
from sklearn.decomposition import PCA
from sklearn.lda import LDA
import pandas as pd


pca_iris = PCA(n_components =2)
principalCompo_iris = pca_iris.fit_transform(iris.data)
principal_iris_df = pd.DataFrame(data = principalCompo_iris, columns = ['PComponent 1','PC2'])
```

```
#ça marche pas D:
```