**LAB 1**

**SUPERVISED AND UNSUPERVISED**

<https://www.kaggle.com/uciml/iris>

import numpy as np

import pandas as pd

from google.colab import files

uploaded = files.upload()

**Read the files**

iris =   pd.read\_csv("Iris.csv")

**Understand the data**

print("Head ==> \n",iris.head())

print("Describe ==> \n",iris.describe())

**Visualizing using Matplot**

from matplotlib import pyplot as plt

iris.plot(kind="scatter", x="SepalLengthCm",   y="SepalWidthCm")

plt.show()

**Visualizing using Pandas Andrew’s Curve**

from pandas.plotting import andrews\_curves

andrews\_curves(iris.drop("Id", axis=1), "Species")

plt.show()

**Preprocessing**

from sklearn.model\_selection import train\_test\_split

x = iris.iloc[:, :-1].values

y = iris.iloc[:,   -1].values

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.2,random\_state=0)

**Supervised Learning**

from sklearn.tree import   DecisionTreeClassifier

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import accuracy\_score

from sklearn.metrics import classification\_report

classifier   = DecisionTreeClassifier()

classifier.fit(x\_train,   y\_train)

y\_pred   = classifier.predict(x\_test)

print(classification\_report(y\_test,   y\_pred))

print(confusion\_matrix(y\_test, y\_pred))

**HeatMap for Confusion matrix**

import seaborn as sns

cm  = confusion\_matrix(y\_test, y\_pred)

cm\_df = pd.DataFrame(cm,index = ['setosa','versicolor','virginica'], columns = ['setosa','versicolor','virginica'])

plt.figure(figsize=(5.5,4))

sns.heatmap(cm\_df,   annot=True)

plt.ylabel('True label')

plt.xlabel('Predicted label')

**Unsupervised Learning**

from sklearn.datasets import load\_iris

from sklearn.cluster import KMeans

iris\_data=load\_iris()

iris\_df = pd.DataFrame(iris\_data.data, columns = iris\_data.feature\_names)

kmeans = KMeans(n\_clusters=3,init = 'k-means++',   max\_iter = 100, n\_init = 10, random\_state = 0)

y\_kmeans = kmeans.fit\_predict(x)

print(kmeans.cluster\_centers\_) #display cluster centers

plt.scatter(x[y\_kmeans   == 0, 0], x[y\_kmeans == 0, 1],s = 100, c = 'red', label = 'Iris-setosa')

plt.scatter(x[y\_kmeans   == 1, 0], x[y\_kmeans == 1, 1],s = 100, c = 'blue', label = 'Iris-versicolour')

plt.scatter(x[y\_kmeans   == 2, 0], x[y\_kmeans == 2, 1],s = 100, c = 'green', label = 'Iris-virginica')

plt.scatter(kmeans.cluster\_centers\_[:,   0], kmeans.cluster\_centers\_[:,1],s = 100, c = 'black', label = 'Centroids')

plt.legend()

plt.show()