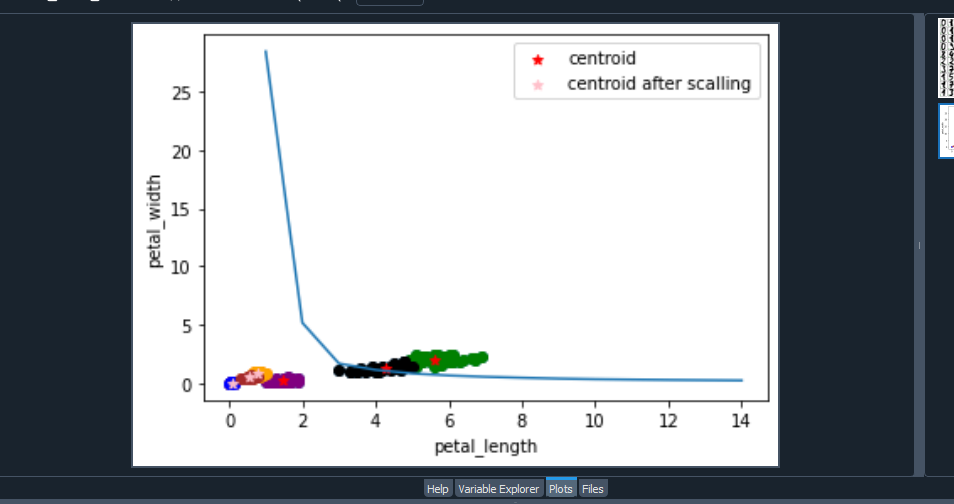
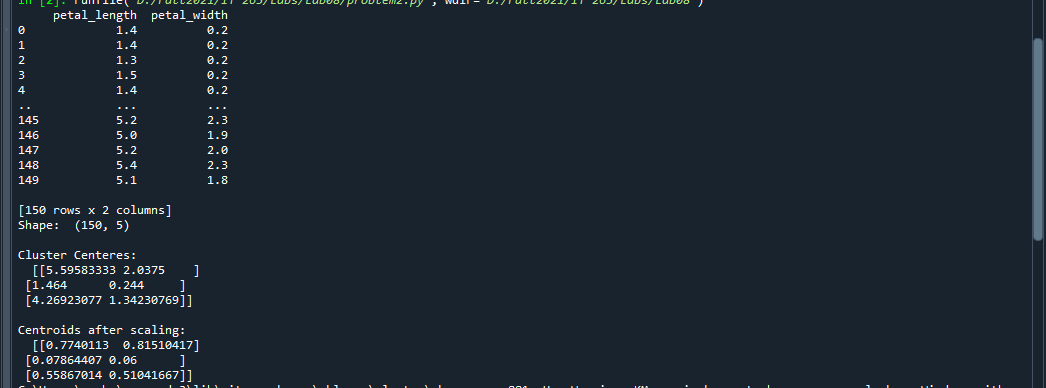
**Student Names: Ghadir Alfadhl & Li Baio**

**Course Name: ITS265** – **Lab08**

**Problem 2:** Download the iris flower dataset as a csv from Kaggle.com.Read the csv iris dataset into a Python SKLearn application. Use unsupervised learning to form clusters of flowers using petal width and length features. Drop the other two features. Figure out if any preprocessing such as scaling would help. Draw elbow plot and from it figure out optimal K value. Use the KMeans algorithm. Determine the centroids from the clustering of the data.

**The K value is 3**





**#Code**

from sklearn.cluster import KMeans

import pandas as pd

#import numpy as np

from sklearn.preprocessing import MinMaxScaler

from matplotlib import pyplot as plt

df = pd.read\_csv("IRIS.csv")

print(df[['petal\_length','petal\_width' ]])

print('Shape: ',df.shape)

plt.scatter(df['petal\_length'],df['petal\_width'])

plt.xlabel('petal\_length')

plt.ylabel('petal\_width')

km = KMeans(n\_clusters=3)

y\_predicted = km.fit\_predict(df[['petal\_length','petal\_width']])

#print(y\_predicted)

df['cluster']=y\_predicted

#print(df.head())

print("\nCluster Centeres:\n ", km.cluster\_centers\_)

#plt.plot(km.cluster\_centers\_)

df1 = df[df.cluster==0]

df2 = df[df.cluster==1]

df3 = df[df.cluster==2]

plt.scatter(df1.petal\_length,df1['petal\_width'],color='green')

plt.scatter(df2.petal\_length,df2['petal\_width'],color='purple')

plt.scatter(df3.petal\_length,df3['petal\_width'],color='black')

plt.scatter(km.cluster\_centers\_[:,0],km.cluster\_centers\_[:,1],color='red',

marker='\*',label='centroid')

plt.xlabel('petal\_length')

plt.ylabel('petal\_width')

plt.legend()

# scal both

scaler = MinMaxScaler() #scaler object

scaler.fit(df[['petal\_width']]) # Scales 0 to 1

df['petal\_width'] = scaler.transform(df[['petal\_width']])

scaler.fit(df[['petal\_length']])

df['petal\_length'] = scaler.transform(df[['petal\_length']])

df.head()

plt.scatter(df.petal\_length,df['petal\_width'])

km = KMeans(n\_clusters=3)

y\_predicted = km.fit\_predict(df[['petal\_length','petal\_width']])

y\_predicted

df['cluster']=y\_predicted

df.head()

print("\nCentroids after scaling:\n ",km.cluster\_centers\_)

df1 = df[df.cluster==0]

df2 = df[df.cluster==1]

df3 = df[df.cluster==2]

plt.scatter(df1.petal\_length,df1['petal\_width'],color='orange')

plt.scatter(df2.petal\_length,df2['petal\_width'],color='blue')

plt.scatter(df3.petal\_length,df3['petal\_width'],color='brown')

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

color='pink',marker='\*',

label='centroid after scalling')

plt.legend()

sse = []

k\_rng = range(1,15)

for k in k\_rng:

km = KMeans(n\_clusters=k)

km.fit(df[['petal\_length','petal\_width']])

sse.append(km.inertia\_)

#sum(np.min(csse(df[['petal\_length','petal\_width']],

# km.cluster\_centers\_,'euclidean'),axis=1)) / (df[['petal\_length','petal\_width']].shape[0])

plt.xlabel('petal\_length')

plt.ylabel('petal\_width')

plt.plot(k\_rng,sse)