Name: Gore Aniket Machhindra

Email: aniket.m.gore.1901@gmail.com

Role: Data Science and Business Analytics Intern

Task 2: Prediction using Unupervised ML

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

```
from sklearn.datasets import load_iris
dataset=load_iris()
dataset
```

```
       [5.4, 3. , 4.5, 1.5],
       [6. , 3.4, 4.5, 1.6],
       [6.7, 3.1, 4.7, 1.5],
       [6.3, 2.3, 4.4, 1.3],
       [5.6, 3. , 4.1, 1.3],
       [5.5, 2.5, 4. , 1.3],
       [5.5, 2.6, 4.4, 1.2],
       [6.1, 3. , 4.6, 1.4],
       [5.8, 2.6, 4. , 1.2],
       [5. , 2.3, 3.3, 1. ],
       [5.6, 2.7, 4.2, 1.3],
       [5.7, 3. , 4.2, 1.2],
       [5.7, 2.9, 4.2, 1.3],
       [6.2, 2.9, 4.3, 1.3],
       [5.1, 2.5, 3. , 1.1],
       [5.7, 2.8, 4.1, 1.3],
       [6.3, 3.3, 6. , 2.5],
       [5.8, 2.7, 5.1, 1.9],
       [7.1, 3. , 5.9, 2.1],
       [6.3, 2.9, 5.6, 1.8],
       [6.5, 3. , 5.8, 2.2],
       [7.6, 3. , 6.6, 2.1],
       [4.9, 2.5, 4.5, 1.7],
       [7.3, 2.9, 6.3, 1.8],
       [6.7, 2.5, 5.8, 1.8],
       [7.2, 3.6, 6.1, 2.5],
       [6.5, 3.2, 5.1, 2. ],
       [6.4, 2.7, 5.3, 1.9],
       [6.8, 3. , 5.5, 2.1],
       [5.7, 2.5, 5. , 2. ],
       [5.8, 2.8, 5.1, 2.4],
       [6.4, 3.2, 5.3, 2.3],
       [6.5, 3. , 5.5, 1.8],
       [7.7, 3.8, 6.7, 2.2],
       [7.7, 2.6, 6.9, 2.3],
       [6. , 2.2, 5. , 1.5],
       [6.9, 3.2, 5.7, 2.3],
       [5.6, 2.8, 4.9, 2. ],
       [7.7, 2.8, 6.7, 2. ],
       [6.3, 2.7, 4.9, 1.8],
       [6.7, 3.3, 5.7, 2.1],
       [7.2, 3.2, 6. , 1.8],
       [6.2, 2.8, 4.8, 1.8],
       [6.1, 3. , 4.9, 1.8],
       [6.4, 2.8, 5.6, 2.1],
       [7.2, 3. , 5.8, 1.6],
       [7.4, 2.8, 6.1, 1.9],
       [7.9, 3.8, 6.4, 2. ],
       [6.4, 2.8, 5.6, 2.2],
       [6.3, 2.8, 5.1, 1.5],
       [6.1, 2.6, 5.6, 1.4],
       [7.7, 3. , 6.1, 2.3],
       [6.3, 3.4, 5.6, 2.4],
       [6.4, 3.1, 5.5, 1.8],
       [6. , 3. , 4.8, 1.8],
       [6.9, 3.1, 5.4, 2.1],
       [6.7, 3.1, 5.6, 2.4],
```

```
#converting dataset into rows and columns form
iris=pd.DataFrame(dataset.data,columns=dataset.feature_names)
```

```
iris.head()

```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

```
iris.shape
```

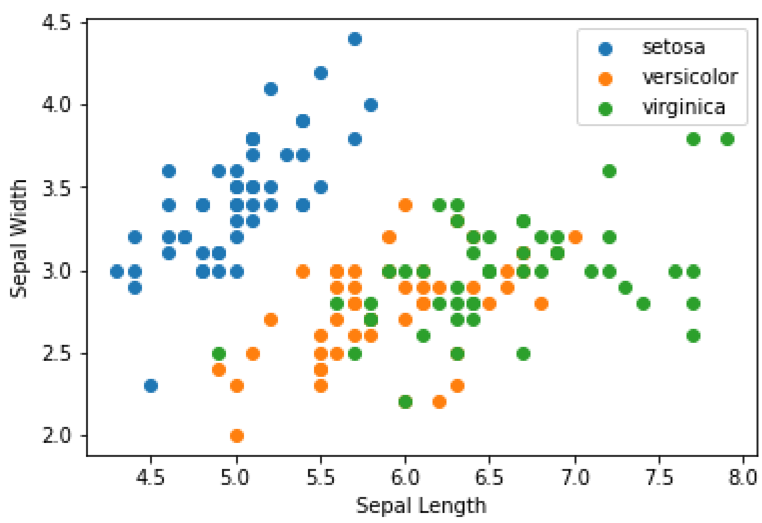
```
(150, 4)
```

```
iris.isnull().sum()
```

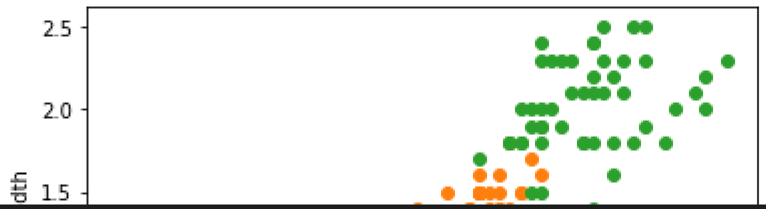
```
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
dtype: int64
```

```
x=dataset.data
y=dataset.target
```

```
#plotting a graph between sepal length and width
plt.scatter(x[y==0,0],x[y==0,1], label='setosa')
plt.scatter(x[y==1,0],x[y==1,1], label='versicolor')
plt.scatter(x[y==2,0],x[y==2,1], label='virginica')
plt.legend(loc='upper right')
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.show()
```

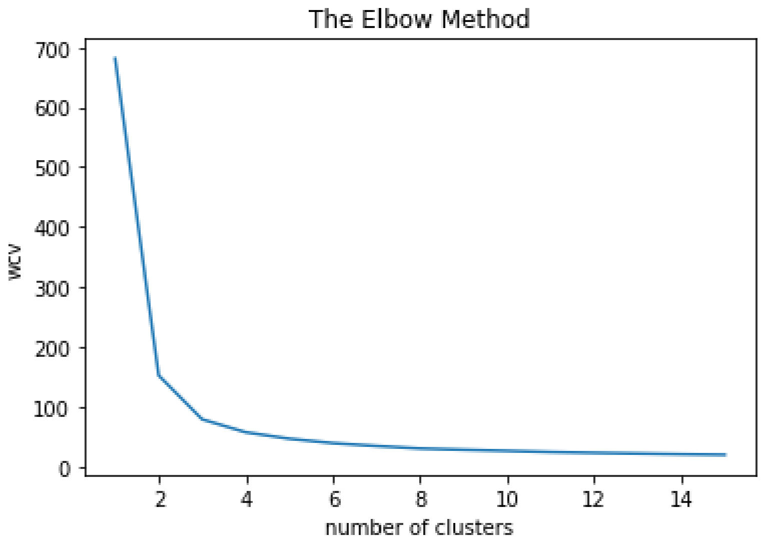


```
#plotting a graph between petal length and width
plt.scatter(x[y==0,2],x[y==0,3], label='setosa')
plt.scatter(x[y==1,2],x[y==1,3], label='versicolor')
plt.scatter(x[y==2,2],x[y==2,3], label='virginica')
plt.legend(loc='lower right')
plt.xlabel('Petal Length')
plt.ylabel('Petal Width')
plt.show()
```



```
#using the K-Means Clustering
from sklearn.cluster import KMeans
#checking the optimum values of 'k' in the cluster variation
wcv=[]
for i in range(1,16):
  kM=KMeans(n_clusters=i)
  kM.fit(x)
  wcv.append(kM.inertia_)

#plotting a graph for getting the elbow point
plt.plot(range(1,16),wcv)
plt.title('The Elbow Method')
plt.xlabel('number of clusters')
plt.ylabel('wcv')
plt.show()
```



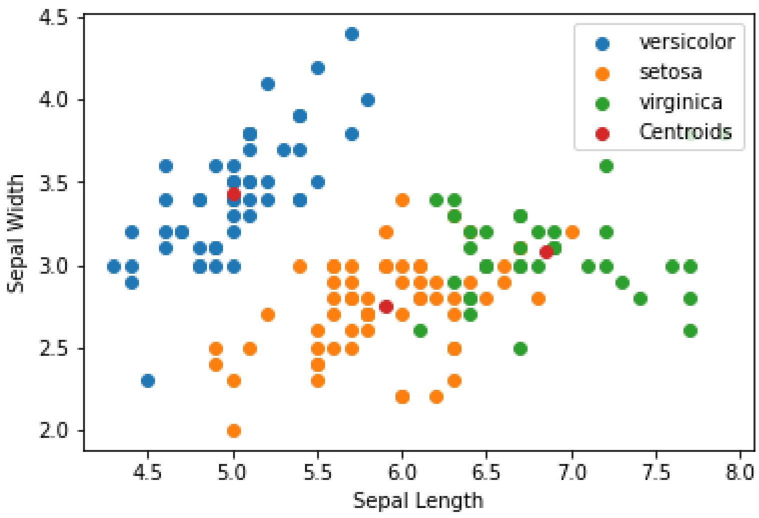
```
#creating the kmeans classifier
kM=KMeans(n_clusters=3)
y_pred=kM.fit_predict(x)
y_pred
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2,
       2, 2, 2, 1, 1, 2, 2, 2, 2, 1, 2, 1, 2, 1, 2, 2, 1, 1, 2, 2, 2, 2,
       2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 1, 2, 2, 2, 1, 2, 2, 1], dtype=int32)
```

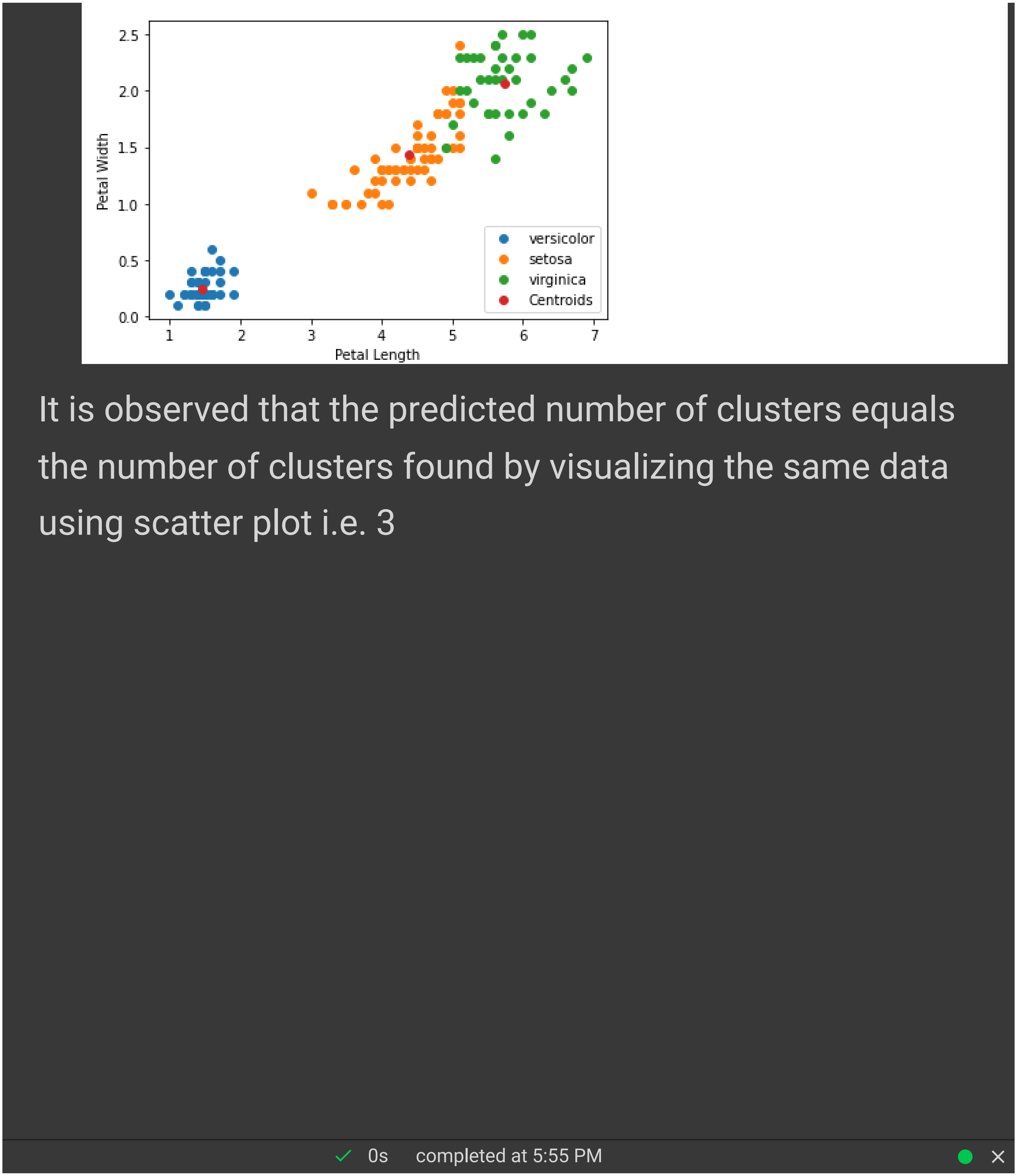
```
iris['clusters']=y_pred
iris.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	clusters
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

```
#plotting the centroids
plt.scatter(x[y_pred==0,0],x[y_pred==0,1], label='versicolor')
plt.scatter(x[y_pred==1,0],x[y_pred==1,1], label='setosa')
plt.scatter(x[y_pred==2,0],x[y_pred==2,1], label='virginica')
plt.scatter(kM.cluster_centers_[:,0],kM.cluster_centers_[:,1], label="Centroids")
plt.legend(loc='upper right')
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.show()
```



```
#plotting the centroids
plt.scatter(x[y_pred==0,2],x[y_pred==0,3], label='versicolor')
plt.scatter(x[y_pred==1,2],x[y_pred==1,3], label='setosa')
plt.scatter(x[y_pred==2,2],x[y_pred==2,3], label='virginica')
plt.scatter(kM.cluster_centers_[:,2],kM.cluster_centers_[:,3], label="Centroids")
plt.legend(loc='lower right')
plt.xlabel('Petal Length')
plt.ylabel('Petal Width')
plt.show()
```



It is observed that the predicted number of clusters equals the number of clusters found by visualizing the same data using scatter plot i.e. 3