LAB NO: 10

K MEANS CLUSTERING

LAB TASK

Q1. Load a dataset (e.g., Mall Customer Segmentation).

```
[1]
✓ 2s
```

```python
#Load Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
```

```
[2]
✓ 0s
```

```python
#Load Dataset (Iris)
iris = load_iris()
X = iris.data      # features
y = iris.target    # labels (for evaluation only)
df = pd.DataFrame(X, columns=iris.feature_names)
print("Dataset Head:")
print(df.head())
```

```
Dataset 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
```

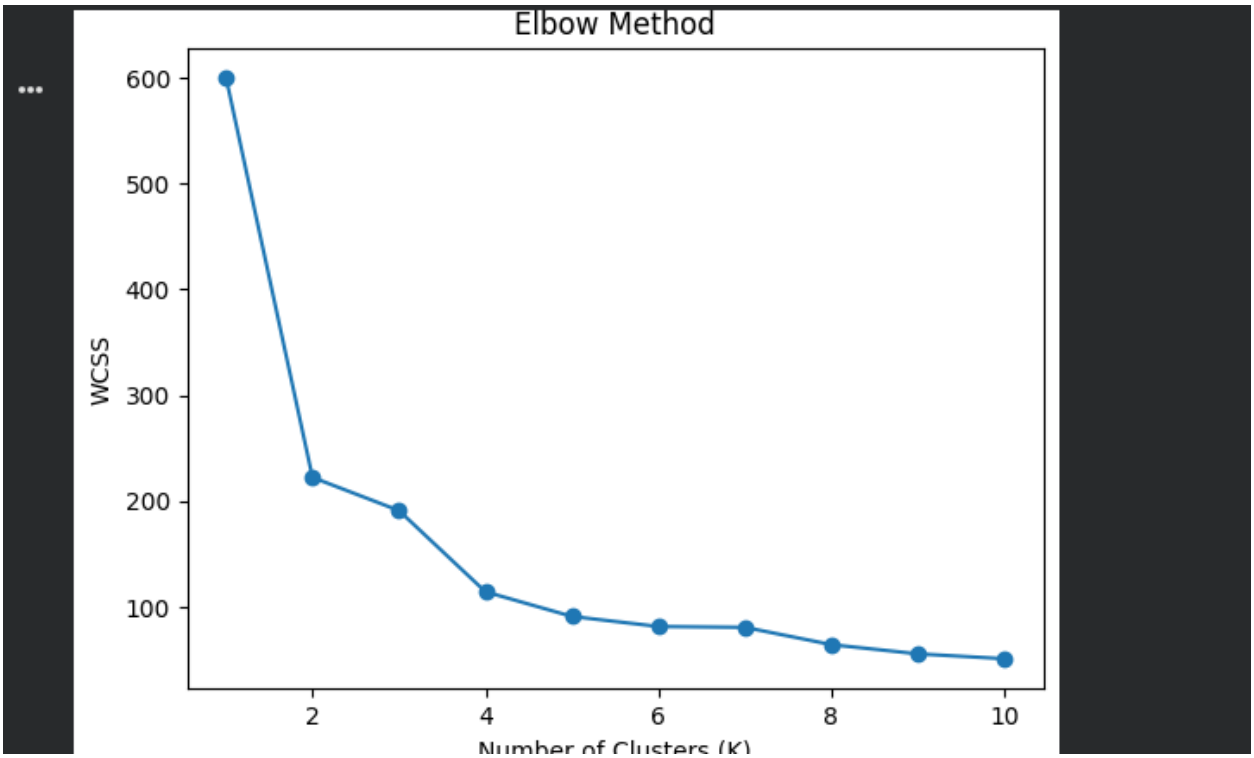
Q2. Apply data preprocessing (normalize features if needed).

```
[3]
✓ 0s
```

```python
#Preprocessing (Normalization)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Q3. Determine the optimal number of clusters using Elbow Method

```
[4]
✓ 0s
#Determine K using Elbow Method
wcss = []  # within-cluster sum of squares
for k in range(1, 11):
    km = KMeans(n_clusters=k, random_state=42)
    km.fit(X_scaled)
    wcss.append(km.inertia_)
plt.plot(range(1, 11), wcss, marker='o')
plt.title("Elbow Method")
plt.xlabel("Number of Clusters (K)")
plt.ylabel("WCSS")
plt.show()
```



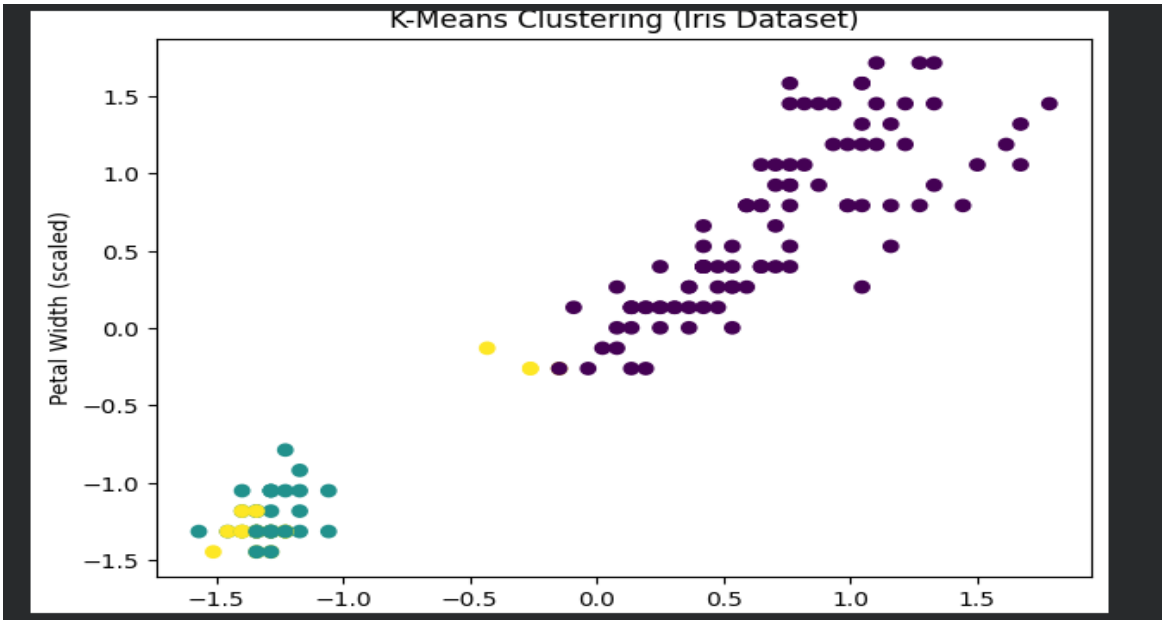
Q4. Implement K-Means clustering for the selected value of K.

```
[5]
✓ 0s
#Apply K-Means (Choose K = 3 for Iris)
kmeans = KMeans(n_clusters=3, random_state=42)
clusters = kmeans.fit_predict(X_scaled)
print("Cluster Labels Assigned by KMeans:")
print(clusters[:10])

Cluster Labels Assigned by KMeans:
[1 2 2 2 1 1 1 1 2 2]
```

Q5. Visualize the clusters in 2D or 3D (if the dataset has 2 or 3 features).

```
[7]
✓ 0s
# 6. Visualize the clusters (2D plot)
plt.figure(figsize=(7,5))
plt.scatter(X_scaled[:, 2], X_scaled[:, 3], c=clusters, cmap='viridis')
plt.title("K-Means Clustering (Iris Dataset)")
plt.xlabel("Petal Length (scaled)")
plt.ylabel("Petal Width (scaled)")
plt.show()
```



Q6. Evaluate the clustering: Although K-Means is unsupervised, since the Iris dataset has labels, we can compare them.

```
[17]
⚠ 0s
#7. Evaluation (Comparing with True Labels)
from scipy.stats import mode

labels = np.zeros_like(clusters)
for i in range(3):
  mask = (clusters == i)
  labels[mask] = mode(y[mask]).mode[0]

print("\nAccuracy of K-Means vs Actual Labels:")
print(accuracy_score(y, labels))

print("\nConfusion Matrix:")
print(confusion_matrix(y, labels))
```