```
import kagglehub
path = kagglehub.dataset_download("uciml/iris")
print("Path to dataset files:", path)
```

```
Warning: Looks like you're using an outdated `kagglehub` version, please consider updating (latest vers
Path to dataset files: /root/.cache/kagglehub/datasets/uciml/iris/versions/2
```

Importing Necessary libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
```

```
df = pd.read_csv("/root/.cache/kagglehub/datasets/uciml/iris/versions/2/Iris.csv")
df.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Next steps: Generate code with df View recommended plots New interactive sheet

Standardizing the data

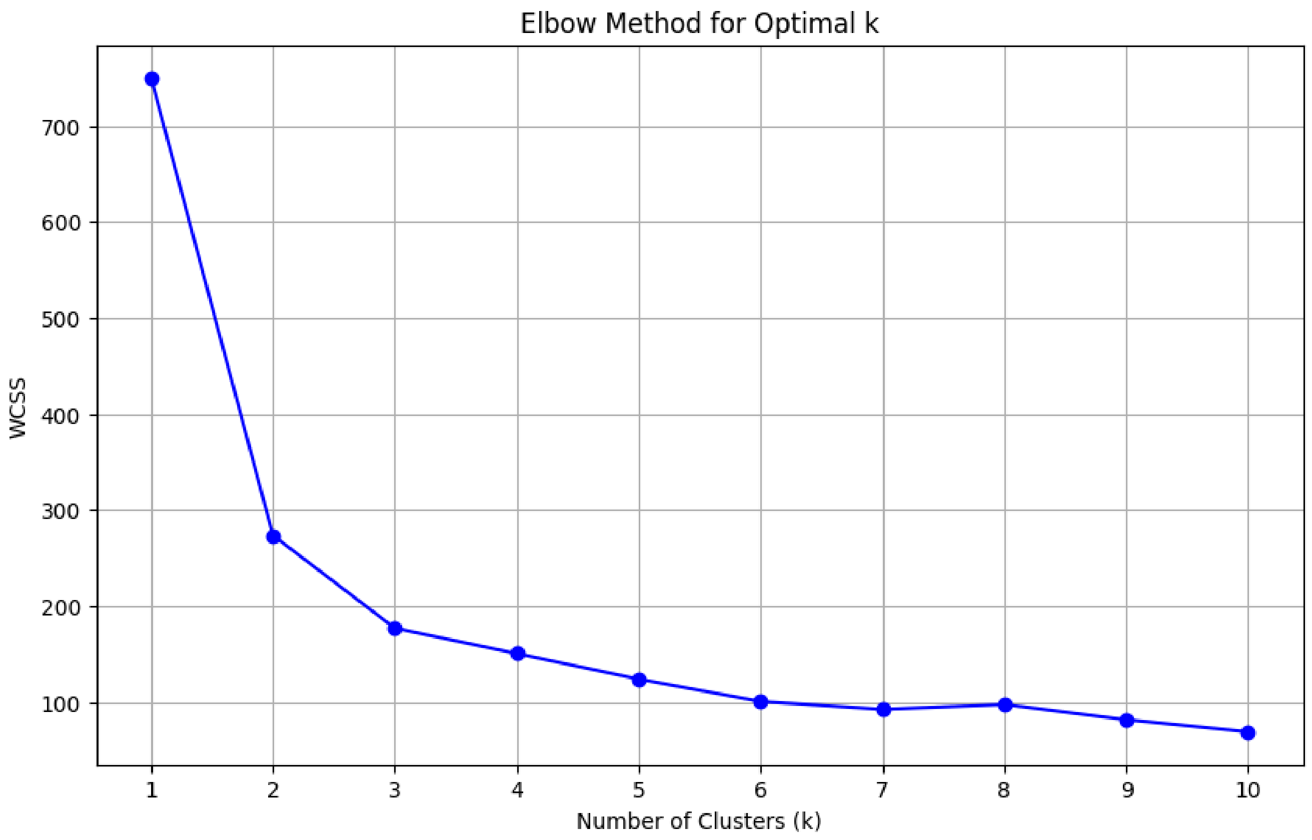
```
X = df.iloc[:, :-1].values
```

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Finding optimal number of clusters using elbow method

```
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)
```

```
plt.figure(figsize=(10, 6))
plt.plot(range(1, 11), wcss, marker='o', color='b')
plt.title('Elbow Method for Optimal k')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('WCSS')
plt.xticks(range(1, 11))
plt.grid()
plt.show()
```



```
optimal_k = 3
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
y_kmeans = kmeans.fit_predict(X_scaled)


df['Cluster'] = y_kmeans
cluster_mapping = {0: 'Iris Setosa', 1: 'Iris Versicolor', 2: 'Iris Virginica'}
df['Cluster'] = df['Cluster'].map(cluster_mapping)
df.head()
```

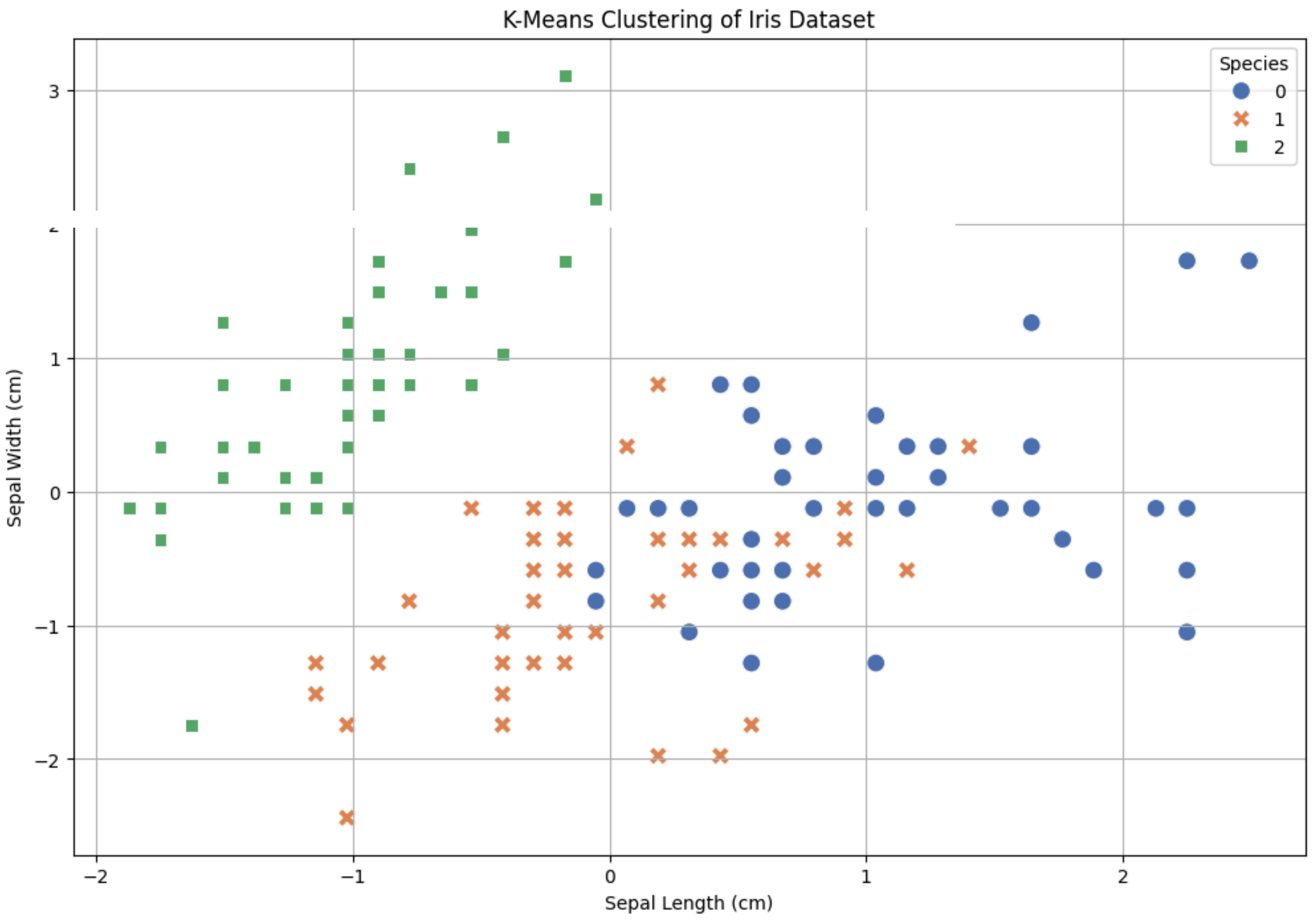
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	Cluster
0	1	5.1	3.5	1.4	0.2	Iris-setosa	Iris Virginica
1	2	4.9	3.0	1.4	0.2	Iris-setosa	Iris Virginica
2	3	4.7	3.2	1.3	0.2	Iris-setosa	Iris Virginica
3	4	4.6	3.1	1.5	0.2	Iris-setosa	Iris Virginica
4	5	5.0	3.6	1.4	0.2	Iris-setosa	Iris Virginica

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
centroids = kmeans.cluster_centers_
centroids_original = scaler.inverse_transform(centroids)
plt.figure(figsize=(12, 8))
sns.scatterplot(data=clustered_data, x='SepalLengthCm', y='SepalWidthCm', hue='Cluster', palette='deep', st
plt.title('K-Means Clustering of Iris Dataset')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
```

```
for i, (x, y) in enumerate(zip(centroids_original[:, 0], centroids_original[:, 1])):
    plt.annotate(cluster_mapping[i], (x, y), textcoords="offset points", xytext=(0,10), ha='center', fontsi

plt.legend(title='Species')
plt.grid()
plt.show()
```


K-Means Clustering of Iris Dataset
Species
Sepal Width (cm)
Sepal Length (cm)