Here's the **procedure** for the university ranking clustering code:

1. Load the Dataset

- Extract the CSV file from the ZIP archive.
- Read the dataset into a Pandas DataFrame.

2. Select Relevant Features

- Choose the columns:
    - `This_site_rank_in_global_internet_engagement` (Global ranking of the site).
    - `Daily_time_on_site` (User engagement time).
- Drop missing values to ensure data consistency.

3. Standardize the Data

- Normalize the selected columns using **StandardScaler** to ensure all values are on the same scale.

4. Determine Optimal Number of Clusters (Elbow Method)

- Perform K-Means clustering for **k = 1 to 10**.
- Store the **inertia (sum of squared distances to cluster centers)** for each k-value.
- Plot the **Elbow Curve** to identify the optimal number of clusters.

5. Apply K-Means Clustering

- Choose the best **K value** (e.g., 3 clusters).
- Apply **K-Means** and assign each university site to a cluster.

6. Analyze the Clusters

- Calculate the **average ranking & engagement time** for each cluster.

- Print the **cluster distribution**.

7. Save the Results

- Save the clustered dataset to a **CSV file** (`university_clusters.csv`) for further analysis.

```
import pandas as pd
import zipfile
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

# Load the dataset
zip_path = r"c:\Users\balaj\Downloads\alexa.com_site_info.csv.zip"
with zipfile.ZipFile(zip_path, 'r') as z:
    with z.open('alexa.com_site_info.csv') as f:
        df = pd.read_csv(f)

# Select relevant columns for clustering
columns = ["This_site_rank_in_global_internet_engagement",
"Daily_time_on_site"]
df_filtered = df[columns].dropna()

# Standardize the data
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df_filtered)

# Determine optimal number of clusters using the Elbow Method
inertia = []
k_range = range(1, 11)
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
    kmeans.fit(df_scaled)
    inertia.append(kmeans.inertia_)

plt.figure(figsize=(8, 5))
plt.plot(k_range, inertia, marker='o')
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.title('Elbow Method for Optimal K')
plt.show()
```

```
# Apply K-Means clustering with optimal K (e.g., 3 for illustration)
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
df_filtered['Cluster'] = kmeans.fit_predict(df_scaled)

# Show cluster distribution
print(df_filtered.groupby('Cluster').mean())

# Save results
df_filtered.to_csv("university_clusters.csv", index=False)
```

OUTPUT:

PS C:\Users\balaj\OneDrive\Desktop> &

'c:\Users\balaj\AppData\Local\Microsoft\WindowsApps\python3.10.exe'

'c:\Users\balaj\.vscode\extensions\ms-python.debugpy-2025.4.1-win32-x64\bundled\libs\debugpy\launcher' '60882' '--' 'c:\Users\balaj\OneDrive\Desktop\un.py'

Cluster	This_site_rank_in_global_internet_engagement	Daily_time_on_site
0	641276.927039	1790.061516
1	88999.313675	897.419355
2	94183.017217	2743.570552

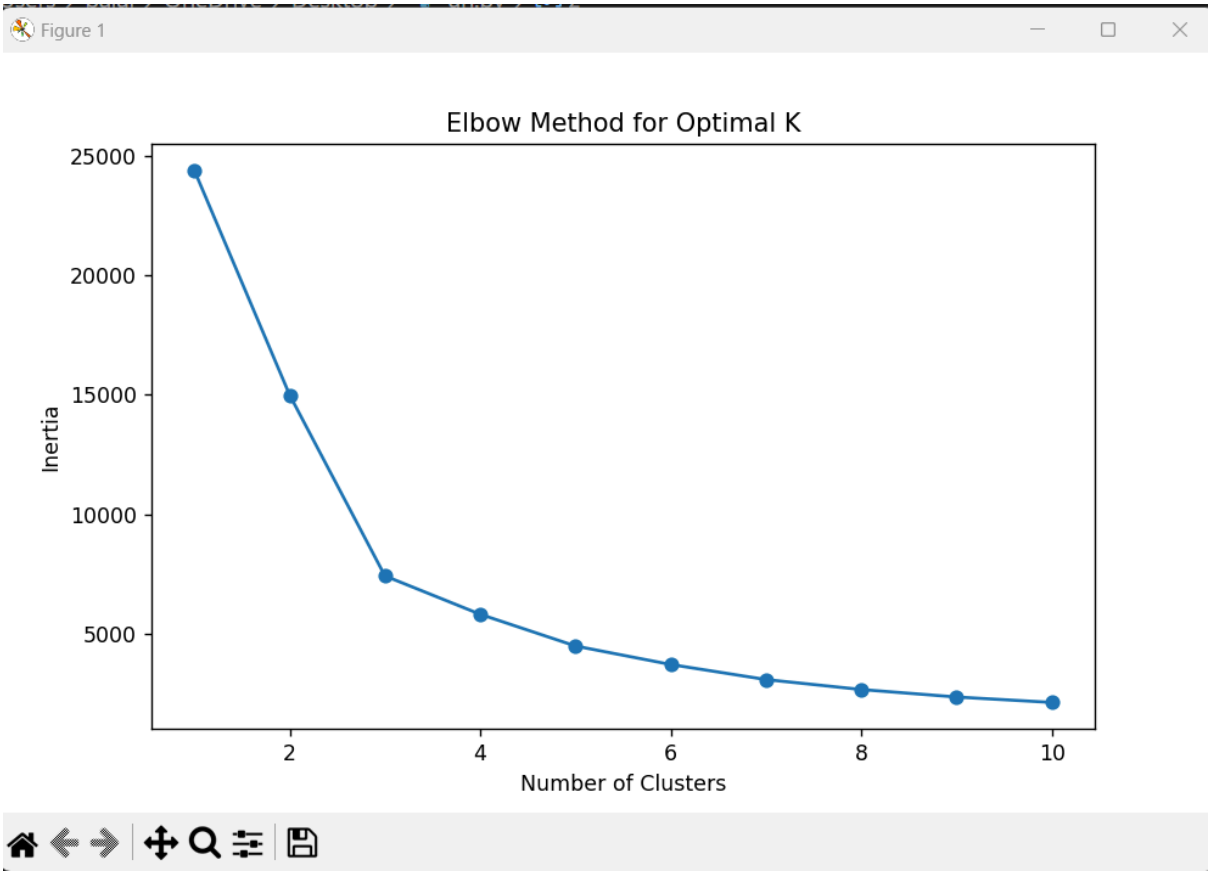
Figure 1
Elbow Method for Optimal K
Inertia
25000
20000
15000
10000
5000
2
4
6
8
10
Number of Clusters