ysjjrfdgo

January 26, 2025

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import davies_bouldin_score, silhouette_score
from sklearn.decomposition import PCA
```

[4]:
```
customers = pd.read_csv(r"C:\Users\vishn\Downloads\Customers - Customers.csv")
transactions = pd.read_csv(r"C:\Users\vishn\Downloads\Transactions -␣
 ↪Transactions.csv")
```

[5]:
```
customers.head()
```

[5]:
```
  CustomerID        CustomerName         Region  SignupDate
0      C0001    Lawrence Carroll  South America  2022-07-10
1      C0002      Elizabeth Lutz           Asia  2022-02-13
2      C0003      Michael Rivera  South America  2024-03-07
3      C0004  Kathleen Rodriguez  South America  2022-10-09
4      C0005         Laura Weber           Asia  2022-08-15
```

[6]:
```
transactions.head()
```

[6]:
```
  TransactionID CustomerID ProductID      TransactionDate  Quantity  \
0        T00001      C0199      P067  2024-08-25 12:38:23         1
1        T00112      C0146      P067  2024-05-27 22:23:54         1
2        T00166      C0127      P067   2024-04-25 7:38:55         1
3        T00272      C0087      P067  2024-03-26 22:55:37         2
4        T00363      C0070      P067  2024-03-21 15:10:10         3

   TotalValue   Price
0      300.68  300.68
1      300.68  300.68
2      300.68  300.68
3      601.36  300.68
4      902.04  300.68
```

[7]:

```
transactions['TransactionDate'] = pd.
 ↪to_datetime(transactions['TransactionDate'], errors='coerce')
```

[8]:

```
transaction_agg = transactions.groupby('CustomerID').agg({
    'TotalValue': 'sum',  # Total spending
    'TransactionID': 'count',  # Transaction frequency
    'TransactionDate': lambda x: (pd.Timestamp.now() - x.max()).days  # Recency␣
 ↪(days since last purchase)
}).rename(columns={
    'TotalValue': 'Total_Spending',
    'TransactionID': 'Frequency',
    'TransactionDate': 'Recency'
}).reset_index()
```

[9]:

```
data = customers.merge(transaction_agg, on='CustomerID', how='left')
```

[10]:

```
data.fillna({'Total_Spending': 0, 'Frequency': 0, 'Recency': data['Recency'].
 ↪max()}, inplace=True)
```

[11]:

```
scaler = StandardScaler()
```

[12]:

```
features = ['Total_Spending', 'Frequency', 'Recency']
data_scaled = scaler.fit_transform(data[features])
```

[22]:

```
inertia = []
range_n_clusters = range(2, 16)
for n_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    kmeans.fit(data_scaled)
    inertia.append(kmeans.inertia_)
```

```
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
```

```
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
```

```
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
```

```
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

[24]:

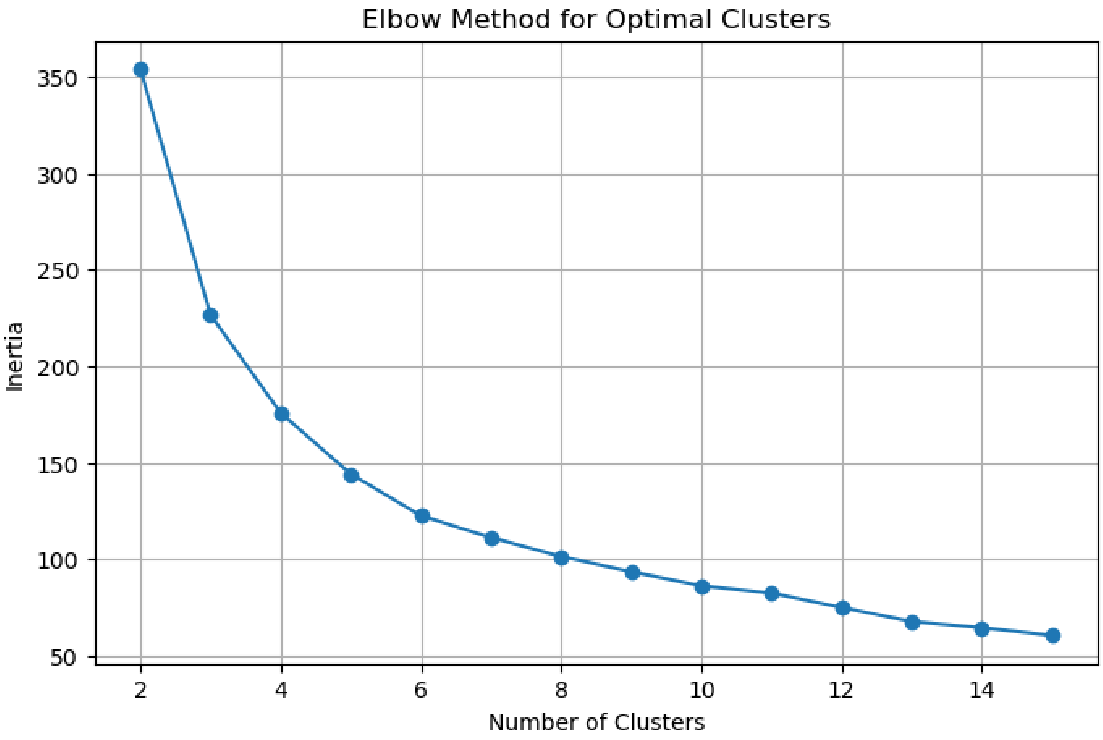
```
num_clusters = 15  # You can choose this based on the Elbow Method
kmeans = KMeans(n_clusters=num_clusters, random_state=42)
data['Cluster'] = kmeans.fit_predict(data_scaled)
```

```
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\vishn\anaconda3\new files\Lib\site-
packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a
memory leak on Windows with MKL, when there are less chunks than available
threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

[25]:

```
plt.figure(figsize=(8, 5))
plt.plot(range_n_clusters, inertia, marker='o')
plt.title("Elbow Method for Optimal Clusters")
```

```python
plt.xlabel("Number of Clusters")
plt.ylabel("Inertia")
plt.grid()
plt.show()
```



[26]:

```python
db_index = davies_bouldin_score(data_scaled, data['Cluster'])
silhouette_avg = silhouette_score(data_scaled, data['Cluster'])
print(f"Davies-Bouldin Index: {db_index}")
print(f"Silhouette Score: {silhouette_avg}")
```

```
Davies-Bouldin Index: 1.002150176920254
Silhouette Score: 0.2751035434966797
```

[14]:

```python
cluster_summary = data.groupby('Cluster').agg({
    'Total_Spending': ['mean', 'median', 'std'],
    'Frequency': ['mean', 'median', 'std'],
    'Recency': ['mean', 'median', 'std'],
    'CustomerID': 'count'
}).rename(columns={'CustomerID': 'Count'})
```

[15]:

```python
cluster_summary.columns = ['_'.join(col).strip() for col in cluster_summary.
 ↪columns]
```

[16]:

```
cluster_summary.to_csv("Cluster_Summary.csv", index=True)
```

[17]:

```
output_dir = "Cluster_Visualizations/"
import os
os.makedirs(output_dir, exist_ok=True)
```

[18]:

```
for feature in features:
    plt.figure(figsize=(10, 6))
    sns.boxplot(x='Cluster', y=feature, data=data, palette='Set3')
    plt.title(f"{feature} Distribution Across Clusters")
    plt.xlabel("Cluster")
    plt.ylabel(feature)
    plt.grid()
    plt.savefig(f"{output_dir}{feature}_Boxplot.png")
    plt.close()
```

[19]:

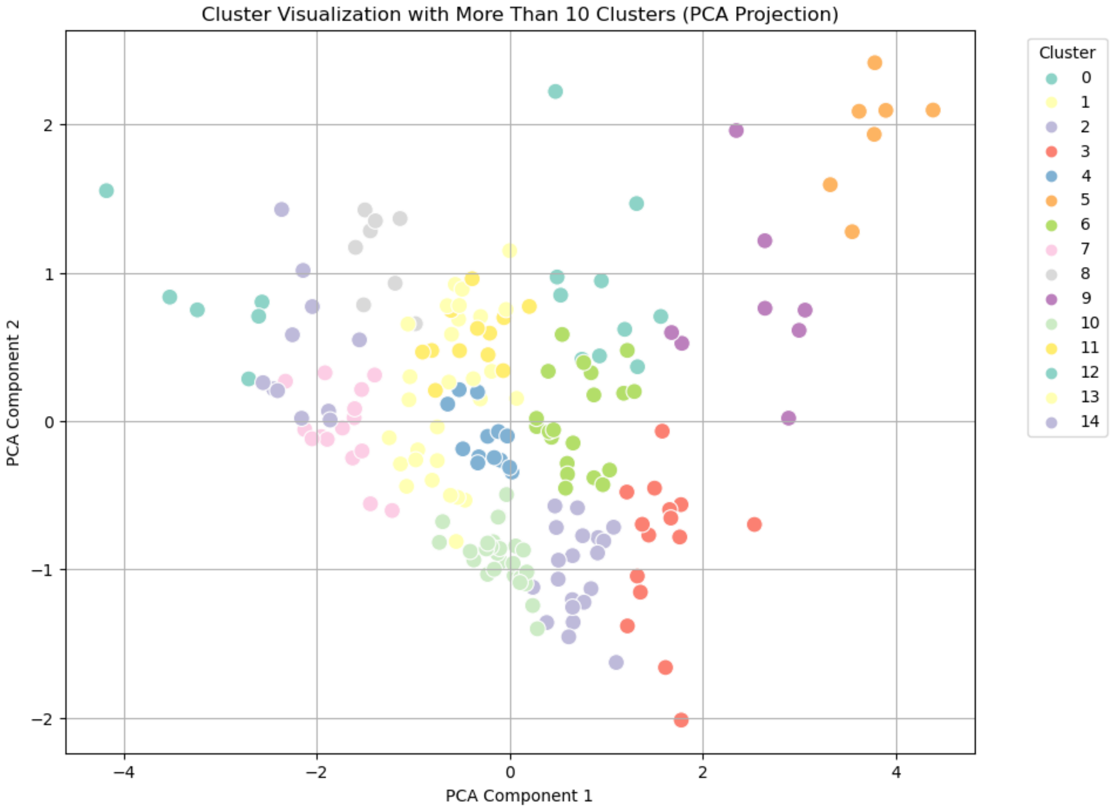
```
plt.figure(figsize=(8, 5))
sns.countplot(x='Cluster', data=data, palette='Set3')
plt.title("Cluster Sizes")
plt.xlabel("Cluster")
plt.ylabel("Number of Customers")
plt.grid()
plt.savefig(f"{output_dir}Cluster_Sizes.png")
plt.close()
```

[20]:

```
pca = PCA(n_components=2)
data_pca = pca.fit_transform(data_scaled)
data['PCA1'] = data_pca[:, 0]
data['PCA2'] = data_pca[:, 1]
```

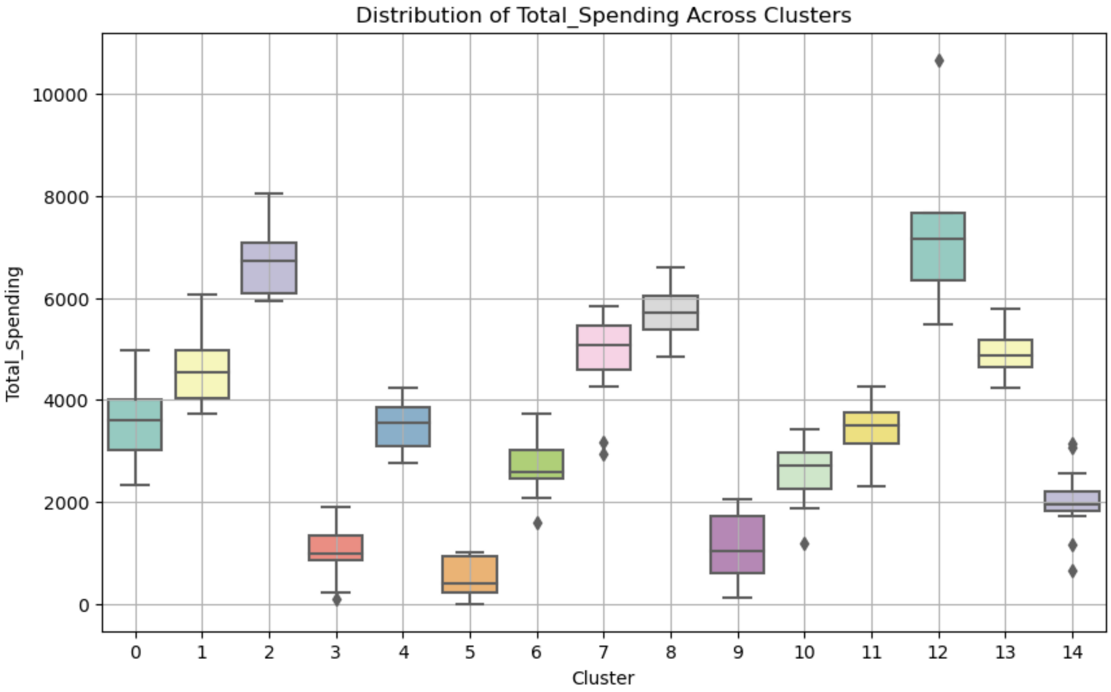
[27]:

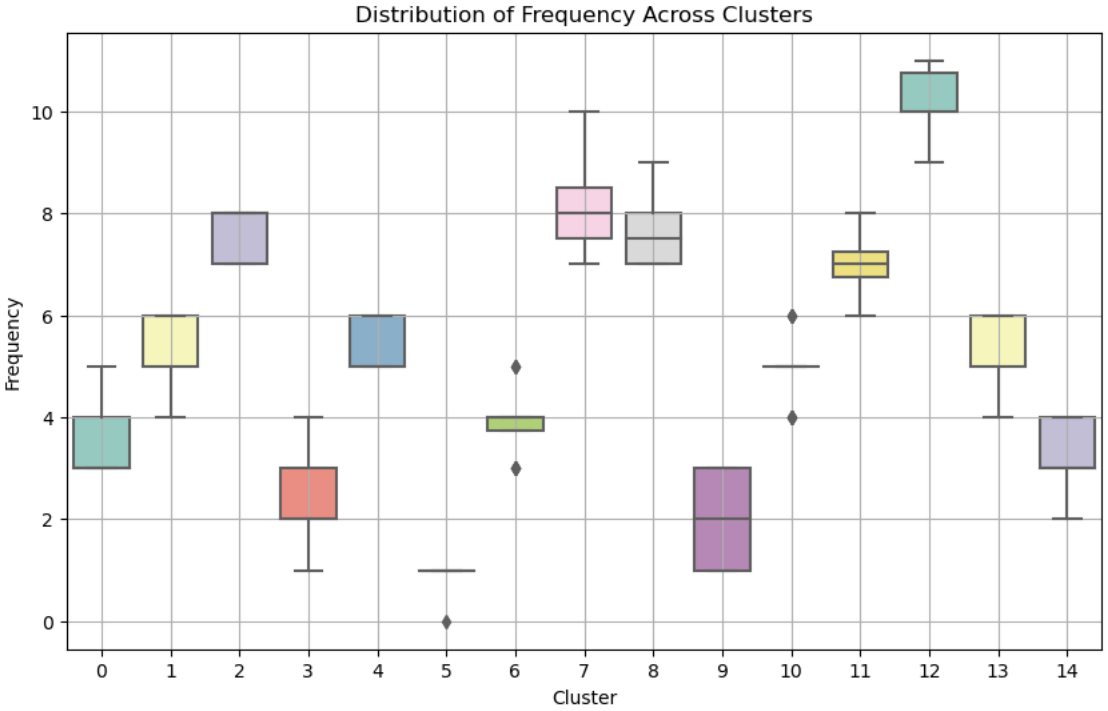
```
plt.figure(figsize=(10, 8))
sns.scatterplot(x='PCA1', y='PCA2', hue='Cluster', data=data, palette='Set3',␣
  ↪s=100)
plt.title("Cluster Visualization with More Than 10 Clusters (PCA Projection)")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.legend(title='Cluster', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid()
plt.show()
```

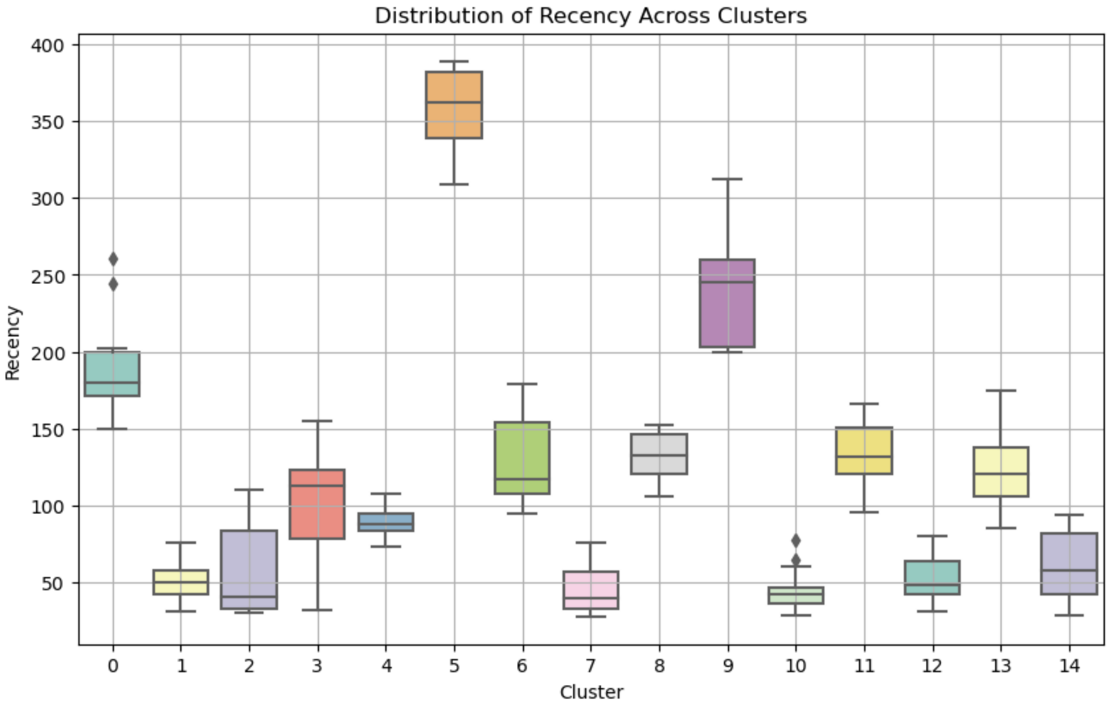


[28]:

```
features_with_cluster = features + ['Cluster']
for feature in features:
    plt.figure(figsize=(10, 6))
    sns.boxplot(x='Cluster', y=feature, data=data, palette='Set3')
    plt.title(f"Distribution of {feature} Across Clusters")
    plt.xlabel("Cluster")
    plt.ylabel(feature)
    plt.grid()
    plt.show()
```


Distribution of Total_Spending Across Clusters
Total_Spending
0
2000
4000
6000
8000
10000
Cluster
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14


Distribution of Frequency Across Clusters
Frequency
0
2
4
6
8
10
Cluster
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14


Distribution of Recency Across Clusters
Recency
400
350
300
250
200
150
100
50
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Cluster

```
[29]:
```

```
[ ]:
```