ml-practical-6-piyusha

October 9, 2025

Piyusha Supe (BE-B-23CO315)

LP3_ML_Practical_6 Implement K-Means clustering/ hierarchical clustering on sales_data_sample.csv dataset. Determine the number of clusters using the elbow method. Dataset link : https://www.kaggle.com/datasets/kyanyoga/sample-sales-data

[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 scipy.cluster.hierarchy import linkage, dendrogram, fcluster
```

[5]:
```
# Load dataset
df = pd.read_csv("/content/sales_data_sample.csv", encoding='unicode_escape')
df.head()
```

[5]:
```
   ORDERNUMBER  QUANTITYORDERED  PRICEEACH  ORDERLINENUMBER    SALES  \
0        10107               30      95.70                2  2871.00
1        10121               34      81.35                5  2765.90
2        10134               41      94.74                2  3884.34
3        10145               45      83.26                6  3746.70
4        10159               49     100.00               14  5205.27

         ORDERDATE   STATUS  QTR_ID  MONTH_ID  YEAR_ID  …  \
0   2/24/2003 0:00  Shipped       1         2     2003  …
1    5/7/2003 0:00  Shipped       2         5     2003  …
2    7/1/2003 0:00  Shipped       3         7     2003  …
3   8/25/2003 0:00  Shipped       3         8     2003  …
4  10/10/2003 0:00  Shipped       4        10     2003  …

                    ADDRESSLINE1  ADDRESSLINE2           CITY STATE  \
0        897 Long Airport Avenue           NaN            NYC    NY
1             59 rue de l'Abbaye           NaN          Reims   NaN
2  27 rue du Colonel Pierre Avia           NaN          Paris   NaN
3             78934 Hillside Dr.           NaN       Pasadena    CA
```

```
4               7734 Strong St.           NaN  San Francisco    CA

  POSTALCODE COUNTRY TERRITORY CONTACTLASTNAME CONTACTFIRSTNAME DEALSIZE
0      10022     USA       NaN              Yu             Kwai    Small
1      51100  France      EMEA         Henriot             Paul    Small
2      75508  France      EMEA        Da Cunha           Daniel   Medium
3      90003     USA       NaN           Young            Julie   Medium
4        NaN     USA       NaN           Brown            Julie   Medium

[5 rows x 25 columns]
```

[6]:

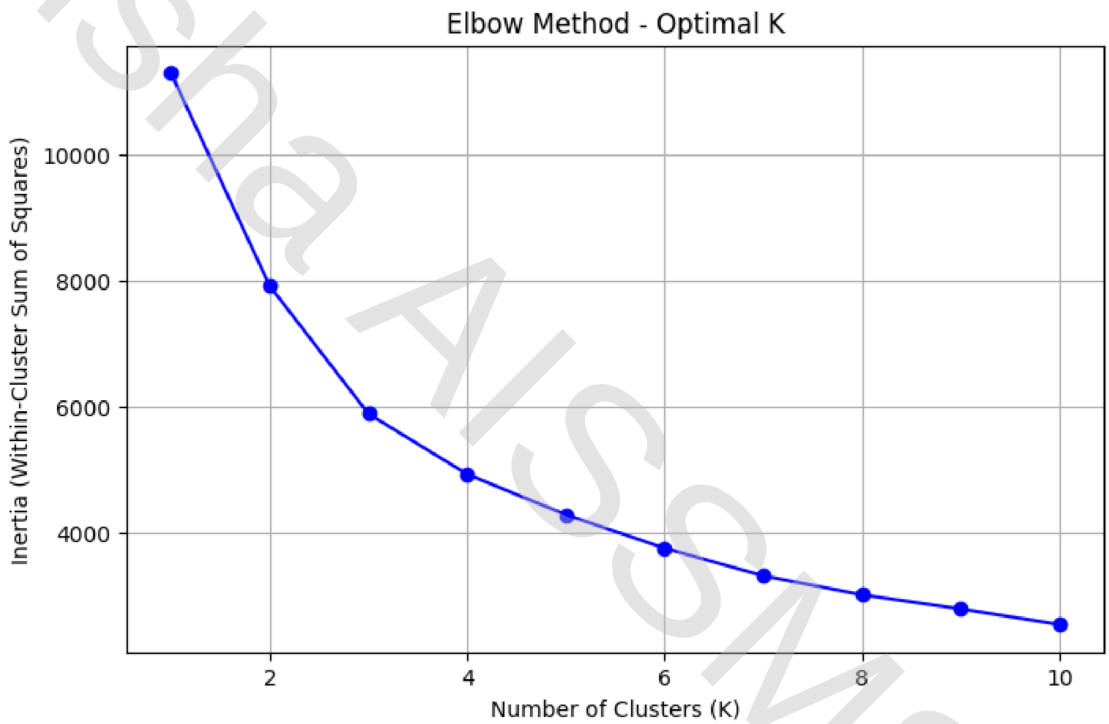
```python
# Basic info
print(df.info())
print(df.describe())
print(df.columns)

# Drop non-numeric and irrelevant columns
df_clean = df.select_dtypes(include=[np.number]).dropna()

# Alternatively, choose a few important numerical features
features = df_clean[['QUANTITYORDERED', 'PRICEEACH', 'SALES', 'ORDERNUMBER']]

# Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(features)

print("Data shape after scaling:", X_scaled.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 25 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   ORDERNUMBER       2823 non-null   int64
 1   QUANTITYORDERED   2823 non-null   int64
 2   PRICEEACH         2823 non-null   float64
 3   ORDERLINENUMBER   2823 non-null   int64
 4   SALES             2823 non-null   float64
 5   ORDERDATE         2823 non-null   object
 6   STATUS            2823 non-null   object
 7   QTR_ID            2823 non-null   int64
 8   MONTH_ID          2823 non-null   int64
 9   YEAR_ID           2823 non-null   int64
 10  PRODUCTLINE       2823 non-null   object
 11  MSRP              2823 non-null   int64
 12  PRODUCTCODE       2823 non-null   object
 13  CUSTOMERNAME      2823 non-null   object
```

```
 14  PHONE             2823 non-null   object
 15  ADDRESSLINE1      2823 non-null   object
 16  ADDRESSLINE2      302 non-null    object
 17  CITY              2823 non-null   object
 18  STATE             1337 non-null   object
 19  POSTALCODE        2747 non-null   object
 20  COUNTRY           2823 non-null   object
 21  TERRITORY         1749 non-null   object
 22  CONTACTLASTNAME   2823 non-null   object
 23  CONTACTFIRSTNAME  2823 non-null   object
 24  DEALSIZE          2823 non-null   object
dtypes: float64(2), int64(7), object(16)
memory usage: 551.5+ KB
None
        ORDERNUMBER  QUANTITYORDERED    PRICEEACH  ORDERLINENUMBER  \
count   2823.000000      2823.000000  2823.000000      2823.000000
mean   10258.725115        35.092809    83.658544         6.466171
std       92.085478         9.741443    20.174277         4.225841
min    10100.000000         6.000000    26.880000         1.000000
25%    10180.000000        27.000000    68.860000         3.000000
50%    10262.000000        35.000000    95.700000         6.000000
75%    10333.500000        43.000000   100.000000         9.000000
max    10425.000000        97.000000   100.000000        18.000000

              SALES       QTR_ID     MONTH_ID     YEAR_ID         MSRP
count   2823.000000  2823.000000  2823.000000  2823.00000  2823.000000
mean    3553.889072     2.717676     7.092455  2003.81509   100.715551
std     1841.865106     1.203878     3.656633     0.69967    40.187912
min      482.130000     1.000000     1.000000  2003.00000    33.000000
25%     2203.430000     2.000000     4.000000  2003.00000    68.000000
50%     3184.800000     3.000000     8.000000  2004.00000    99.000000
75%     4508.000000     4.000000    11.000000  2004.00000   124.000000
max    14082.800000     4.000000    12.000000  2005.00000   214.000000
Index(['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'ORDERLINENUMBER',
       'SALES', 'ORDERDATE', 'STATUS', 'QTR_ID', 'MONTH_ID', 'YEAR_ID',
       'PRODUCTLINE', 'MSRP', 'PRODUCTCODE', 'CUSTOMERNAME', 'PHONE',
       'ADDRESSLINE1', 'ADDRESSLINE2', 'CITY', 'STATE', 'POSTALCODE',
       'COUNTRY', 'TERRITORY', 'CONTACTLASTNAME', 'CONTACTFIRSTNAME',
       'DEALSIZE'],
      dtype='object')
Data shape after scaling: (2823, 4)
```

[7]:
```python
inertia = []
K = range(1, 11)

for k in K:
    model = KMeans(n_clusters=k, random_state=42)
```

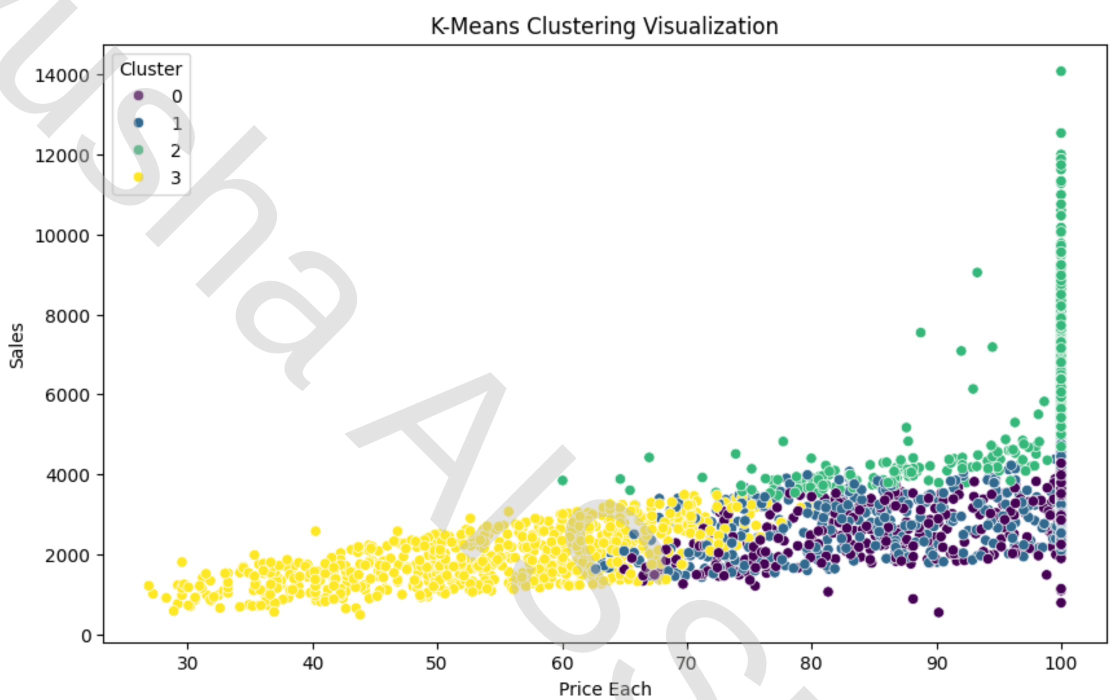
```
    model.fit(X_scaled)
    inertia.append(model.inertia_)

# Plot Elbow Curve
plt.figure(figsize=(8,5))
plt.plot(K, inertia, marker='o', color='blue')
plt.title("Elbow Method - Optimal K")
plt.xlabel("Number of Clusters (K)")
plt.ylabel("Inertia (Within-Cluster Sum of Squares)")
plt.grid(True)
plt.show()
```



[8]:
```
# Choose K based on elbow result (example: K=4)
kmeans = KMeans(n_clusters=4, random_state=42)
df['Cluster'] = kmeans.fit_predict(X_scaled)

# Cluster centers
print("Cluster Centers:\n", kmeans.cluster_centers_)
```

```
Cluster Centers:
 [[-0.70854061  0.46259617 -0.26929189  0.82525847]
 [-0.43685254  0.41359923 -0.13496959 -1.01228217]
 [ 1.01181348  0.67408353  1.27088991  0.22337467]
 [ 0.04001807 -1.44645947 -0.87598836  0.05091556]]
```

[9]:
```
plt.figure(figsize=(10,6))
sns.scatterplot(x='PRICEEACH', y='SALES', hue='Cluster', data=df,␣
 ↪palette='viridis')
plt.title('K-Means Clustering Visualization')
plt.xlabel('Price Each')
plt.ylabel('Sales')
plt.legend(title='Cluster')
plt.show()
```



[10]:
```
cluster_summary = df.groupby('Cluster')[['SALES', 'PRICEEACH',␣
 ↪'QUANTITYORDERED']].mean()
print("Cluster Summary:\n", cluster_summary)
```

```
Cluster Summary:
                SALES  PRICEEACH  QUANTITYORDERED
Cluster
0        3056.839670  93.000440        28.183673
1        3304.355601  91.949036        30.842179
2        5894.282221  97.255283        44.947586
3        1941.139275  54.471409        35.495302
```

[ ]: