In [ ]:
```
Name : Hanuman bavane
Roll No : 14108
```

In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [5]:
```
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
```

In [7]:
```
df = pd.read_csv("sales_data_sample.csv", encoding ="Latin-1")
```

In [ ]:
```

```

In [10]:
```
df.head()
```

Out[10]:

| | ORDERNUMBER | QUANTITYORDERED | PRICEEACH | ORDERLINENUMBER | SALES | OR |
|---|---|---|---|---|---|---|
| **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 | 1 |

5 rows × 25 columns

In [12]:
```
df.shape
```

Out[12]:
```
(2823, 25)
```

In [14]:
```
df.describe()
```

Out[14]:

| | ORDERNUMBER | QUANTITYORDERED | PRICEEACH | ORDERLINENUMBER | |
|---|---|---|---|---|---|
| **count** | 2823.000000 | 2823.000000 | 2823.000000 | 2823.000000 | 2823.0 |
| **mean** | 10258.725115 | 35.092809 | 83.658544 | 6.466171 | 3553.8 |
| **std** | 92.085478 | 9.741443 | 20.174277 | 4.225841 | 1841.8 |
| **min** | 10100.000000 | 6.000000 | 26.880000 | 1.000000 | 482.1 |
| **25%** | 10180.000000 | 27.000000 | 68.860000 | 3.000000 | 2203.4 |
| **50%** | 10262.000000 | 35.000000 | 95.700000 | 6.000000 | 3184.8 |
| **75%** | 10333.500000 | 43.000000 | 100.000000 | 9.000000 | 4508.0 |
| **max** | 10425.000000 | 97.000000 | 100.000000 | 18.000000 | 14082.8 |

In [16]:

```
df.info()
```

```
<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
```

In [18]:

```
df.info()
```

```
<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
```

In [20]:
```
df.isnull().sum()
```

```
Out[20]: ORDERNUMBER            0
         QUANTITYORDERED        0
         PRICEEACH              0
         ORDERLINENUMBER        0
         SALES                  0
         ORDERDATE              0
         STATUS                 0
         QTR_ID                 0
         MONTH_ID               0
         YEAR_ID                0
         PRODUCTLINE            0
         MSRP                   0
         PRODUCTCODE            0
         CUSTOMERNAME           0
         PHONE                  0
         ADDRESSLINE1           0
         ADDRESSLINE2        2521
         CITY                   0
         STATE               1486
         POSTALCODE            76
         COUNTRY                0
         TERRITORY           1074
         CONTACTLASTNAME        0
         CONTACTFIRSTNAME       0
         DEALSIZE               0
         dtype: int64
```

```
In [22]: df.dtypes
```

```
Out[22]: ORDERNUMBER            int64
         QUANTITYORDERED        int64
         PRICEEACH            float64
         ORDERLINENUMBER        int64
         SALES                float64
         ORDERDATE             object
         STATUS                object
         QTR_ID                 int64
         MONTH_ID               int64
         YEAR_ID                int64
         PRODUCTLINE           object
         MSRP                   int64
         PRODUCTCODE           object
         CUSTOMERNAME          object
         PHONE                 object
         ADDRESSLINE1          object
         ADDRESSLINE2          object
         CITY                  object
         STATE                 object
         POSTALCODE            object
         COUNTRY               object
         TERRITORY             object
         CONTACTLASTNAME       object
         CONTACTFIRSTNAME      object
         DEALSIZE              object
         dtype: object
```

```
In [26]: df_drop = ['ADDRESSLINE1', 'ADDRESSLINE2', 'STATUS', 'POSTALCODE', 'CITY']
```

```
In [28]: df = df.drop(df_drop, axis=1)
```

In [30]:
```
df.isnull().sum()
```

Out[30]:
```
ORDERNUMBER            0
QUANTITYORDERED        0
PRICEEACH              0
ORDERLINENUMBER        0
SALES                  0
ORDERDATE              0
QTR_ID                 0
MONTH_ID               0
YEAR_ID                0
PRODUCTLINE            0
MSRP                   0
PRODUCTCODE            0
CUSTOMERNAME           0
PHONE                  0
STATE               1486
COUNTRY                0
TERRITORY           1074
CONTACTLASTNAME        0
CONTACTFIRSTNAME       0
DEALSIZE               0
dtype: int64
```

In [32]:
```
df['COUNTRY'].unique()
```

Out[32]:
```
array(['USA', 'France', 'Norway', 'Australia', 'Finland', 'Austria', 'UK',
       'Spain', 'Sweden', 'Singapore', 'Canada', 'Japan', 'Italy',
       'Denmark', 'Belgium', 'Philippines', 'Germany', 'Switzerland',
       'Ireland'], dtype=object)
```

In [34]:
```
 df['PRODUCTLINE'].unique()
```

Out[34]:
```
array(['Motorcycles', 'Classic Cars', 'Trucks and Buses', 'Vintage Cars',
       'Planes', 'Ships', 'Trains'], dtype=object)
```

In [36]:
```
df['DEALSIZE'].unique()
```

Out[36]:
```
array(['Small', 'Medium', 'Large'], dtype=object)
```

In [40]:
```
productline = pd.get_dummies(df['PRODUCTLINE'])
Dealsize = pd.get_dummies(df['DEALSIZE'])
```

In [43]:
```
df = pd.concat([df, productline,Dealsize],axis=1)
```

In [45]:
```
df_drop = ['COUNTRY', 'PRODUCTLINE', 'DEALSIZE']
df = df.drop(df_drop, axis =1 )
```

In [47]:
```
 df['PRODUCTCODE'] = pd.Categorical(df[ 'PRODUCTCODE']).codes
```

In [49]:
```
df.drop('ORDERDATE', axis = 1, inplace=True)
```

In [51]:
```
df.dtypes
```

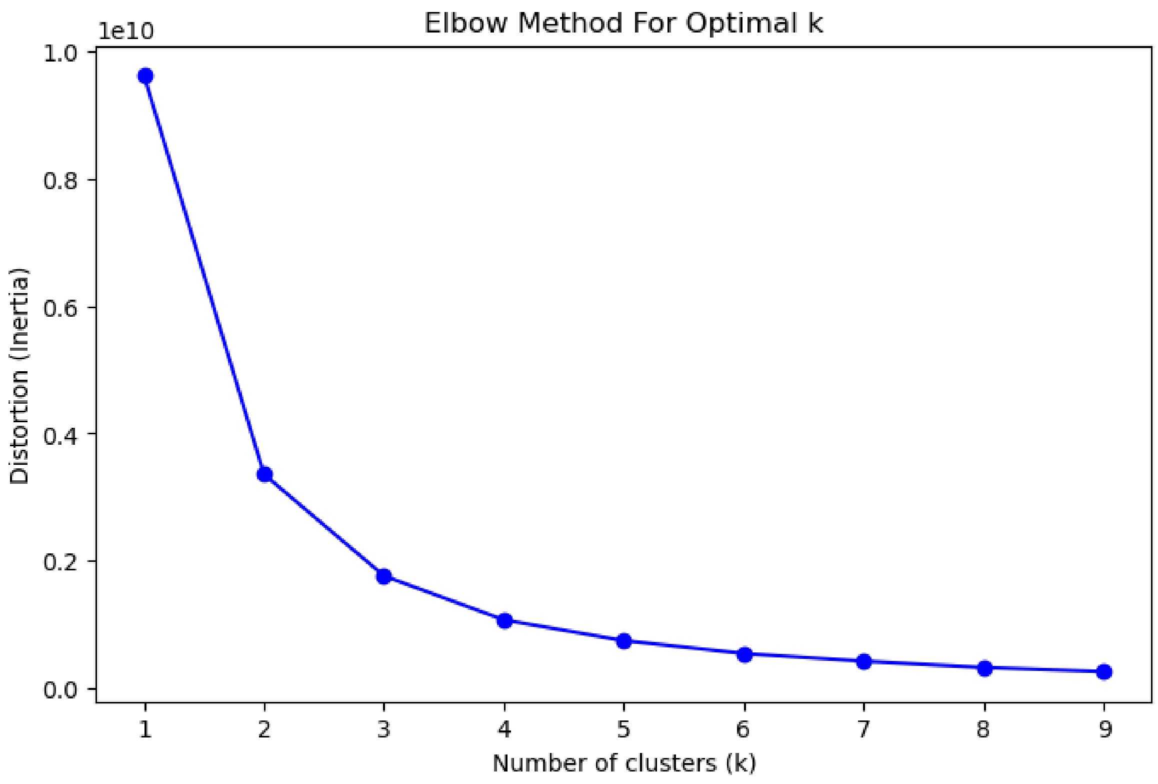
```
Out[51]: ORDERNUMBER            int64
         QUANTITYORDERED        int64
         PRICEEACH            float64
         ORDERLINENUMBER        int64
         SALES                float64
         QTR_ID                 int64
         MONTH_ID               int64
         YEAR_ID                int64
         MSRP                   int64
         PRODUCTCODE             int8
         CUSTOMERNAME          object
         PHONE                 object
         STATE                 object
         TERRITORY             object
         CONTACTLASTNAME       object
         CONTACTFIRSTNAME      object
         Classic Cars            bool
         Motorcycles             bool
         Planes                  bool
         Ships                   bool
         Trains                  bool
         Trucks and Buses        bool
         Vintage Cars            bool
         Large                   bool
         Medium                  bool
         Small                   bool
         dtype: object
```

```
In [114…
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

df_encoded = df.copy()
for col in df_encoded.select_dtypes(include=['object']):
    df_encoded[col] = LabelEncoder().fit_transform(df_encoded[col])

df_encoded = df_encoded.fillna(0)
distortions = []
K = range(1, 10)

for k in K:
    kmeanModel = KMeans(n_clusters=k, random_state=42)
    kmeanModel.fit(df_encoded)
    distortions.append(kmeanModel.inertia_)  # Fixed spacing
plt.figure(figsize=(8,5))
plt.plot(K, distortions, 'bo-')
plt.xlabel('Number of clusters (k)')
plt.ylabel('Distortion (Inertia)')
plt.title('Elbow Method For Optimal k')
plt.show()
```



In [61]:
```
x_train = df.values
```

In [63]:
```
 x_train.shape
```

Out[63]:
```
(2823, 26)
```

In [112…
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.cluster import KMeans
if not isinstance(x_train, pd.DataFrame):
    x_train = pd.DataFrame(x_train)

for col in x_train.select_dtypes(include=['object']):
    x_train[col] = LabelEncoder().fit_transform(x_train[col])
x_train = x_train.fillna(0)
model = KMeans(n_clusters=3, random_state=2)
model.fit(x_train)
predictions = model.predict(x_train)
x_train['Cluster'] = predictions
print(x_train.head())
```

```
    0   1     2   3     4  5  6  7   8  9  ...  17  18  19  20  21  22  23  \
0   7  19   943   1  1176  0  1  0  39  0  ...   1   0   0   0   0   0   0
1  21  23   696   4  1091  1  4  0  39  0  ...   1   0   0   0   0   0   0
2  33  30   928   1  1800  2  6  0  39  0  ...   1   0   0   0   0   0   0
3  43  34   735   5  1723  2  7  0  39  0  ...   1   0   0   0   0   0   0
4  56  38  1015  13  2284  3  9  0  39  0  ...   1   0   0   0   0   0   0

   24  25  Cluster
0   0   1        2
1   0   1        2
2   1   0        2
3   1   0        2
4   1   0        1

[5 rows x 27 columns]
```

In [82]:
```python
unique, counts = np.unique(predictions, return_counts=True)
```

In [84]:
```python
counts = counts.reshape(1,3)
```

In [88]:
```python
counts_df = pd.DataFrame(counts, columns=['Cluster', 'Cluster2', 'Cluster3'])
```

In [90]:
```python
counts_df.head()
```

Out[90]:

| | Cluster | Cluster2 | Cluster3 |
|---|---|---|---|
| **0** | 886 | 971 | 966 |

In [92]:
```python
pca = PCA(n_components=2)
```

In [110…
```python
import pandas as pd
from sklearn.decomposition import PCA

# If x_train is a NumPy array, convert to DataFrame first
if not isinstance(x_train, pd.DataFrame):
    x_train = pd.DataFrame(x_train)

x_train.columns = x_train.columns.astype(str)
x_train = x_train.apply(pd.to_numeric, errors='coerce').fillna(0)
pca = PCA(n_components=2)
reduced_X = pd.DataFrame(pca.fit_transform(x_train), columns=['PCA1', 'PCA2'])
reduced_X.head()
```

Out[110…

| | PCA1 | PCA2 |
|---|---|---|
| **0** | -128.681718 | 258.430435 |
| **1** | -288.637142 | 51.966292 |
| **2** | 457.159455 | 42.470316 |
| **3** | 322.179951 | -115.634814 |
| **4** | 943.332505 | -31.903135 |

In [ ]:

In [ ]:
```
```
In [ ]:
```
```