Machine Learning - Assignment 6

In [ ]:
```
# Implement K-Means clustering/ hierarchical clustering on sales_data_sample.
# Determine the number of clusters using the elbow method.
```

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

In [ ]:
```
data = pd.read_csv("sales_data_sample.csv", encoding='Latin-1')
data.head()

# While utf-8 supports all languages according to pandas' documentation, utf-
```

Out[ ]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	C
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	

5 rows × 25 columns

In [ ]:
```
data.shape
```

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

In [ ]:
```
# Number of NAN values per column in the dataset
data.isnull().sum()
```

Out[ ]:
```
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 [ ]:
```
data.drop(["ORDERNUMBER", "PRICEEACH", "ORDERDATE", "PHONE", "ADDRESSLINE1",
```

In [ ]:
```
data.head()
```

Out[ ]:

	QUANTITYORDERED	ORDERLINENUMBER	SALES	STATUS	QTR_ID	MONTH_ID
0	30	2	2871.00	Shipped	1	2
1	34	5	2765.90	Shipped	2	5
2	41	2	3884.34	Shipped	3	7
3	45	6	3746.70	Shipped	3	8
4	49	14	5205.27	Shipped	4	10

In [ ]:
```
data.isnull().sum()
```

Out[ ]:
```
QUANTITYORDERED    0
ORDERLINENUMBER    0
SALES              0
STATUS             0
QTR_ID             0
MONTH_ID           0
YEAR_ID            0
PRODUCTLINE        0
MSRP               0
PRODUCTCODE        0
CUSTOMERNAME       0
COUNTRY            0
DEALSIZE           0
dtype: int64
```

Exploratary Data Analysis

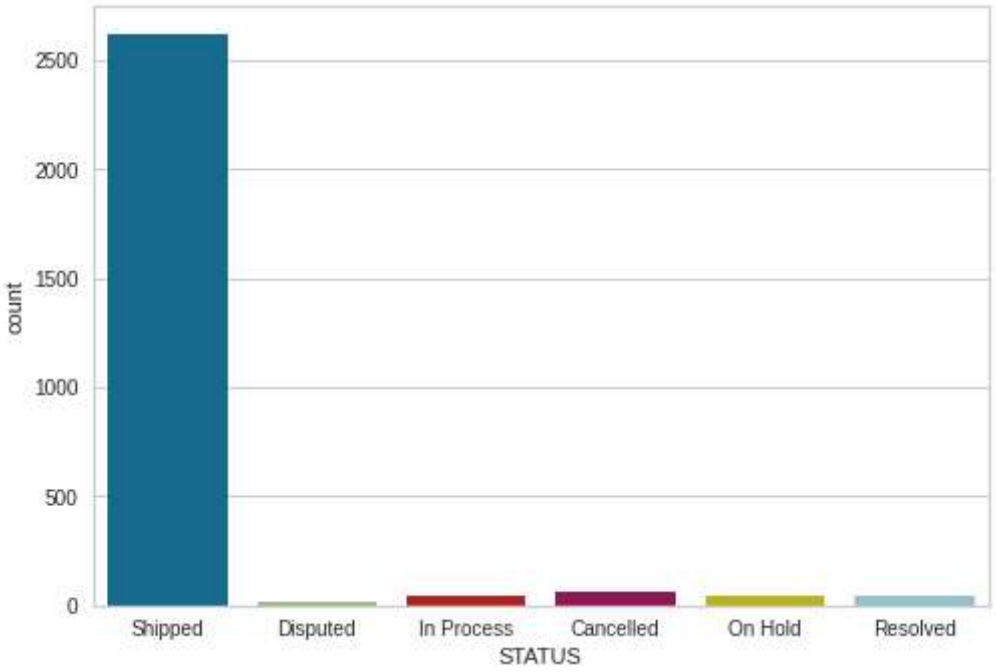
In [ ]:
```
data.describe()
```

Out[ ]:

	QUANTITYORDERED	ORDERLINENUMBER	SALES	QTR_ID	MONTH
count	2823.000000	2823.000000	2823.000000	2823.000000	2823.000
mean	35.092809	6.466171	3553.889072	2.717676	7.092
std	9.741443	4.225841	1841.865106	1.203878	3.656
min	6.000000	1.000000	482.130000	1.000000	1.000
25%	27.000000	3.000000	2203.430000	2.000000	4.000
50%	35.000000	6.000000	3184.800000	3.000000	8.000
75%	43.000000	9.000000	4508.000000	4.000000	11.000
max	97.000000	18.000000	14082.800000	4.000000	12.000

In [ ]:
```
sns.countplot(data = data , x = 'STATUS')
```

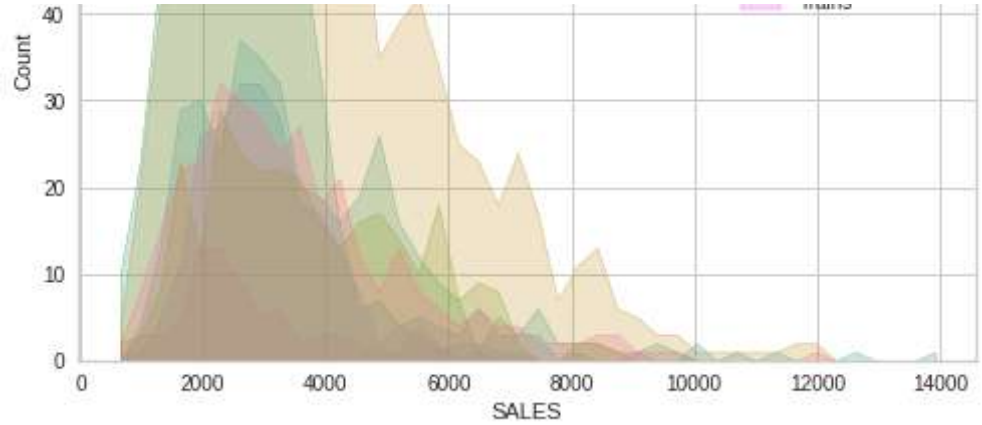
Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f27c8bbbd50>
```



In [ ]:
```
import seaborn as sns
```

In [ ]:
```
sns.histplot(x = 'SALES' , hue = 'PRODUCTLINE', data = data,
             element="poly")
```

Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f27c8b6f210>
```



Here we can see all the catagory lies in the range of price and hence in this we be creating a cluster on targeting the same

In [ ]:
```
data['PRODUCTLINE'].unique()
```

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

In [ ]:
```
#checking the duplicated values
data.drop_duplicates(inplace=True)
```

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2823 entries, 0 to 2822
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   QUANTITYORDERED  2823 non-null   int64
 1   ORDERLINENUMBER  2823 non-null   int64
 2   SALES            2823 non-null   float64
 3   STATUS           2823 non-null   object
 4   QTR_ID           2823 non-null   int64
 5   MONTH_ID         2823 non-null   int64
 6   YEAR_ID          2823 non-null   int64
 7   PRODUCTLINE      2823 non-null   object
 8   MSRP             2823 non-null   int64
 9   PRODUCTCODE      2823 non-null   object
 10  CUSTOMERNAME     2823 non-null   object
 11  COUNTRY          2823 non-null   object
 12  DEALSIZE         2823 non-null   object
dtypes: float64(1), int64(6), object(6)
memory usage: 308.8+ KB
```

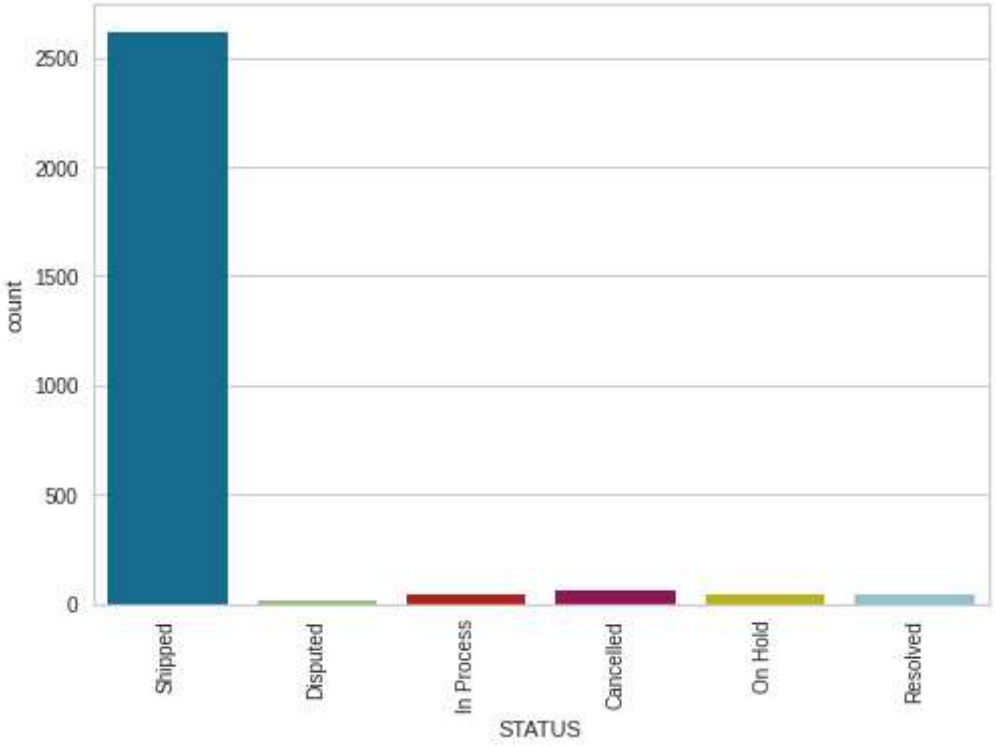
In [ ]:
```
list_cat = data.select_dtypes(include=['object']).columns.tolist()
```

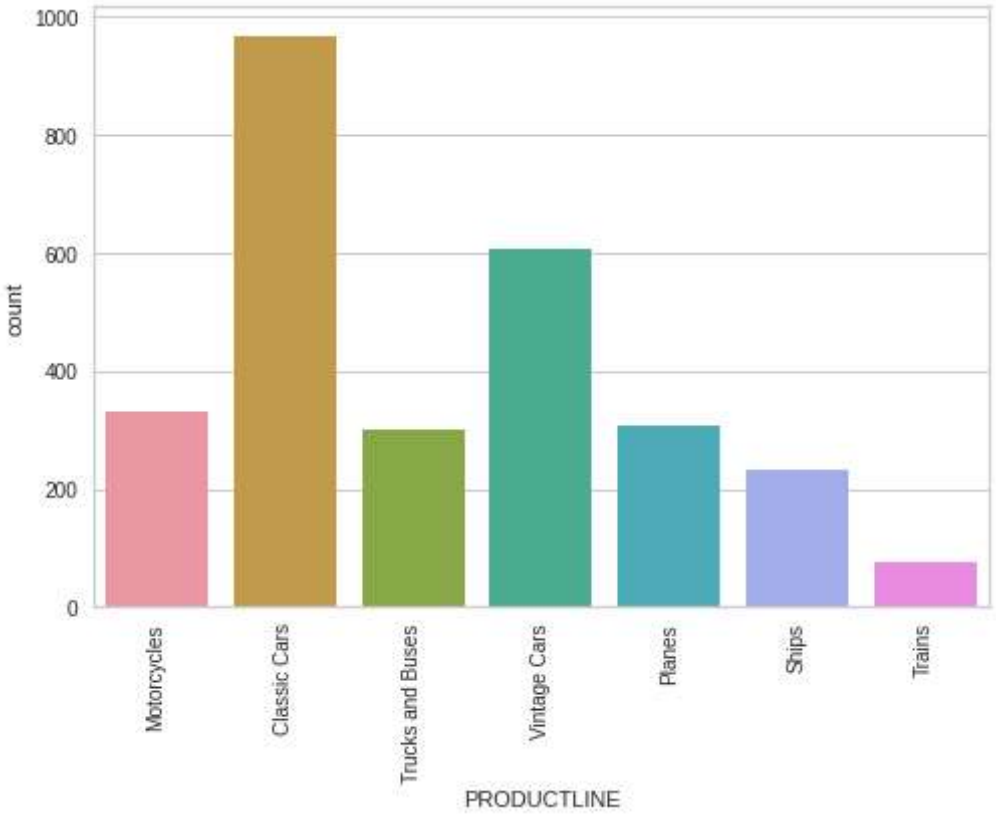
In [ ]:
```
list_cat
```

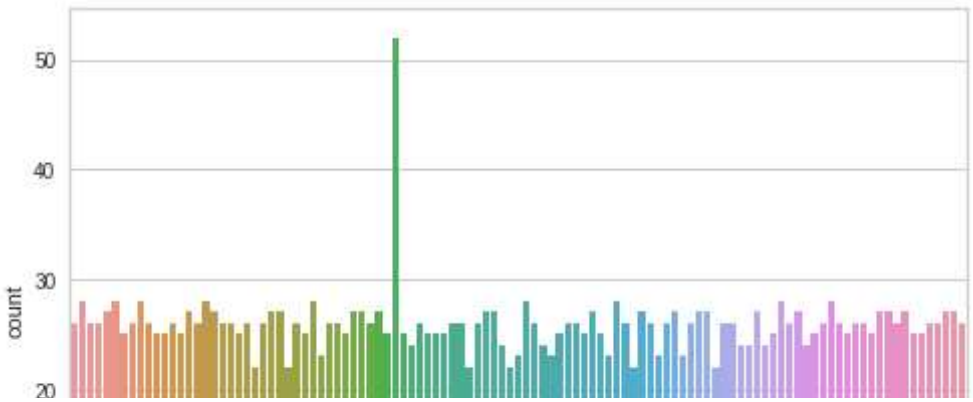
Out[ ]:
```
['STATUS', 'PRODUCTLINE', 'PRODUCTCODE', 'CUSTOMERNAME', 'COUNTRY', 'DEALSIZ
E']
```

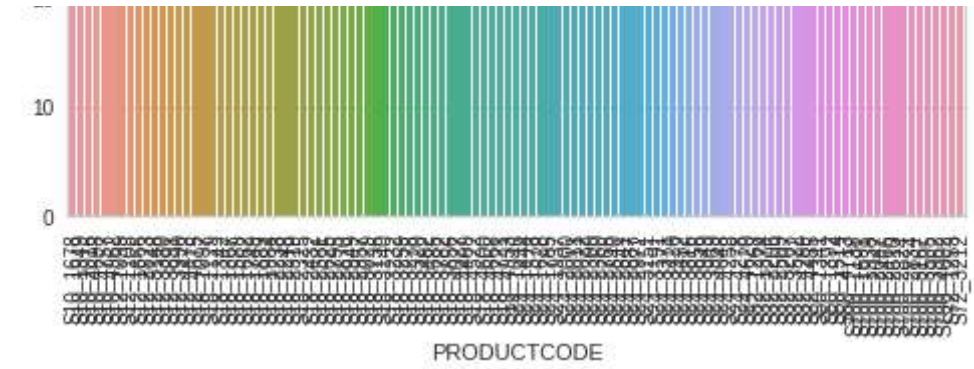
In [ ]:
```
for i in list_cat:
  sns.countplot(data = data ,x = i)
```

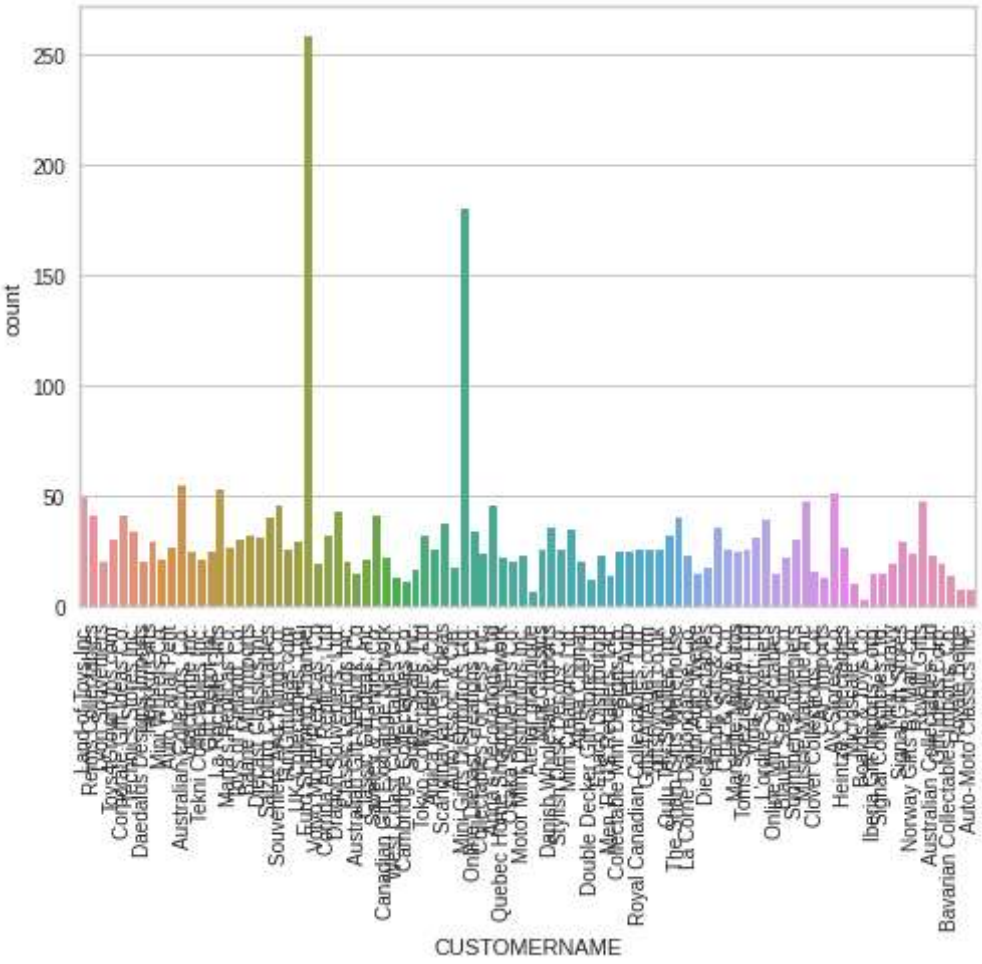
```
plt.xticks(rotation = 90)
plt.show()
```

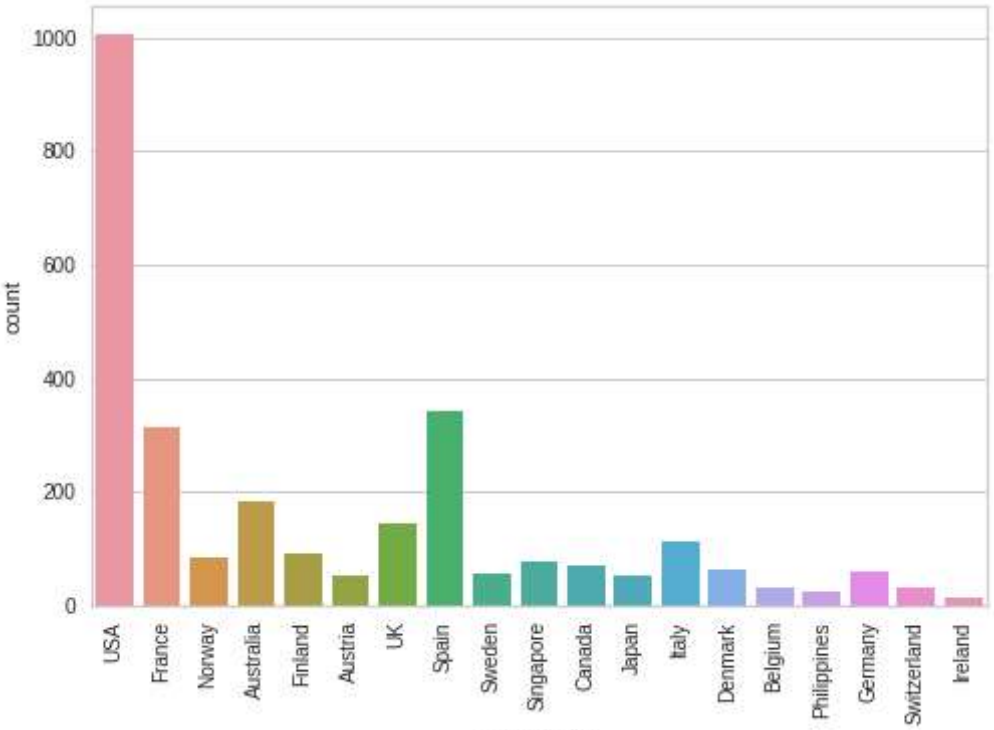


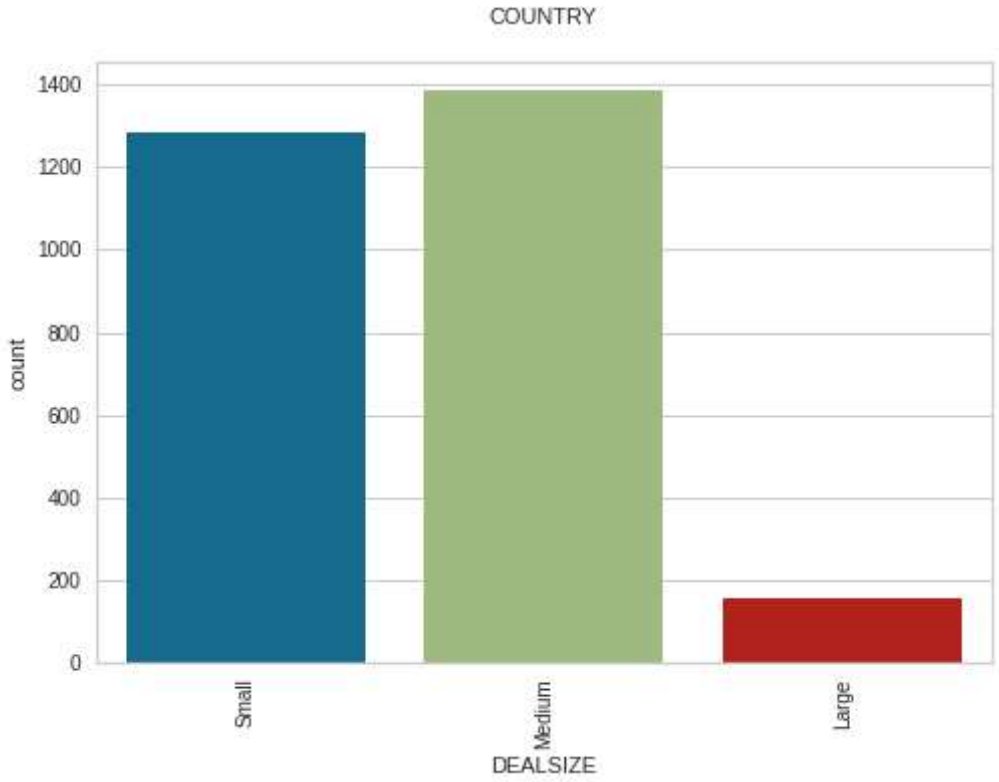





10
0
PRODUCTCODE


250
200
150
100
50
0
count
CUSTOMERNAME


1000
800
600
400
200
0
count
USA
France
Norway
Australia
Finland
Austria
UK
Spain
Sweden
Singapore
Canada
Japan
Italy
Denmark
Belgium
Philippines
Germany
Switzerland
Ireland



In [ ]:
```
#dealing with the catagorical features
from sklearn import preprocessing
le = preprocessing.LabelEncoder()

# Encode labels in column 'species'.
for i in list_cat:
  data[i]= le.fit_transform(data[i])
```

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2823 entries, 0 to 2822
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   QUANTITYORDERED  2823 non-null   int64
 1   ORDERLINENUMBER  2823 non-null   int64
 2   SALES            2823 non-null   float64
 3   STATUS           2823 non-null   int64
 4   QTR_ID           2823 non-null   int64
 5   MONTH_ID         2823 non-null   int64
 6   YEAR_ID          2823 non-null   int64
 7   PRODUCTLINE      2823 non-null   int64
 8   MSRP             2823 non-null   int64
 9   PRODUCTCODE      2823 non-null   int64
 10  CUSTOMERNAME     2823 non-null   int64
 11  COUNTRY          2823 non-null   int64
 12  DEALSIZE         2823 non-null   int64
dtypes: float64(1), int64(12)
memory usage: 373.3 KB
```

In [ ]:
```
data['SALES'] = data['SALES'].astype(int)
```

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2823 entries, 0 to 2822
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   QUANTITYORDERED  2823 non-null   int64
 1   ORDERLINENUMBER  2823 non-null   int64
 2   SALES            2823 non-null   int64
 3   STATUS           2823 non-null   int64
 4   QTR_ID           2823 non-null   int64
 5   MONTH_ID         2823 non-null   int64
 6   YEAR_ID          2823 non-null   int64
 7   PRODUCTLINE      2823 non-null   int64
 8   MSRP             2823 non-null   int64
 9   PRODUCTCODE      2823 non-null   int64
 10  CUSTOMERNAME     2823 non-null   int64
 11  COUNTRY          2823 non-null   int64
 12  DEALSIZE         2823 non-null   int64
dtypes: int64(13)
memory usage: 373.3 KB
```

In [ ]:
```
data.describe()
```

Out[ ]:

	QUANTITYORDERED	ORDERLINENUMBER	SALES	STATUS	QTR
count	2823.000000	2823.000000	2823.000000	2823.000000	2823.000
mean	35.092809	6.466171	3553.421537	4.782501	2.717
std	9.741443	4.225841	1841.865754	0.879416	1.203
min	6.000000	1.000000	482.000000	0.000000	1.000
25%	27.000000	3.000000	2203.000000	5.000000	2.000
50%	35.000000	6.000000	3184.000000	5.000000	3.000
75%	43.000000	9.000000	4508.000000	5.000000	4.000
max	97.000000	18.000000	14082.000000	5.000000	4.000

In [ ]:
```
## taget feature are Sales and productline
X = data[['SALES','PRODUCTCODE']]
```

In [ ]:
```
data.columns
```

Out[ ]:
```
Index(['QUANTITYORDERED', 'ORDERLINENUMBER', 'SALES', 'STATUS', 'QTR_ID',
       'MONTH_ID', 'YEAR_ID', 'PRODUCTLINE', 'MSRP', 'PRODUCTCODE',
       'CUSTOMERNAME', 'COUNTRY', 'DEALSIZE'],
      dtype='object')
```

K Means implementation

In [ ]:
```
from yellowbrick.cluster import KElbowVisualizer
model = KMeans()
visualizer = KElbowVisualizer(model, k=(1,12)).fit(X)
visualizer.show()
```