

In [4]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [5]:

```
df = pd.read_csv("sales_data_sample.csv")
```

In [6]:

```
df.head()
```

Out[6]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDATE	STATUS	QTR_ID	MONTH_ID
0	10107	30	95.70	2	2871.00	2/24/2003 0:00	Shipped	1	
1	10121	34	81.35	5	2765.90	5/7/2003 0:00	Shipped	2	
2	10134	41	94.74	2	3884.34	7/1/2003 0:00	Shipped	3	
3	10145	45	83.26	6	3746.70	8/25/2003 0:00	Shipped	3	
4	10159	49	100.00	14	5205.27	10/10/2003 0:00	Shipped	4	

5 rows x 25 columns

In [7]:

```
df.dtypes
```

Out[7]:

```
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 [8]:

```
df.isnull().sum()
```

Out[8]:

```
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 [9]:

```
df.info()
```

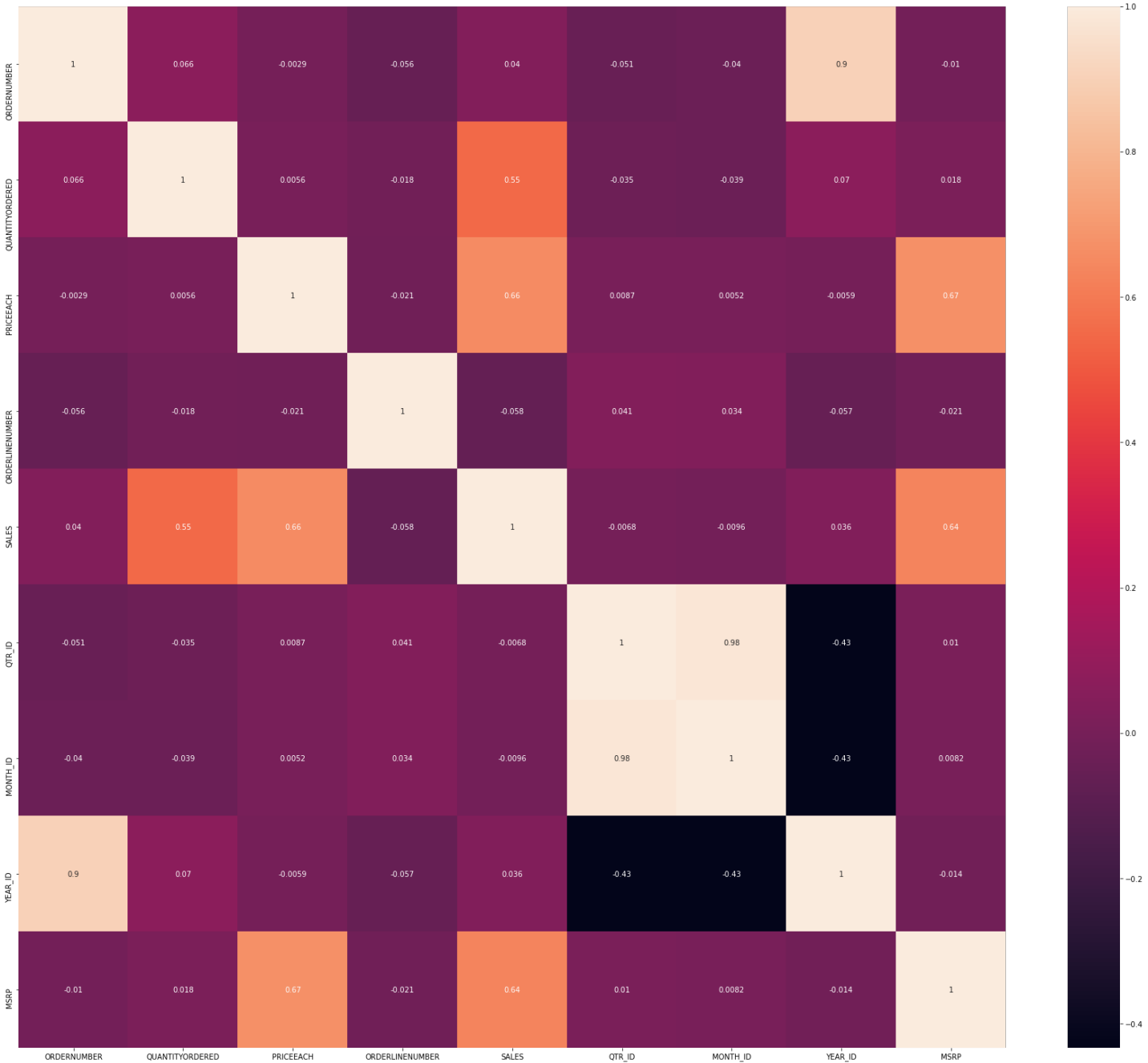
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype  
---  -
 0   ORDERNUMBER           2823 non-null  int64  
 1   QUANTITYORDERED       2823 non-null  int64  
 2   PRICEEACH             2823 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 [10]:

```
plt.figure(figsize = (30,26))
sns.heatmap(df.corr(),annot = True)
```

Out[10]:

<AxesSubplot:>



In [11]:

```
df_drop = ['ADDRESSLINE1', 'ADDRESSLINE2', 'STATUS','POSTALCODE', 'CITY', 'TERRITORY',
'PHONE', 'STATE', 'CONTACTFIRSTNAME', 'CONTACTLASTNAME', 'CUSTOMERNAME', 'ORDERNUMBER']
df = df.drop(df_drop, axis=1)
```

In [12]:

```
df.head()
```

Out[12]:

	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDATE	QTR_ID	MONTH_ID	YEAR_ID	PRODUCTI
0	30	95.70	2	2871.00	2/24/2003 0:00	1	2	2003	Motorcy
1	34	81.35	5	2765.90	5/7/2003 0:00	2	5	2003	Motorcy
2	41	94.74	2	3884.34	7/1/2003 0:00	3	7	2003	Motorcy
3	45	83.26	6	3746.70	8/25/2003 0:00	3	8	2003	Motorcy

QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDATE	QTR_ID	MONTH_ID	YEAR_ID	PRODUCTI
4	49	100.00	14 5205.27	10/10/2003 0:00	4	10	2003	Motorcy

In [13]:

```
df.shape
```

Out[13]:

```
(2823, 13)
```

In [14]:

```
df.isnull().sum()
```

Out[14]:

```
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
COUNTRY             0
DEALSIZE            0
dtype: int64
```

In [15]:

```
df.dtypes
```

Out[15]:

```
QUANTITYORDERED    int64
PRICEEACH          float64
ORDERLINENUMBER    int64
SALES              float64
ORDERDATE          object
QTR_ID             int64
MONTH_ID           int64
YEAR_ID            int64
PRODUCTLINE        object
MSRP               int64
PRODUCTCODE        object
COUNTRY            object
DEALSIZE           object
dtype: object
```

In []:

In [16]:

```
country = pd.get_dummies(df['COUNTRY'])
productline = pd.get_dummies(df['PRODUCTLINE'])
Dealsize = pd.get_dummies(df['DEALSIZE'])
```

In [17]:

```
df = pd.concat([df,country,productline,Dealsize], axis = 1)
```

In [18]:

```
df.head()
```

Out[18]:

	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDATE	QTR_ID	MONTH_ID	YEAR_ID	PRODUCTI
0	30	95.70	2	2871.00	2/24/2003 0:00	1	2	2003	Motorcy
1	34	81.35	5	2765.90	5/7/2003 0:00	2	5	2003	Motorcy
2	41	94.74	2	3884.34	7/1/2003 0:00	3	7	2003	Motorcy
3	45	83.26	6	3746.70	8/25/2003 0:00	3	8	2003	Motorcy
4	49	100.00	14	5205.27	10/10/2003 0:00	4	10	2003	Motorcy

5 rows x 42 columns

In [19]:

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

In [20]:

```
df.dtypes
```

Out[20]:

```
QUANTITYORDERED      int64
PRICEEACH             float64
ORDERLINENUMBER      int64
SALES                 float64
ORDERDATE             object
QTR_ID               int64
MONTH_ID             int64
YEAR_ID             int64
MSRP                 int64
PRODUCTCODE          object
Australia            uint8
Austria              uint8
Belgium              uint8
Canada               uint8
Denmark              uint8
Finland              uint8
France               uint8
Germany              uint8
Ireland              uint8
Italy                uint8
Japan                uint8
Norway               uint8
Philippines          uint8
Singapore            uint8
Spain                uint8
Sweden               uint8
Switzerland          uint8
UK                   uint8
USA                  uint8
Classic Cars         uint8
Motorcycles           uint8
Planes               uint8
Ships                uint8
Trains               uint8
Trucks and Buses     uint8
Vintage Cars         uint8
Large                uint8
Medium               uint8
Small                uint8
dtype: object
```

```
dtype: object
```

```
In [21]:
```

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

```
In [22]:
```

```
df.dtypes
```

```
Out[22]:
```

```
QUANTITYORDERED    int64
PRICEEACH           float64
ORDERLINENUMBER     int64
SALES               float64
ORDERDATE           object
QTR_ID              int64
MONTH_ID            int64
YEAR_ID             int64
MSRP                int64
PRODUCTCODE         int8
Australia            uint8
Austria              uint8
Belgium              uint8
Canada               uint8
Denmark              uint8
Finland              uint8
France               uint8
Germany              uint8
Ireland              uint8
Italy                uint8
Japan                uint8
Norway               uint8
Philippines          uint8
Singapore            uint8
Spain                uint8
Sweden               uint8
Switzerland          uint8
UK                   uint8
USA                  uint8
Classic Cars         uint8
Motorcycles          uint8
Planes               uint8
Ships                uint8
Trains               uint8
Trucks and Buses     uint8
Vintage Cars         uint8
Large                uint8
Medium               uint8
Small                uint8
dtype: object
```

```
In [23]:
```

```
df.drop('ORDERDATE', axis=1, inplace=True)
```

```
In [24]:
```

```
df.dtypes
```

```
Out[24]:
```

```
QUANTITYORDERED    int64
PRICEEACH           float64
ORDERLINENUMBER     int64
SALES               float64
QTR_ID              int64
MONTH_ID            int64
YEAR_ID             int64
MSRP                int64
PRODUCTCODE         int8
Australia            uint8
```

Austria	uint8
Belgium	uint8
Canada	uint8
Denmark	uint8
Finland	uint8
France	uint8
Germany	uint8
Ireland	uint8
Italy	uint8
Japan	uint8
Norway	uint8
Philippines	uint8
Singapore	uint8
Spain	uint8
Sweden	uint8
Switzerland	uint8
UK	uint8
USA	uint8
Classic Cars	uint8
Motorcycles	uint8
Planes	uint8
Ships	uint8
Trains	uint8
Trucks and Buses	uint8
Vintage Cars	uint8
Large	uint8
Medium	uint8
Small	uint8
dtype:	object

In [25]:

```
from sklearn.cluster import KMeans
```

In [26]:

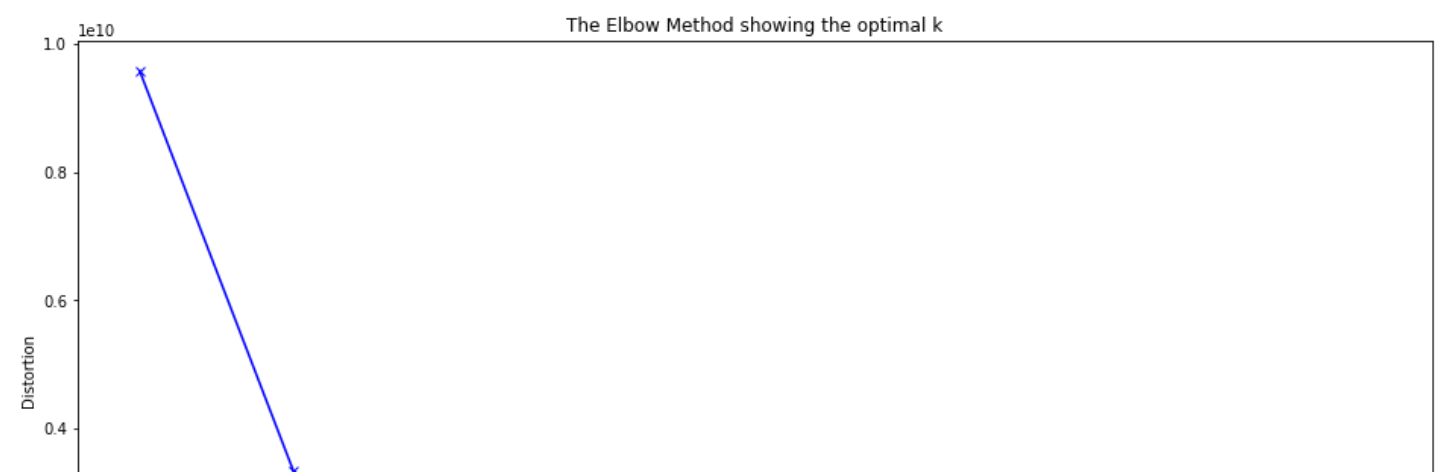
```
WCSS = [] # Within Cluster Sum of Squares from the centroid
```

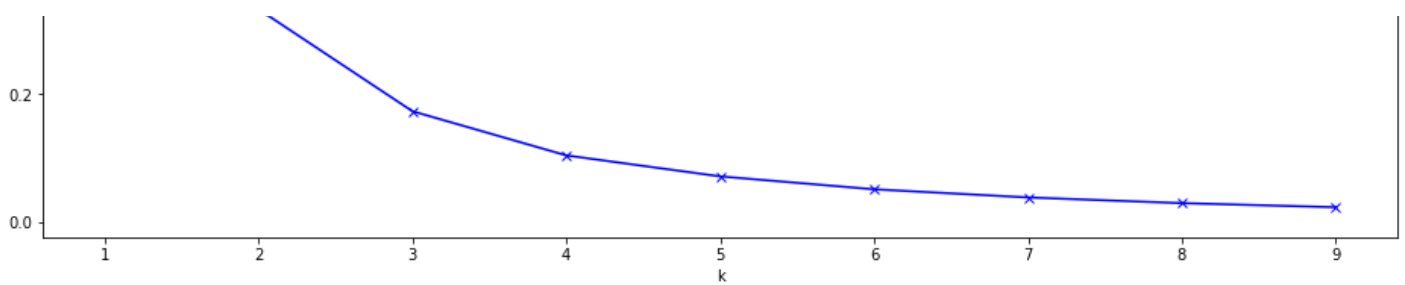
In [27]:

```
distortions = []
K = range(1,10)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(df)
    distortions.append(kmeanModel.inertia_)
```

In [28]:

```
plt.figure(figsize=(16,8))
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal k')
plt.show()
```





In [29]:

```
kmeanModel = KMeans(n_clusters=3)
y_kmeans = kmeanModel.fit_predict
```

In [30]:

```
plt.scatter(df['y'])
```

```
-----
KeyError                                Traceback (most recent call last)
~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
    3079         try:
-> 3080             return self._engine.get_loc(casted_key)
    3081         except KeyError as err:

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'y'
```

The above exception was the direct cause of the following exception:

```
KeyError                                Traceback (most recent call last)
<ipython-input-30-00540c767b35> in <module>
----> 1 plt.scatter(df['y'])

~\anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
    3022         if self.columns.nlevels > 1:
    3023             return self._getitem_multilevel(key)
-> 3024         indexer = self.columns.get_loc(key)
    3025         if is_integer(indexer):
    3026             indexer = [indexer]

~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
    3080             return self._engine.get_loc(casted_key)
    3081         except KeyError as err:
-> 3082             raise KeyError(key) from err
    3083
    3084         if tolerance is not None:
```

KeyError: 'y'

In []:

```
print(y_kmeans)
```

In []:

```
plt.figure(figsize = (30,26))
sns.heatmap(df.corr(),annot = True)
```


In []:

```
pip install yellowbrick
```

In []:

```
from yellowbrick.cluster import KElbowVisualizer
```

In []:

```
model = KMeans()  
visualizer = KElbowVisualizer(model, k=(1, 10), timings = False)  
visualizer.fit(df)  
visualizer.show()
```

In []:

In []:

In [31]:

```
df.head()
```

Out[31]:

	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	QTR_ID	MONTH_ID	YEAR_ID	MSRP	PRODUCTCODE
0	30	95.70	2	2871.00	1	2	2003	95	0
1	34	81.35	5	2765.90	2	5	2003	95	0
2	41	94.74	2	3884.34	3	7	2003	95	0
3	45	83.26	6	3746.70	3	8	2003	95	0
4	49	100.00	14	5205.27	4	10	2003	95	0

5 rows x 38 columns

In [32]:

```
from sklearn.preprocessing import Normalizer
```

In [33]:

```
df_scaled = Normalizer(df)
```

In [34]:

```
df_x = pd.DataFrame(df_scaled, columns = df.columns )
```

```
-----  
ValueError                                Traceback (most recent call last)  
<ipython-input-34-7343a6fbcd9a> in <module>  
----> 1 df_x = pd.DataFrame(df_scaled, columns = df.columns )  
  
~\anaconda3\lib\site-packages\pandas\core\frame.py in __init__(self, data, index, columns  
, dtype, copy)  
    588         else:  
    589             if index is None or columns is None:  
--> 590                 raise ValueError("DataFrame constructor not properly called!")  
    591  
    592             if not dtype:
```

ValueError: DataFrame constructor not properly called!

In []: