K-Means Clustering

Name : Satbhai Anket Satbhai

Roll No. : 4286, Batch : B7

In [1]:

```
import pandas as pd
df=pd.read_csv('C:\\Users\\DELL\\Desktop\\sales_data_sample.csv', encoding='lati
df.head()
```

Out[1]:

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

5 rows × 25 columns

In [2]:

```
cf=df.iloc[:,[1,4]]
cf.head(3)
```

Out[2]:

	QUANTITYORDERED	SALES
0	30	2871.00
1	34	2765.90
2	41	3884.34

In [3]:

```
# k-mean algorithm
from sklearn.cluster import KMeans
```

In [4]:

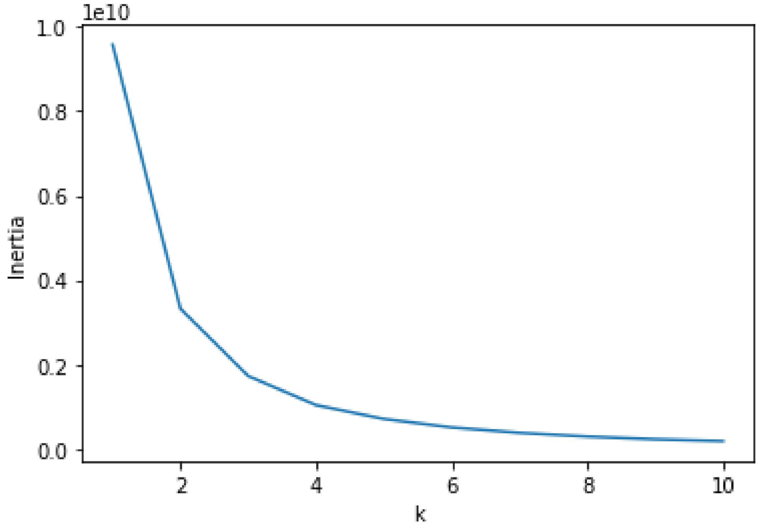
```
# elbow method
inertia=[]
k=[]
for i in range(1,11):
    km=KMeans(n_clusters=i)
    km.fit(cf)
    inertia.append(km.inertia_)
    k.append(i)
```

In [5]:

```
from matplotlib import pyplot as plt
%matplotlib inline
plt.xlabel('k')
plt.ylabel('Inertia')
plt.plot(k,inertia)
```

Out[5]:

```
[<matplotlib.lines.Line2D at 0x1b0ff020d60>]
```

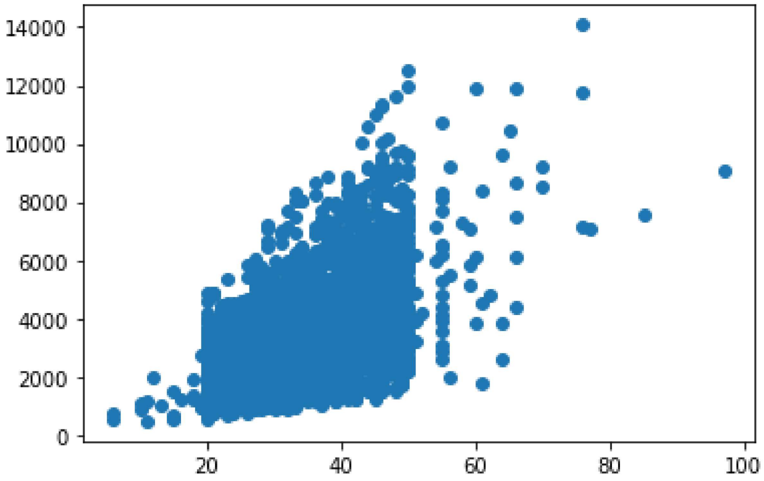


In [6]:

```
plt.scatter(df['QUANTITYORDERED'],df['SALES'])
```

Out[6]:

```
<matplotlib.collections.PathCollection at 0x1b0fd0c0df0>
```



```
In [7]: kmean=KMeans(n_clusters=3)
        y_pred=kmean.fit_predict(df[['QUANTITYORDERED','SALES']])
        y_pred

Out[7]: array([0, 0, 2, ..., 2, 0, 0])
```

```
In [8]: cf['cluster']=y_pred
        cf.head(2)

        C:\Users\DELL\AppData\Local\Temp\ipykernel_5948\3023818644.py:1: SettingWithCop
        yWarning:
        A value is trying to be set on a copy of a slice from a DataFrame.
        Try using .loc[row_indexer,col_indexer] = value instead

        See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
        ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
        ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
        opy)
          cf['cluster']=y_pred

Out[8]:
```

	QUANTITYORDERED	SALES	cluster
0	30	2871.0	0
1	34	2765.9	0

```
In [9]: cf0=cf[cf.cluster==0]
        cf1=cf[cf.cluster==1]
        cf2=cf[cf.cluster==2]
```

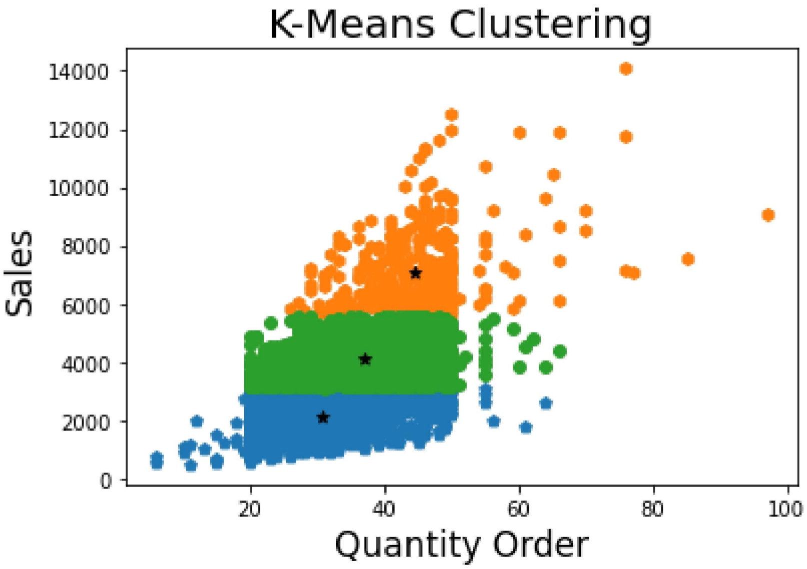
```
In [10]: kmean.cluster_centers_

Out[10]: array([[  30.80235988, 2115.98150442],
                [  44.61096606, 7056.82060052],
                [  37.09686347, 4114.94062731]])
```

In [11]:

```
plt.title('K-Means Clustering',size=20)
plt.xlabel('Quantity Order',size=17)
plt.ylabel('Sales',size=17)
plt.scatter(cf0['QUANTITYORDERED'],cf0['SALES'],marker="p")
plt.scatter(cf1['QUANTITYORDERED'],cf1['SALES'],marker='h')
plt.scatter(cf2['QUANTITYORDERED'],cf2['SALES'])
plt.scatter(kmean.cluster_centers_[:,0],kmean.cluster_centers_[:,1],color='black'
```

Out[11]:

```
<matplotlib.collections.PathCollection at 0x1b0804d7ca0>
```



In [12]:

```
# min max scaler
from sklearn.preprocessing import MinMaxScaler
slr=MinMaxScaler()
```

In [13]:

```
slr.fit(cf[['QUANTITYORDERED']])
cf['QUANTITYORDERED']=slr.transform(cf[['QUANTITYORDERED']])
slr.fit(cf[['SALES']])
cf['SALES']=slr.transform(cf[['SALES']])
cf.head()
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_5948\827238070.py:2: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  cf['QUANTITYORDERED']=slr.transform(cf[['QUANTITYORDERED']])
C:\Users\DELL\AppData\Local\Temp\ipykernel_5948\827238070.py:4: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  cf['SALES']=slr.transform(cf[['SALES']])
```

Out[13]:

	QUANTITYORDERED	SALES	cluster
0	0.263736	0.175644	0
1	0.307692	0.167916	0
2	0.384615	0.250150	2
3	0.428571	0.240030	2
4	0.472527	0.347273	2

In [14]:

```
y_pred_new=kmean.fit_predict(cf[['QUANTITYORDERED','SALES']])
y_pred_new
```

Out[14]:

```
array([1, 1, 0, ..., 2, 1, 0])
```

In [15]:

```
cf['cluster2']=y_pred_new
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_5948\1195066258.py:1: SettingWithCop
yWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  cf['cluster2']=y_pred_new
```

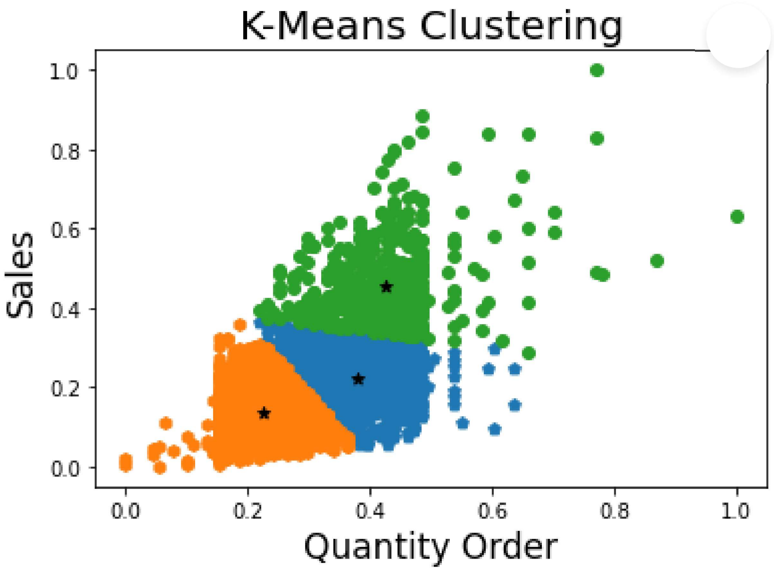
```
In [16]: cfn0=cf[cf.cluster2==0]
         cfn1=cf[cf.cluster2==1]
         cfn2=cf[cf.cluster2==2]
```

```
In [17]: kmean.cluster_centers_
```

```
Out[17]: array([[0.38076717, 0.22462815],
                [0.22516923, 0.13535002],
                [0.42454575, 0.45246352]])
```

```
In [18]: plt.title('K-Means Clustering',size=20)
         plt.xlabel('Quantity Order',size=17)
         plt.ylabel('Sales',size=17)
         plt.scatter(cfn0['QUANTITYORDERED'],cfn0['SALES'],marker="p")
         plt.scatter(cfn1['QUANTITYORDERED'],cfn1['SALES'],marker='h')
         plt.scatter(cfn2['QUANTITYORDERED'],cfn2['SALES'])
         plt.scatter(kmean.cluster_centers_[:,0],kmean.cluster_centers_[:,1],color='black'
```

```
Out[18]: <matplotlib.collections.PathCollection at 0x1b080558e80>
```


K-Means Clustering
Sales
Quantity Order