Analysis Of Mobile Phones Sales

A mini-project submitted for

Business Intelligence Lab (Semester VI)

by

SAP ID	Name
60003180056	Vaibhav Gandhi
60003180060	Vedant Gandhi

Problem Definition :

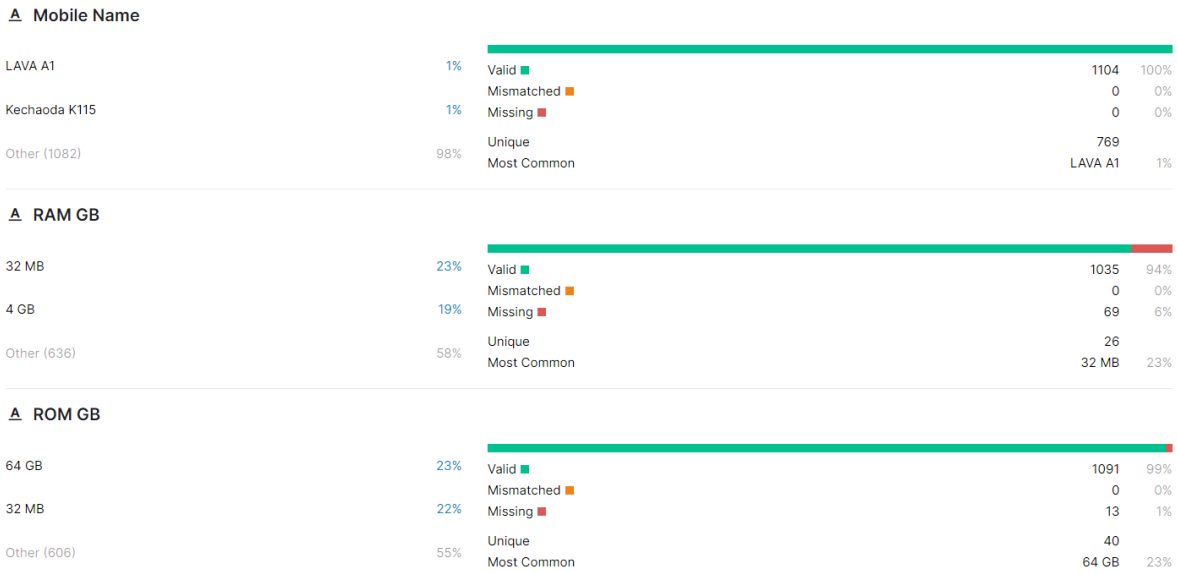
With the increase in competition in the mobile phone industry, many businesses are trying to device plans to boost their sales and survive in this competitive environment. The dataset used includes various parameters of mobile phones that act as a deciding factor in purchasing a new device. After analysing these factors, we can provide different decisions to the businesses that could help them boost their sales.

Dataset Link :

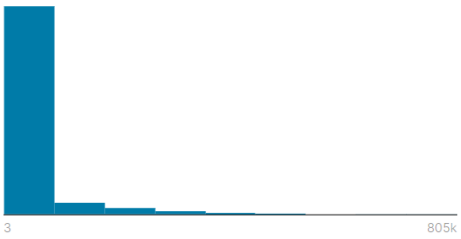
https://www.kaggle.com/ginelledsouza/mobilephone

Data Exploration Steps:

Visualizing the data, for a better understanding of the dataset and it's features:

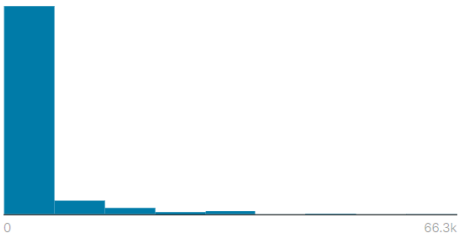


Ratings



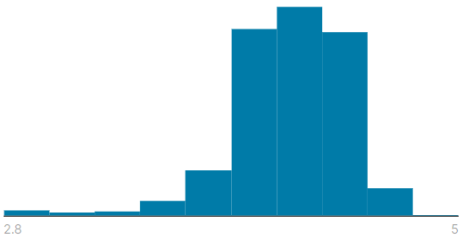
Valid	1095	99%
Mismatched	0	0%
Missing	9	1%
Mean	37.9k	
Std. Deviation	84.4k	
Quantiles	3	Min
	840	25%
	5941	50%
	29.8k	75%
	805k	Max

Reviews



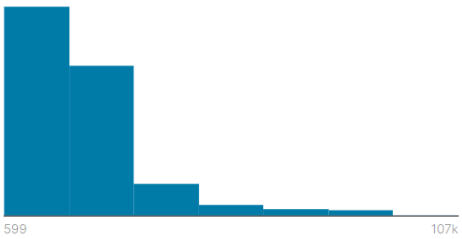
Valid	1095	99%
Mismatched	0	0%
Missing	9	1%
Mean	3.33k	
Std. Deviation	7.45k	
Quantiles	0	Min
	71	25%
	531	50%
	2815	75%
	66.3k	Max

Stars



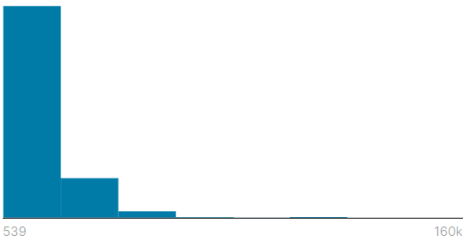
Valid	1104	100%
Mismatched	0	0%
Missing	0	0%
Mean	4.18	
Std. Deviation	0.29	
Quantiles	2.8	Min
	4	25%
	4.3	50%
	4.4	75%
	5	Max

List Price



Valid	817	74%
Mismatched	0	0%
Missing	287	26%
Mean	13k	
Std. Deviation	12.1k	
Quantiles	599	Min
	1889	25%
	11.0k	50%
	18.0k	75%
	107k	Max

Sales Price



Valid	1104	100%
Mismatched	0	0%
Missing	0	0%
Mean	10.2k	
Std. Deviation	12.9k	
Quantiles	539	Min
	1202	25%
	8490	50%
	14.0k	75%
	160k	Max

Importing Packages

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

Loading the data

```
[2]  df = pd.read_csv('/content/MobilePhones.csv')
```

Exploring the Dataset

```
df.head()
```

	MobileName	RAM_GB	ROM_GB	Sales	Reviews	Stars	ListPrice	SalesPrice
0	Redmi 8 (Onyx Black, 64 GB)	4	64	674638	50064	4.4	10999	9999
1	Realme 5i (Forest Green, 64 GB)	4	64	243106	16497	4.5	10999	10999
2	Realme 5i (Aqua Blue, 64 GB)	4	64	243106	16497	4.5	10999	10999
3	Redmi 8 (Sapphire Blue, 64 GB)	4	32	674638	50064	4.4	10999	9999
4	POCO X2 (Matrix Purple, 128 GB)	6	64	133486	14732	4.5	19999	18499

```
df.describe()
```

	RAM_GB	ROM_GB	Sales	Reviews	Stars	ListPrice	SalesPrice
count	125.000000	125.000000	125.000000	125.000000	125.000000	125.000000	125.000000
mean	4.752000	67.744000	56420.080000	4912.256000	4.270400	17790.256000	15931.280000
std	3.903381	38.021196	133546.887408	10762.940593	0.455437	14604.167266	12579.825331
min	2.000000	4.000000	53.000000	3.000000	2.800000	609.000000	609.000000
25%	3.000000	32.000000	5815.000000	438.000000	4.300000	9990.000000	8990.000000
50%	4.000000	64.000000	17040.000000	1298.000000	4.400000	12999.000000	11490.000000
75%	6.000000	64.000000	42968.000000	3745.000000	4.500000	18999.000000	17499.000000
max	32.000000	256.000000	805006.000000	66292.000000	4.600000	106600.000000	79999.000000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125 entries, 0 to 124
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   MobileName  125 non-null    object 
 1   RAM_GB      125 non-null    int64  
 2   ROM_GB      125 non-null    int64  
 3   Sales       125 non-null    int64  
 4   Reviews     125 non-null    int64  
 5   Stars       125 non-null    float64
 6   ListPrice   125 non-null    int64  
 7   SalesPrice  125 non-null    int64  
dtypes: float64(1), int64(6), object(1)
memory usage: 7.9+ KB
```

Data Preprocessing

Removing duplicates if any:

```
df.drop_duplicates(inplace=True)
df.shape
```

```
(117, 14)
```

Removing outliers:

```
print("*** RAM *** ")
print(df['RAM_GB'].value_counts())
print("\n*** ROM *** ")
print(df['ROM_GB'].value_counts())
```

```
*** RAM ***
4     52
3     28
6     19
2     12
8     11
32     2
12     1
Name: RAM_GB, dtype: int64

*** ROM ***
64     64
32     30
128    24
16      5
256     1
4       1
Name: ROM_GB, dtype: int64
```

```
[10] print(df[df['RAM_GB'] == 32].index)
     print(df[df['ROM_GB'] == 4].index)

     Int64Index([115, 118], dtype='int64')
     Int64Index([80], dtype='int64')
```

```
[11] df.drop([115,118,80], inplace=True,axis=0)
```

Creating a separate column for attribute 'color'

```
df['Color'] = df['MobileName'].apply(lambda x : x.split(",")[0].split("(")[1]
                                     if len(x.split(",")[0].split("(")) > 1 else 'No Color')

df.head()
```

	MobileName	RAM_GB	ROM_GB	Sales	Reviews	Stars	ListPrice	SalesPrice	Color
0	Redmi 8 (Onyx Black, 64 GB)	4	64	674638	50064	4.4	10999	9999	Onyx Black
1	Realme 5i (Forest Green, 64 GB)	4	64	243106	16497	4.5	10999	10999	Forest Green
2	Realme 5i (Aqua Blue, 64 GB)	4	64	243106	16497	4.5	10999	10999	Aqua Blue
3	Redmi 8 (Sapphire Blue, 64 GB)	4	32	674638	50064	4.4	10999	9999	Sapphire Blue
4	POCO X2 (Matrix Purple, 128 GB)	6	64	133486	14732	4.5	19999	18499	Matrix Purple

Creating a separate column for attribute 'brand'

```
df['MobileName'] = df['MobileName'].apply(lambda x : x.split("(")[0])
df.head()
```

	MobileName	RAM_GB	ROM_GB	Sales	Reviews	Stars	ListPrice	SalesPrice	Color	Brand
0	Redmi 8	4	64	674638	50064	4.4	10999	9999	Onyx Black	Redmi
1	Realme 5i	4	64	243106	16497	4.5	10999	10999	Forest Green	Realme
2	Realme 5i	4	64	243106	16497	4.5	10999	10999	Aqua Blue	Realme
3	Redmi 8	4	32	674638	50064	4.4	10999	9999	Sapphire Blue	Redmi
4	POCO X2	6	64	133486	14732	4.5	19999	18499	Matrix Purple	POCO

```
df['Brand'] = df['MobileName'].apply(lambda x : x.split()[0])
df['Brand'] = df['Brand'].apply(lambda x : 'I Kall' if x == 'I' else x)
df.head()
```

	MobileName	RAM_GB	ROM_GB	Sales	Reviews	Stars	ListPrice	SalesPrice	Color	Brand
0	Redmi 8 (Onyx Black, 64 GB)	4	64	674638	50064	4.4	10999	9999	Onyx Black	Redmi
1	Realme 5i (Forest Green, 64 GB)	4	64	243106	16497	4.5	10999	10999	Forest Green	Realme
2	Realme 5i (Aqua Blue, 64 GB)	4	64	243106	16497	4.5	10999	10999	Aqua Blue	Realme
3	Redmi 8 (Sapphire Blue, 64 GB)	4	32	674638	50064	4.4	10999	9999	Sapphire Blue	Redmi
4	POCO X2 (Matrix Purple, 128 GB)	6	64	133486	14732	4.5	19999	18499	Matrix Purple	POCO

Creating a separate column for attribute ‘discount’

```
df['Discount'] = df['ListPrice'] - df['SalesPrice']
df.head()
```

	MobileName	RAM_GB	ROM_GB	Sales	Reviews	Stars	ListPrice	SalesPrice	Color	Brand	Discount
0	Redmi 8	4	64	674638	50064	4.4	10999	9999	Onyx Black	Redmi	1000
1	Realme 5i	4	64	243106	16497	4.5	10999	10999	Forest Green	Realme	0
2	Realme 5i	4	64	243106	16497	4.5	10999	10999	Aqua Blue	Realme	0
3	Redmi 8	4	32	674638	50064	4.4	10999	9999	Sapphire Blue	Redmi	1000
4	POCO X2	6	64	133486	14732	4.5	19999	18499	Matrix Purple	POCO	1500

Creating a separate column for attribute ‘percent category’

```
df['PercentDiscount'] = df['Discount']/df['ListPrice']*100
bins = [-1, 5, 10, 15, 20, 25, 30, np.inf]
names = ['0', '1-5%', '6-10%', '11-15%', '15-20%', '21-25%', '26-30%']
df['PercentCategory'] = pd.cut(df['PercentDiscount'], bins, labels=names)
df.head()
```

	MobileName	RAM_GB	ROM_GB	Sales	Reviews	Stars	ListPrice	SalesPrice	Color	Brand	Discount	PercentDiscount	PercentCategory
0	Redmi 8	4	64	674638	50064	4.4	10999	9999	Onyx Black	Redmi	1000	9.091736	1-5%
1	Realme 5i	4	64	243106	16497	4.5	10999	10999	Forest Green	Realme	0	0.000000	0
2	Realme 5i	4	64	243106	16497	4.5	10999	10999	Aqua Blue	Realme	0	0.000000	0
3	Redmi 8	4	32	674638	50064	4.4	10999	9999	Sapphire Blue	Redmi	1000	9.091736	1-5%
4	POCO X2	6	64	133486	14732	4.5	19999	18499	Matrix Purple	POCO	1500	7.500375	1-5%

Creating a new attribute of ‘Specs’

```
df['Specs'] = (df['RAM_GB']/max(df['RAM_GB'])+df['ROM_GB']/max(df['ROM_GB']))*50
```

```
BestSellers = df.loc[(df['Brand'] == 'Apple')|(df['Brand']=='Redmi')|(df['Brand']=='Realme')|(df['Brand']=='POCO')
              |(df['Brand']=='OnePlus')|(df['Brand']=='Google')|(df['Brand']=='Motorola')]
print(BestSellers)
```

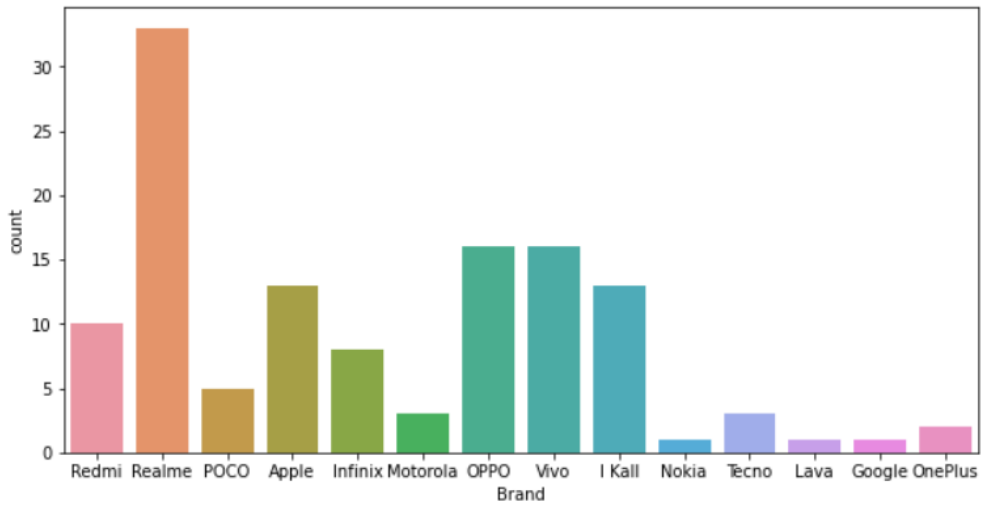
```
              MobileName  RAM_GB  ...  PercentCategory   Specs
0                Redmi 8       4  ...              1-5%  18.750
1              Realme 5i       4  ...                 0  18.750
2              Realme 5i       4  ...                 0  18.750
3                Redmi 8       4  ...              1-5%  12.500
4                POCO X2       6  ...              1-5%  21.875
..                   ...     ...  ...               ...     ...
120      Google Pixel 4a       6  ...              1-5%  34.375
121      Apple iPhone 11       4  ...             6-10%  18.750
122           OnePlus 8T       8  ...              1-5%  37.500
123  Apple iPhone 11 Pro       4  ...            15-20%  18.750
124      OnePlus Nord 5G      12  ...                 0  68.750

[67 rows x 14 columns]
```

Data Modelling

Relationship between ‘Brand’ and ‘Count’

```
plt.figure(figsize=(10,5))
sns.countplot('Brand', data=df)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f03e83efd10>
```

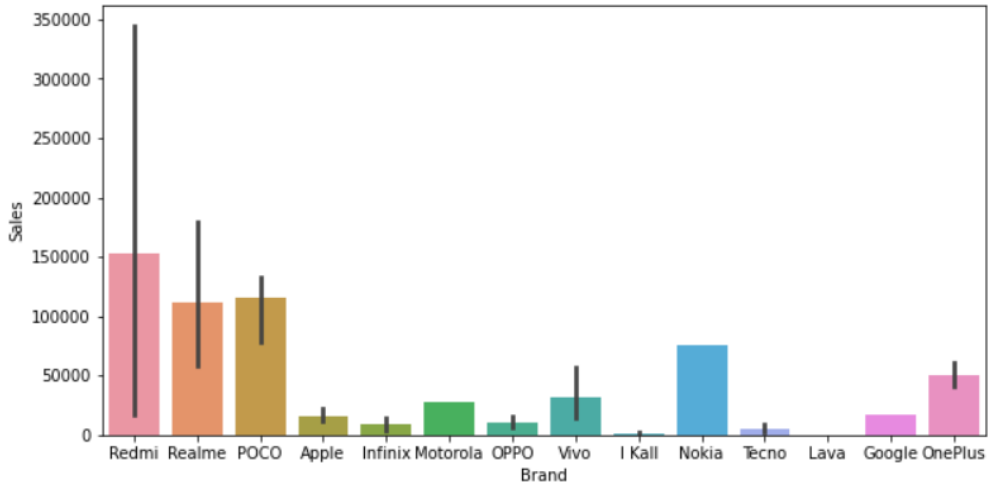


From the graph we can interpret that the number of devices of the Realme brand are too high, while Nokia, Lava and Google have a pretty low number.

Relationship between 'Brand' and 'Sales'

```
plt.figure(figsize=(10,5))
sns.barplot(df['Brand'],df['Sales'],data=df)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the fo
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f03e78d0ad0>
```

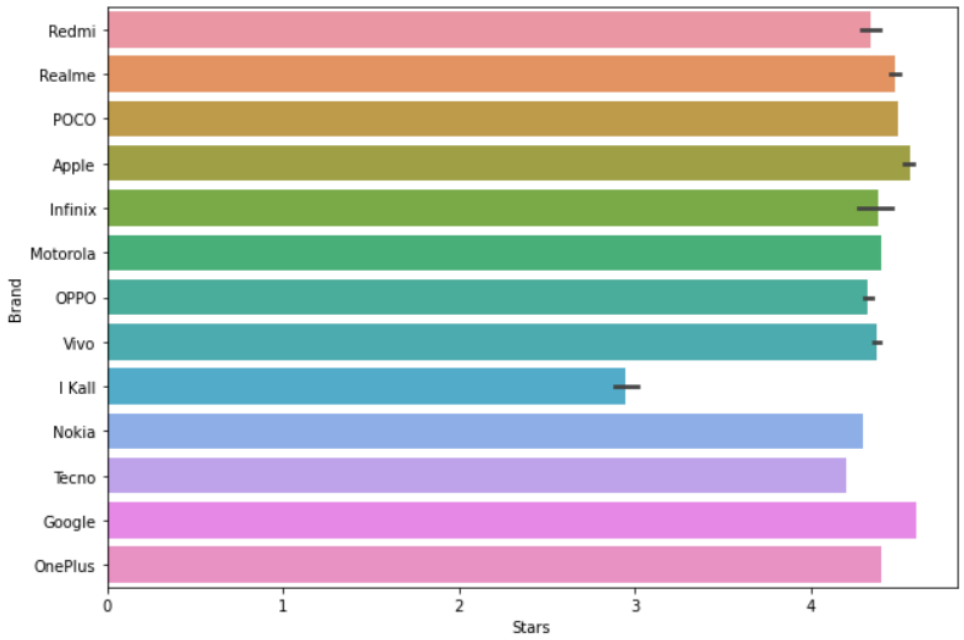


Here, we can see that Realme sells the highest number of devices. They outsell their closest competitor by almost 50%. Companies like Realme, POCO and Nokia also have good sales figures while companies like Lava and I Kall are struggling to sell devices.

Relationship between 'Brand' and 'Stars'

```
[ ]  plt.figure(figsize=(10,7))
     sns.barplot(y=df['Brand'],x=df['Stars'],data=df,orient='h')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69a53a7310>
```

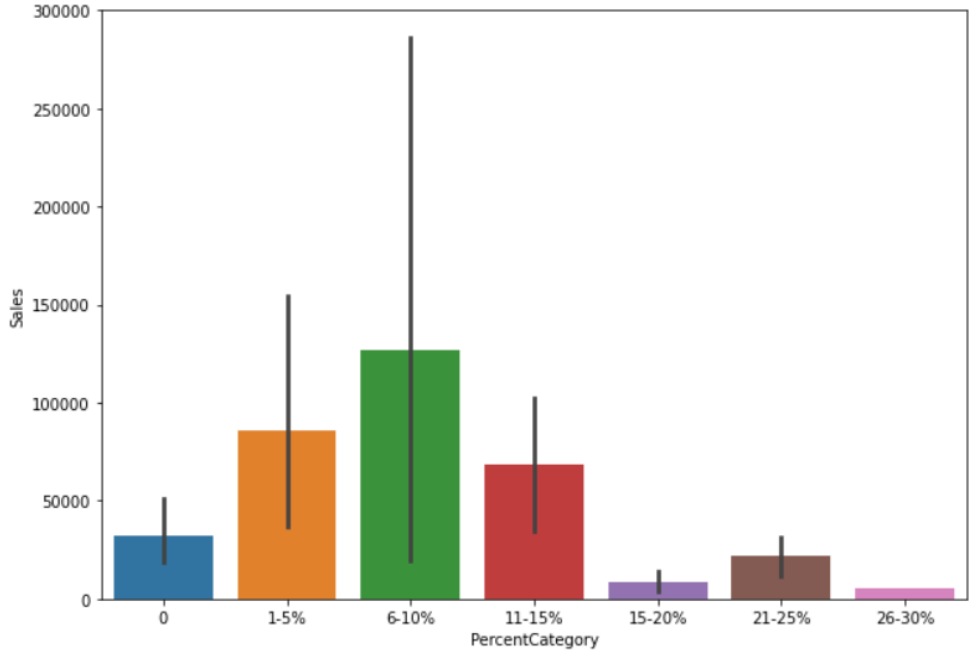


From the graph, we can see that Google has received the highest stars among the smartphones. Apple and POCO are just behind Google, but I Kall has the lowest rating which falls just near the borderline of 3.

Relationship between 'PercentCategory' and 'Sales'

```
[ ]  plt.figure(figsize=(10,7))
     sns.barplot(y=df['Sales'], x=df['PercentCategory'], data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69a52e78d0>
```

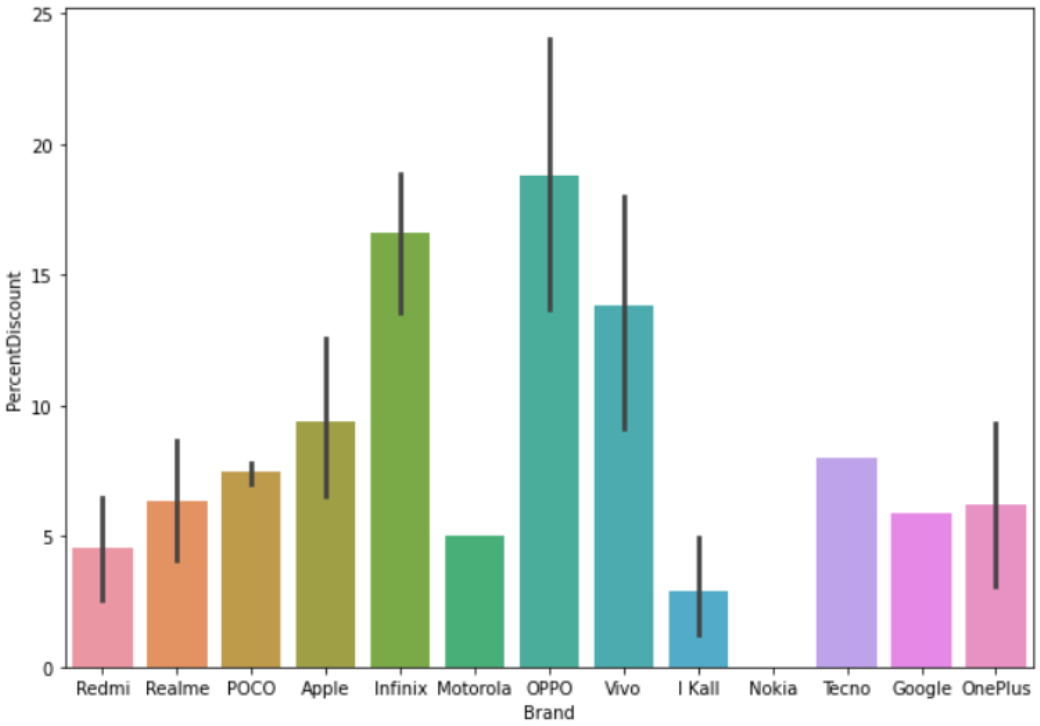


Devices that have discounts from 6-10% range have shown the maximum number of sales. It is interesting to note that as the percent discount increases, the sales figures show a dip. Devices with no discount have shown more appreciation than devices with 15+ discounts. It should also be noted that this discount range is for all devices, and not for any one device in particular.

Relationship between 'Brand' and 'PercentDiscount'

```
[ ]  plt.figure(figsize=(10,7))
     sns.barplot(y=df['PercentDiscount'], x=df['Brand'], data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69a527c450>
```

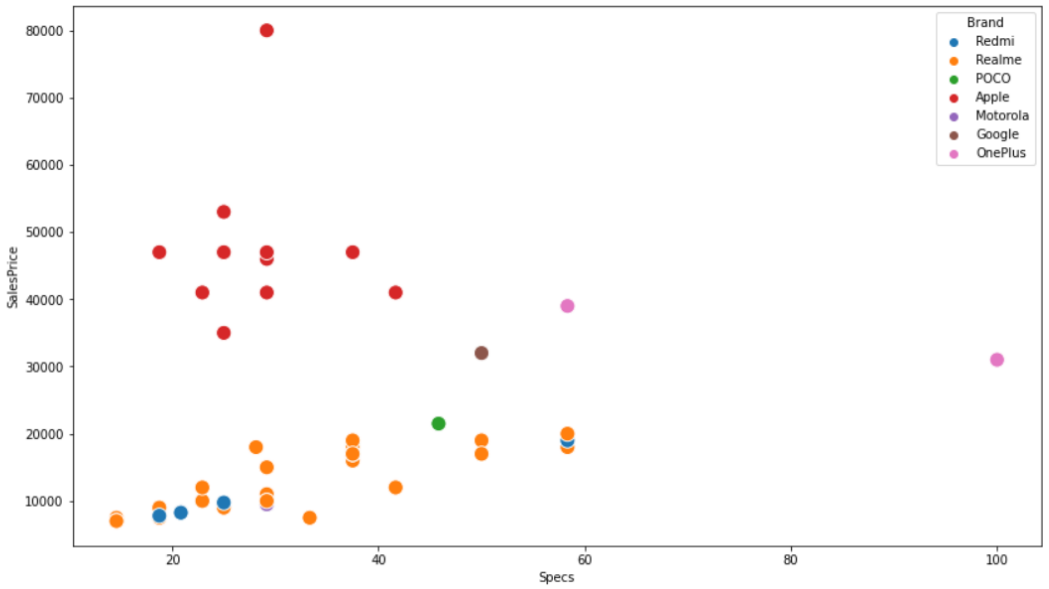


Here we can see that brand OPPO is offering the highest discount which is around 20%. Infinix is providing the second best discount while Nokia is not offering any discount.

Relationship between 'Specs' and 'SalesPrice'

```
plt.figure(figsize=(14,8))
sns.scatterplot(x=BestSellers['Specs'], y=BestSellers['SalesPrice'], hue=BestSellers['Brand'], data=BestSellers, s=150)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fac750d9610>
```

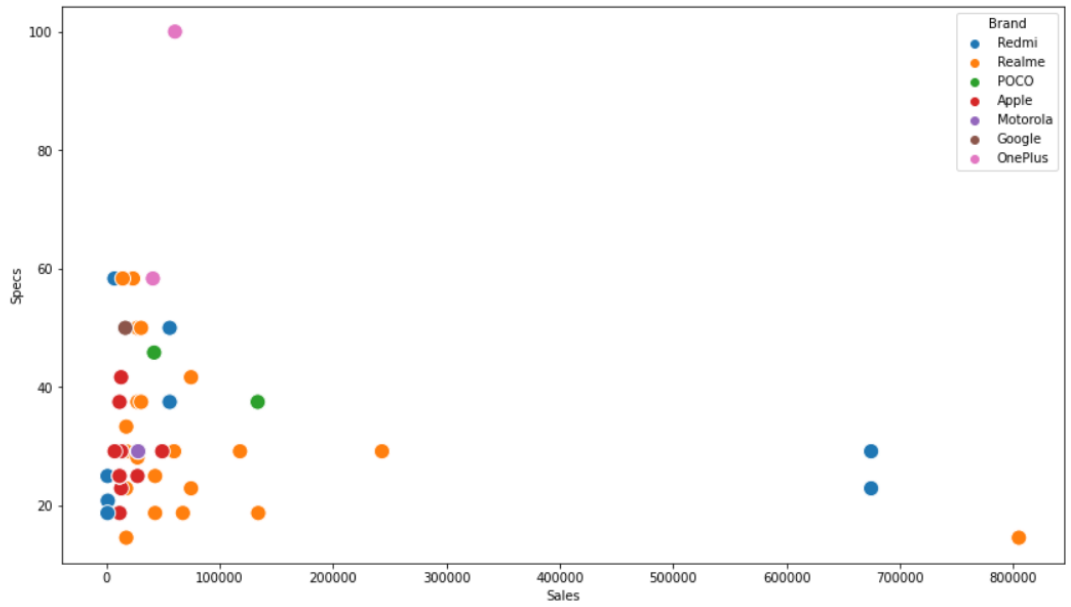


Here we can see that OnePlus provides the best specs though its sales price being in the region of Rs 30000-40000 whereas in the case of Apple, the salesprice is quite high though its specs are in the region of 20-40. Most of the smartphones having specs in the range of 20-40 have a quite low sales price.

Relationship between ‘Specs’ and ‘Sales’

```
plt.figure(figsize=(14,8))
sns.scatterplot(x=BestSellers['Sales'], y=BestSellers['Specs'], hue=BestSellers['Brand'], data=BestSellers, s=150)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fac75089b50>
```



Here, we can see that OnePlus 8T device has the best specs. Most devices fall under the 60 Specs score. It can be seen that Realme and Redmi devices have devices for all needs.

Justification on why a particular data mining task is chosen. Also justify the algorithm selected:

The aim of this project is to provide the best possible solution that a business can adopt to increase their sales. This dataset consists of 7 different features which can help us decide the necessary factors to boost the sale of a particular device. We will use a few features to derive our own columns to give us a new perspective.

When considering mobile phones, there is no one-size-fits-all formula. The sale of a particular device may depend on several factors such as the budget of the customer, past experience with the company, specifications etc. Hence, we can use clustering to group various devices offering the same things and use those clusters to analyse and make decisions.

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

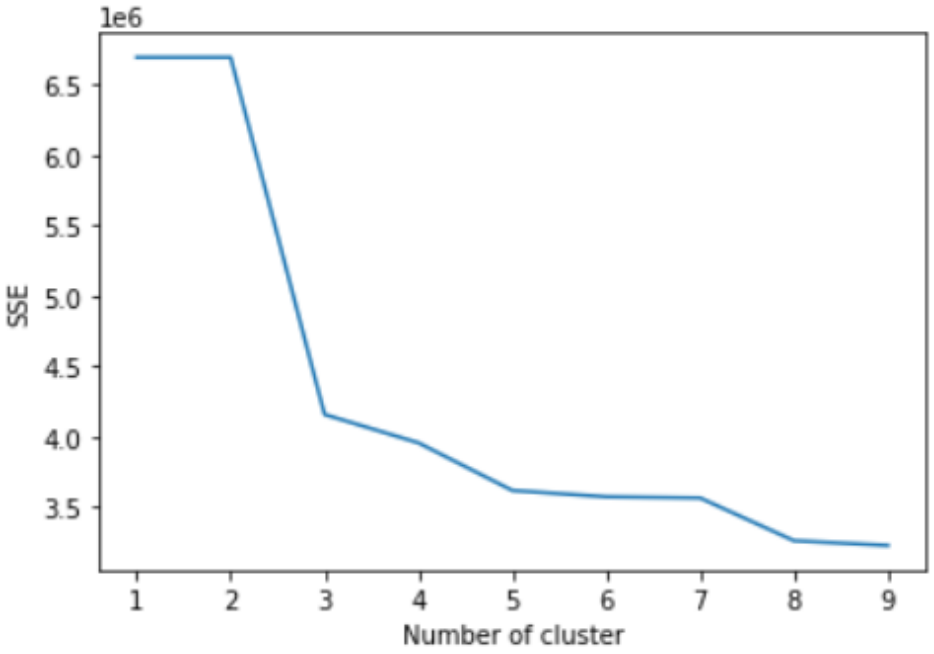
We will use KMedoids clustering for our analysis. Advantages of K-medoids Algorithms are:

- As compared to other Partitioning algorithms, it effectively dealt with the noise and outliers present in data; because it uses medoid for the partitioning of objects into clusters.
- Easily Implementable and simple to understand.
- K-Medoid Algorithm is comparably fast as compared to other partitional algorithms.
- It outputs the final clusters of objects in a fixed number of iterations.

Code and Visualization of Decision Tree Algorithm:

A fundamental step for any unsupervised algorithm is to determine the optimal number of clusters into which the data may be clustered. The Elbow Method is one of the most popular methods to determine this optimal value of k.

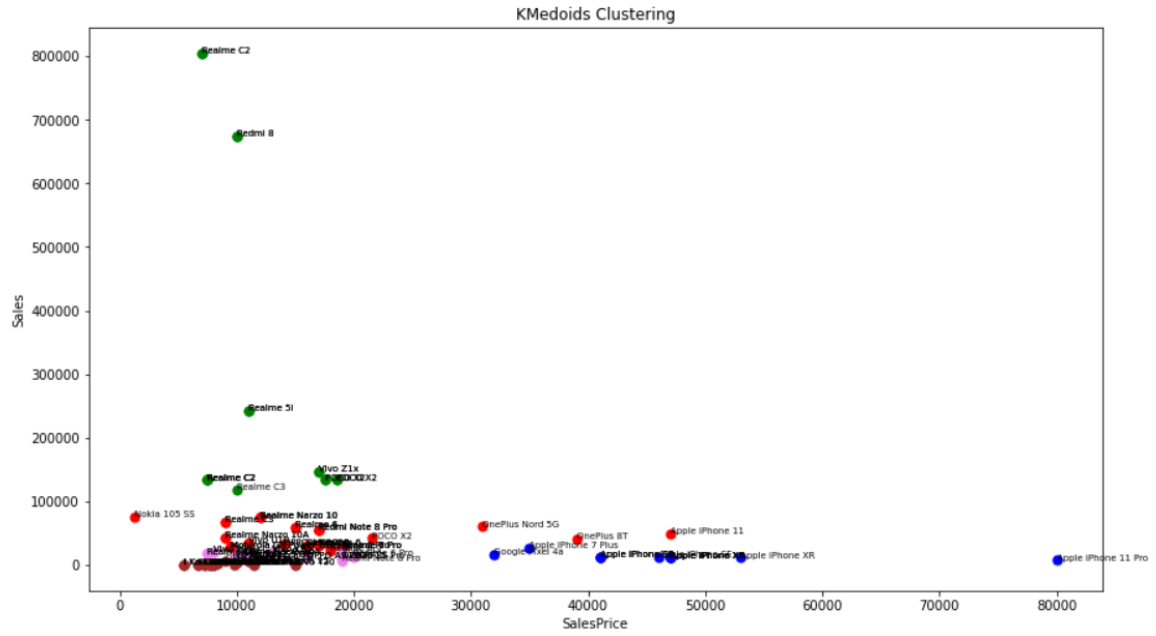
```
from sklearn_extra.cluster import KMedoids
dataSet = df[['SalesPrice','Sales']].copy(deep=True)
dataSet.dropna(axis=0, inplace=True)
data = np.array(dataSet)
sse = {}
for k in range(1, 10):
    k_med = KMedoids(n_clusters=k, max_iter=1000).fit(data)
    sse[k] = k_med.inertia_ # Inertia: Sum of distances of samples to their closest cluster center
plt.figure()
plt.plot(list(sse.keys()), list(sse.values()))
plt.xlabel("Number of cluster")
plt.ylabel("SSE")
plt.show()
```



For annotations, we need to create an array of device names in order.

```
mobName = df[['MobileName']].copy(deep=True)
mobName.dropna(axis=0, inplace=True)
name = np.array(mobName)
```

```
k_med = KMedoids(n_clusters = 5)
k_medoid = k_med.fit_predict(data)
plt.figure(figsize=(14,8))
plt.scatter(
    data[k_medoid == 0, 0], data[k_medoid == 0, 1],
    s=50, c='red'
)
plt.scatter(
    data[k_medoid == 1, 0], data[k_medoid == 1, 1],
    s=50, c='blue'
)

plt.scatter(
    data[k_medoid == 2, 0], data[k_medoid == 2, 1],
    s=50, c='green'
)

plt.scatter(
    data[k_medoid == 3, 0], data[k_medoid == 3, 1],
    s=50, c='brown'
)

plt.scatter(
    data[k_medoid == 4, 0], data[k_medoid == 4, 1],
    s=50, c='violet'
)

for i, txt in enumerate(name):
    plt.annotate(*(name[i]), (data[:,0][i], data[:,1][i])).set_fontsize(7)
plt.xlabel('SalesPrice')
plt.ylabel('Sales')
plt.title('KMedoids Clustering')
plt.show()
```



Observations:

- The devices in the green cluster have great sales figures. None of these devices cost above Rs. 20,000. Most of these devices are from the companies Realme and Redmi. These companies provide 4-7% discount on their devices.
- The devices in the blue cluster are very costly ranging from Rs 30,000 to 80,000. Most of these devices are from companies like Apple and Google. Although these devices cost more, they have the highest star rating and their sales figures are similar to the pink and brown cluster which are not as costly as these ones.
- The devices in the red cluster have good sales. Although some of these devices cost more, none exceed the Rs. 50,000 mark. Companies like OnePlus, POCO fall into this cluster, while it also contains some devices from Realme and Nokia. OnePlus devices have the best specs, while POCO devices have lower specs but with a significant price drop.
- The devices in the pink and brown cluster have very low sales figures even though they don't cost much. Devices from companies like Motorola, I Kall, Infinix, OPPO, Vivo, Techno, Lava. OPPO, Infinix and Vivo provide a discount of more than 15% on their devices. These companies have a low customer star rating(below 4.2). I Kall's rating is below 3.

Business Intelligence Decisions obtained:

- The most important factor in the sale of a device is the price. Consumers want a well-rounded phone with good specifications for a relatively low cost. The devices in the pink and brown cluster also cost less but do not sell well because of a low specs score. Any company has to make sure to provide a decent on-paper package.
- Realme is a well-known company because it boosts its brand name by producing devices that cater to every need. By having something to offer for every demographic, any company can boost its sale.
- As a company releases a new model, it should give a small discount on its older devices. We see the highest sales in the discount range of 6-10%. The brands that fall into this figure are Realme and POCO. We can see that companies like Infinix, Techno and OPPO, despite having a lower star-rating, they have generated decent sales figure by offering very high discounts. While companies like Lava and I Kall having a low user-rating and offering minimal discounts, see very low sales. So, any company has to make sure it has a good user-rating, else it will have to cut down their profit margins by offering higher discounts.
- Devices from Apple and Google have good sales figures despite having a higher price. This can be attributed to their user-experience, brand value and star-rating. They have the highest star rating. This allows such companies to have a higher margin on their devices in the future. OnePlus devices, although cost more, do not see such a rise in user-rating and sales. Any company has to make sure to refine its craft to get the satisfaction of their users before increasing their price.