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

Reading the csv file into pandas Dataframe

```
df=pd.read_csv('/content/drive/MyDrive/Data Sets/Mobilephone specifications and prices.csv')
```

```
df
```

	Unnamed: 0	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	...	Rear camera	Front camera
0	0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	...	48.0	16.0
1	1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	...	64.0	16.0
2	2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	...	12.0	12.0
3	3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	...	12.0	12.0
4	4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	...	12.0	32.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1354	1354	Intex Aqua A2	Intex	Aqua A2	1500	4.00	Yes	480	800	4	...	5.0	0.3
1355	1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.00	Yes	480	854	4	...	8.0	5.0
1356	1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.50	Yes	480	854	2	...	5.0	2.0
1357	1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.00	Yes	480	800	1	...	2.0	0.3
1358	1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.00	Yes	480	854	4	...	8.0	0.0

1359 rows × 22 columns

Understanding the data

```
df.head()
```

	Unnamed: 0	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	...	Rear camera	Front camera	Oper s
0	0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	...	48.0	16.0	A
1	1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	...	64.0	16.0	A
2	2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	...	12.0	12.0	
3	3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	...	12.0	12.0	
4	4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	...	12.0	32.0	A

5 rows × 22 columns

```
df.tail()
```

	Unnamed: 0	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	...	Rear camera	Front camera
1354	1354	Intex Aqua A2	Intex	Aqua A2	1500	4.0	Yes	480	800	4	...	5.0	0.3
1355	1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.0	Yes	480	854	4	...	8.0	5.0
1356	1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.5	Yes	480	854	2	...	5.0	2.0
1357	1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.0	Yes	480	800	1	...	2.0	0.3
1358	1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.0	Yes	480	854	4	...	8.0	0.0

5 rows × 22 columns

```
df.shape
```

```
(1359, 22)
```

missing value and duplicates handling

```
df.isna().sum()
```

	0
Unnamed: 0	0
Name	0
Brand	0
Model	0
Battery capacity (mAh)	0
Screen size (inches)	0
Touchscreen	0
Resolution x	0
Resolution y	0
Processor	0
RAM (MB)	0
Internal storage (GB)	0
Rear camera	0
Front camera	0
Operating system	0
Wi-Fi	0
Bluetooth	0
GPS	0
Number of SIMs	0
3G	0
4G/ LTE	0
Price	0

```
df.duplicated().sum()
```

```
0
```

```
df.dtypes
```

	0
Unnamed: 0	int64
Name	object
Brand	object
Model	object
Battery capacity (mAh)	int64
Screen size (inches)	float64
Touchscreen	object
Resolution x	int64
Resolution y	int64
Processor	int64
RAM (MB)	int64
Internal storage (GB)	float64
Rear camera	float64
Front camera	float64
Operating system	object
Wi-Fi	object
Bluetooth	object
GPS	object
Number of SIMs	int64
3G	object
4G/ LTE	object
Price	int64

dtype: object

```
df.describe()
```

	Unnamed: 0	Battery capacity (mAh)	Screen size (inches)	Resolution x	Resolution y	Processor	RAM (MB)	Internal storage (GB)	Rear camera	Front camera
count	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000
mean	679.000000	2938.489330	5.291310	811.543046	1490.777778	5.551141	2488.777778	30.654864	12.070199	7.037969
std	392.453819	873.514133	0.671357	270.707271	557.780120	2.196562	1664.440386	36.950241	8.948337	6.295448
min	0.000000	1010.000000	2.400000	240.000000	320.000000	1.000000	64.000000	0.064000	0.000000	0.000000
25%	339.500000	2300.000000	5.000000	720.000000	1280.000000	4.000000	1000.000000	8.000000	8.000000	2.000000
50%	679.000000	3000.000000	5.200000	720.000000	1280.000000	4.000000	2000.000000	16.000000	12.200000	5.000000
75%	1018.500000	3500.000000	5.700000	1080.000000	1920.000000	8.000000	3000.000000	32.000000	13.000000	8.000000

```
df.drop(columns=['Unnamed: 0'],inplace=True)
```

```
df
```

	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	RAM (MB)	...	Rear camera	Front camera	Ope
0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	12000	...	48.0	16.0	
1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	6000	...	64.0	16.0	
2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	4000	...	12.0	12.0	
3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	4000	...	12.0	12.0	
4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	6000	...	12.0	32.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1354	Intex Aqua A2	Intex	Aqua A2	1500	4.00	Yes	480	800	4	512	...	5.0	0.3	
1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.00	Yes	480	854	4	1000	...	8.0	5.0	
1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.50	Yes	480	854	2	512	...	5.0	2.0	
1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.00	Yes	480	800	1	256	...	2.0	0.3	
1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.00	Yes	480	854	4	1000	...	8.0	0.0	

1359 rows × 21 columns

```
df['Operating system'].unique()
```

```
array(['Android', 'iOS', 'Cyanogen', 'BlackBerry', 'Windows', 'Tizen',
       'Sailfish'], dtype=object)
```

```
df['Wi-Fi'].unique()
```

```
array(['Yes', 'No'], dtype=object)
```

```
df.describe()
```

	Battery capacity (mAh)	Screen size (inches)	Resolution x	Resolution y	Processor	RAM (MB)	Internal storage (GB)	Rear camera	Front camera	Number of SIMs
count	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000
mean	2938.489330	5.291310	811.543046	1490.777778	5.551141	2488.777778	30.654864	12.070199	7.037969	1.833701
std	873.514133	0.671357	270.707271	557.780120	2.196562	1664.440386	36.950241	8.948337	6.295448	0.374457
min	1010.000000	2.400000	240.000000	320.000000	1.000000	64.000000	0.064000	0.000000	0.000000	1.000000
25%	2300.000000	5.000000	720.000000	1280.000000	4.000000	1000.000000	8.000000	8.000000	2.000000	2.000000
50%	3000.000000	5.200000	720.000000	1280.000000	4.000000	2000.000000	16.000000	12.200000	5.000000	2.000000
75%	3500.000000	5.700000	1080.000000	1920.000000	8.000000	3000.000000	32.000000	13.000000	8.000000	2.000000

```
# Replace values less than 1 with 1 GB in 'Internal Storage (GB)'
df['Internal storage (GB)'] = df['Internal storage (GB)'].apply(lambda x: 1 if x < 1 else x)
```

```
df.describe()
```

	Battery capacity (mAh)	Screen size (inches)	Resolution x	Resolution y	Processor	RAM (MB)	Internal storage (GB)	Rear camera	Front camera	Number of SIMs
count	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000
mean	2938.489330	5.291310	811.543046	1490.777778	5.551141	2488.777778	30.660044	12.070199	7.037969	1.833701
std	873.514133	0.671357	270.707271	557.780120	2.196562	1664.440386	36.946035	8.948337	6.295448	0.374457
min	1010.000000	2.400000	240.000000	320.000000	1.000000	64.000000	1.000000	0.000000	0.000000	1.000000
25%	2300.000000	5.000000	720.000000	1280.000000	4.000000	1000.000000	8.000000	8.000000	2.000000	2.000000
50%	3000.000000	5.200000	720.000000	1280.000000	4.000000	2000.000000	16.000000	12.200000	5.000000	2.000000
75%	3500.000000	5.700000	1080.000000	1920.000000	8.000000	3000.000000	32.000000	13.000000	8.000000	2.000000

```
df['Internal storage (GB)'].min()
```

```
1.0
```

```
df['RAM (MB)'].min()
```

```
64
```

```
df['Price'].mean()
```

```
11465.825607064018
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1359 entries, 0 to 1358
Data columns (total 21 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Name                    1359 non-null   object
 1   Brand                   1359 non-null   object
 2   Model                   1359 non-null   object
 3   Battery capacity (mAh)  1359 non-null   int64
 4   Screen size (inches)    1359 non-null   float64
 5   Touchscreen             1359 non-null   object
 6   Resolution x            1359 non-null   int64
 7   Resolution y            1359 non-null   int64
 8   Processor               1359 non-null   int64
 9   RAM (MB)                1359 non-null   int64
 10  Internal storage (GB)   1359 non-null   float64
 11  Rear camera             1359 non-null   float64
 12  Front camera            1359 non-null   float64
 13  Operating system        1359 non-null   object
 14  Wi-Fi                   1359 non-null   object
 15  Bluetooth               1359 non-null   object
 16  GPS                     1359 non-null   object
 17  Number of SIMs          1359 non-null   int64
 18  3G                      1359 non-null   object
 19  4G/ LTE                 1359 non-null   object
 20  Price                   1359 non-null   int64
dtypes: float64(4), int64(7), object(10)
memory usage: 223.1+ KB
```

```
df.isna().sum()
```

	0
Name	0
Brand	0
Model	0
Battery capacity (mAh)	0
Screen size (inches)	0
Touchscreen	0
Resolution x	0
Resolution y	0
Processor	0
RAM (MB)	0
Internal storage (GB)	0
Rear camera	0
Front camera	0
Operating system	0
Wi-Fi	0
Bluetooth	0
GPS	0
Number of SIMs	0
3G	0
4G/ LTE	0
Price	0

```
df.dropna(inplace=True)
```

```
df.isna().sum()
```

	0
Name	0
Brand	0
Model	0
Battery capacity (mAh)	0
Screen size (inches)	0
Touchscreen	0
Resolution x	0
Resolution y	0
Processor	0
RAM (MB)	0
Internal storage (GB)	0
Rear camera	0
Front camera	0
Operating system	0
Wi-Fi	0
Bluetooth	0
GPS	0
Number of SIMs	0
3G	0
4G/ LTE	0
Price	0

```
df.shape
```

```
(1359, 21)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1359 entries, 0 to 1358
Data columns (total 21 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Name                    1359 non-null   object
 1   Brand                   1359 non-null   object
 2   Model                   1359 non-null   object
 3   Battery capacity (mAh)  1359 non-null   int64
 4   Screen size (inches)    1359 non-null   float64
 5   Touchscreen             1359 non-null   object
 6   Resolution x            1359 non-null   int64
 7   Resolution y            1359 non-null   int64
 8   Processor               1359 non-null   int64
 9   RAM (MB)                1359 non-null   int64
 10  Internal storage (GB)   1359 non-null   float64
 11  Rear camera             1359 non-null   float64
 12  Front camera            1359 non-null   float64
 13  Operating system        1359 non-null   object
 14  Wi-Fi                   1359 non-null   object
 15  Bluetooth               1359 non-null   object
 16  GPS                     1359 non-null   object
 17  Number of SIMs          1359 non-null   int64
 18  3G                      1359 non-null   object
 19  4G/ LTE                 1359 non-null   object
 20  Price                   1359 non-null   int64
dtypes: float64(4), int64(7), object(10)
memory usage: 223.1+ KB
```

```
df.describe()
```

	Battery capacity (mAh)	Screen size (inches)	Resolution x	Resolution y	Processor	RAM (MB)	Internal storage (GB)	Rear camera	Front camera	Number of SIMs
count	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000	1359.000000
mean	2938.489330	5.291310	811.543046	1490.777778	5.551141	2488.777778	30.660044	12.070199	7.037969	1.833701
std	873.514133	0.671357	270.707271	557.780120	2.196562	1664.440386	36.946035	8.948337	6.295448	0.374457
min	1010.000000	2.400000	240.000000	320.000000	1.000000	64.000000	1.000000	0.000000	0.000000	1.000000
25%	2300.000000	5.000000	720.000000	1280.000000	4.000000	1000.000000	8.000000	8.000000	2.000000	2.000000
50%	3000.000000	5.200000	720.000000	1280.000000	4.000000	2000.000000	16.000000	12.200000	5.000000	2.000000
75%	3500.000000	5.700000	1080.000000	1920.000000	8.000000	3000.000000	32.000000	13.000000	8.000000	2.000000

```
df.count()
```

	0
Name	1359
Brand	1359
Model	1359
Battery capacity (mAh)	1359
Screen size (inches)	1359
Touchscreen	1359
Resolution x	1359
Resolution y	1359
Processor	1359
RAM (MB)	1359
Internal storage (GB)	1359
Rear camera	1359
Front camera	1359
Operating system	1359
Wi-Fi	1359
Bluetooth	1359
GPS	1359
Number of SIMs	1359
3G	1359
4G/ LTE	1359
Price	1359

```
#labelled index value change
df.reset_index(drop=True,inplace=True)
```

```
df
```

	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	RAM (MB)	...	Rear camera	Front camera	Ope
0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	12000	...	48.0	16.0	
1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	6000	...	64.0	16.0	
2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	4000	...	12.0	12.0	
3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	4000	...	12.0	12.0	
4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	6000	...	12.0	32.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1354	Intex Aqua A2	Intex	Aqua A2	1500	4.00	Yes	480	800	4	512	...	5.0	0.3	
1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.00	Yes	480	854	4	1000	...	8.0	5.0	
1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.50	Yes	480	854	2	512	...	5.0	2.0	
1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.00	Yes	480	800	1	256	...	2.0	0.3	
1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.00	Yes	480	854	4	1000	...	8.0	0.0	

1359 rows × 21 columns

feature engineering

```
for col in df:
 if df[col].dtype=='int' or 'float':
  print(df[col].value_counts())
  print('-----------------------------------')
  print()
```

```
Name
OnePlus 7T Pro McLaren Edition    1
Lava A77                          1
Karbonn Aura 4G                   1
Panasonic Eluga Ray               1
Panasonic P85                     1
                                 ..
Lenovo P780                       1
BlackBerry Q10                    1
BlackBerry Q5                     1
Lava Iris 504q                    1
iBall Andi Avonte 5               1
Name: count, Length: 1359, dtype: int64
-----------------------------------

Brand
Intex        117
Samsung      101
Micromax      71
Lava          59
Panasonic     55
             ...
Onida          1
Aqua           1
Jio            1
Razer          1
Philips        1
Name: count, Length: 76, dtype: int64
-----------------------------------

Model
V5               4
5                3
Z10              3
```

```
3                3
2                3
                ..
Camon i4         1
Moto E6s         1
S5               1
S5 Lite          1
Andi Avonte 5    1
Name: count, Length: 1321, dtype: int64
------------------------------------

Battery capacity (mAh)
3000    182
4000    145
2000    115
2500     91
5000     49
       ...
3250      1
3550      1
3730      1
5100      1
1960      1
```

```
for col in df:
 if df[col].dtype=='object':
  print(df[col].value_counts())
  print('------------------------------------')
  print()
```

```
Name
OnePlus 7T Pro McLaren Edition    1
Lava A77                          1
Karbonn Aura 4G                   1
Panasonic Eluga Ray               1
Panasonic P85                     1
                                 ..
Lenovo P780                       1
BlackBerry Q10                    1
BlackBerry Q5                     1
Lava Iris 504q                    1
iBall Andi Avonte 5               1
Name: count, Length: 1359, dtype: int64
------------------------------------

Brand
Intex        117
Samsung      101
Micromax      71
Lava          59
Panasonic     55
             ...
Onida          1
Aqua           1
Jio            1
Razer          1
Philips        1
Name: count, Length: 76, dtype: int64
------------------------------------

Model
V5               4
5                3
Z10              3
3                3
2                3
                ..
Camon i4         1
Moto E6s         1
S5               1
S5 Lite          1
Andi Avonte 5    1
Name: count, Length: 1321, dtype: int64
------------------------------------

Touchscreen
Yes    1342
No       17
Name: count, dtype: int64
------------------------------------

Operating system
Android       1299
Windows         19
iOS             17
Cyanogen        10
BlackBerry      10
Tizen            3
```

```
df
```

	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	RAM (MB)	...	Rear camera	Front camera	Ope
0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	12000	...	48.0	16.0	
1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	6000	...	64.0	16.0	
2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	4000	...	12.0	12.0	
3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	4000	...	12.0	12.0	
4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	6000	...	12.0	32.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1354	Intex Aqua A2	Intex	Aqua A2	1500	4.00	Yes	480	800	4	512	...	5.0	0.3	
1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.00	Yes	480	854	4	1000	...	8.0	5.0	
1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.50	Yes	480	854	2	512	...	5.0	2.0	
1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.00	Yes	480	800	1	256	...	2.0	0.3	
1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.00	Yes	480	854	4	1000	...	8.0	0.0	

1359 rows × 21 columns

```
df[['Name','Price']].groupby(by='Name').mean().sort_values(by='Price',ascending=True)
```

Name	Price
Lyf Water 7	494.0
LG K7	994.0
Gionee A1 Plus	994.0
Jio Phone	1249.0
Lava A32	1999.0
...	...
Samsung Galaxy S20 Ultra	92999.0
iPhone 11 Pro	96900.0
iPhone 11 Pro Max	106900.0
Samsung Galaxy Fold	164999.0
Samsung Galaxy Z Flip	174990.0

1359 rows × 1 columns

```
#highest price mobile
df[['Name','Price']].sort_values(by='Price',ascending=False)
```

	Name	Price
617	Samsung Galaxy Z Flip	174990
651	Samsung Galaxy Fold	164999
2	iPhone 11 Pro Max	106900
630	iPhone 11 Pro	96900
614	Samsung Galaxy S20 Ultra	92999
...	...	...
1112	Lyf Flame 6	1999
153	Jio Phone	1249
1225	LG K7	994
343	Gionee A1 Plus	994
1101	Lyf Water 7	494

1359 rows × 2 columns

```
df['Price'].max()
```

```
174990
```

```
df['Price'].min()
```

```
494
```

```
df['Price'].value_counts()
```

Price	count
4999	38
6999	32
3999	28
8999	28
7999	25
...	...
17172	1
19660	1
5555	1
21490	1
2498	1

627 rows × 1 columns

```
df['Brand'].nunique()
```

```
76
```

```
df[df['Brand']=='HP']
```

	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	RAM (MB)	...	Rear camera	Front camera	Operating system	W
1186	HP Elite x3	HP	Elite x3	4150	5.96	Yes	1440	2560	4	4000	...	15.0	8.0	Windows	Y

1 rows × 21 columns

```
df.groupby('Brand')['Price'].mean().sort_values(ascending=False)
```

Brand	Price
HP	88719.000000
Razer	54990.000000
Apple	45510.470588
Cat	45219.750000
Huawei	38564.750000
...	...
Tambo	3595.000000
Phicomm	3238.000000
Philips	3199.000000
Reach	2529.666667
Jio	1249.000000

76 rows × 1 columns

```
#highest price brand
df[['Brand','Price']].sort_values(by='Price',ascending=False)
```

	Brand	Price
617	Samsung	174990
651	Samsung	164999
2	Apple	106900
630	Apple	96900
614	Samsung	92999
...	...	...
1112	Lyf	1999
153	Jio	1249
1225	LG	994
343	Gionee	994
1101	Lyf	494

1359 rows × 2 columns

```
df[['Name','Price']].sort_values(by='Price',ascending=False)
```

	Name	Price
617	Samsung Galaxy Z Flip	174990
651	Samsung Galaxy Fold	164999
2	iPhone 11 Pro Max	106900
630	iPhone 11 Pro	96900
614	Samsung Galaxy S20 Ultra	92999
...	...	...
1112	Lyf Flame 6	1999
153	Jio Phone	1249
1225	LG K7	994
343	Gionee A1 Plus	994
1101	Lyf Water 7	494

1359 rows × 2 columns

```
#highest price model
df[['Model','Price']].sort_values(by='Price',ascending=False)
```

	Model	Price
617	Galaxy Z Flip	174990
651	Galaxy Fold	164999
2	iPhone 11 Pro Max	106900
630	iPhone 11 Pro	96900
614	Galaxy S20 Ultra	92999
...	...	...
1112	Flame 6	1999
153	Phone	1249
1225	K7	994
343	A1 Plus	994
1101	Water 7	494

1359 rows × 2 columns

```
#highest battery capacity mobile
df[['Name','Battery capacity (mAh)',]].sort_values(by='Battery capacity (mAh)',ascending=False)
```

	Name	Battery capacity (mAh)
78	Samsung Galaxy M30s	6000
8	Asus ROG Phone 2	6000
621	Tecno Spark Power	6000
157	Xiaomi Mi Max 2	5300
358	Lenovo P2	5100
...	...	...
587	Nokia Asha 501	1200
1166	Lava A52	1200
585	Samsung Galaxy Pocket	1200
908	Samsung Z4	1050
443	Nokia Asha 502	1010

1359 rows × 2 columns

```
#lowest battery capacity mobile
df[['Name','Battery capacity (mAh)']].sort_values(by='Battery capacity (mAh)')
```

	Name	Battery capacity (mAh)
443	Nokia Asha 502	1010
908	Samsung Z4	1050
585	Samsung Galaxy Pocket	1200
1166	Lava A52	1200
587	Nokia Asha 501	1200
...	...	...
358	Lenovo P2	5100
157	Xiaomi Mi Max 2	5300
621	Tecno Spark Power	6000
8	Asus ROG Phone 2	6000
78	Samsung Galaxy M30s	6000

1359 rows × 2 columns

```
#highest and lowest screen size mobile
df[['Name','Screen size (inches)']].sort_values(by='Screen size (inches)',ascending=False)
```

	Name	Screen size (inches)
651	Samsung Galaxy Fold	7.30
1238	Asus ZenPad C 7.0 (Z170MG)	7.00
614	Samsung Galaxy S20 Ultra	6.90
7	Samsung Galaxy Note 10+	6.80
629	Samsung Galaxy A70s	6.70
...	...	...
611	BlackBerry 9720	2.80
454	Samsung Galaxy Y Pro Duos	2.60
754	Nokia 8110 4G	2.45
591	BlackBerry Curve 9220	2.44
153	Jio Phone	2.40

1359 rows × 2 columns

```
#highest and lowest RAM mobile
df[['Name','RAM (MB)']].sort_values(by='RAM (MB)',ascending=False)
```

	Name	RAM (MB)
0	OnePlus 7T Pro McLaren Edition	12000
614	Samsung Galaxy S20 Ultra	12000
7	Samsung Galaxy Note 10+	12000
651	Samsung Galaxy Fold	12000
119	Oppo R17 Pro	8000
...	...	...
1320	Lava Flair P1i	256
1074	Lava A32	256
1357	iBall Andi4 B20	256
587	Nokia Asha 501	64
443	Nokia Asha 502	64

1359 rows × 2 columns

```
df[['Name','Number of SIMs']].sort_values(by='Number of SIMs')
```

	Name	Number of SIMs
220	HTC Desire Eye	1
238	Google Nexus 5	1
235	Gionee Elife E7	1
234	Sony Xperia Z1 Compact	1
233	Sony Xperia Z2	1
...	...	...
503	Vivo Y55s	2
502	Zopo Flash X Plus	2
500	Lava Z10	2
1358	iBall Andi Avonte 5	2
508	Coolpad Mega 3	3

1359 rows × 2 columns

```
#high battery capacity in budget models
df[['Name', 'Brand', 'Price', 'Battery capacity (mAh)']].sort_values(by='Battery capacity (mAh)', ascending=False)
```

	Name	Brand	Price	Battery capacity (mAh)
78	Samsung Galaxy M30s	Samsung	12999	6000
8	Asus ROG Phone 2	Asus	37999	6000
621	Tecno Spark Power	Tecno	8499	6000
157	Xiaomi Mi Max 2	Xiaomi	8999	5300
358	Lenovo P2	Lenovo	16999	5100
...	...	...	...	...
587	Nokia Asha 501	Nokia	4999	1200
1166	Lava A52	Lava	1999	1200
585	Samsung Galaxy Pocket	Samsung	3190	1200
908	Samsung Z4	Samsung	4790	1050
443	Nokia Asha 502	Nokia	3499	1010

1359 rows × 4 columns

```
#Large Screen Size in Premium Models
df[['Name', 'Brand', 'Price', 'Screen size (inches)']].sort_values(by='Screen size (inches)', ascending=False)
```

	Name	Brand	Price	Screen size (inches)
651	Samsung Galaxy Fold	Samsung	164999	7.30
1238	Asus ZenPad C 7.0 (Z170MG)	Asus	6199	7.00
614	Samsung Galaxy S20 Ultra	Samsung	92999	6.90
7	Samsung Galaxy Note 10+	Samsung	79699	6.80
629	Samsung Galaxy A70s	Samsung	25999	6.70
...	...	...	...	...
611	BlackBerry 9720	BlackBerry	7500	2.80
454	Samsung Galaxy Y Pro Duos	Samsung	5555	2.60
754	Nokia 8110 4G	Nokia	2999	2.45
591	BlackBerry Curve 9220	BlackBerry	6999	2.44
153	Jio Phone	Jio	1249	2.40

1359 rows × 4 columns

```
#High Megapixel Rear Cameras in Mid-Range Phones
df[['Name', 'Brand', 'Price', 'Rear camera']].sort_values(by='Rear camera', ascending=False)
```

	Name	Brand	Price	Rear camera
614	Samsung Galaxy S20 Ultra	Samsung	92999	108.0
68	Poco X2	Poco	15999	64.0
1	Realme X2 Pro	Realme	27999	64.0
75	Realme X2	Realme	16999	64.0
82	Redmi Note 8 Pro	Xiaomi	13999	64.0
...	...	...	...	...
1258	Intex Aqua Wave	Intex	2199	0.3
1266	Intex Aqua Play	Intex	4299	0.3
1137	Intex Aqua Joy	Intex	2666	0.3
922	Zopo Color M4	Zopo	4888	0.0
1000	Intex Aqua Power M	Intex	3499	0.0

1359 rows × 4 columns

```
df[['Name', 'Brand', 'Price', 'RAM (MB)', 'Screen size (inches)']].sort_values(by=['RAM (MB)', 'Screen size (inches)'], ascending=False)
```

	Name	Brand	Price	RAM (MB)	Screen size (inches)
651	Samsung Galaxy Fold	Samsung	164999	12000	7.30
614	Samsung Galaxy S20 Ultra	Samsung	92999	12000	6.90
7	Samsung Galaxy Note 10+	Samsung	79699	12000	6.80
0	OnePlus 7T Pro McLaren Edition	OnePlus	58998	12000	6.67
69	Samsung Galaxy S10 Lite	Samsung	39999	8000	6.70
...	...	...	...	...	...
1357	iBall Andi4 B20	iBall	2498	256	4.00
588	Nokia Lumia 610	Nokia	4990	256	3.70
1347	iBall Andi 3.5V Genius2	iBall	2499	256	3.50
443	Nokia Asha 502	Nokia	3499	64	3.00
587	Nokia Asha 501	Nokia	4999	64	3.00

1359 rows × 5 columns

```
df['Touchscreen'].value_counts()
```

Touchscreen	count
Yes	1342
No	17

```
df['Operating system'].value_counts()
```

Operating system	count
Android	1299
Windows	19
iOS	17
Cyanogen	10
BlackBerry	10
Tizen	3
Sailfish	1

```
df['Wi-Fi'].value_counts()
```

Wi-Fi	count
Yes	1351
No	8

```
df['Bluetooth'].value_counts()
```

Bluetooth	count
Yes	1344
No	15

```
df['GPS'].value_counts()
```

GPS	count
Yes	1251
No	108

```
df['3G'].value_counts()
```

3G	count
Yes	1214
No	145

```
df['4G/ LTE'].value_counts()
```

4G/ LTE	count
Yes	1012
No	347

```
df['Battery capacity (mAh)'].min()
```

```
1010
```

```
df['Price'].sort_values(ascending=False)
```

	Price
617	174990
651	164999
2	106900
630	96900
614	92999
...	...
1112	1999
153	1249
1225	994
343	994
1101	494

1359 rows × 1 columns

```
df['Price'].sort_values()
```

	Price
1101	494
343	994
1225	994
153	1249
1089	1999
...	...
614	92999
630	96900
2	106900
651	164999
617	174990

1359 rows × 1 columns

```
df['price_segment']=0
df['price_segment'][df['Price']>75000]='Premium'
df['price_segment'][df['Price']<30000]='Budget'
df['price_segment'][(df['Price']>=30000)&(df['Price']<=75000)]='Mid-Range'
df
```

```
<ipython-input-64-7f4b41fe6534>:2: FutureWarning: ChainedAssignmentError: behaviour will change in pandas 3.0!
You are setting values through chained assignment. Currently this works in certain cases, but when using Copy-on-Write (which will b
A typical example is when you are setting values in a column of a DataFrame, like:

df["col"][row_indexer] = value

Use `df.loc[row_indexer, "col"] = values` instead, to perform the assignment in a single step and ensure this keeps updating the ori

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus

  df['price_segment'][df['Price']>75000]='Premium'
<ipython-input-64-7f4b41fe6534>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  df['price_segment'][df['Price']>75000]='Premium'
<ipython-input-64-7f4b41fe6534>:2: FutureWarning: Setting an item of incompatible dtype is deprecated and will raise an error in a f
  df['price_segment'][df['Price']>75000]='Premium'
```

	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	RAM (MB)	...	Front camera	Operating system
0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	12000	...	16.0	Android
1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	6000	...	16.0	Android
2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	4000	...	12.0	iOS
3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	4000	...	12.0	iOS
4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	6000	...	32.0	Android
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1354	Intex Aqua A2	Intex	Aqua A2	1500	4.00	Yes	480	800	4	512	...	0.3	Android
1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.00	Yes	480	854	4	1000	...	5.0	Android
1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.50	Yes	480	854	2	512	...	2.0	Android
1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.00	Yes	480	800	1	256	...	0.3	Android
1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.00	Yes	480	854	4	1000	...	0.0	Android

```
df['Bluetooth'].value_counts()
```

Bluetooth	count
Yes	1344
No	15

```
df['price_segment'].value_counts()
```

price_segment	count
Budget	1274
Mid-Range	76
Premium	9

```
df['Brand'].value_counts()
```

Brand	count
Intex	117
Samsung	101
Micromax	71
Lava	59
Panasonic	55
...	...
Onida	1
Aqua	1
Jio	1
Razer	1
Philips	1

76 rows × 1 columns

univariate analysis

```
df.columns
```

```
Index(['Name', 'Brand', 'Model', 'Battery capacity (mAh)',
       'Screen size (inches)', 'Touchscreen', 'Resolution x', 'Resolution y',
       'Processor', 'RAM (MB)', 'Internal storage (GB)', 'Rear camera',
       'Front camera', 'Operating system', 'Wi-Fi', 'Bluetooth', 'GPS',
       'Number of SIMs', '3G', '4G/ LTE', 'Price', 'price_segment'],
      dtype='object')
```

```
for col in df:
 if df[col].dtype=='object':
  print(df[col].value_counts())
  print('-----------------------------------')
  print()
```

```
Name
OnePlus 7T Pro McLaren Edition    1
Lava A77                          1
Karbonn Aura 4G                   1
Panasonic Eluga Ray               1
Panasonic P85                     1
                                 ..
Lenovo P780                       1
BlackBerry Q10                    1
BlackBerry Q5                     1
Lava Iris 504q                    1
iBall Andi Avonte 5               1
Name: count, Length: 1359, dtype: int64
-----------------------------------

Brand
Intex        117
Samsung      101
Micromax      71
Lava          59
Panasonic     55
             ...
Onida          1
Aqua           1
Jio            1
Razer          1
Philips        1
Name: count, Length: 76, dtype: int64
-----------------------------------

Model
V5               4
5                3
Z10              3
3                3
2                3
                ..
Camon i4         1
Moto E6s         1
S5               1
S5 Lite          1
```

```
Andi Avonte 5    1
Name: count, Length: 1321, dtype: int64
------------------------------------

Touchscreen
Yes    1342
No       17
Name: count, dtype: int64
------------------------------------

Operating system
Android       1299
Windows         19
iOS             17
Cyanogen        10
BlackBerry      10
Tizen            3
```

```
for col in df:
 if df[col].dtype=='int' or 'float':
  print(df[col].value_counts())
  print('------------------------------------')
  print()
```

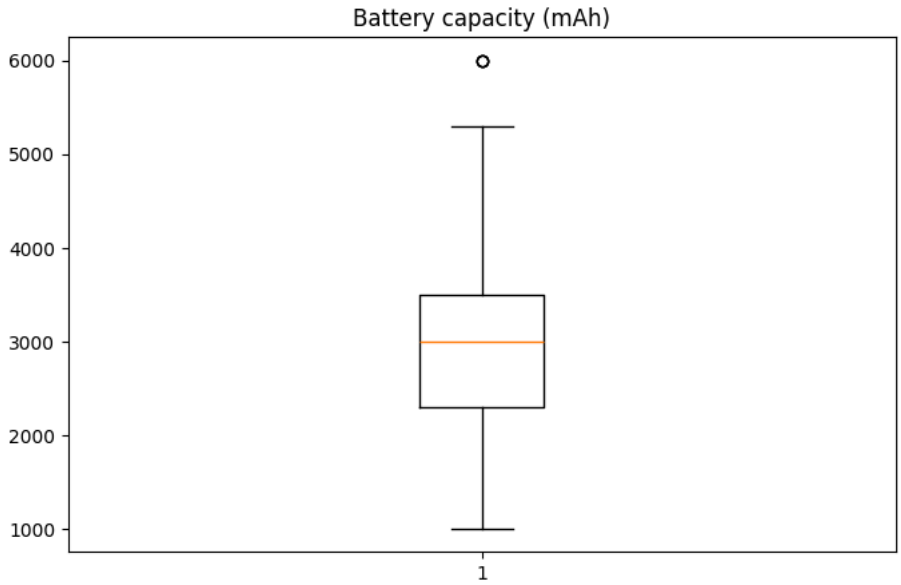
```
Name
OnePlus 7T Pro McLaren Edition    1
Lava A77                          1
Karbonn Aura 4G                   1
Panasonic Eluga Ray               1
Panasonic P85                     1
                                 ..
Lenovo P780                       1
BlackBerry Q10                    1
BlackBerry Q5                     1
Lava Iris 504q                    1
iBall Andi Avonte 5               1
Name: count, Length: 1359, dtype: int64
------------------------------------

Brand
Intex        117
Samsung      101
Micromax      71
Lava          59
Panasonic     55
            ...
Onida          1
Aqua           1
Jio            1
Razer          1
Philips        1
Name: count, Length: 76, dtype: int64
------------------------------------

Model
V5               4
5                3
Z10              3
3                3
2                3
                ..
Camon i4         1
Moto E6s         1
S5               1
S5 Lite          1
Andi Avonte 5    1
Name: count, Length: 1321, dtype: int64
------------------------------------

Battery capacity (mAh)
3000    182
4000    145
2000    115
2500     91
5000     49
       ...
3250      1
3550      1
3730      1
5100      1
1960      1
Name: count, Length: 165, dtype: int64
```

univariate analysis of numerical columns

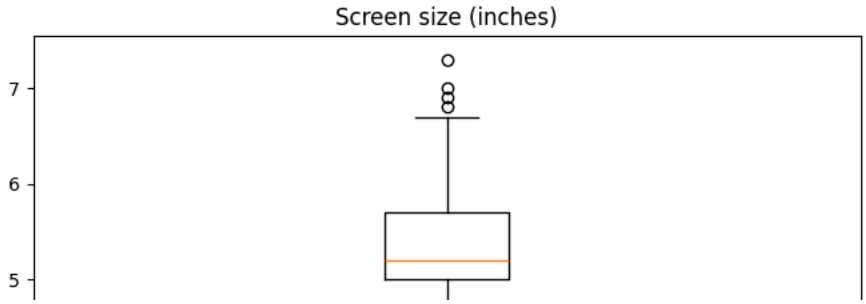
```
for col in df:
  if df[col].dtype=='int' or df[col].dtype=='float':
```

```
    print(col.capitalize())
    print('-------------------------')
    print(f'Mean:{df[col].mean()}')
    print(f'Median:{df[col].median()}')

    print(f'Minimum of {col}:{df[col].min()}')
    print(f'Maximum of {col}: {df[col].max()}')
    print(f'Variance of {col}: {df[col].var()}')
    print(f'Standard deviation:{df[col].std()}')
    plt.figure(figsize=(8,5))
    plt.boxplot(df[col])
    plt.title(col)
    plt.show()
    print('------------------------------------------------------------------------------------------')
    print()
```

```
Battery capacity (mah)
-------------------------
Mean:2938.4893303899926
Median:3000.0
Minimum of Battery capacity (mAh):1010
Maximum of Battery capacity (mAh): 6000
Variance of Battery capacity (mAh): 763026.9407918218
Standard deviation:873.5141331379943
```




```
------------------------------------------------------------------------------------------

Screen size (inches)
-------------------------
Mean:5.2913097866078
Median:5.2
Minimum of Screen size (inches):2.4
Maximum of Screen size (inches): 7.3
Variance of Screen size (inches): 0.45071963812947974
Standard deviation:0.6713565655666739
```




```
df.columns
```

```
Index(['Name', 'Brand', 'Model', 'Battery capacity (mAh)',
       'Screen size (inches)', 'Touchscreen', 'Resolution x', 'Resolution y',
       'Processor', 'RAM (MB)', 'Internal storage (GB)', 'Rear camera',
       'Front camera', 'Operating system', 'Wi-Fi', 'Bluetooth', 'GPS',
       'Number of SIMs', '3G', '4G/ LTE', 'Price', 'price_segment'],
      dtype='object')
```

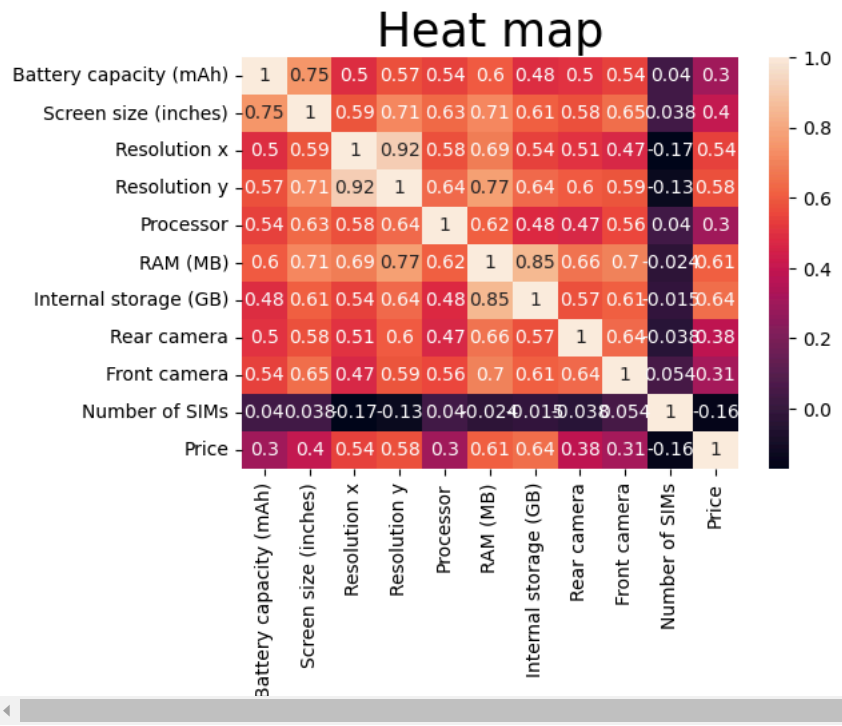
```
df[['Screen size (inches)','Price']].groupby('Screen size (inches)').agg(['mean','median','max','min'])
```

	Price			
	mean	median	max	min
Screen size (inches)				
2.40	1249.000000	1249.0	1249	1249
2.44	6999.000000	6999.0	6999	6999
2.45	2999.000000	2999.0	2999	2999
2.60	5555.000000	5555.0	5555	5555
2.80	5345.000000	5345.0	7500	3190
...	...	...	...	...
6.70	58896.428571	38999.0	174990	22299
6.80	79699.000000	79699.0	79699	79699
6.90	92999.000000	92999.0	92999	92999
7.00	6199.000000	6199.0	6199	6199
7.30	164999.000000	164999.0	164999	164999

80 rows × 4 columns

heat map

```
plt.figure(figsize=(6,4))
plt.title('Heat map',size=25)
sns.heatmap(df.corr(numeric_only=True),annot=True)
plt.show()
```

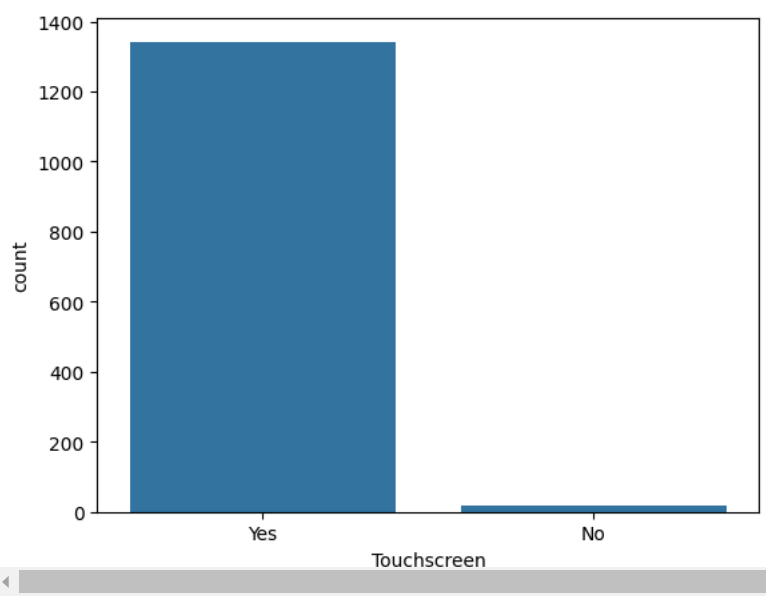



```
df.columns
```

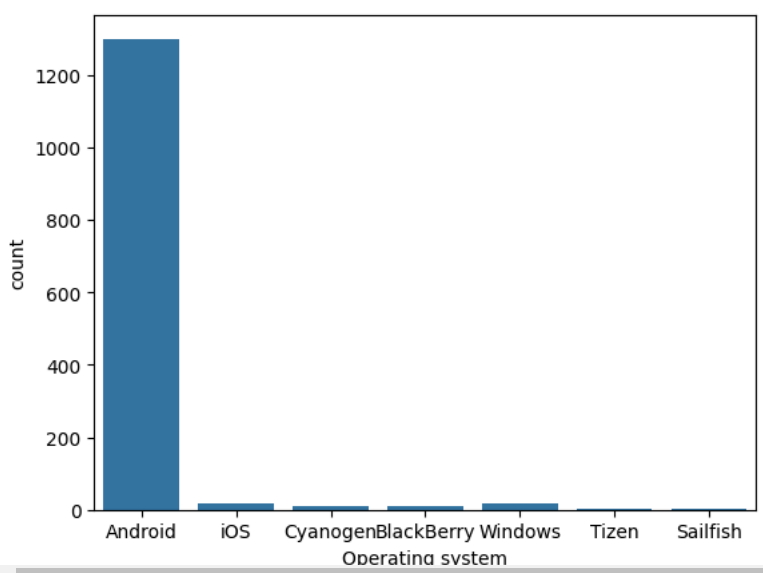
```
Index(['Name', 'Brand', 'Model', 'Battery capacity (mAh)',
       'Screen size (inches)', 'Touchscreen', 'Resolution x', 'Resolution y',
       'Processor', 'RAM (MB)', 'Internal storage (GB)', 'Rear camera',
       'Front camera', 'Operating system', 'Wi-Fi', 'Bluetooth', 'GPS',
       'Number of SIMs', '3G', '4G/ LTE', 'Price', 'price_segment'],
      dtype='object')
```

countplot

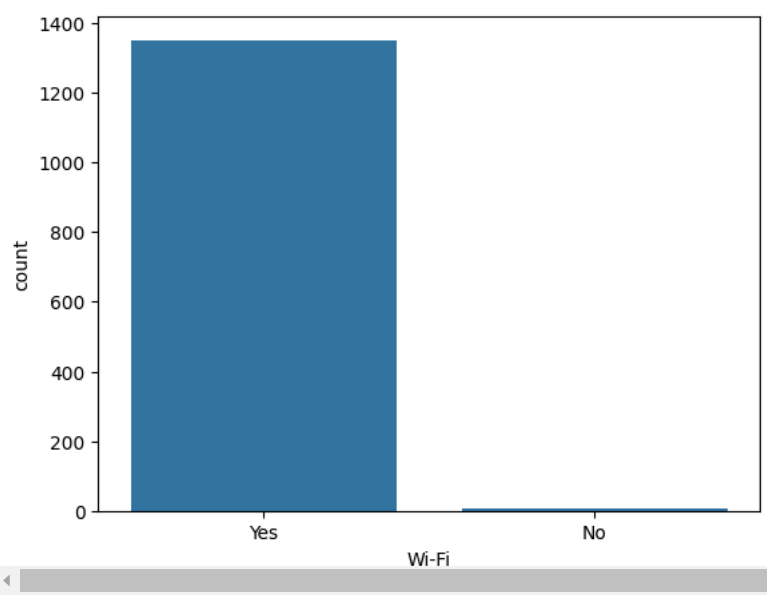
```
sns.countplot(data=df,x='Touchscreen')
plt.show()
```



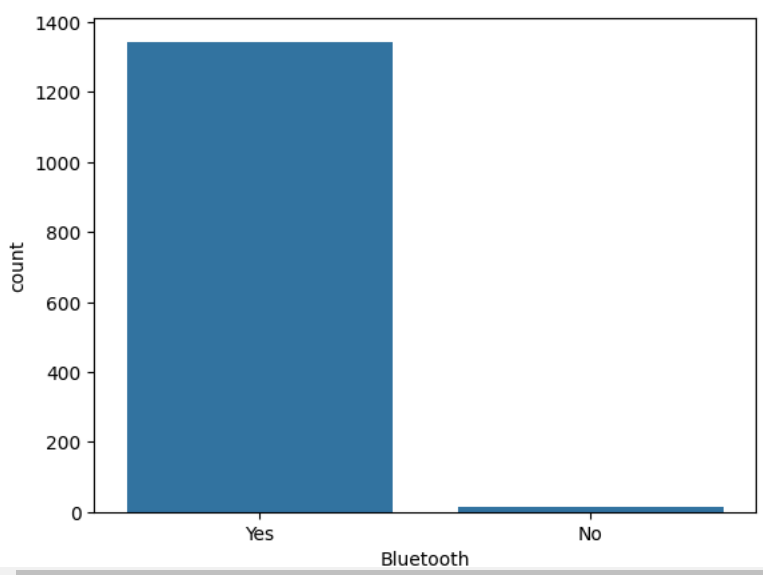

```
sns.countplot(data=df,x='Operating system')
plt.show()
```



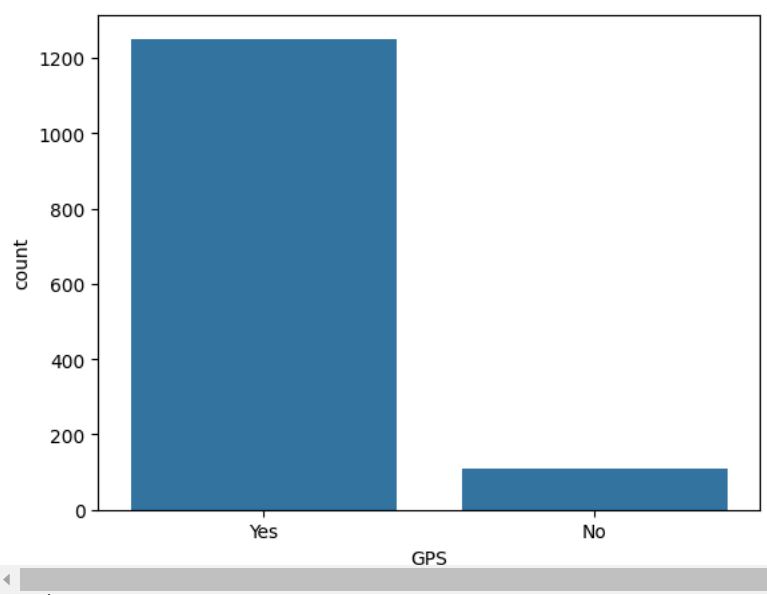

```
sns.countplot(data=df,x='Wi-Fi')
plt.show()
```



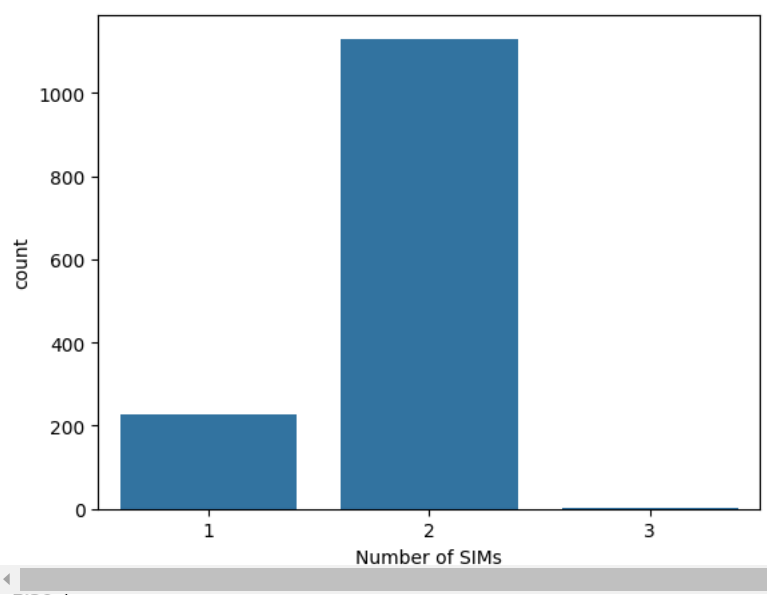

```
sns.countplot(data=df,x='Bluetooth')
plt.show()
```



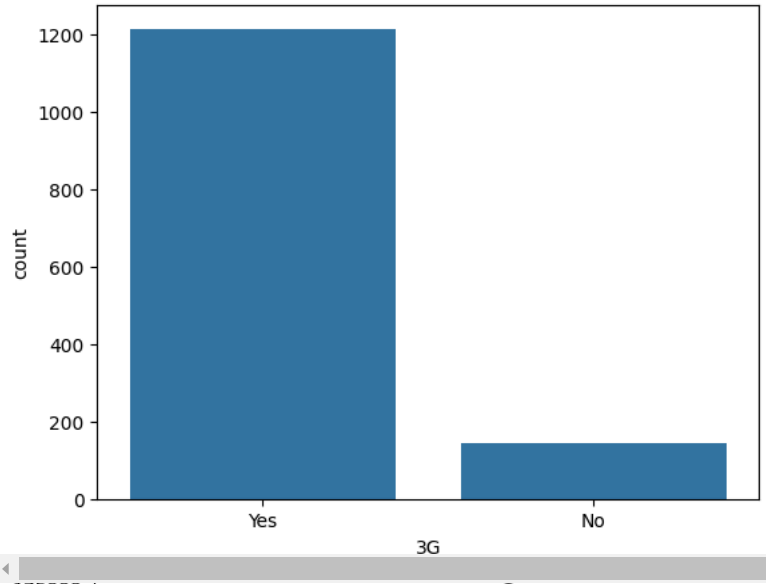

```
sns.countplot(data=df,x='GPS')
plt.show()
```



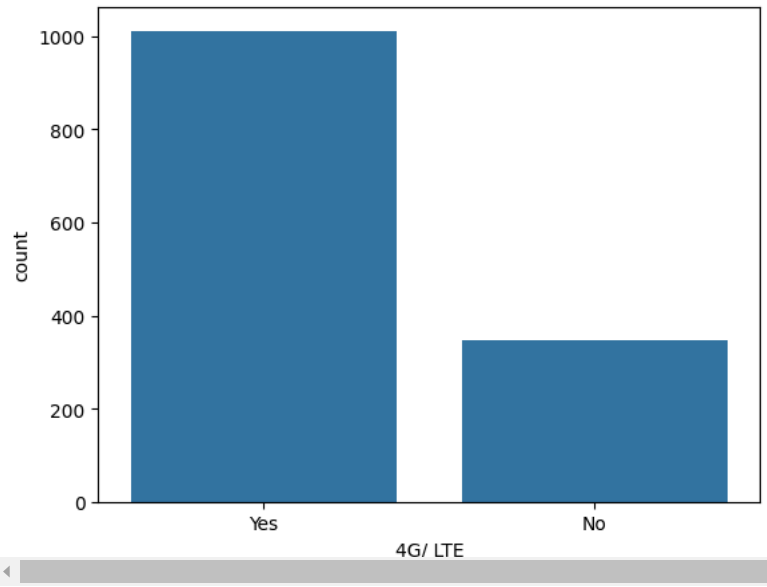

```
sns.countplot(data=df,x='Number of SIMs')
plt.show()
```



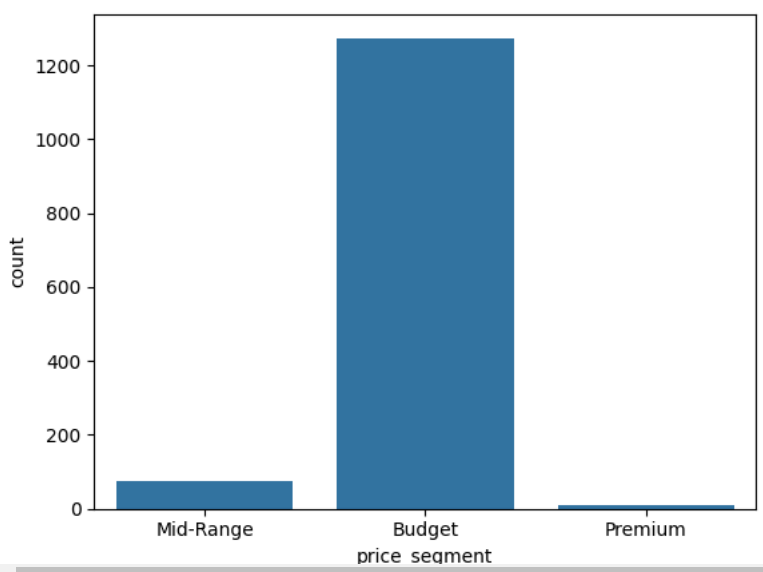

```
sns.countplot(data=df,x='3G')
plt.show()
```




```
sns.countplot(data=df,x='4G/ LTE')
plt.show()
```




```
sns.countplot(data=df,x='price_segment')
plt.show()
```



pie plot

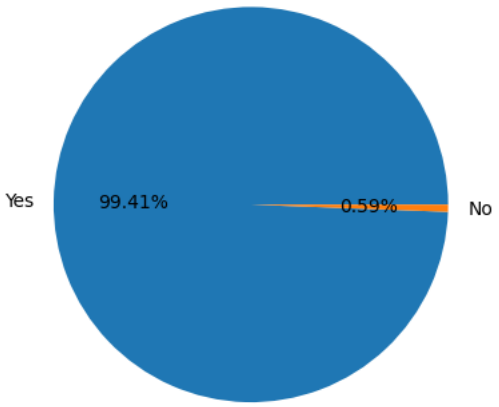
```
df
```

	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	RAM (MB)	...	Front camera	Operating system
0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	12000	...	16.0	Android
1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	6000	...	16.0	Android
2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	4000	...	12.0	iOS
3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	4000	...	12.0	iOS
4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	6000	...	32.0	Android
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1354	Intex Aqua A2	Intex	Aqua A2	1500	4.00	Yes	480	800	4	512	...	0.3	Android
1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.00	Yes	480	854	4	1000	...	5.0	Android
1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.50	Yes	480	854	2	512	...	2.0	Android
1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.00	Yes	480	800	1	256	...	0.3	Android
1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.00	Yes	480	854	4	1000	...	0.0	Android

1359 rows × 22 columns

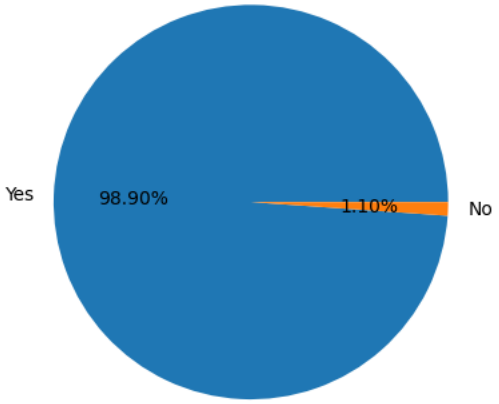
```
plt.pie(df['Wi-Fi'].value_counts(),labels=df['Wi-Fi'].unique(),autopct='%1.2f%%')
plt.plot()
```

```
[]
```



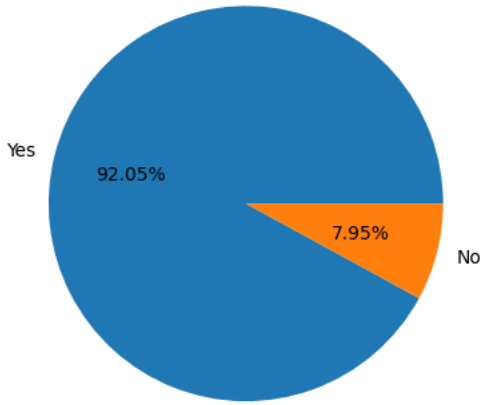

```
plt.pie(df['Bluetooth'].value_counts(),labels=df['Bluetooth'].unique(),autopct='%1.2f%%')
plt.plot()
```

[]



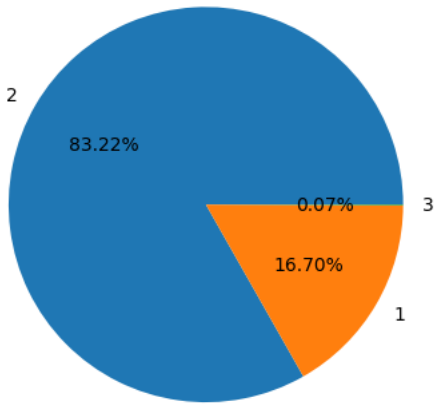

```
plt.pie(df['GPS'].value_counts(),labels=df['GPS'].unique(),autopct='%1.2f%%')
plt.plot()
```

[]



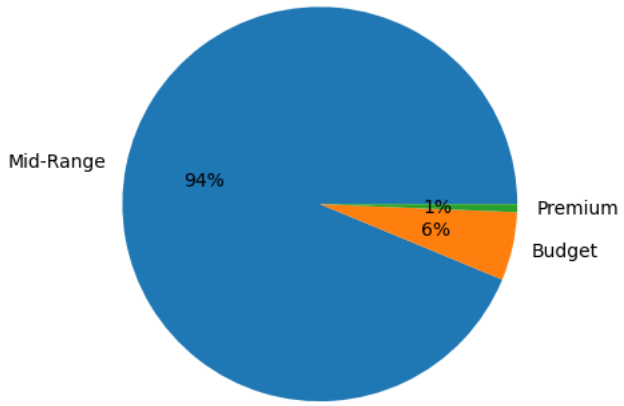

```
plt.pie(df['Number of SIMs'].value_counts(),labels=df['Number of SIMs'].unique(),autopct='%1.2f%%')
plt.plot()
```

[]




```
plt.pie(df['price_segment'].value_counts(),labels=df['price_segment'].unique(),autopct='%1.f%%')
plt.plot()
```

```
[]
```



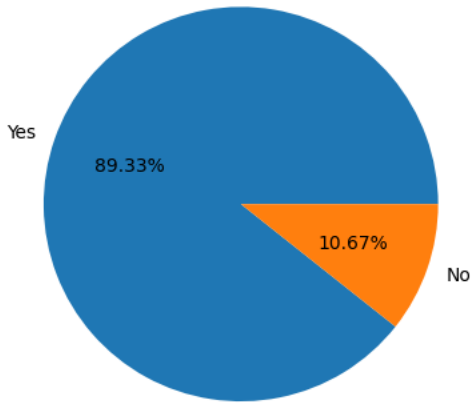

```
df
```

	Name	Brand	Model	Battery capacity (mAh)	Screen size (inches)	Touchscreen	Resolution x	Resolution y	Processor	RAM (MB)	...	Front camera	Operating system
0	OnePlus 7T Pro McLaren Edition	OnePlus	7T Pro McLaren Edition	4085	6.67	Yes	1440	3120	8	12000	...	16.0	Android
1	Realme X2 Pro	Realme	X2 Pro	4000	6.50	Yes	1080	2400	8	6000	...	16.0	Android
2	iPhone 11 Pro Max	Apple	iPhone 11 Pro Max	3969	6.50	Yes	1242	2688	6	4000	...	12.0	iOS
3	iPhone 11	Apple	iPhone 11	3110	6.10	Yes	828	1792	6	4000	...	12.0	iOS
4	LG G8X ThinQ	LG	G8X ThinQ	4000	6.40	Yes	1080	2340	8	6000	...	32.0	Android
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1354	Intex Aqua A2	Intex	Aqua A2	1500	4.00	Yes	480	800	4	512	...	0.3	Android
1355	Videocon Infinium Z51 Nova+	Videocon	Infinium Z51 Nova+	2000	5.00	Yes	480	854	4	1000	...	5.0	Android
1356	Intex Aqua Y4	Intex	Aqua Y4	1700	4.50	Yes	480	854	2	512	...	2.0	Android
1357	iBall Andi4 B20	iBall	Andi4 B20	1250	4.00	Yes	480	800	1	256	...	0.3	Android
1358	iBall Andi Avonte 5	iBall	Andi Avonte 5	2150	5.00	Yes	480	854	4	1000	...	0.0	Android

1359 rows × 22 columns

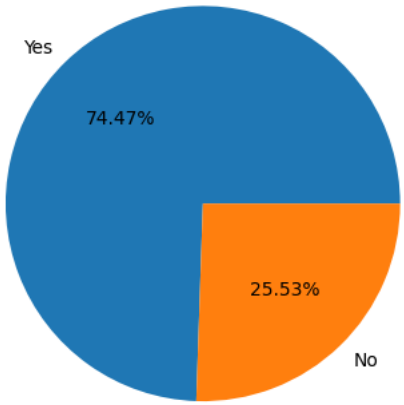
```
plt.pie(df['3G'].value_counts(),labels=df['3G'].unique(),autopct='%1.2f%%')
plt.plot()
```

```
[]
```



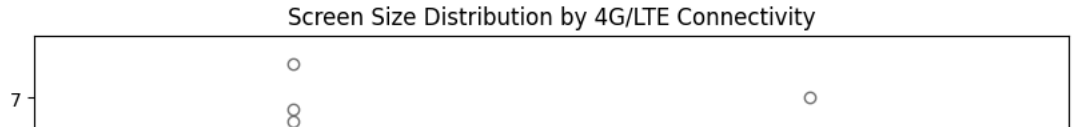

```
plt.pie(df['4G/ LTE'].value_counts(),labels=df['4G/ LTE'].unique(),autopct='%1.2f%%')
plt.plot()
```

```
[]
```




```
plt.figure(figsize=(10, 6))
sns.boxplot(x='4G/ LTE', y='Screen size (inches)', data=df, palette='pastel')
plt.title('Screen Size Distribution by 4G/LTE Connectivity')
plt.xlabel('4G/LTE Support')
plt.ylabel('Screen Size (inches)')
plt.show()
```

```
<ipython-input-94-957d153ab53f>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.boxplot(x='4G/ LTE', y='Screen size (inches)', data=df, palette='pastel')
```




```
plt.figure(figsize=(14, 8))
avg_price_by_brand = df.groupby('Brand')['Price'].mean().sort_values()
sns.barplot(x=avg_price_by_brand.index, y=avg_price_by_brand.values, palette='viridis')
plt.xticks(rotation=90)
plt.title('Average Price by Brand')
plt.xlabel('Brand')
plt.ylabel('Average Price')
plt.show()
```

```
<ipython-input-95-16492c31914d>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(x=avg_price_by_brand.index, y=avg_price_by_brand.values, palette='viridis')
```