```
#load the file
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/train.csv')
df
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...
0	842	0	2.2	0	1	0	7	0.6	188	2	..
1	1021	1	0.5	1	0	1	53	0.7	136	3	..
2	563	1	0.5	1	2	1	41	0.9	145	5	..
3	615	1	2.5	0	0	0	10	0.8	131	6	..
4	1821	1	1.2	0	13	1	44	0.6	141	2	..
...	...	...	...	...	...	...	...	...	...	...	..
1995	794	1	0.5	1	0	1	2	0.8	106	6	..
1996	1965	1	2.6	1	0	0	39	0.2	187	4	..
1997	1911	0	0.9	1	1	1	36	0.7	108	8	..
1998	1512	0	0.9	0	4	1	46	0.1	145	5	..
1999	510	1	2.0	1	5	1	45	0.9	168	6	..

2000 rows × 21 columns

```
#First 5 rows of data
df.head()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	p
0	842	0	2.2	0	1	0	7	0.6	188	2	...	
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	
2	563	1	0.5	1	2	1	41	0.9	145	5	...	
3	615	1	2.5	0	0	0	10	0.8	131	6	...	
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	

5 rows × 21 columns

```
#Last 5 rows of data
df.tail()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...
1995	794	1	0.5	1	0	1	2	0.8	106	6	..
1996	1965	1	2.6	1	0	0	39	0.2	187	4	..
1997	1911	0	0.9	1	1	1	36	0.7	108	8	..
1998	1512	0	0.9	0	4	1	46	0.1	145	5	..
1999	510	1	2.0	1	5	1	45	0.9	168	6	..

5 rows × 21 columns

```
#Shape of the dataset
df.shape
```

```
(2000, 21)
```

```
#Description of the dataset
df.describe()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_
count	2000.000000	2000.0000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000
mean	1238.518500	0.4950	1.522250	0.509500	4.309500	0.521500	32.046500	0.501
std	439.418206	0.5001	0.816004	0.500035	4.341444	0.499662	18.145715	0.288
min	501.000000	0.0000	0.500000	0.000000	0.000000	0.000000	2.000000	0.100
25%	851.750000	0.0000	0.700000	0.000000	1.000000	0.000000	16.000000	0.200
50%	1226.000000	0.0000	1.500000	1.000000	3.000000	1.000000	32.000000	0.500
75%	1615.250000	1.0000	2.200000	1.000000	7.000000	1.000000	48.000000	0.800
max	1998.000000	1.0000	3.000000	1.000000	19.000000	1.000000	64.000000	1.000

8 rows × 21 columns

```
#checking null values of the data
df.isnull().sum()
```

	0
battery_power	0
blue	0
clock_speed	0
dual_sim	0
fc	0
four_g	0
int_memory	0
m_dep	0
mobile_wt	0
n_cores	0
pc	0
px_height	0
px_width	0
ram	0
sc_h	0
sc_w	0
talk_time	0
three_g	0
touch_screen	0
wifi	0
price_range	0

dtype: int64

```
#Info of the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   battery_power  2000 non-null   int64  
 1   blue           2000 non-null   int64  
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64  
 4   fc             2000 non-null   int64  
 5   four_g         2000 non-null   int64  
 6   int_memory     2000 non-null   int64  
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64  
 9   n_cores        2000 non-null   int64  
 10  pc             2000 non-null   int64  
 11  px_height      2000 non-null   int64  
 12  px_width       2000 non-null   int64  
 13  ram            2000 non-null   int64
 14  sc_h           2000 non-null   int64
 15  sc_w           2000 non-null   int64
 16  talk_time      2000 non-null   int64
 17  three_g        2000 non-null   int64
 18  touch_screen   2000 non-null   int64
 19  wifi           2000 non-null   int64
 20  price_range    2000 non-null   int64
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

```
#change column Names
df=df.rename(columns={"blue":"bluetooth","clock_speed":"time_speed","touch_screen":"smart_screen"})
df.head()
```

	battery_power	bluetooth	time_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	..
0	842	0	2.2	0	1	0	7	0.6	188	2	.
1	1021	1	0.5	1	0	1	53	0.7	136	3	.
2	563	1	0.5	1	2	1	41	0.9	145	5	.
3	615	1	2.5	0	0	0	10	0.8	131	6	.
4	1821	1	1.2	0	13	1	44	0.6	141	2	.

5 rows × 21 columns

```
#change the data
mobile_df=pd.DataFrame()
mobile_df["bluetooth"]=np.where(df["bluetooth"]<1,"No","yes")
mobile_df["dual_sim"]=np.where(df["dual_sim"]<1,"No","yes")
mobile_df["four_g"]=np.where(df["four_g"]<1,"No","yes")
mobile_df["three_g"]=np.where(df["three_g"]<1,"No","yes")
mobile_df["smart_screen"]=np.where(df["smart_screen"]<1,"No","yes")
mobile_df["wifi"]=np.where(df["wifi"]<1,"No","yes")
print("Column Data has updated Successfully")
```

```
Column Data has updated Successfully
```

```
mobile_df.head()
```

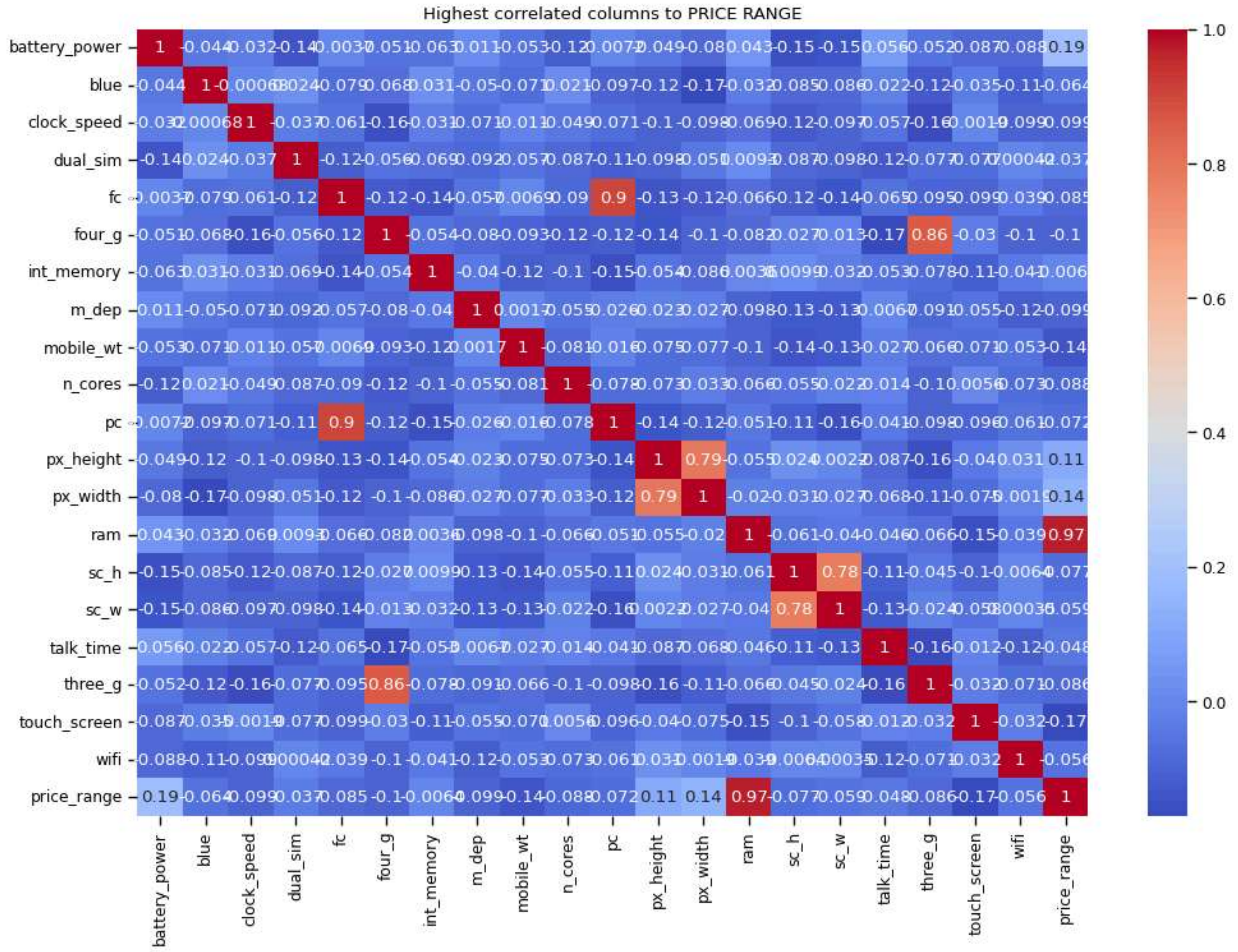
```
df["price_range"].replace({"Low Cost":1, "Median Cost":2, "High Cost":3})
print("Column Data has updated Successfully")
```

```
Column Data has updated Successfully
```

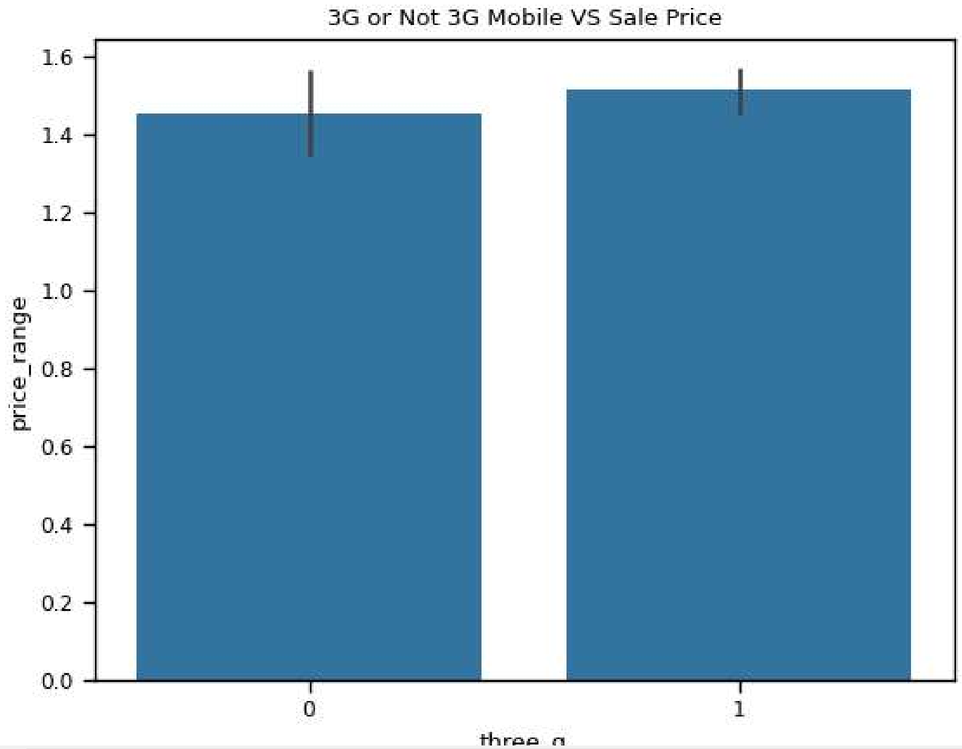
```
df.head()
```

_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	..
2.2	0	1	0	7	0.6	188	2	..
0.5	1	0	1	53	0.7	136	3	..
0.5	1	2	1	41	0.9	145	5	..
2.5	0	0	0	10	0.8	131	6	..
1.2	0	13	1	44	0.6	141	2	..

```
#VISUALIZATION
#Get the highest correlated columns to price range
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(12,8))
sns.heatmap(mobile_corr.corr(),cmap="coolwarm",annot=True)
plt.title("Highest correlated columns to PRICE RANGE")
plt.show()
```



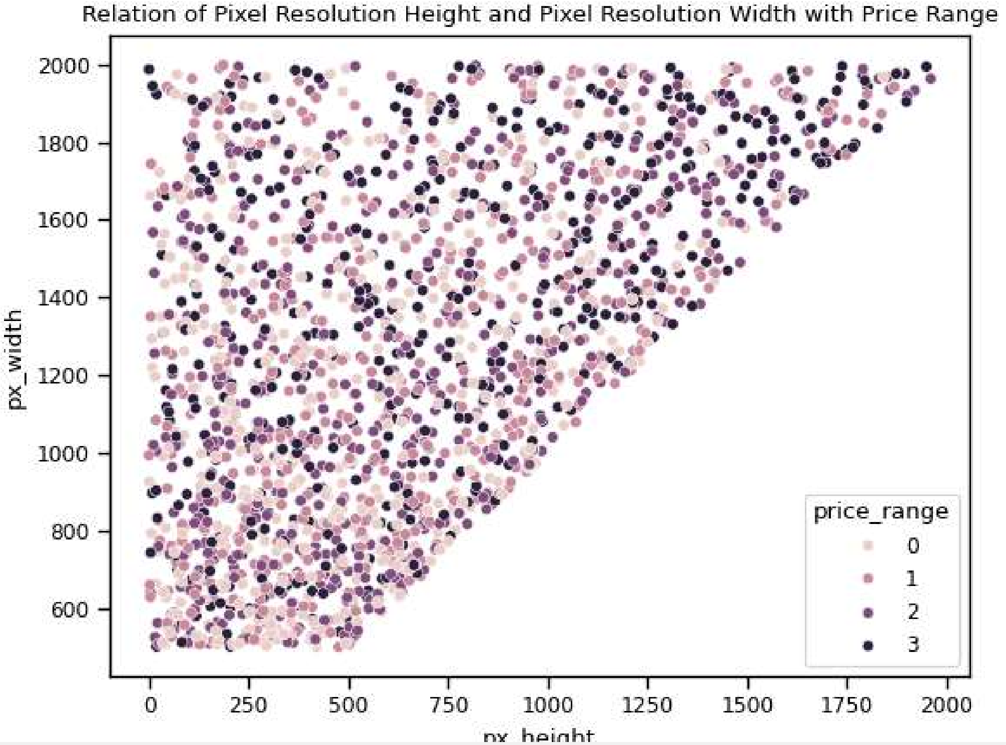
```
#show the 3G or not 3G mobile vs sale price using bar plot
sns.barplot(x='three_g',y='price_range',data=df)
plt.title("3G or Not 3G Mobile VS Sale Price")
plt.show()
```



```
#Show the count plot for supporting bluetooth or not vs price
sns.countplot(x='bluetooth',hue='price_range',data=df)
plt.title("Supporting Bluetooth or Not VS Price")
plt.show()
```

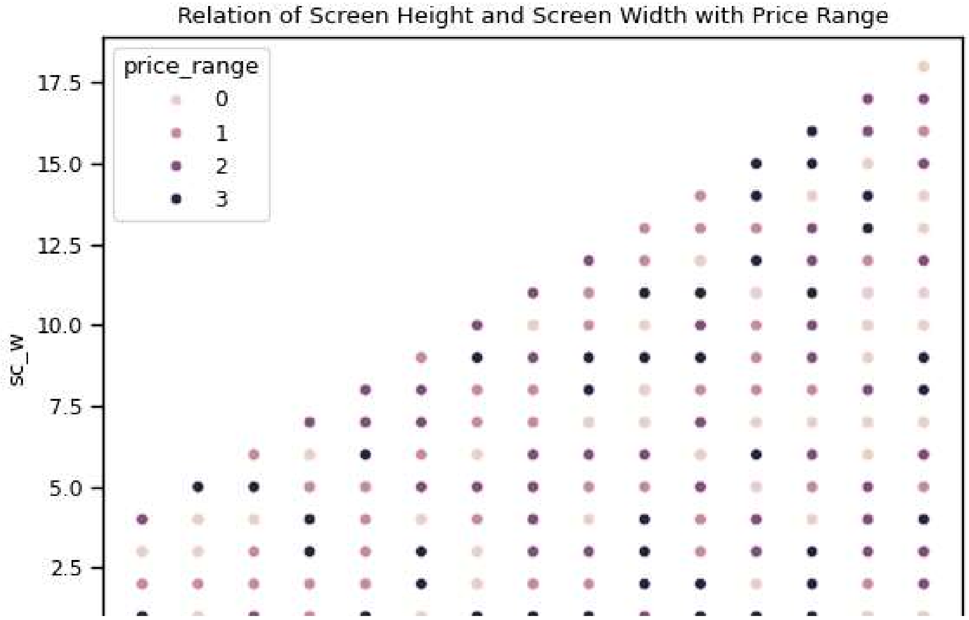
Supporting Bluetooth or Not VS Price

```
#Use scatterplot to show the relation of pixel resolution height and pixel resolution width
 #with price range
sns.scatterplot(x="px_height",y="px_width",hue="price_range",data=df)
plt.title("Relation of Pixel Resolution Height and Pixel Resolution Width with Price Range")
plt.show()
```



Relation of Pixel Resolution Height and Pixel Resolution Width with Price Range

```
#Use scatterplot to show the relation of screen height and screen width
#with price ranges
sns.scatterplot(x="sc_h",y="sc_w",hue='price_range',data=df)
plt.title("Relation of Screen Height and Screen Width with Price Range")
plt.show()
```



Relation of Screen Height and Screen Width with Price Range