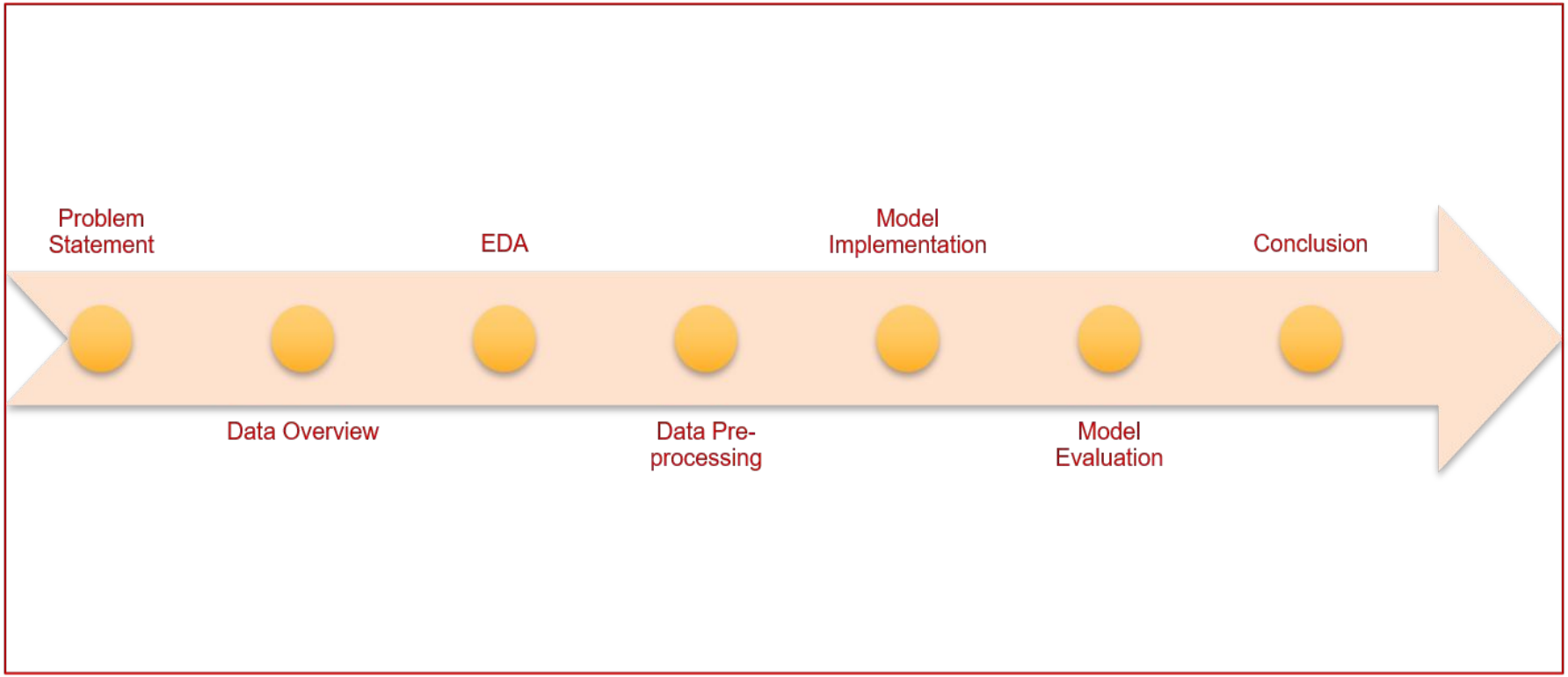
Capstone Project-3

Mobile Price Range Prediction

Prepared By
Akshay Nikam

Points for Discussion



Problem Statement

- In the competitive mobile phone market companies want to understand sales data of mobile phones and factors which drive the prices. The objective is to find out some relation between features of a mobile phone(eg:- RAM, Internal Memory, etc) and its selling price. In this problem, we do not have to predict the actual price but a price range indicating how high the price is.

- Perform Classification analysis using multiple models and predict the Mobile Price Range and in the end will compare the evaluation metrics for all different models to find the best model.

Data Overview

In this dataset We have 21 columns and 2000 rows which includes following columns:

Columns

Battery_power - Total energy a battery can store in one time measured in mAh.
Blue - Has bluetooth or not.
Clock_speed - speed at which microprocessor executes instructions.
Dual_sim - Has dual sim support or not.
Fc - Front Camera mega pixels.
Four_g - Has 4G or not.
Int_memory - Internal Memory in Gigabytes.
M_dep - Mobile Depth in cm.
Mobile_wt - Weight of mobile phone.
N_cores - Number of cores of processor.
Pc - Primary Camera mega pixels.
Px_height - Pixel Resolution Height.
Px_width - Pixel Resolution Width.
Ram - Random Access Memory in Mega Bytes.
Sc_h - Screen Height of mobile in cm.
Sc_w - Screen Width of mobile in cm.
Talk_time - longest time that a single battery charge will last when you are.
Three_g - Has 3G or not.
Touch_screen - Has touch screen or not.
Wifi - Has wifi or not.
Price_range - This is the target variable with value of 0(low cost), 1(medium cost), 2(high cost) and 3(very high cost).

Data Overview

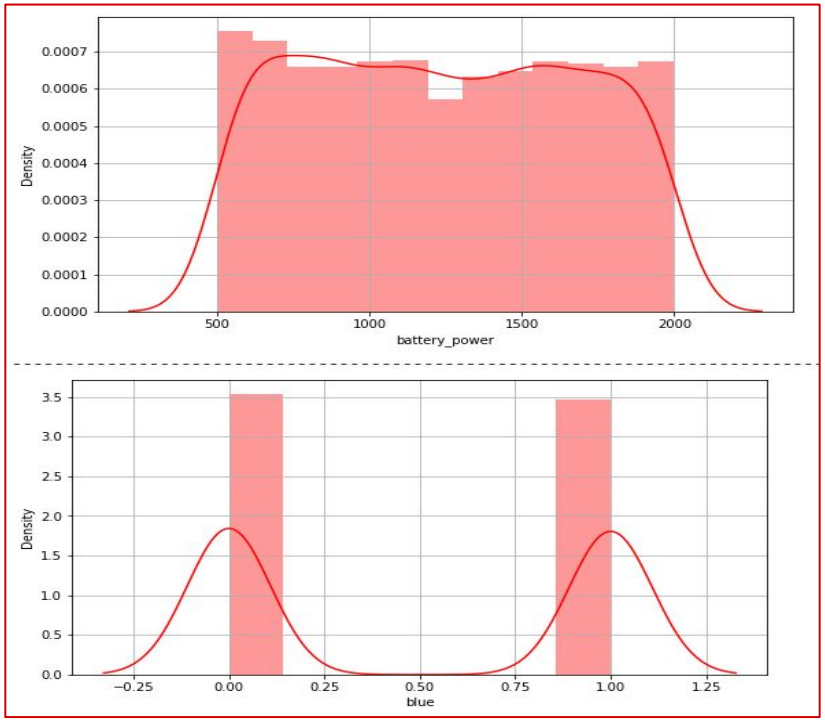
Data Summary:

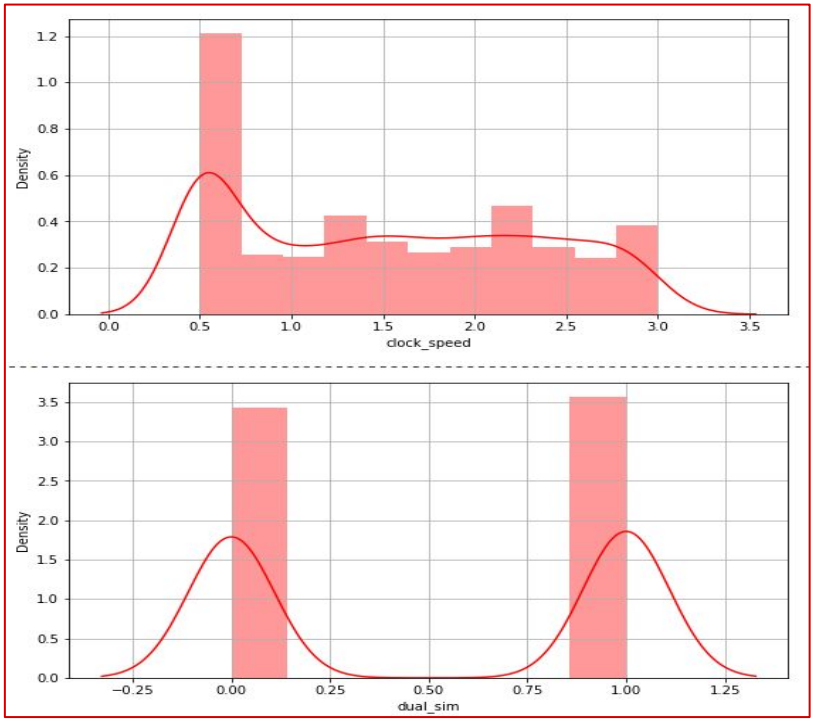
px_height	px_width	ram	sc_h	sc_w
2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
645.108000	1251.515500	2124.213000	12.306500	5.767000
443.780811	432.199447	1084.732044	4.213245	4.356398
0.000000	500.000000	256.000000	5.000000	0.000000
282.750000	874.750000	1207.500000	9.000000	2.000000
564.000000	1247.000000	2146.500000	12.000000	5.000000
947.250000	1633.000000	3064.500000	16.000000	9.000000
1960.000000	1998.000000	3998.000000	19.000000	18.000000

```
<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
```

EDA

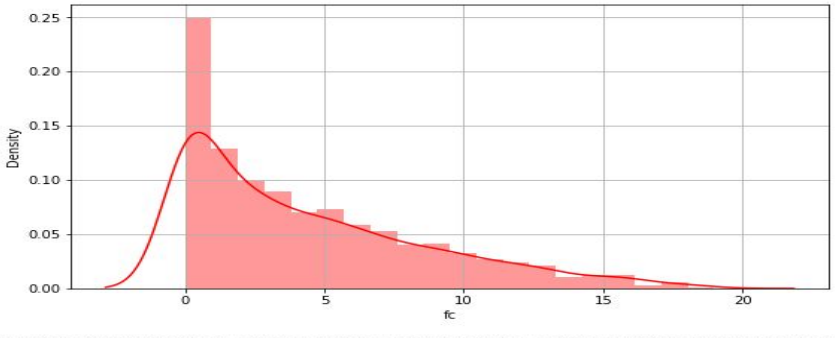
Univariate Analysis :

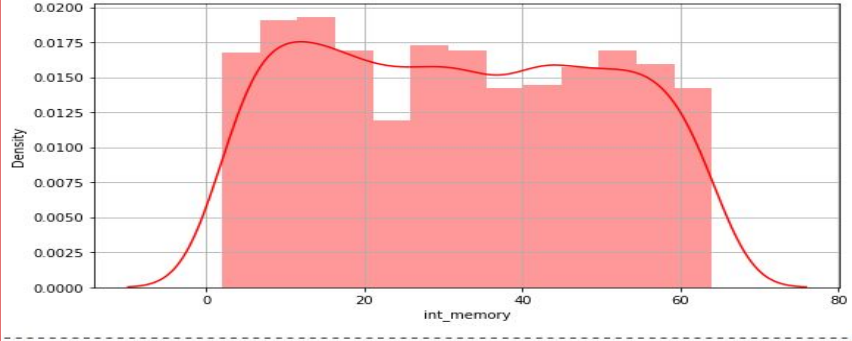


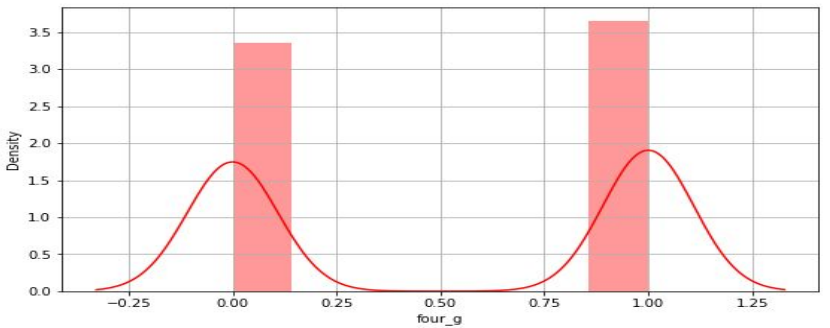


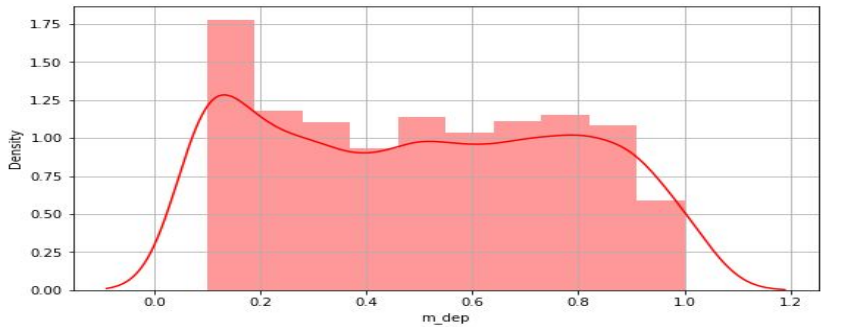
EDA

Univariate Analysis :



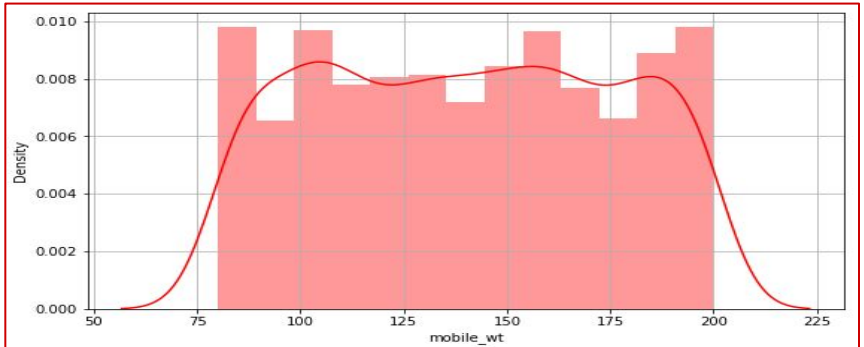


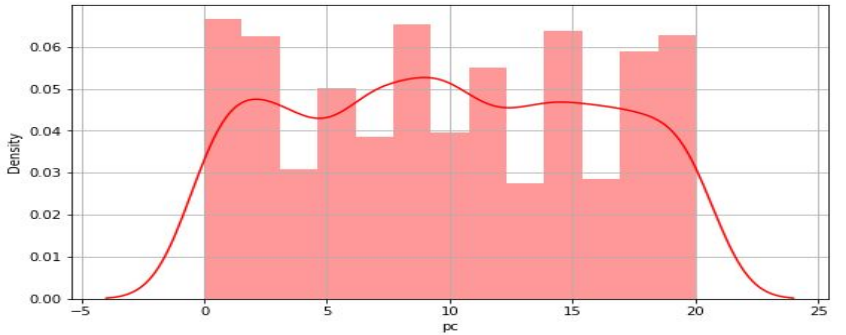


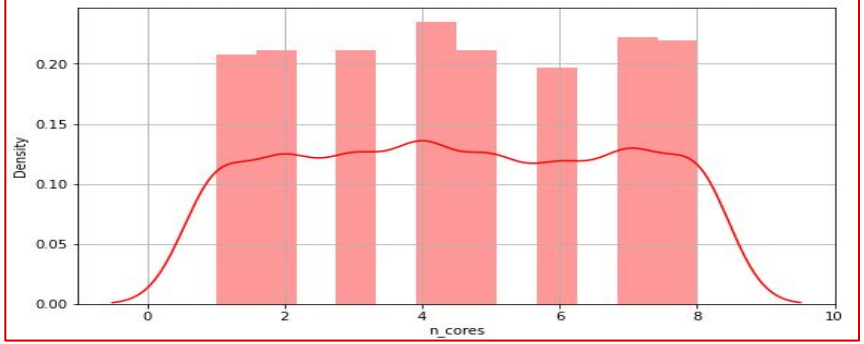


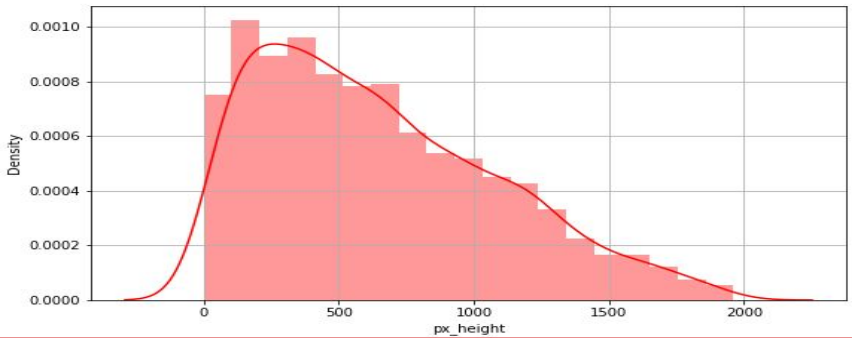
EDA

Univariate Analysis :



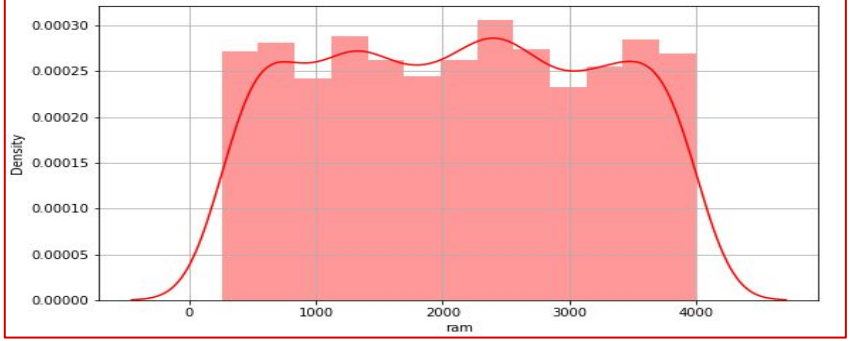


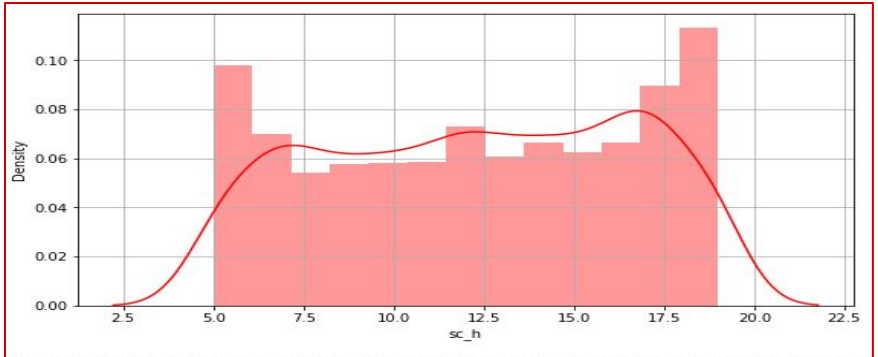


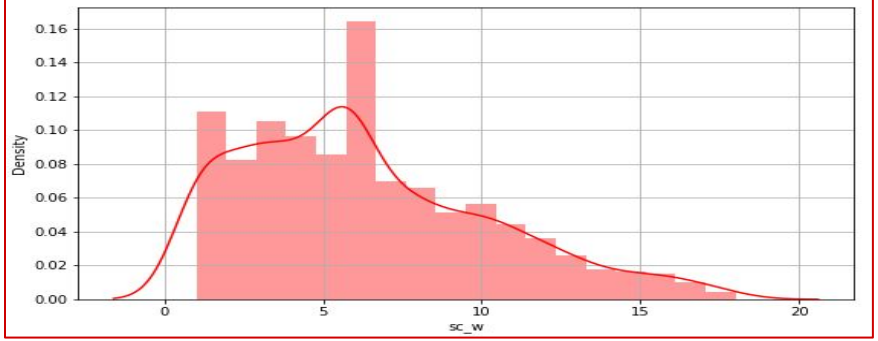


EDA

Univariate Analysis :

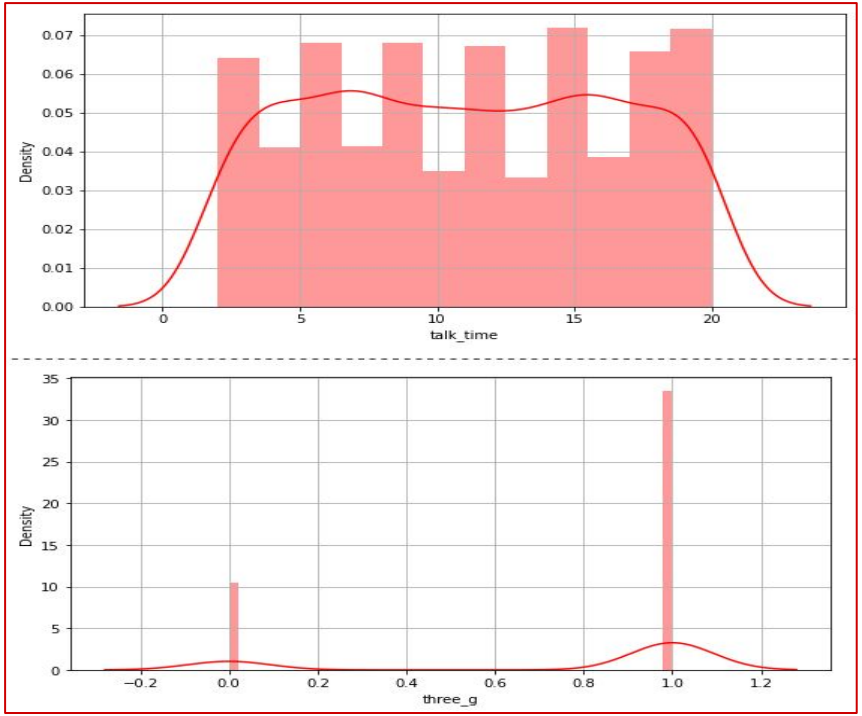


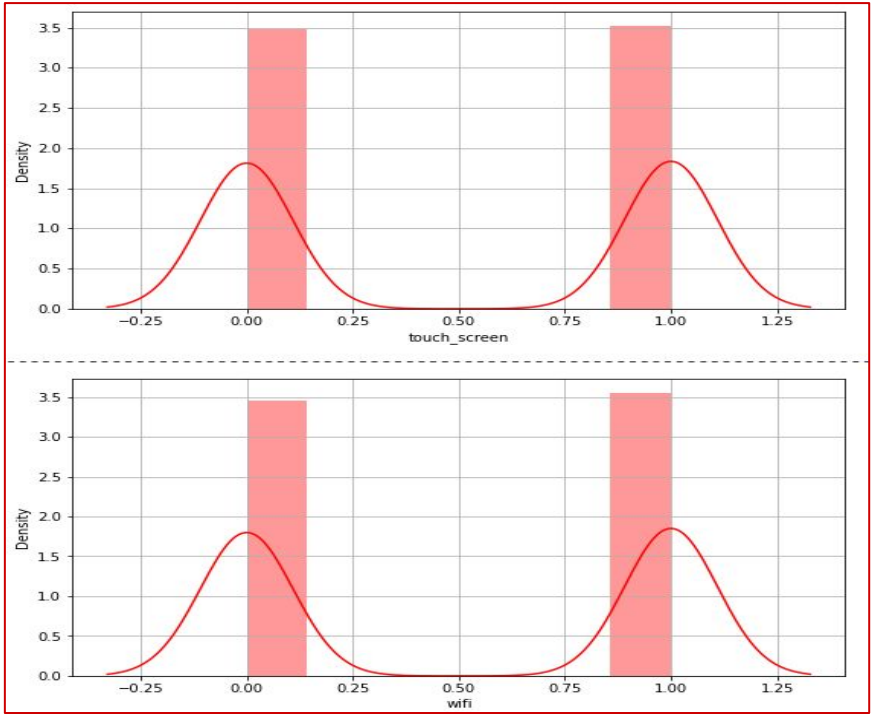




EDA

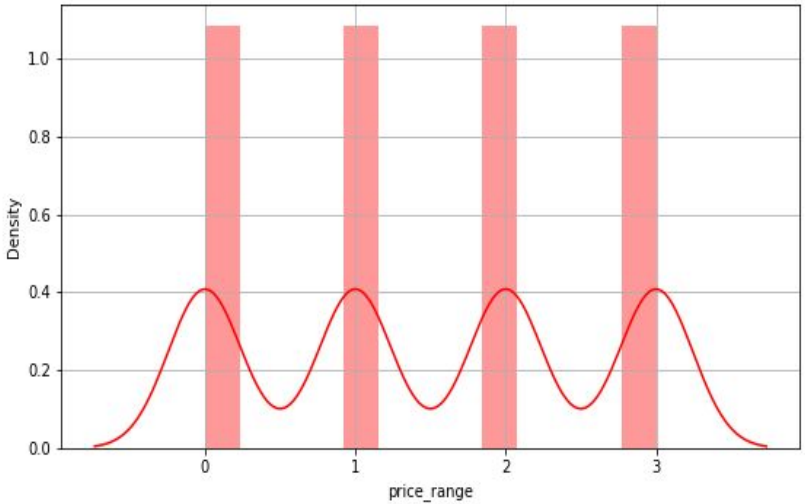
Univariate Analysis :

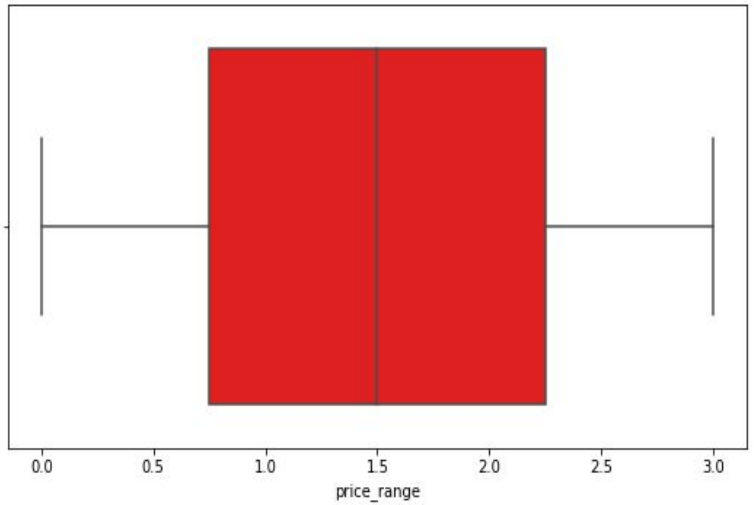




EDA

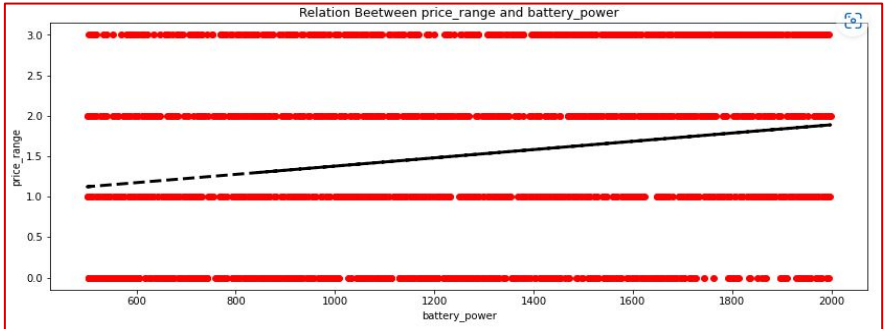
Univariate Analysis : Dependent Variable

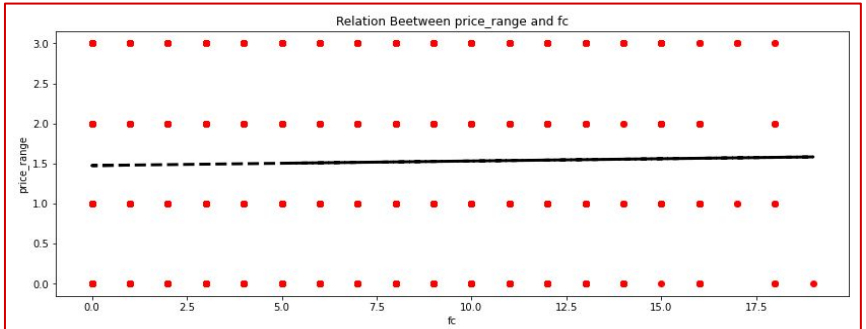


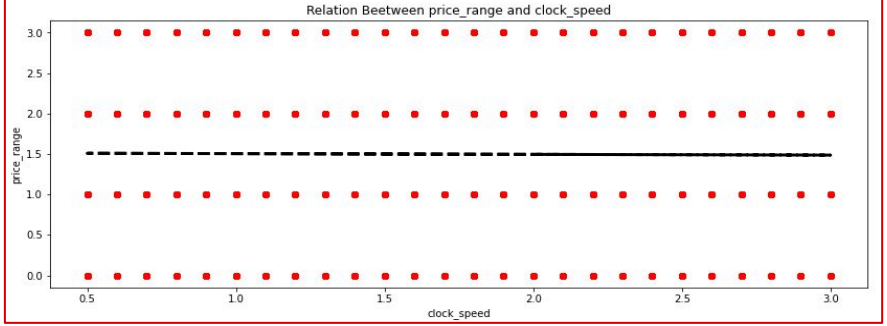


EDA

Bivariate Analysis :

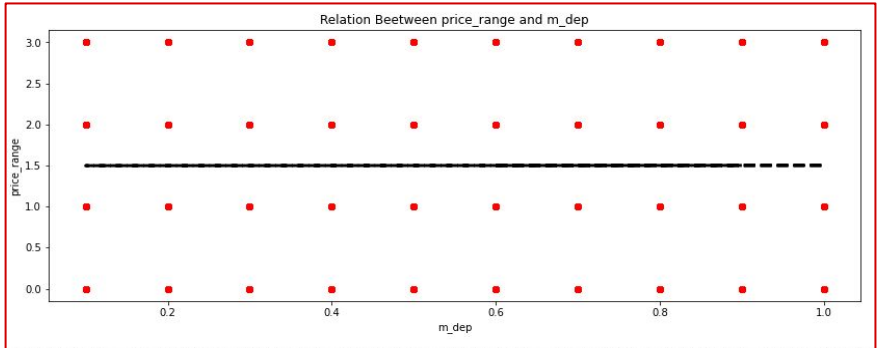


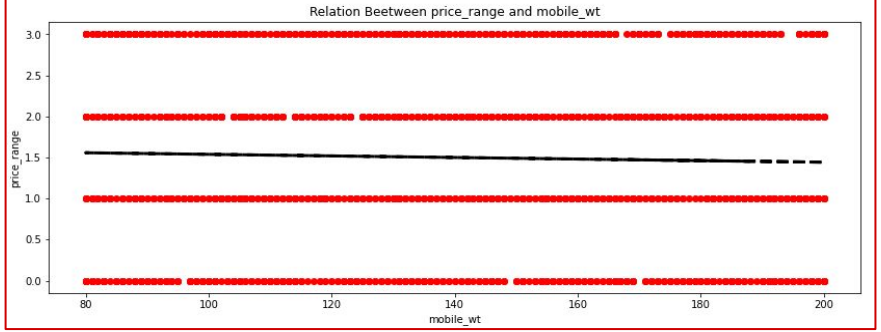


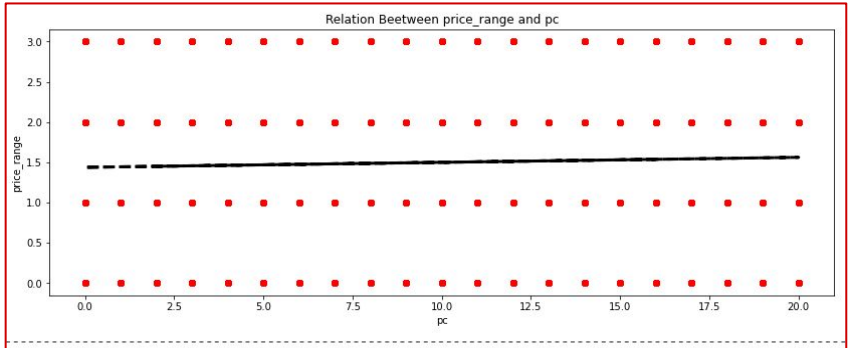


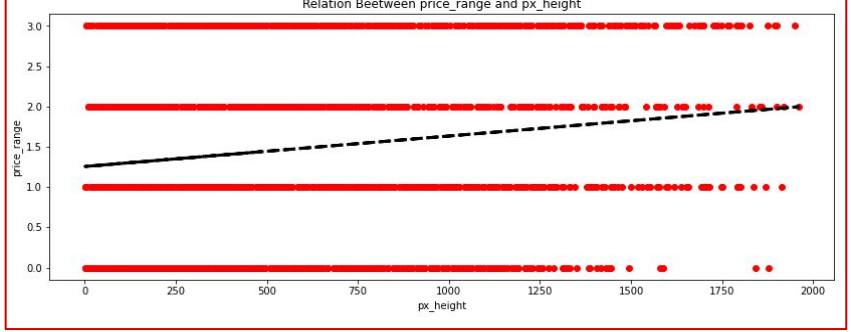
EDA

Univariate Analysis :



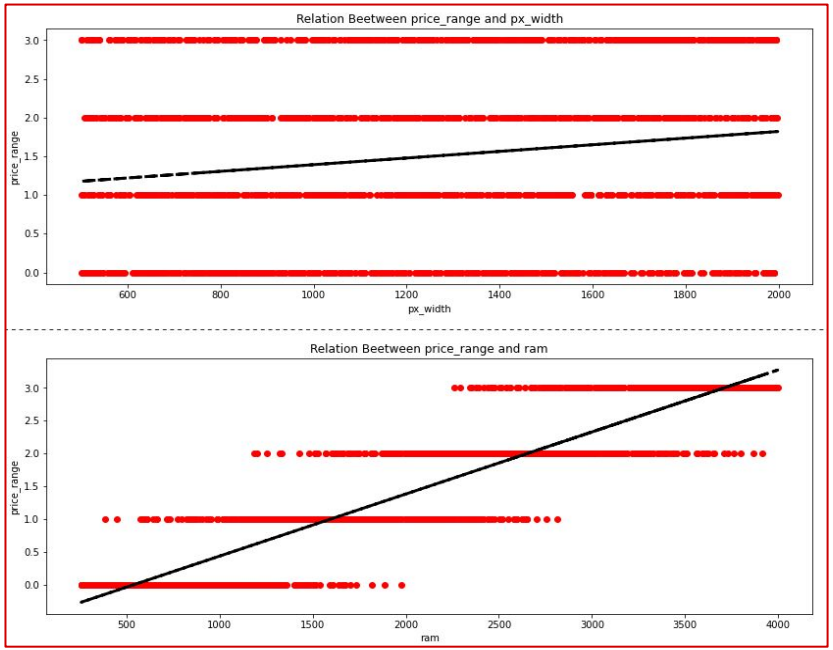


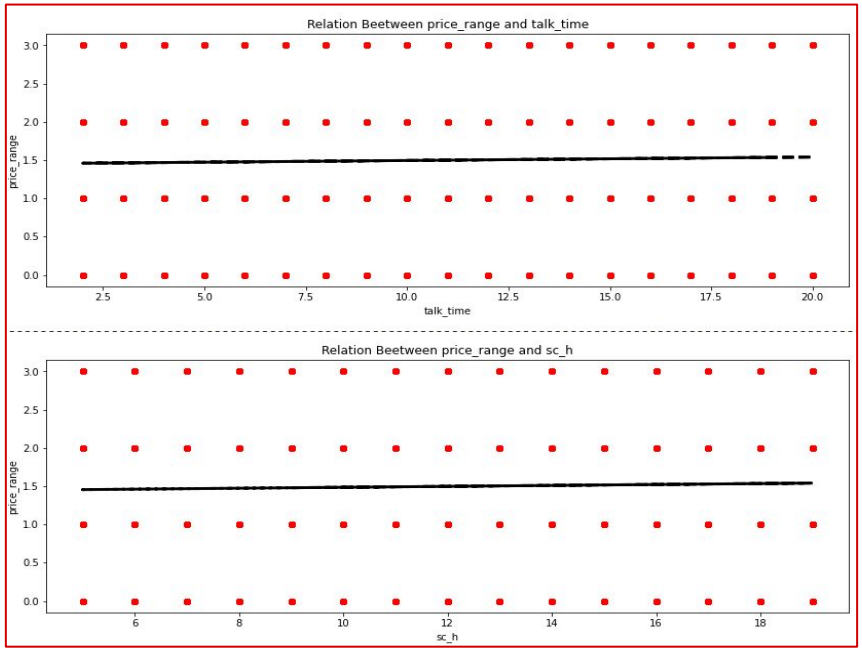




EDA

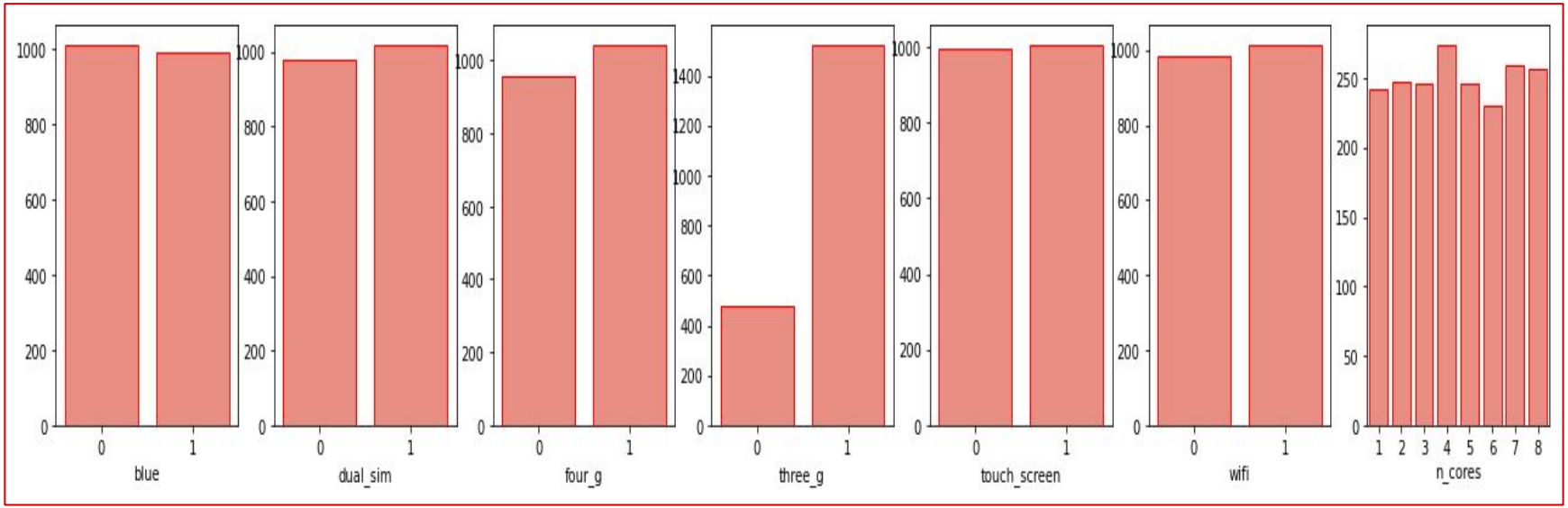
Univariate Analysis :





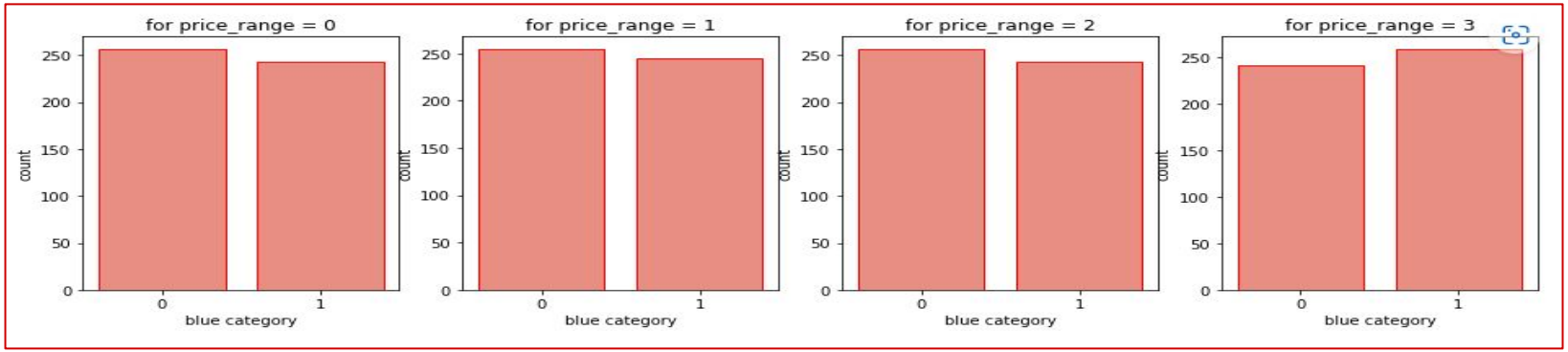
EDA

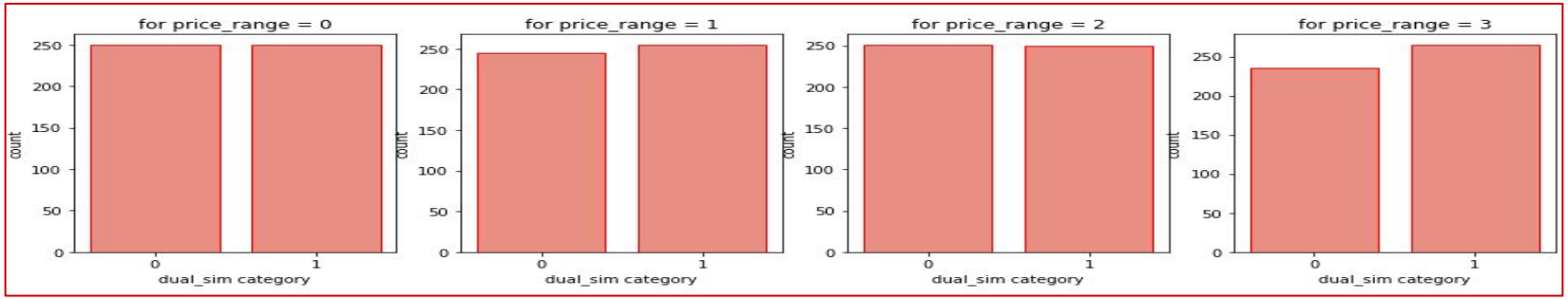
Univariate Analysis :



EDA

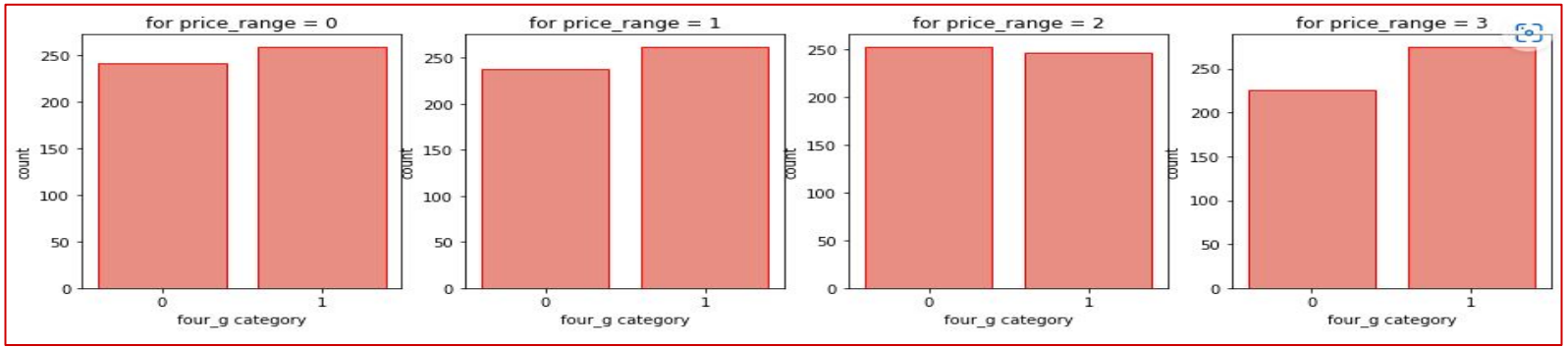
Bivariate Analysis :

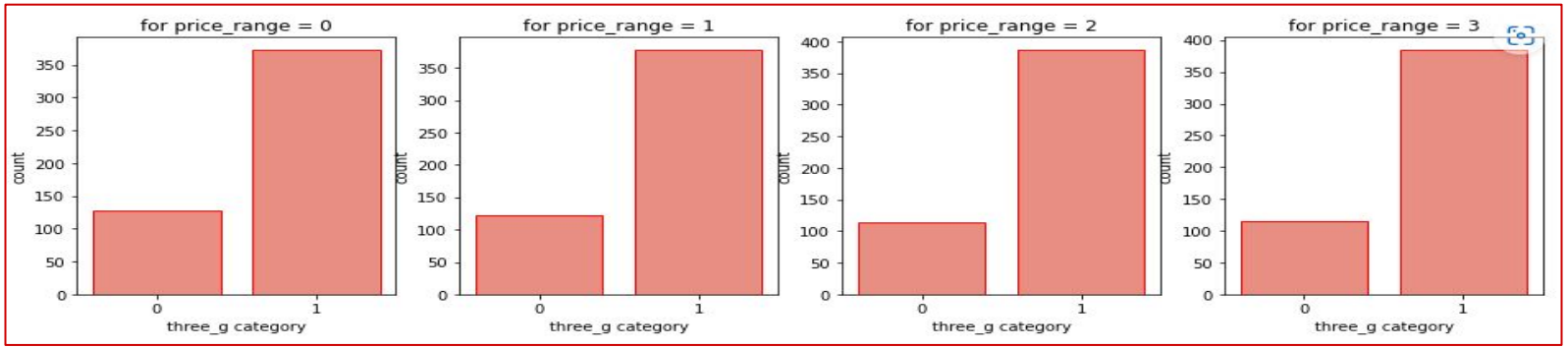




EDA

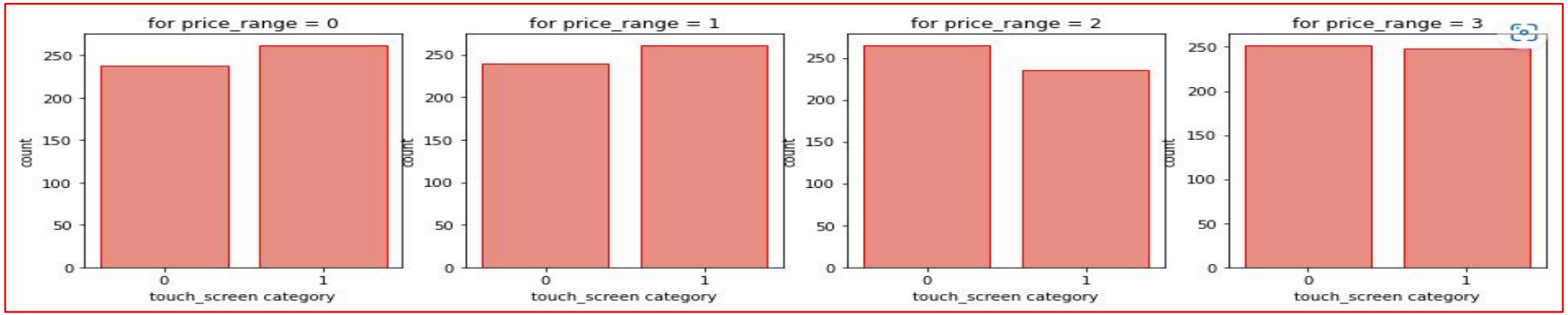
Bivariate Analysis :

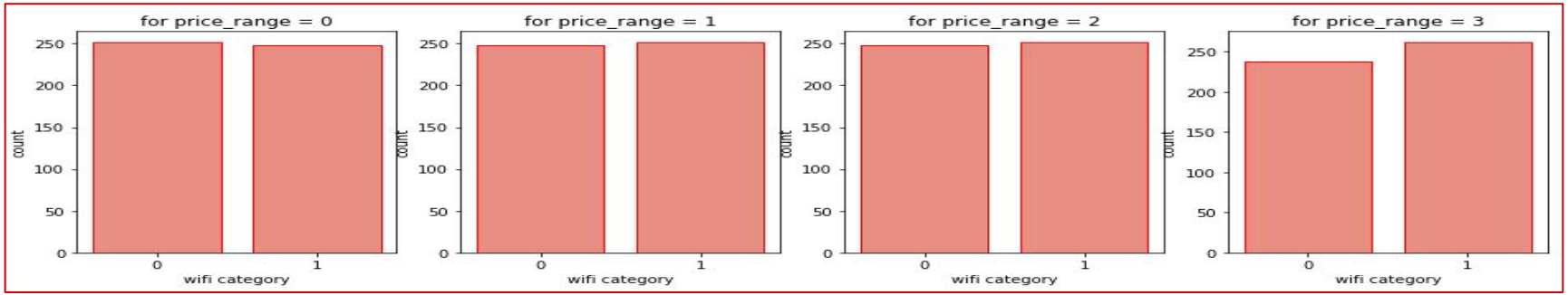




EDA

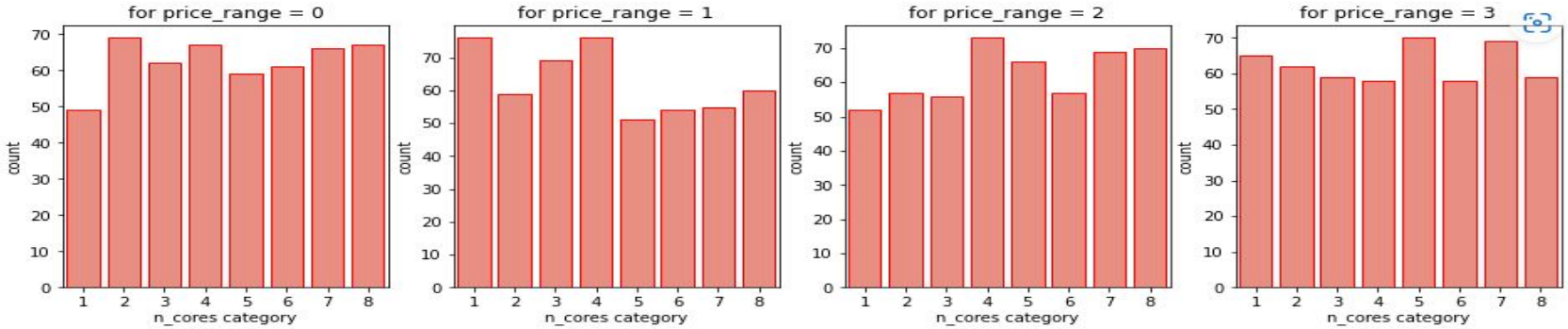
Bivariate Analysis :





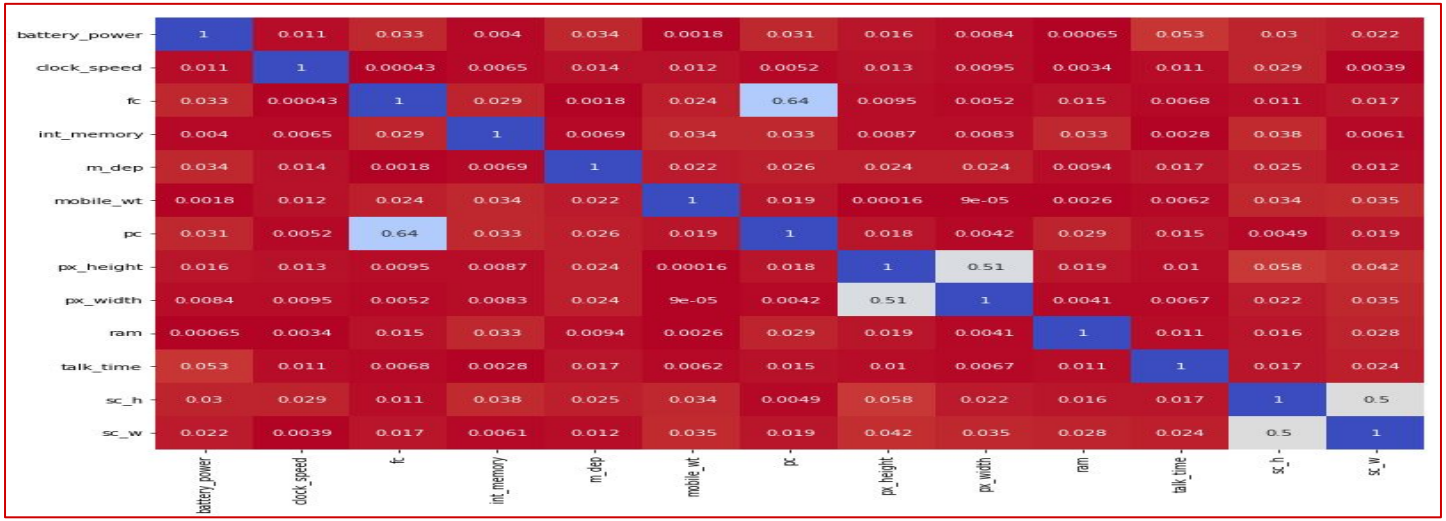
EDA

Bivariate Analysis :



EDA

Multivariate Analysis :



- There is some collinearity in feature pairs ('pc', 'fc'), ('px_width', 'px_height') and ('sc_h','sc_w'). Both correlations are justified since there are good chances that if front camera of a phone is good, the back camera would also be good.

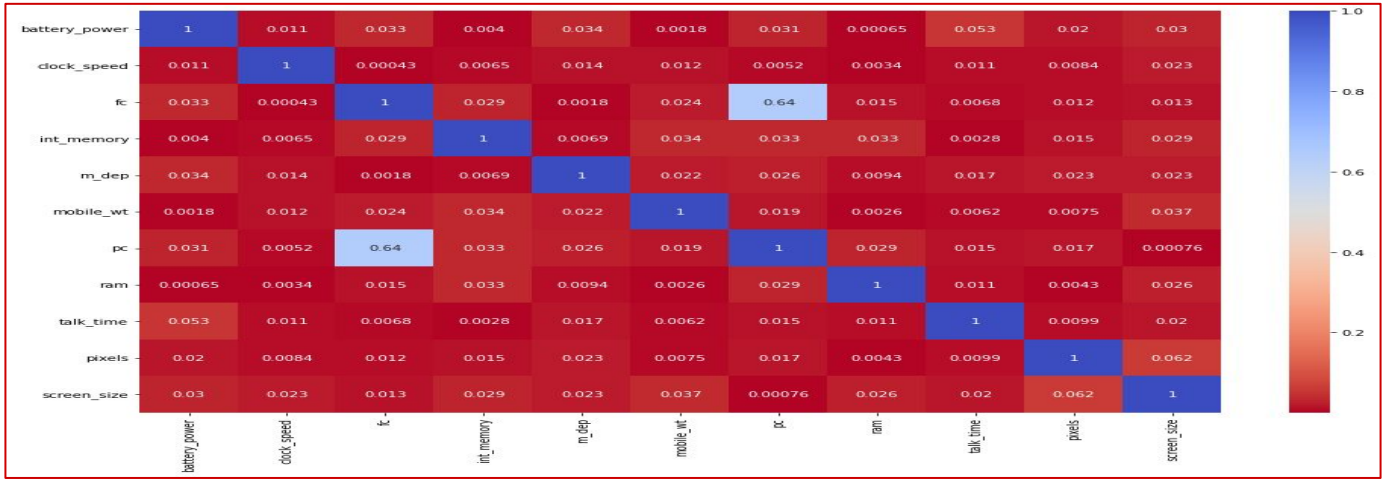
Data Pre-processing

Feature Engineering:

- I have created a new feature by multiplying px_height and px_width as they both denotes the pixel dimensions and dropped the both features px_height and px_width as they no more required for the model
- In the above analysis section, we can look at the combination of feature 3G and 4G. Interestingly, we have observed that there are zero mobile phones that supports 4G but not 3G. That means every phone that supports 4G, will support 3G by default. So, I'm gonna make a single feature called 'network' by adding 3G and 4G features
- Converting columns 'sc_h'(screen height) and 'sc_w'(screen width in a single column 'screen_size'(total screen size) , it will be the vertical length of the mobile.

EDA

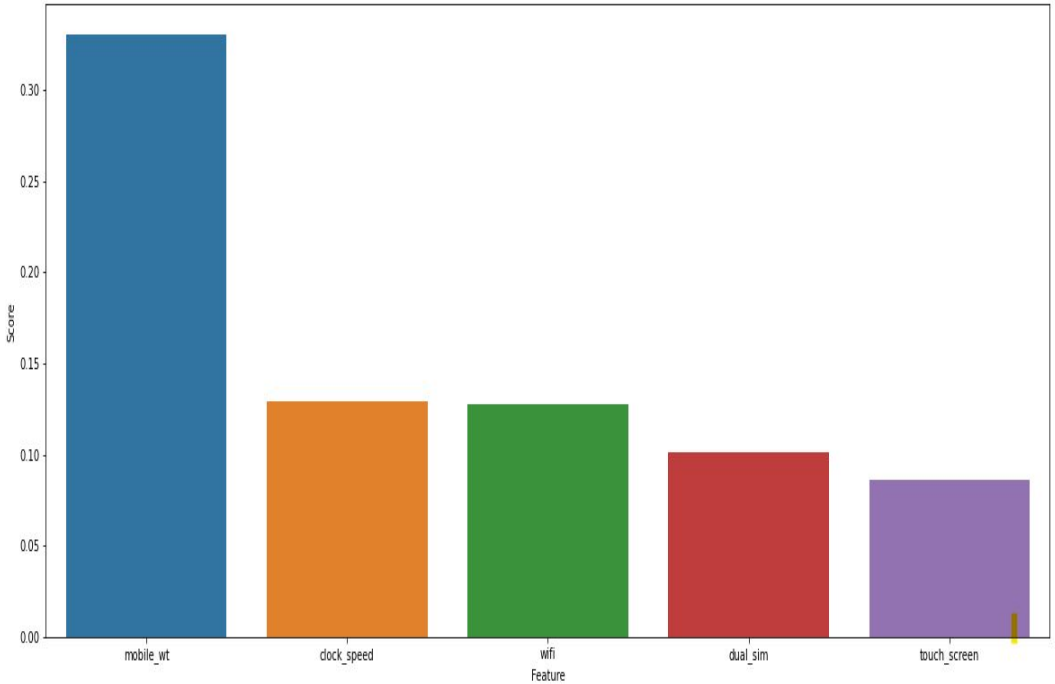
Multivariate Analysis :



- We can say that after understanding heatmap there is next to no multicollinearity exists in the dataset. every feature has distinct relationship with the dependent variable. variables like pixel dimensions and screen dimensions has kind of collinearity which we have treated above in this section.
- Front Camera megapixels and Primary camera megapixels are different variables and showing collinearity with each other. So we'll be keeping them as they are.

Model Implementation

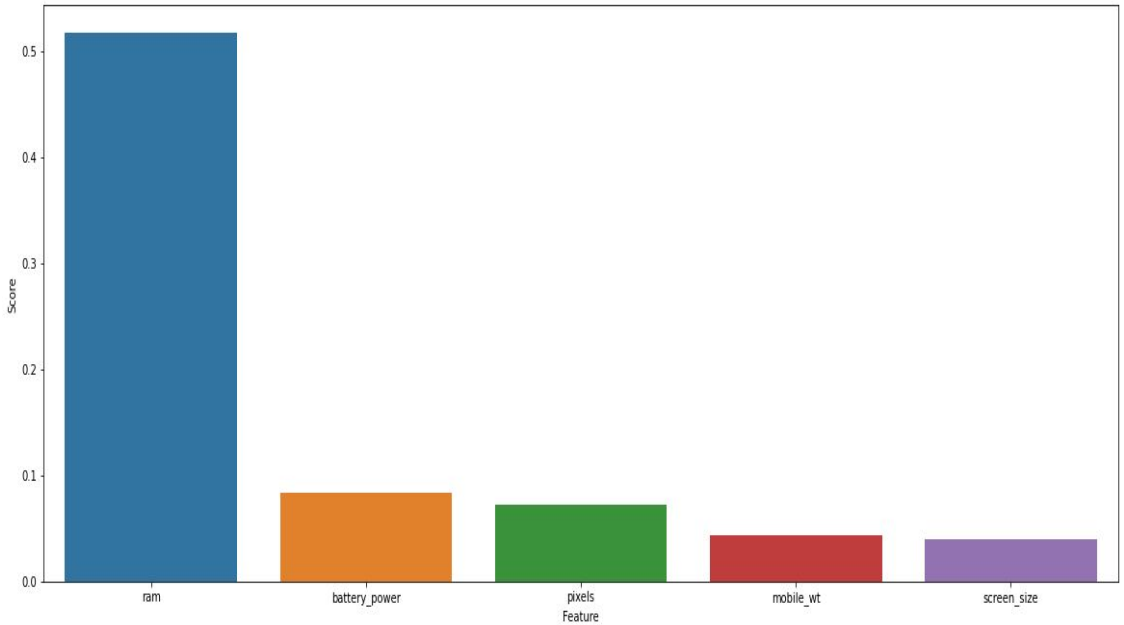
Logistic Regression:



	precision	recall	f1-score	support
0	0.93	0.92	0.93	93
1	0.85	0.81	0.83	101
2	0.82	0.81	0.81	108
3	0.87	0.94	0.90	98
accuracy			0.87	400
macro avg	0.87	0.87	0.87	400
weighted avg	0.87	0.87	0.87	400

Model Implementation

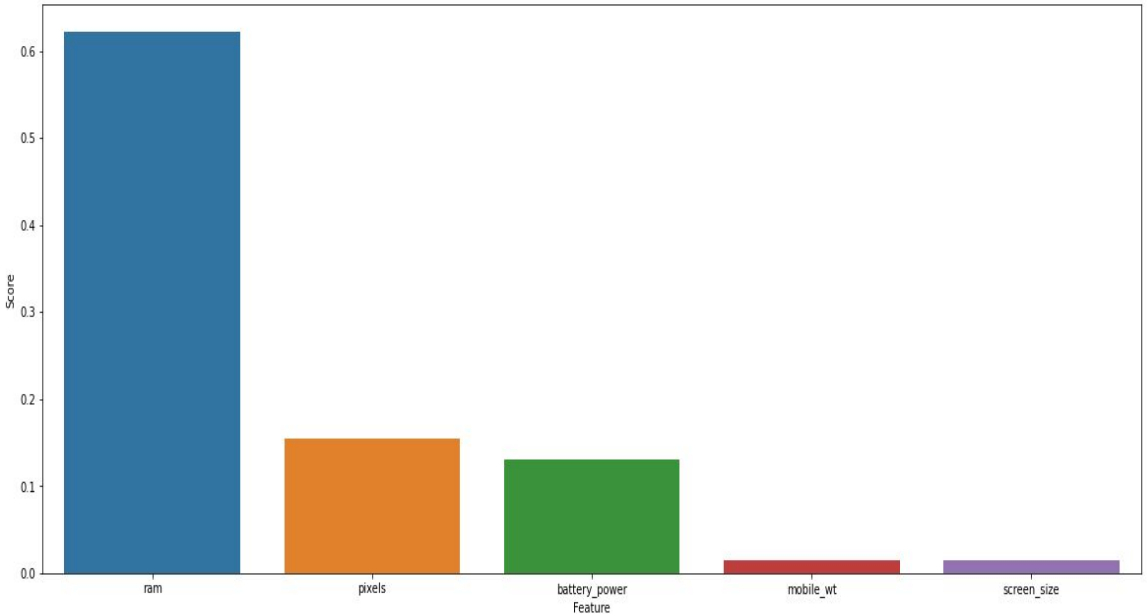
Random Forest Classifier:



	precision	recall	f1-score	support
0	0.95	0.97	0.96	90
1	0.88	0.81	0.84	104
2	0.77	0.76	0.77	108
3	0.84	0.91	0.87	98
accuracy			0.85	400
macro avg	0.86	0.86	0.86	400
weighted avg	0.85	0.85	0.85	400

Model Implementation

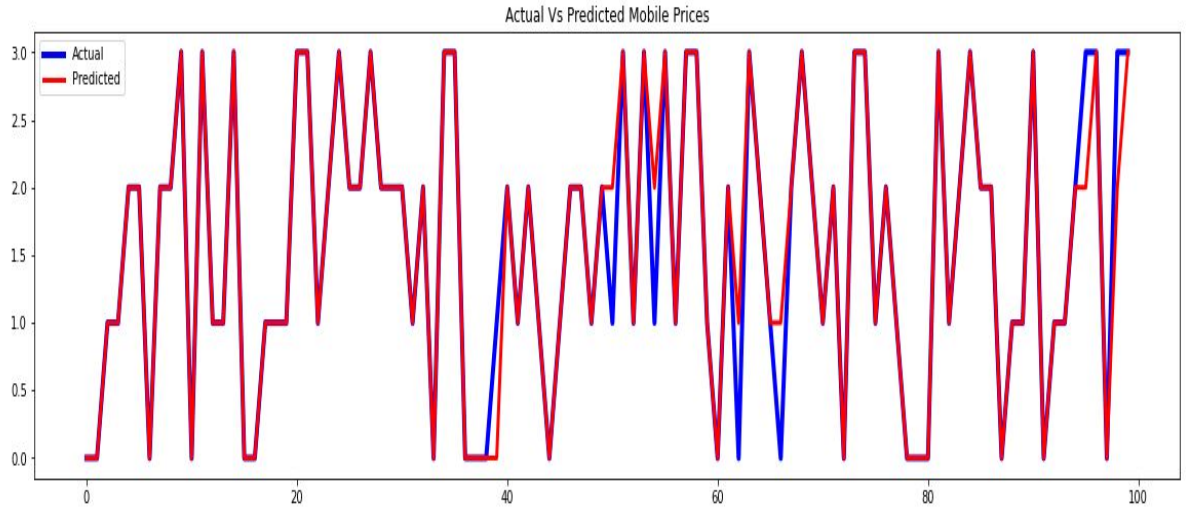
Decision Tree Classifier:



	precision	recall	f1-score	support
0	0.92	0.93	0.93	91
1	0.85	0.77	0.81	107
2	0.75	0.77	0.76	103
3	0.85	0.91	0.88	99
accuracy			0.84	400
macro avg	0.84	0.84	0.84	400
weighted avg	0.84	0.84	0.84	400

Model Implementation

Support Vector Machine Classifier:



	precision	recall	f1-score	support
0	0.98	0.96	0.97	94
1	0.91	0.86	0.88	101
2	0.86	0.87	0.86	105
3	0.92	0.97	0.94	100
accuracy			0.91	400
macro avg	0.91	0.91	0.91	400
weighted avg	0.91	0.91	0.91	400

Model Evaluation

Model Evaluation :

		0	1	2	3	accuracy	macro avg	weighted avg
Logistic Regression	precision	0.918367	0.794118	0.828283	0.950495	0.8725	0.872816	0.873189
	recall	0.978261	0.843750	0.773585	0.905660	0.8725	0.875314	0.872500
	f1-score	0.947368	0.818182	0.800000	0.927536	0.8725	0.873272	0.872055
	support	92.000000	96.000000	106.000000	106.000000	0.8725	400.000000	400.000000
Random Forest Classifier	precision	0.896907	0.785714	0.782178	0.894231	0.8400	0.839758	0.839108
	recall	0.945652	0.802083	0.745283	0.877358	0.8400	0.842594	0.840000
	f1-score	0.920635	0.793814	0.763285	0.885714	0.8400	0.840862	0.839246
	support	92.000000	96.000000	106.000000	106.000000	0.8400	400.000000	400.000000
Decision Tree Classifier	precision	0.934066	0.766355	0.766990	0.909091	0.8400	0.844126	0.842922
	recall	0.923913	0.854167	0.745283	0.849057	0.8400	0.843105	0.840000
	f1-score	0.928962	0.807882	0.755981	0.878049	0.8400	0.842718	0.840571
	support	92.000000	96.000000	106.000000	106.000000	0.8400	400.000000	400.000000
Support Vector Machine Classifier	precision	0.957447	0.861386	0.866667	0.970000	0.9125	0.913875	0.913662
	recall	0.978261	0.906250	0.858491	0.915094	0.9125	0.914524	0.912500
	f1-score	0.967742	0.883249	0.862559	0.941748	0.9125	0.913824	0.912702
	support	92.000000	96.000000	106.000000	106.000000	0.9125	400.000000	400.000000

Conclusion

Conclusion :

- From EDA we can see that there is no imbalance in classes of output variable which is a good thing. also there is very less multicollinearity between features which is kind of reduces our work still we did some changes in variables which ultimately improve our performance.
- It has been observed that expensive mobiles can have fewer features. Like you can find expensive mobile phones with no wifi, no touch screen, no 4G support, no dual sim and even with no bluetooth support. and that's mainly because of the quality and the brand of the mobile phone.
- As We have used predictive models, It is being observed that the data distribution of each variable to the each classes of output variable is almost similar which might be the reason for worst performance of tree based models.
- Mainly features like RAM, battery power, pixels and Mobile weight plays the significant role in deciding the price range of mobile phone.
- On the basis of the classification report, we can say that the SVM using hyperparameters we got the best results out of all other followed by logistic regression model.

Thank You