Capstone Project - 3

Supervised ML - Classification

Topic - Mobile Price Range Prediction

By - Avilash Srivastava

CONTENTS OF THE PRESENTATION

- Problem Statement
- Data Summary
- Exploratory data analysis
- Data wrangling
- Machine Learning models
- Model Explanation
- Challenges
- Conclusion

Problem Statement

- Mobile phones have become the greatest necessity for almost all individuals nowadays. People want more features and best specifications in a phone and that too at cheaper prices. The demand for phones is so high that there is a huge competition prevailing between mobile manufacturers. To stay ahead in the race, these companies try to bring in new features and innovations so that people are lured towards buying their brand smartphones.
- Price of a mobile phone is influenced by various factors. Brand name, newness of the model, specifications such as internal memory, camera, ram, sizes, connectivity etc., are some of the important factors in determining the price. As a business point of view, it becomes an utmost priority to analyse these factors from time to time and come up with best set of specifications and price ranges so that people buy their mobile phones.
- Hence, through this exercise and our predictions we will try to help companies estimate price of mobiles to give tough competition to other mobile manufacturer and also it will be useful for customers to verify that they are paying best price for a mobile.

Data Summary

The contents of the data had these features:

- Battery_power - Total energy a battery can store in one time measured in mAh
- Clock_speed - speed at which microprocessor executes instructions
- Fc , Pc - Front and Primary Camera megapixels
- 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
- Px_height, Px_width - Pixel Resolution Height and Width
- Ram - Random Access Memory in Megabytes
- Sc_h, Sc_w - Screen Height and width of mobile in cm
- Talk_time - longest time that a single battery charge will last when you are on call
- Blue, 4g, 3g, dual_sim, touchscreen, wifi - Some supported and unsupported categories
- Price_range - This is the target variable with value of 0(low cost), 1(medium cost), 2(high cost) and 3(very high cost).

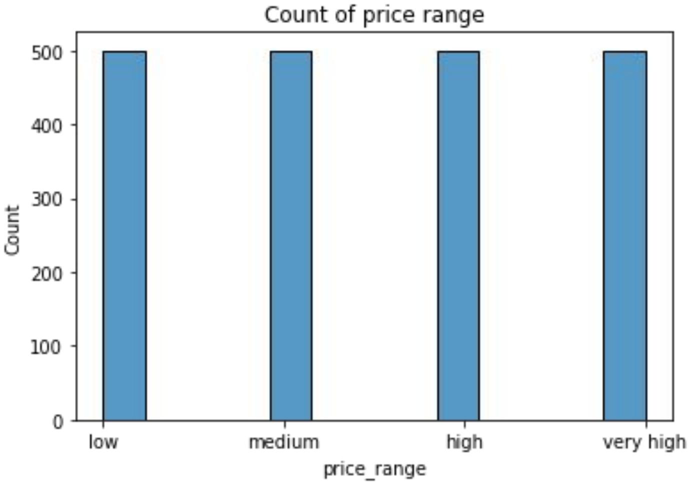
Total rows = 2000

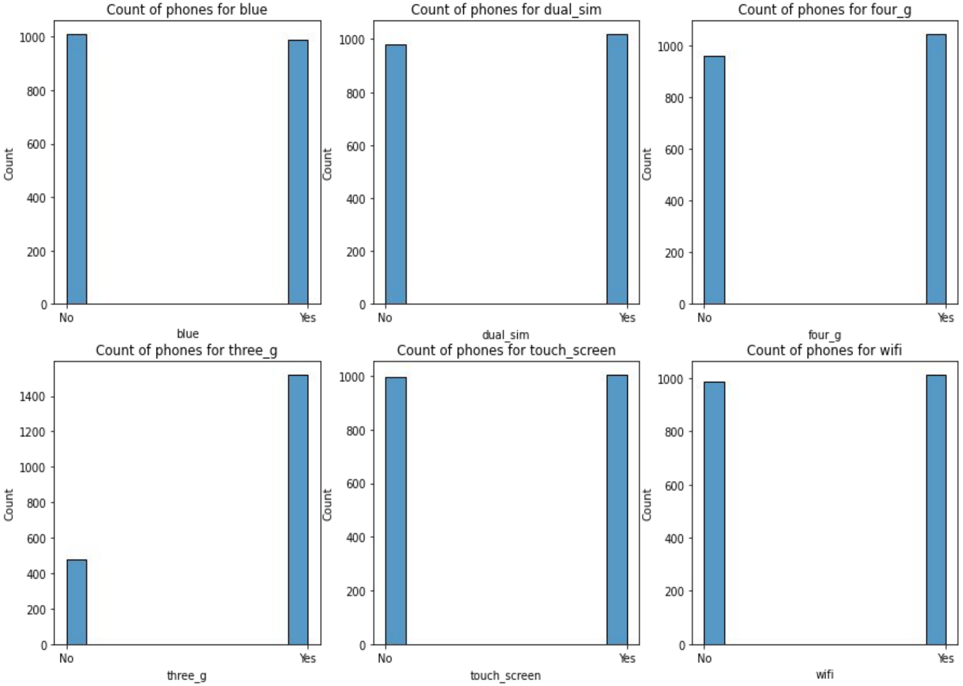
Samsung Mobile
RAM (2 GB)

RAM inside Mobile
Motherboard

EDA - Univariate analysis

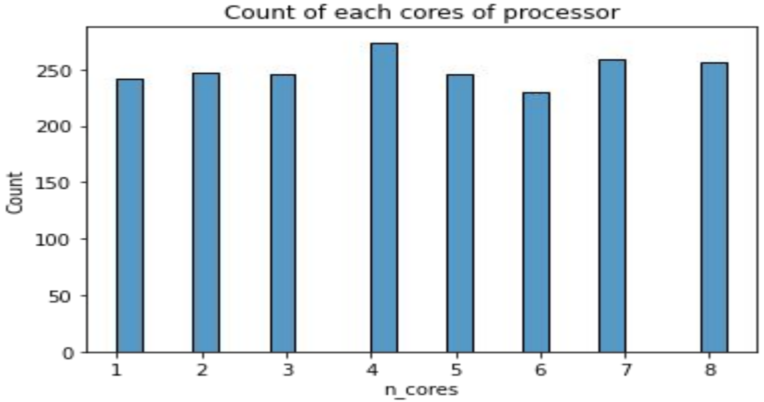
- Our dependent variable - Price range has equal no of observations in each bucket.
- Other dichotomous types have equal no of observations for each category, except for 3g

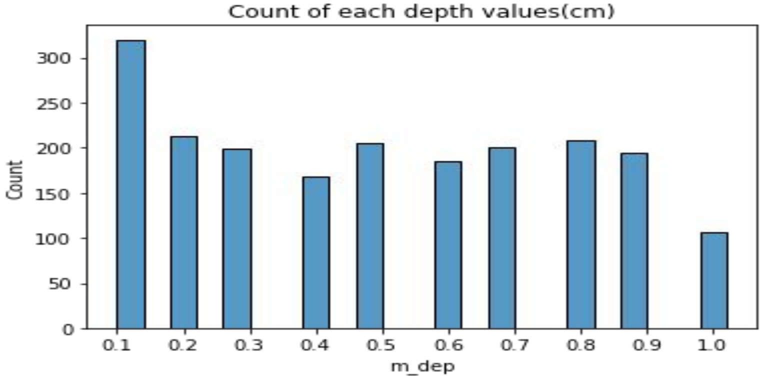




EDA - Univariate analysis

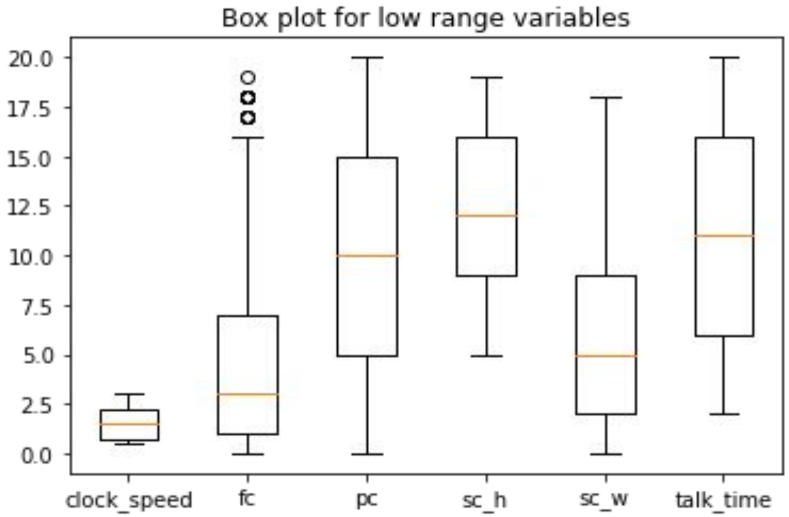
- Equal no of observations for n_cores, highest being 4 cores.
- Almost equal no. of observations for mobile depth values from 0.2 to 0.9 cm
- 0.1 cm depth had higher no. of observations.
- 1 cm depth had least no. of observations.

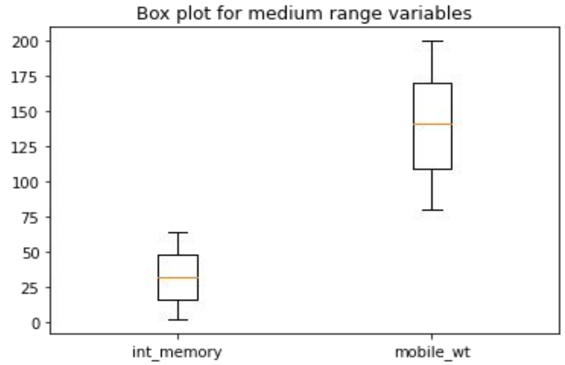


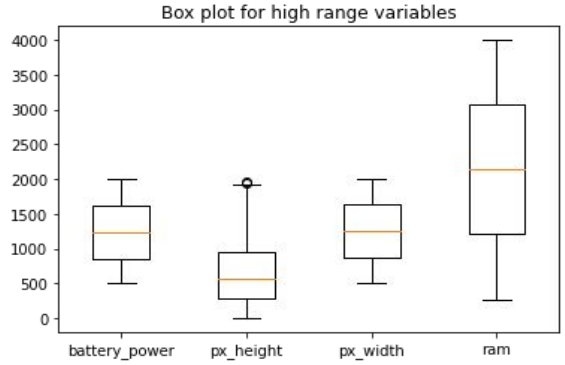


Univariate analysis - Numerical variables

Analysed descriptive stats using boxplots

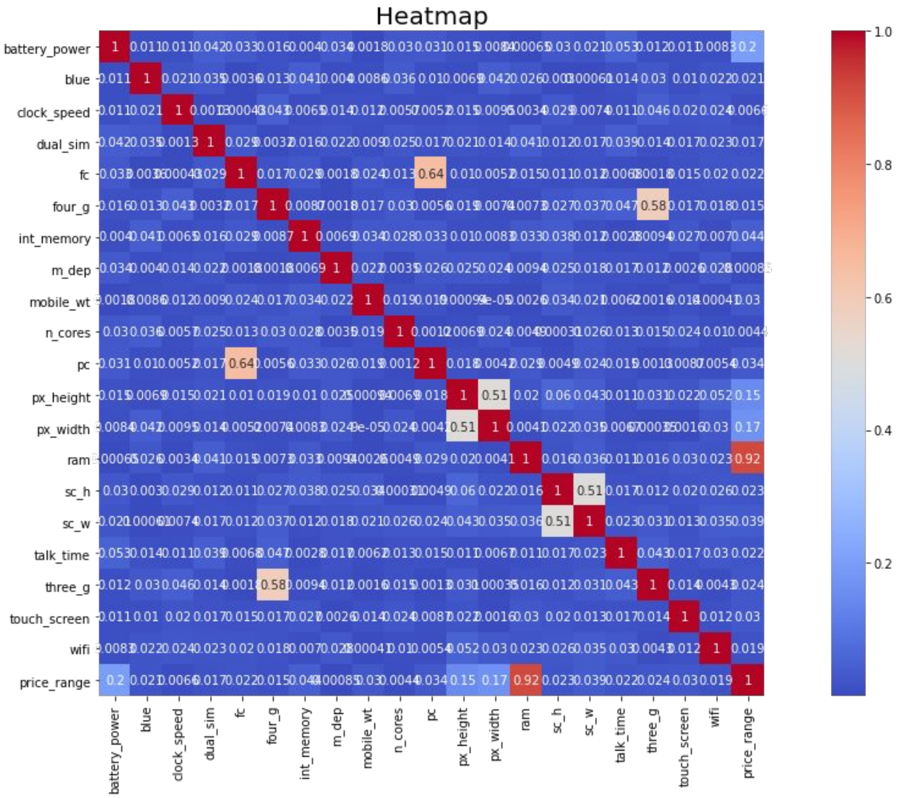

Box plot for low range variables


Box plot for medium range variables


Box plot for high range variables

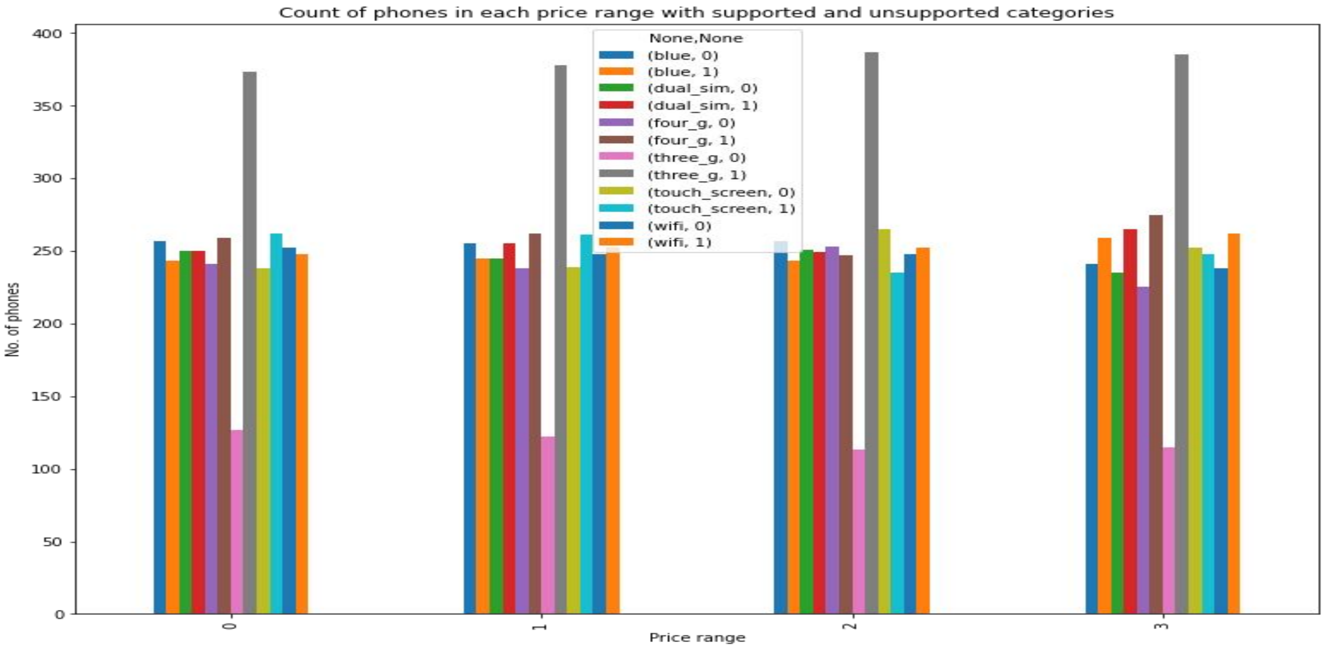
Multivariate analysis

- Visualized correlation between each variables using heatmap.
- Pc is correlated with Fc.
- Px_height and Px_width are moderately correlated.
- Sc_h and Sc_w are moderately correlated.
- Ram is highly correlated with price range.



Multivariate analysis - Categorical variables

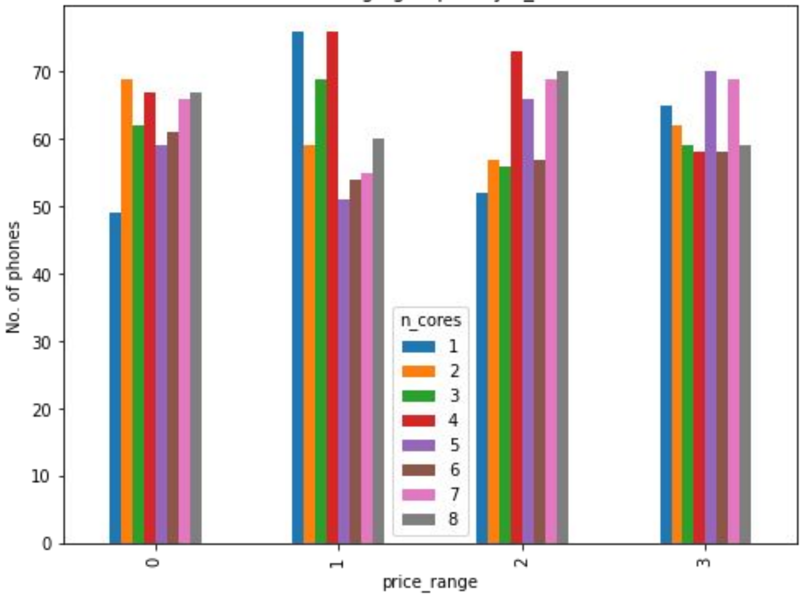
Almost equal no. of observations for each price range for each category.



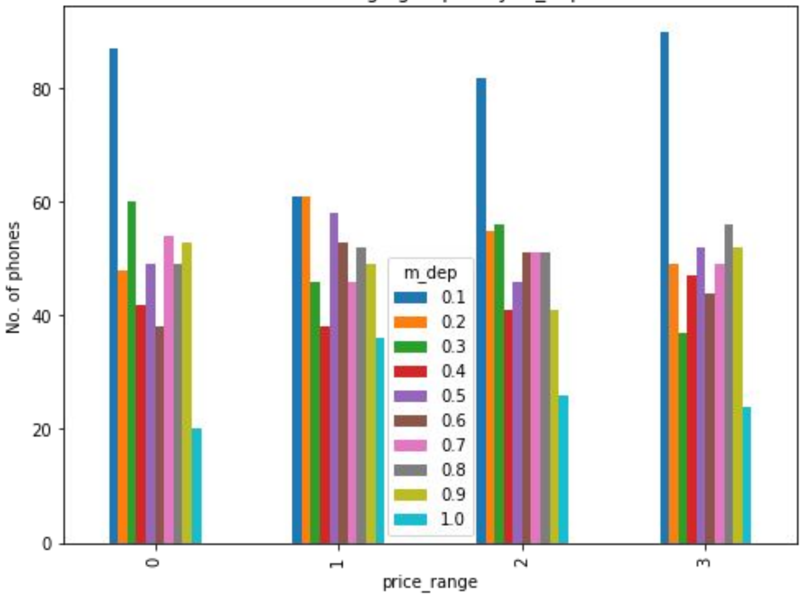
Multivariate analysis - n_cores and m_dep

1. Count of less n_cores is high for 0 and 1 price range.
2. Count of high n_cores is high for 2 and 3 price range.
3. Count of phones with less thickness is high and count of phones with high thickness is low.

Price range grouped by n_cores

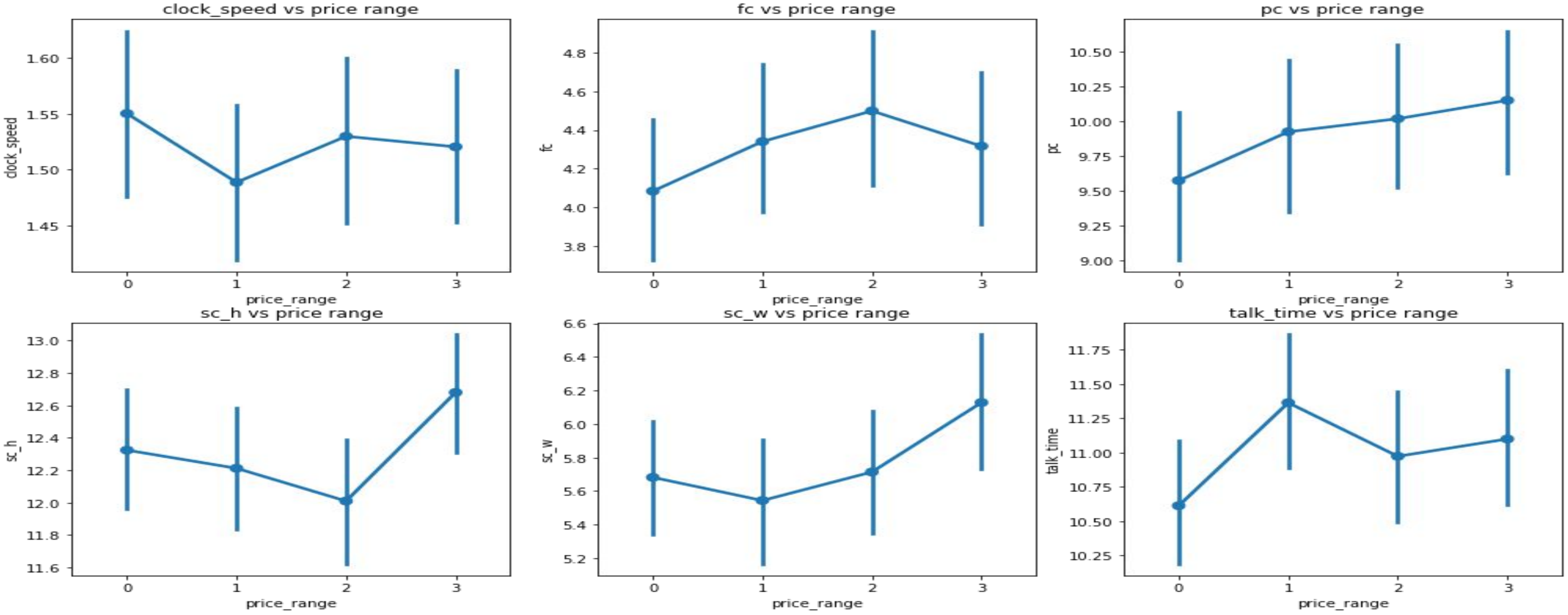


Price range grouped by m_dep



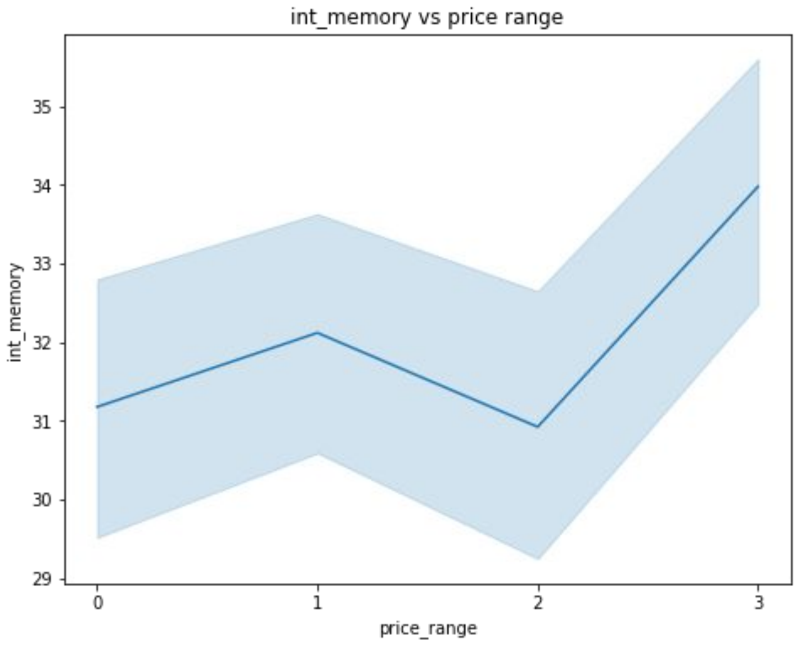
Multivariate analysis - Numerical variables

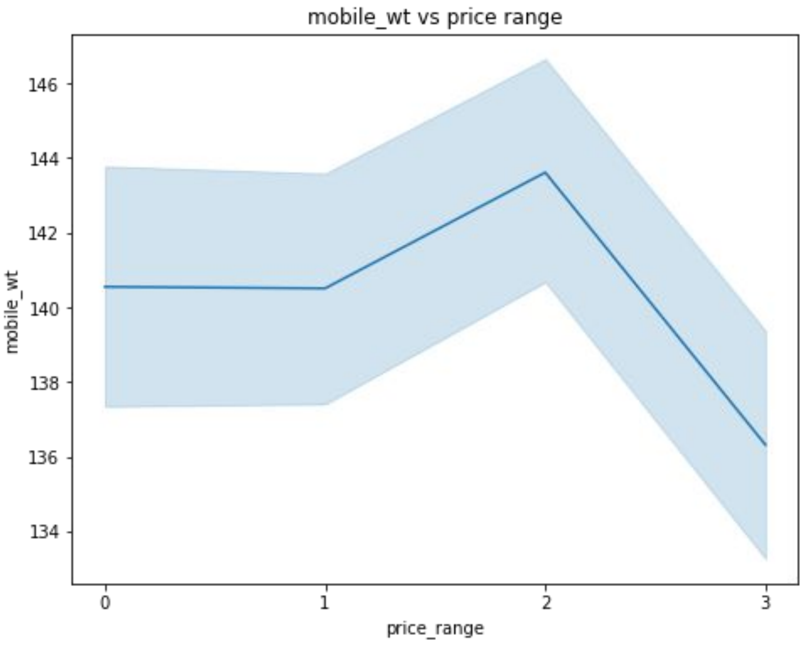
- Clock speed is high for low price range phones, Talk time is less.
- Pc, fc , sc_w are in increasing trend.



Multivariate analysis - int_memory, mobile_wt

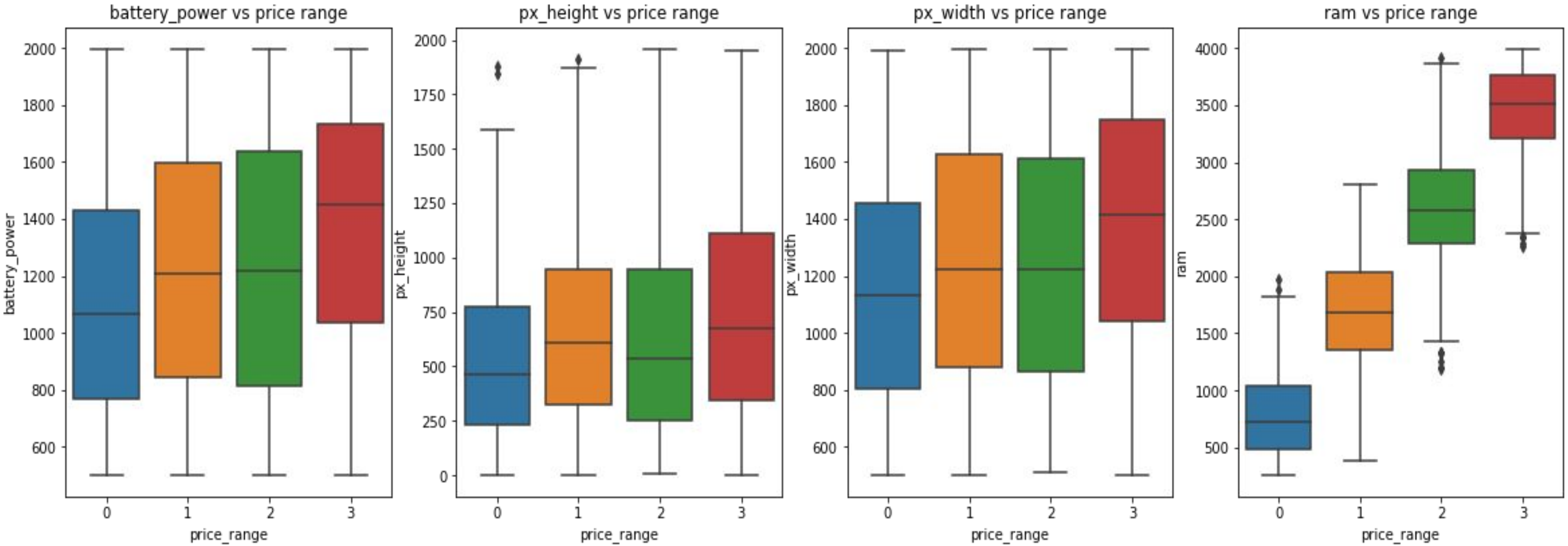
1. We can observe drastic increase in internal memory for very high prices.
2. Also there is drastic decrease in mobile weight for very high prices.



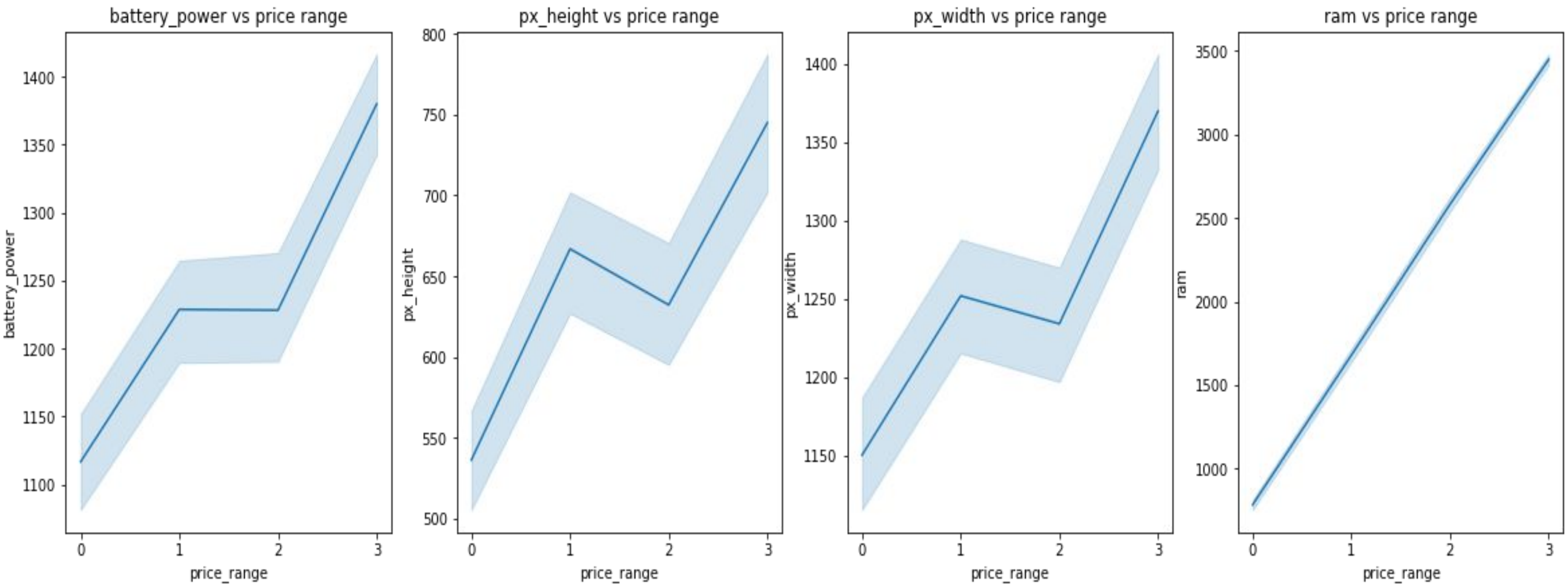


Multivariate analysis - battery power, ram

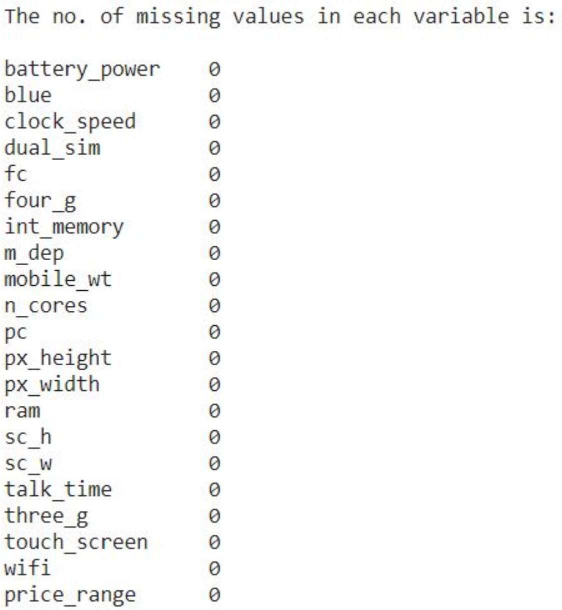
Mean values of battery power, px_height, px_width, ram is increasing with increase in prices.



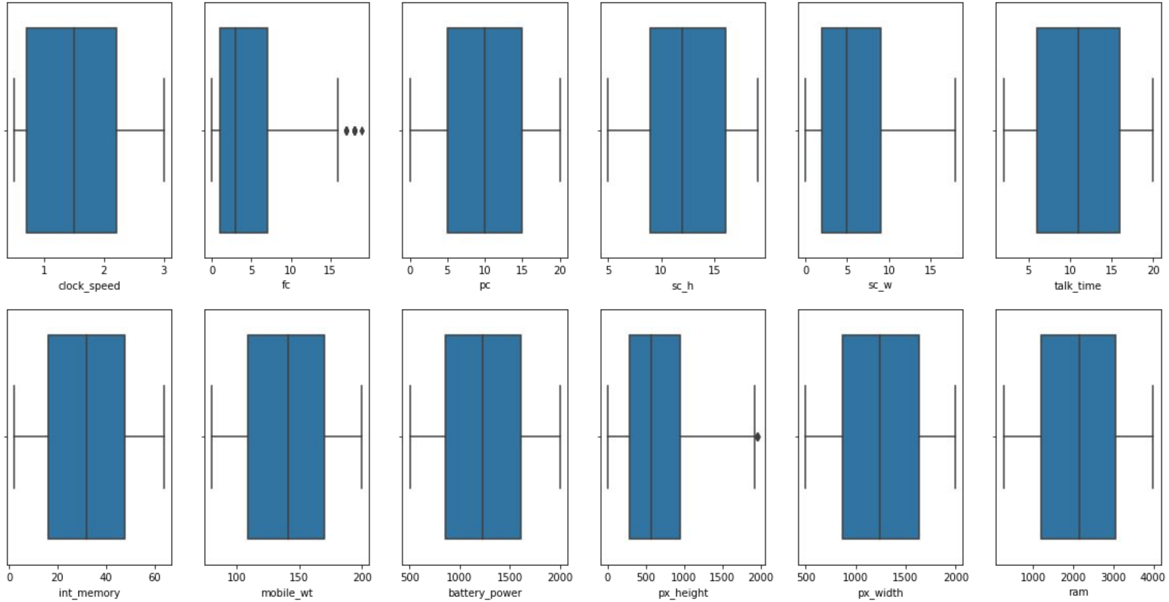
Multivariate analysis - Continued



Data Wrangling



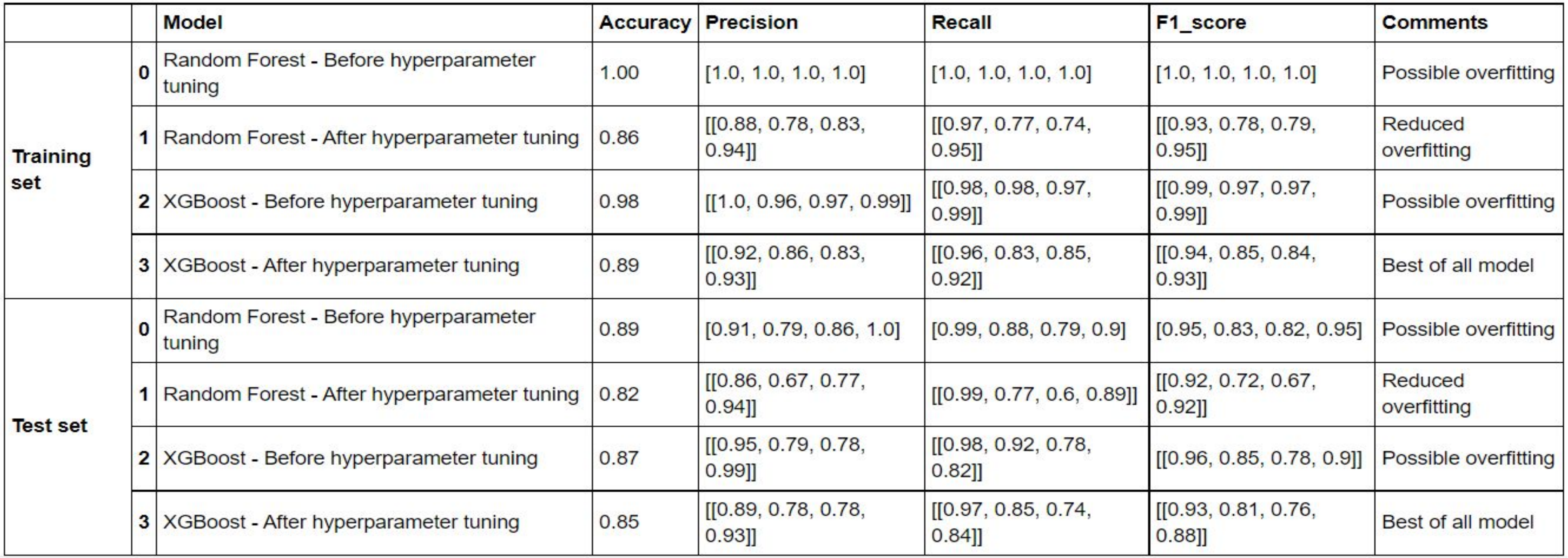
```
The no. of missing values in each variable is:

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
```



- No null values.
- Handled outliers.
- Modified px_height, px_width and sc_h, sc_w into single columns.

Machine Learning Models

- 2 models experimented: Random Forest Classifier and XGBoost Classifier. The evaluation results are:
- Training set shape - 1600,18
- Test set shape - 400,18



		Model	Accuracy	Precision	Recall	F1_score	Comments
Training set	0	Random Forest - Before hyperparameter tuning	1.00	[1.0, 1.0, 1.0, 1.0]	[1.0, 1.0, 1.0, 1.0]	[1.0, 1.0, 1.0, 1.0]	Possible overfitting
	1	Random Forest - After hyperparameter tuning	0.86	[[0.88, 0.78, 0.83, 0.94]]	[[0.97, 0.77, 0.74, 0.95]]	[[0.93, 0.78, 0.79, 0.95]]	Reduced overfitting
	2	XGBoost - Before hyperparameter tuning	0.98	[[1.0, 0.96, 0.97, 0.99]]	[[0.98, 0.98, 0.97, 0.99]]	[[0.99, 0.97, 0.97, 0.99]]	Possible overfitting
	3	XGBoost - After hyperparameter tuning	0.89	[[0.92, 0.86, 0.83, 0.93]]	[[0.96, 0.83, 0.85, 0.92]]	[[0.94, 0.85, 0.84, 0.93]]	Best of all model
Test set	0	Random Forest - Before hyperparameter tuning	0.89	[0.91, 0.79, 0.86, 1.0]	[0.99, 0.88, 0.79, 0.9]	[0.95, 0.83, 0.82, 0.95]	Possible overfitting
	1	Random Forest - After hyperparameter tuning	0.82	[[0.86, 0.67, 0.77, 0.94]]	[[0.99, 0.77, 0.6, 0.89]]	[[0.92, 0.72, 0.67, 0.92]]	Reduced overfitting
	2	XGBoost - Before hyperparameter tuning	0.87	[[0.95, 0.79, 0.78, 0.99]]	[[0.98, 0.92, 0.78, 0.82]]	[[0.96, 0.85, 0.78, 0.9]]	Possible overfitting
	3	XGBoost - After hyperparameter tuning	0.85	[[0.89, 0.78, 0.78, 0.93]]	[[0.97, 0.85, 0.74, 0.84]]	[[0.93, 0.81, 0.76, 0.88]]	Best of all model

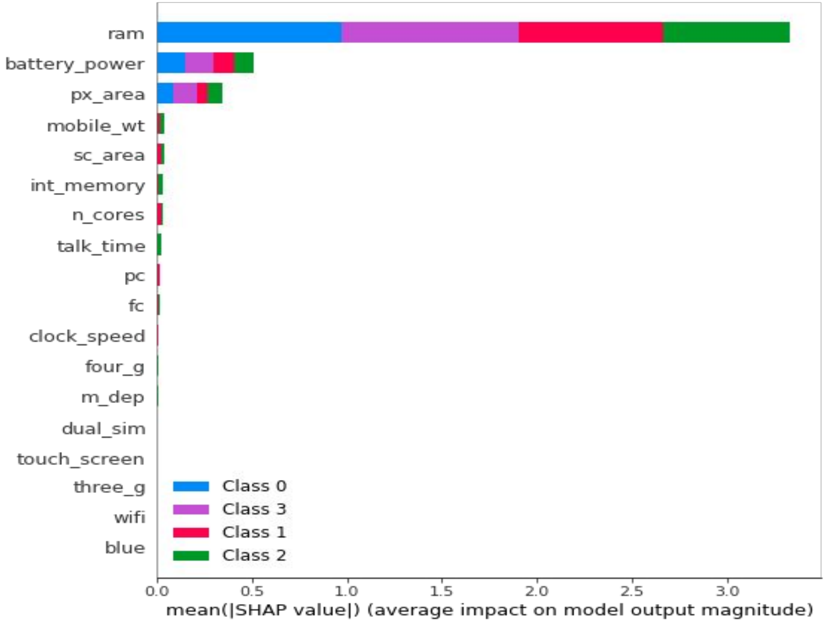
Model selection and validation

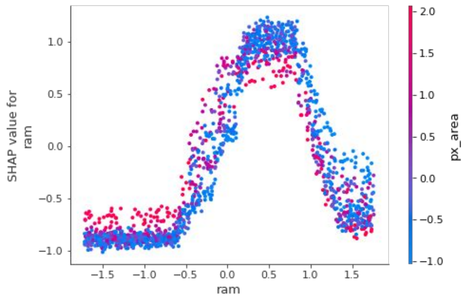
- Random Forest and XGBoost initially overfitted.
- Overfitting was tackled using hyperparameter tuning.
- The best performance was given by XGBoost model, with accuracy of 0.89 and 0.85 for training and test set respectively.
- The best hyperparameter values were:
  - learning rate = 0.13
  - n_estimators = 13
  - max_depth = 13
  - min_child_weight = 10
  - gamma = 1
  - subsample = 0.5

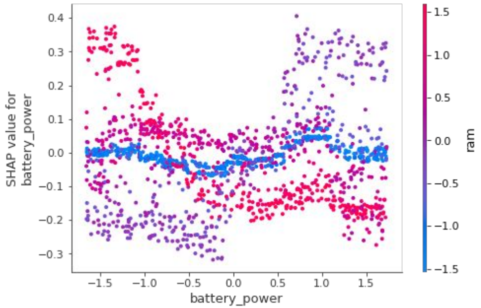
```
The evaluation metric values for training set - XGBoost:
The accuracy of training set =  0.89
The precision of training set =  [0.92 0.86 0.83 0.93]
The recall of training set =  [0.96 0.83 0.85 0.92]
The f1 score of training set =  [0.94 0.85 0.84 0.93]
The confusion matrix of training set =
 [[387  18   0   0]
 [ 34 339  35   0]
 [  0  37 339  25]
 [  0   0  32 354]]

The evaluation metric values for test set - XGBoost:
The accuracy of test set =  0.85
The precision of test set =  [0.89 0.78 0.78 0.93]
The recall of test set =  [0.97 0.85 0.74 0.84]
The f1 score of test set =  [0.93 0.81 0.76 0.88]
The confusion matrix of test set =
 [[92  3  0  0]
 [11 78  3  0]
 [ 0 19 73  7]
 [ 0  0 18 96]]
```

Model Explanation

- Shap techniques were implemented to understand the working of the best model.
- The most important features were Ram, battery power, pixel height and px width.



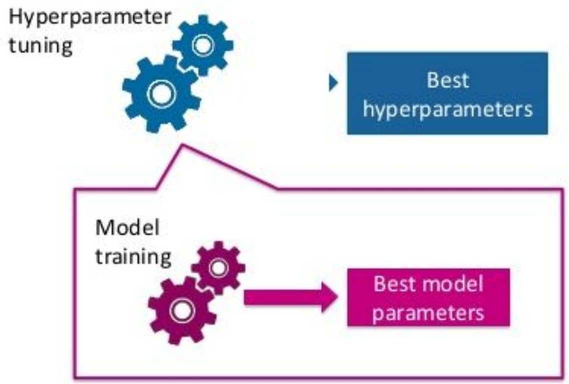




Challenges

- The most challenging part in this exercise was to find an optimal set of parameters that could give us the best performance.
- It took hours to try every combinations and finally selecting the best values.
- The model could even perform better with even finer tunings.

Hyperparameter tuning vs. model training



Conclusion

- Throughout the analysis we went through various steps to determine our predictions for the mobile price range. We started with simple eda where we analysed our dependent variable as well as other independent variables. We found out the correlation, count, relationships with the dependent variable. We looked for missing values and outliers and did some feature modifications.
- Finally we implemented 2 machine learning algorithms namely; RandomForest and XGBoost. We tried hyperparameter tuning to reduce overfitting and increase model performance. The best performance was given by our XGBoost model.
- We also implemented shap techniques to identify the important features impacting our model predictions. We saw ram, battery power, px_height and px_weight were the major contributors. Higher the values of these led to higher predicted values.
- The accuracy of our best model was 0.89 and 0.85 for training and test set respectively. Although, the difference is still 4, considering the simplicity and less no. of observations, this can be considered a good model. Performance can be improved even further by applying fine tunings and gathering more amount of observations so that the models can identify more patterns and become less prone to overfitting. With evolution of new technology, these numbers can change in future hence there will always be a need to check on the model from time to time. I hope this exercise will help you to take a step forward!