CAPSTONE PROJECT-3

Mobile Price Range Prediction

Team Members

Jaya Vishwakarma
Priyvrat Sharma
Richa Pandya
Kavya Sharma

Content

1. Problem Statement
2. Data Summary
3. Data Pre-processing
4. Feature Engineering
5. EDA
6. Model Implementation & Evaluation Metrics
7. Challenges & Conclusion

Data 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
```

Problem Statement

The problem statement is to predict the price range of mobile phones based on the features available (price range indicating how high the price is).

Here is the description of target classes:

- 0 - Low cost phones
- 1 - Medium cost phones
- 2 - High cost phones
- 3 - Very high cost phones

This will basically help companies to estimate price of mobiles to give tough competition to other mobile manufacturer. Also, it will be useful for consumers to verify that they are paying best price for a mobile.

A Quick Data Summary

- We have a record of 2000 mobile phones with 20 features.
- We have **perfectly balanced dataset** with 500 observations for each class. Each column represents the feature of the mobile.
- Interestingly, we had **zero null** values.
- We started with importing all the required python libraries.

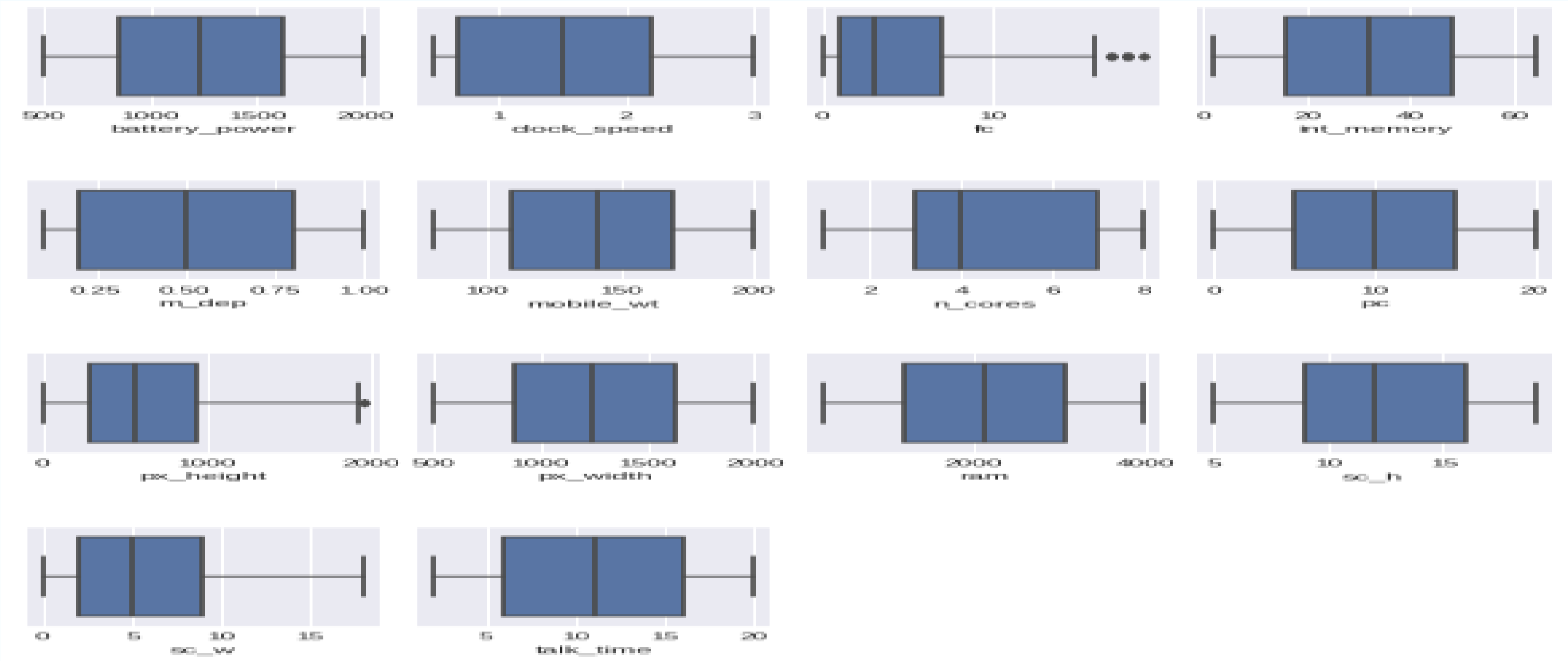
- We implemented different model to find out best model to predict the mobile price range with respect to the mobile features. We have **applied - Decision Tree, Random Forest, Naïve Bayes, KNN, Logistic Regression and XG Boost**.
- At last we conclude that **Logistic Regression** is performing **better** than any other model.

Python Libraries & Classification Model

Preparing Our Data for Deep Analysis

- Step 1: Problem Description & Data Description
- Step 2: Understanding & Pre-processing of data: Null values and Duplicate values
- Step 3: Creating numerical and categorical Columns
- Step 4: Univariate analysis and check multicollinearity
- Step 5: Splitting the dataset into the training and test sets.
- Step 6: Model Implementation
- Step 7: Model performance
- Step 8: Challenges
- Step 9: Conclusion

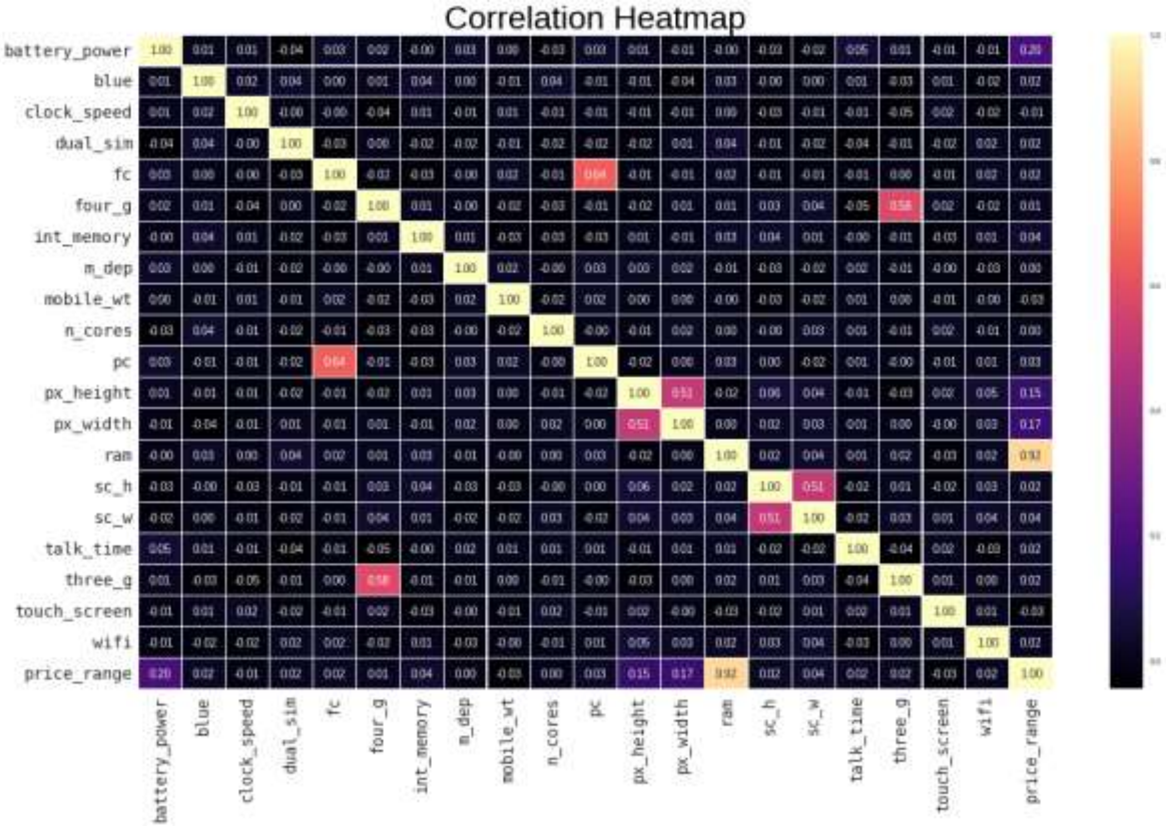
Outliers Detection



The main reason for this detection is that the outliers can **cause serious issues in statistical analysis**. Hence we have checked this before starting analysing our data. So that we can visualise data properly without having measurement errors, data entry or processing errors.

In our data, there seems to be **no outlier**.

Multi Collinearity

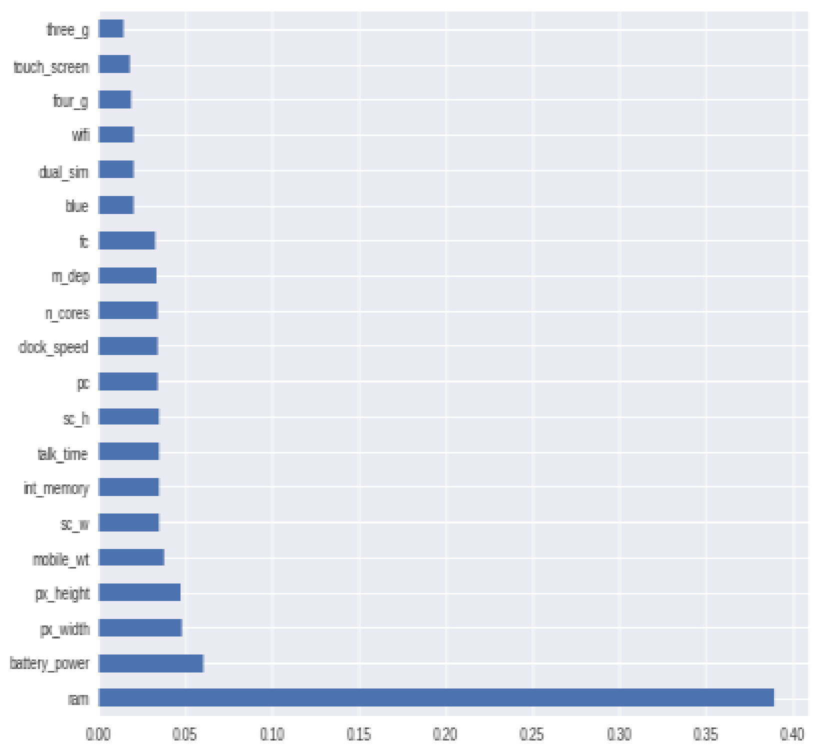


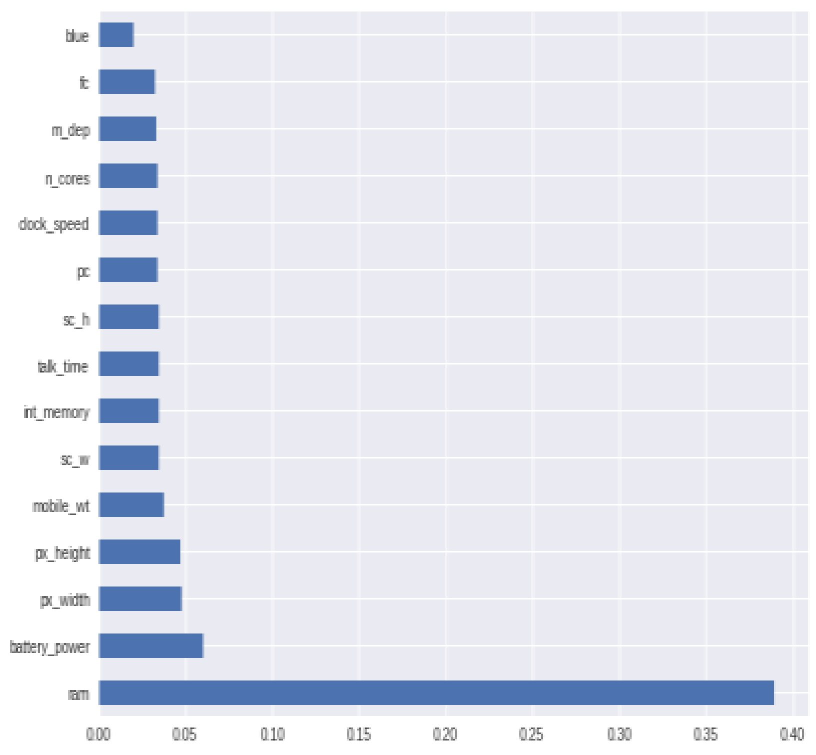
From this heatmap , we can observe correlation between all of our feature.

Here, some of the feature shows **high positive correlation**.

which means that our dependent feature is highly related with other independent features, In simple word that shows **multicollinearity does exist** in our dataset .

Feature Selection

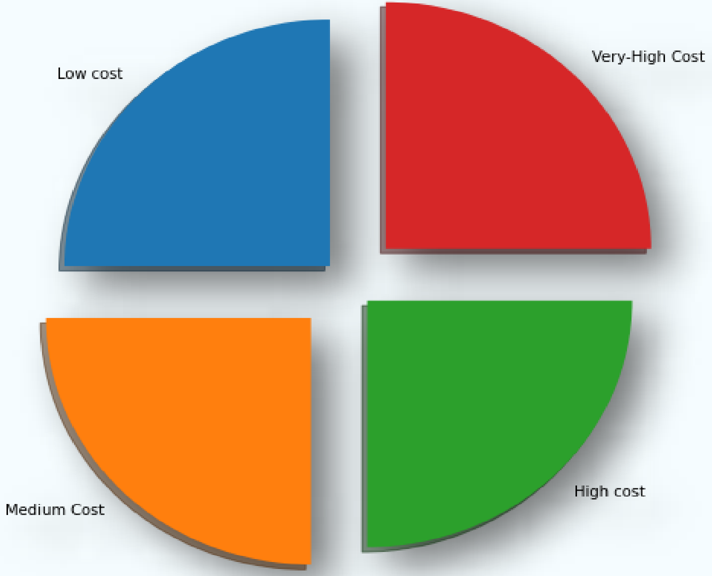


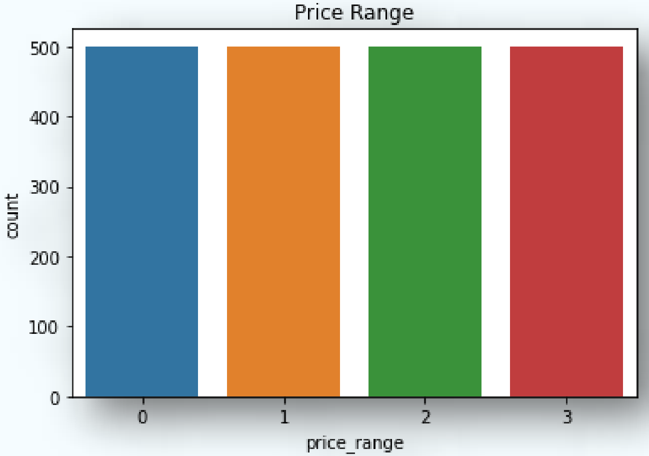


Exploratory Data Analysis

We are done with pre-processing our data.
Our next step is to perform EDA.

Price Range

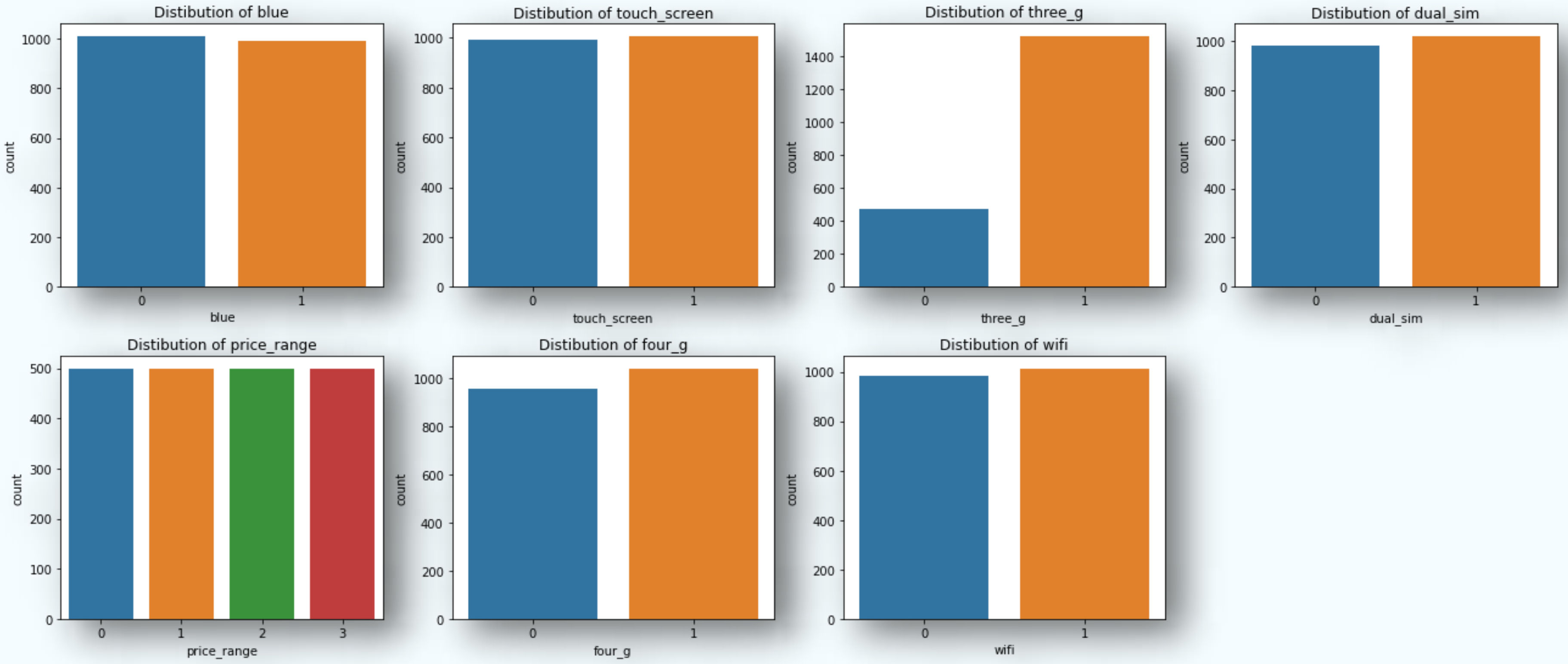




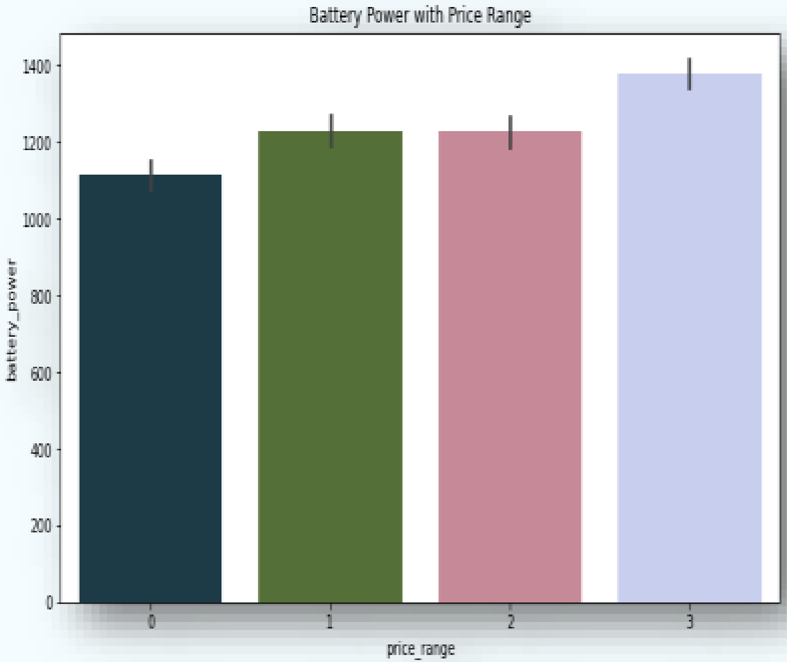
Our data is equally distributed. Prince range are as follows:

- 0 - Low cost phones
- 1 - Medium cost phones
- 2 - High cost phones
- 3 - Very high cost phones

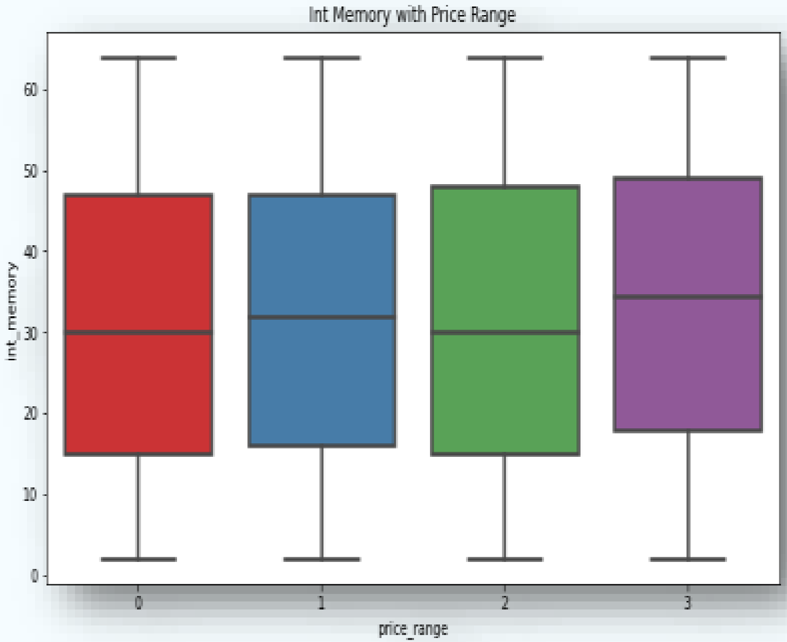
Categorical Analysis



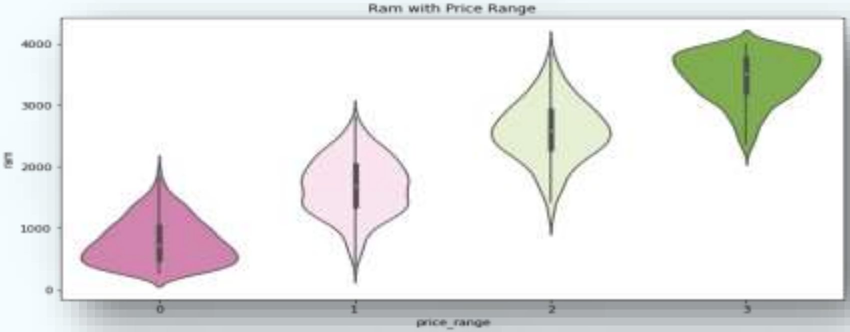
Price with Battery Power & Int Memory

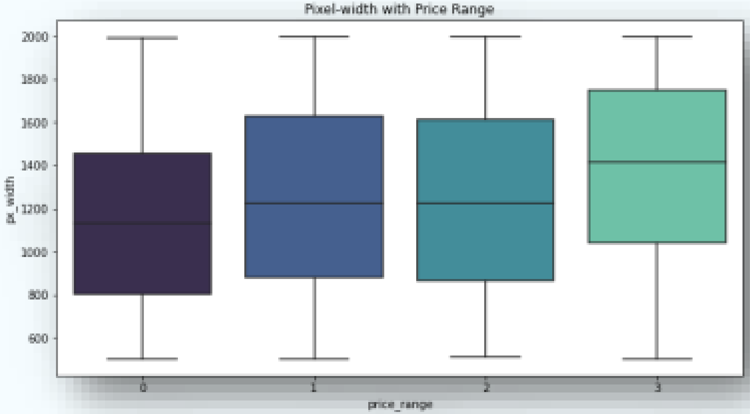


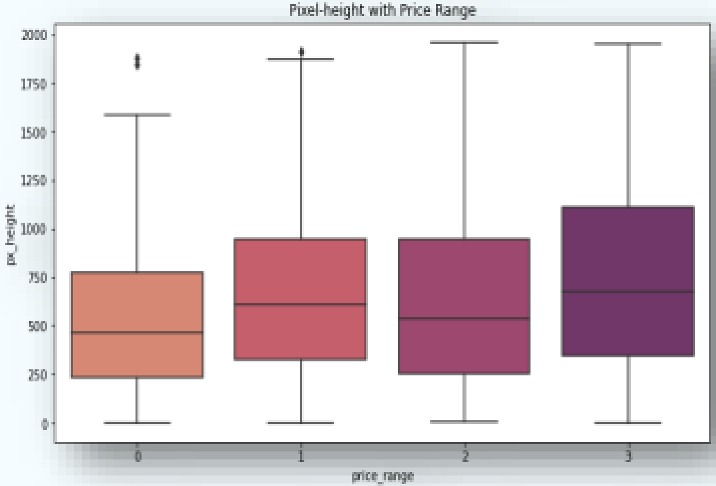
We can see from bar plot of battery against price range that more expensive the phone is higher the battery power is .



Price Range with Pixel Height-width & Ram

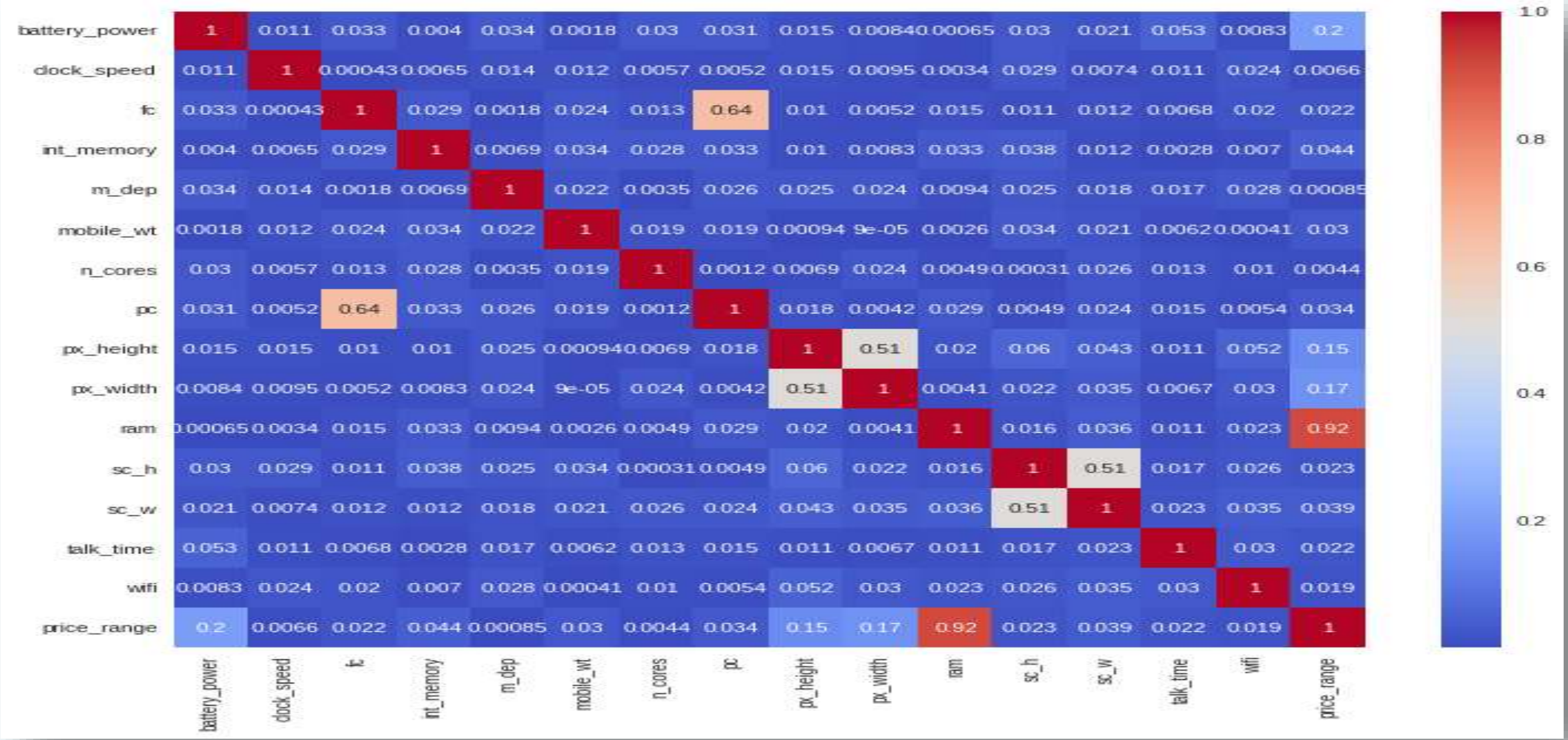






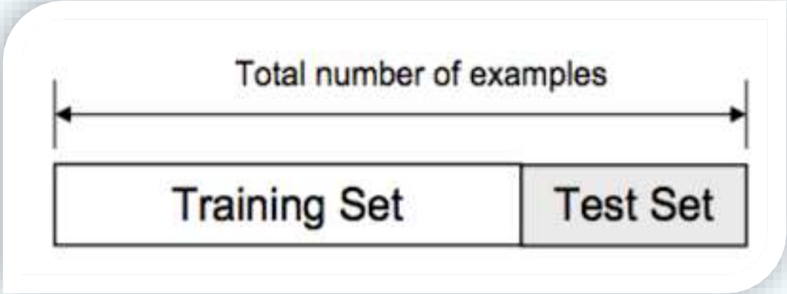
Expensive mobile phone's pixel width is more.
Price is increasing with Ram.

Correlation



Train Test Split

Before, fitting any model it is a **rule of thumb** to split the dataset into a training and test set.

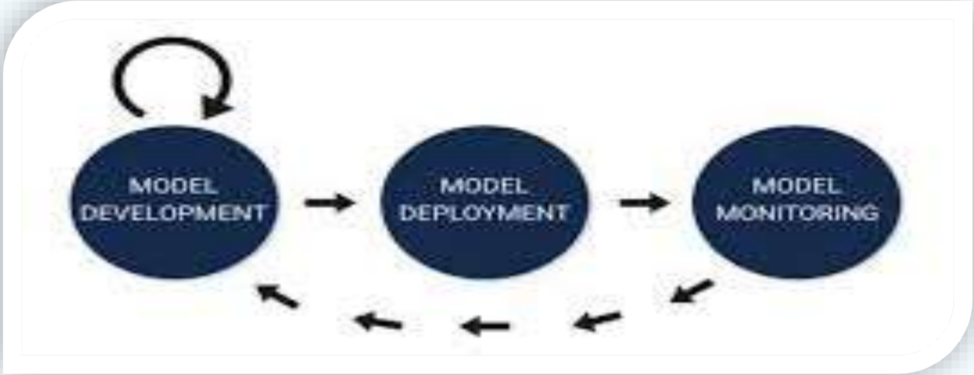


This means some proportions of the data will go into training the model and some portion will be used to evaluate how our **model is performing** on any unseen data.

The proportions may vary from 60:40, 70:30, 80:20 depending on the person. But mostly used is **80:20** for training and testing respectively. In this step we will split our data into **training and testing set using Sickit learn library**.

ML Model Development

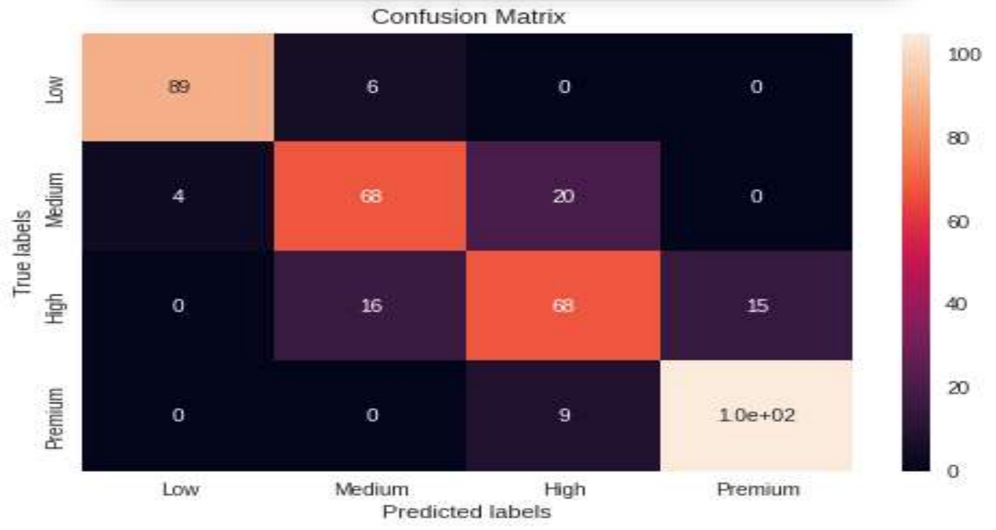
Build the model using classification algorithms **Decision Tree, Random Forest, Naïve Bayes, KNN, Logistic Regression, XG Boost**.
And then we checked performance of these model using score metrics.



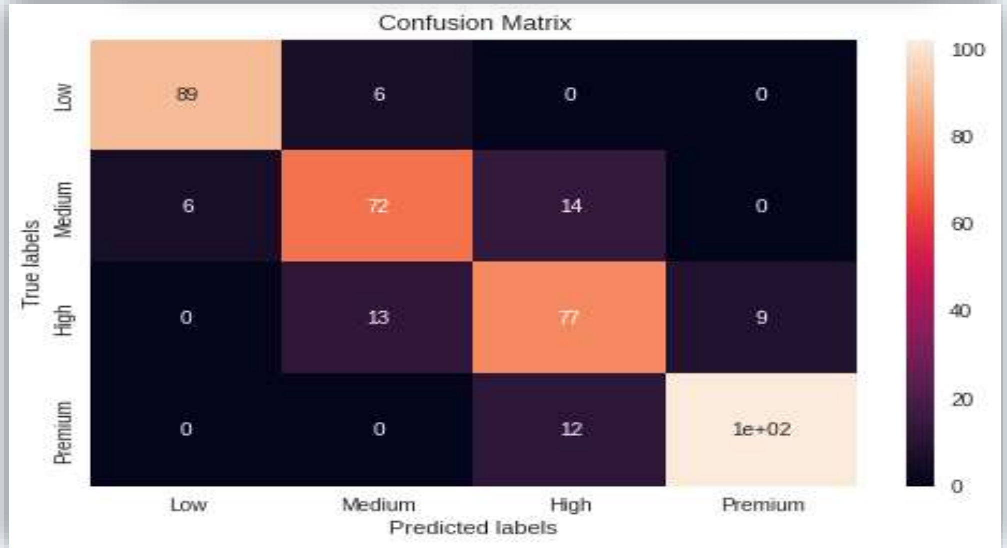
Naive Bayes

Naive Bayes is a supervised machine learning model majorly used in solving classification problems. Supervised machine learning models are those where we use in *text classification* that includes a high-dimensional training dataset.

```
Evaluation metrices on the test data
Accuracy : 0.825
Recall : 0.825
Precision : 0.8239428786535121
F1 : 0.8242390777915095
[0.825, 0.825, 0.8239428786535121, 0.8242390777915095]
```



```
Evaluation metrices on the test data
Accuracy : 0.84
Recall : 0.84
Precision : 0.8406531109445278
F1 : 0.8402461172404688
[0.84, 0.84, 0.8406531109445278, 0.8402461172404688]
```



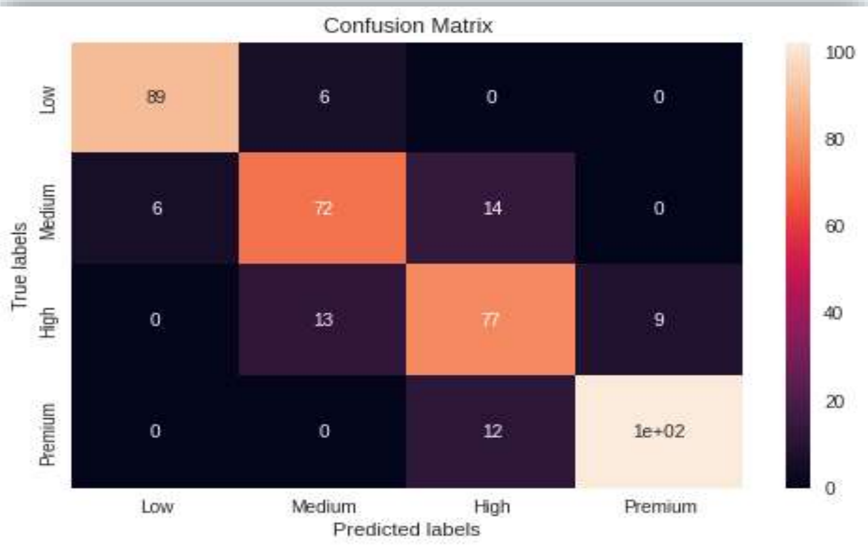
Decision Tree

Decision tree is the most powerful and popular tool to deal with classification problem. A Decision tree is a flowchart like tree structure, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

Random Forest

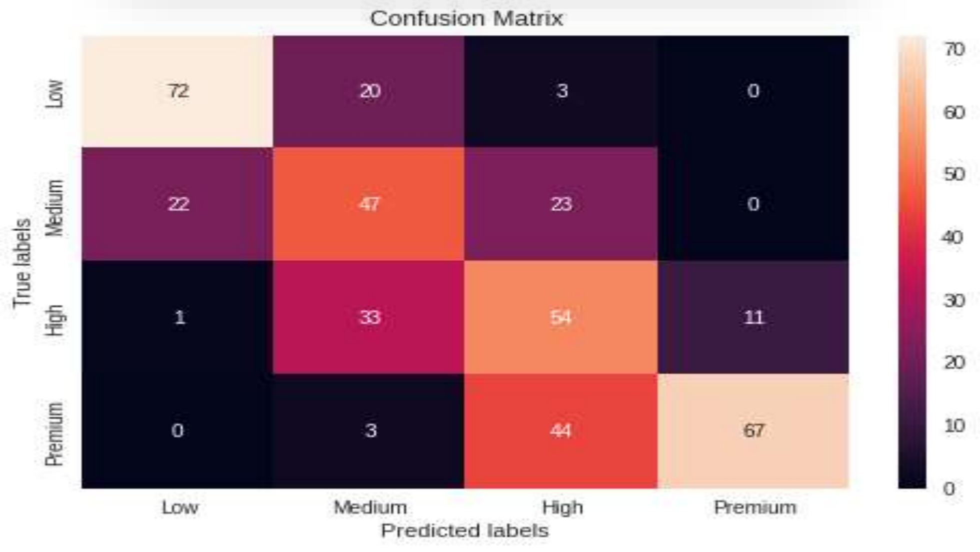
Random Forest is a bagging type of Decision Tree Algorithm that creates a number of decision trees from a randomly selected subset of the training set, collects the labels from these subsets and then averages the final prediction depending on the most number of times a label has been predicted out of all.

```
Evaluation metrices on the test data
Accuracy : 0.84
Recall : 0.84
Precision : 0.8406531109445278
F1 : 0.8402461172404688
[0.84, 0.84, 0.8406531109445278, 0.8402461172404688]
```



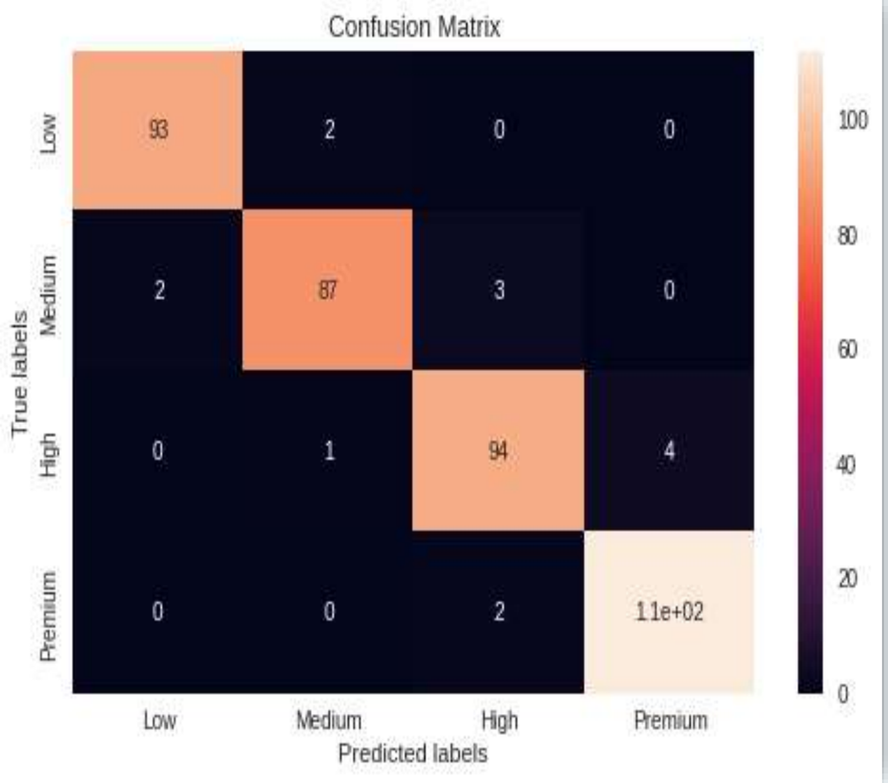
KNN

```
Evaluation metrices on the test data
Accuracy : 0.6
Recall : 0.6
Precision : 0.637541406682888
F1 : 0.6096435157238129
[0.6, 0.6, 0.637541406682888, 0.6096435157238129]
```



KNN is a method which is used for classifying objects based on closest training examples in the feature space. KNN is the most basic type of instance-based learning or lazy learning. It assumes all instances are points in n-dimensional space. A distance measure is needed to determine the “closeness” of instances. It classifies an instance by finding its nearest neighbours and picking the most popular class among the neighbours.

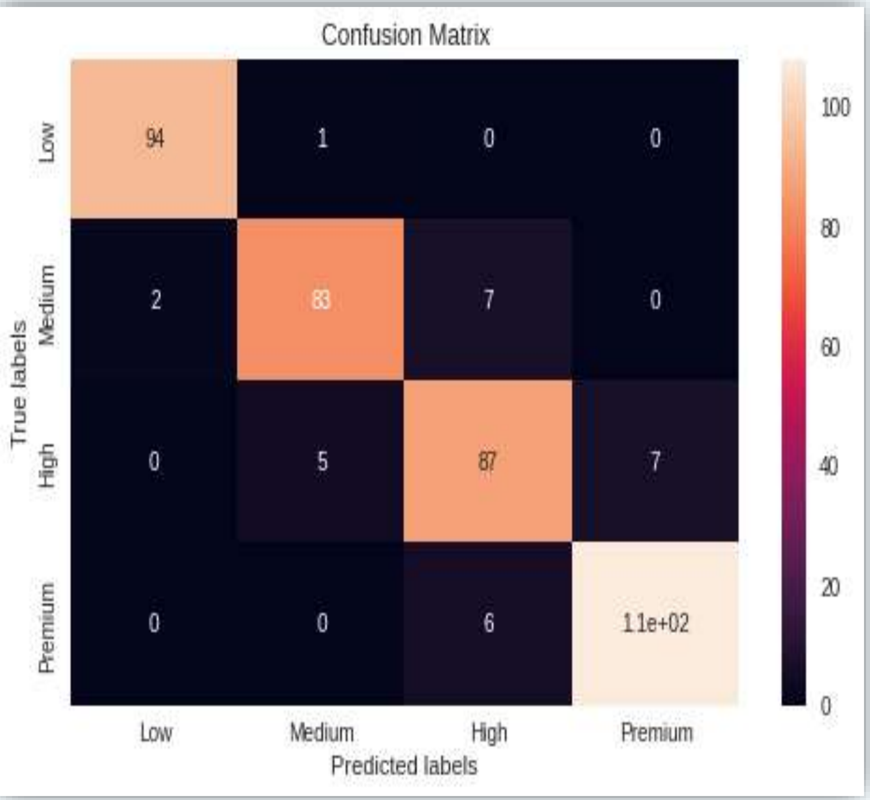
Logistic Regression




```
Evaluation metrices on the test data
Accuracy : 0.965
Recall : 0.965
Precision : 0.9650057471264368
F1 : 0.9649553272814143
[0.965, 0.965, 0.9650057471264368, 0.9649553272814143]
```

Logistic regression estimates the probability of an event occurring. Since the outcome is a probability, the dependent variable is bounded between 0 and 1

XG Boost



XG Boost stands for Extreme Gradient Boosting. The term gradient boosting consists of two sub-terms gradient and boosting.

```
Evaluation metrices on the test data
Accuracy : 0.93
Recall : 0.93
Precision : 0.9300236392688487
F1 : 0.9299368559017597
[0.93, 0.93, 0.9300236392688487, 0.9299368559017597]
```

Model Performance

		Model	Accuracy Score	Recall	Precision	F1
Training set	0	GNB	0.82	0.82	0.82	0.82
	1	KNN	0.74	0.74	0.75	0.74
	2	Decision tree	0.92	0.92	0.92	0.92
	3	Random Forest	0.92	0.92	0.92	0.92
	4	Logistic Regression	0.98	0.98	0.98	0.98
	5	XG Boost	1.00	1.00	1.00	1.00
Test set	0	GNB	0.82	0.82	0.82	0.82
	1	KNN	0.60	0.60	0.64	0.61
	2	Decision tree	0.84	0.84	0.84	0.84
	3	Random Forest	0.84	0.84	0.84	0.84
	4	Logistic Regression	0.96	0.96	0.97	0.96
	5	XG Boost	0.93	0.93	0.93	0.93

Challenges

Most of the models are not able to get good accuracy for each class of target variable.

With hyperparameter tuning, even after assigning different parameters values **XG boost** performed **not so good** on test data but It works really well on training set.

Conclusion

- **XG Boost** is giving us good overall accuracy but they didn't perform well on Individual classes.
- Out of all the model we have tried **logistic regression** is performing **well** on overall as well as Individual classes.
- Ram, Battery power, Mobile weight, Screen size and pixels are **key features** in predicting the mobile price range.
- Most of the mis-classifications were encountered between **Medium range phones** and **high range phones**. To counter that we can train a specific model for these two classes and can reclassify the cases when base model predicts the result as Medium range or High range

THANK YOU!