# Smartphone Price Range Classification

## Description

The aim of this project is to classify the mobile phone according to its price in Bangladesh market. This kind of prediction will help companies estimate price of mobiles to give tough competition to other mobile manufacturer specially in Bangladesh’s market. Also it will be useful for consumers to verify that they are paying best price for a mobile with respect to the features they received.

We will use machine learning technique to classify mobile phones according to some features and attributes in few price range classes like -

1. Low range ( <10k)
2. Lower mid range (10-25k)
3. Mid range (25-40k)
4. Higher mid range (40-60k)
5. Higher range (>60k)

## 

## Dataset

The dataset we will use to train our classification model is collected from a popular website in Bangladesh named *Mobile Dokan.*

The website link - [www.mobiledokan.com](https://www.mobiledokan.com/all-brands/)

### Collecting data from the website

We scrapped and dumped all the necessary data from the website in json format with a python script. One function from the python script and a sample raw data are in the following.

def get\_product\_details(driver, product\_link):  
 """This method fetches detailed information of the product and store those into a list of dictionaries and returns that  
  
 Args:  
 driver (obj): Passing the chromedriver that was initalized  
 product\_link (str): The link of the product details page

"""  
  
 driver.get(product\_link)  
 scroll\_down\_the\_page(driver)  
 data = {}  
 name = driver.find\_element\_by\_css\_selector('.entry-header').text  
 data['Name'] = name  
 data['Brand'] = product\_link.split('/')[3]  
 data['Link'] = product\_link  
 try:  
 spec\_tables = driver.find\_elements\_by\_css\_selector(  
 '.table-is-responsive')  
 print('Table found')  
 price\_tds = spec\_tables[0].find\_elements\_by\_tag\_name('td')  
 data[price\_tds[0].text] = price\_tds[1].text  
 rows = spec\_tables[1].find\_elements\_by\_tag\_name('tr')  
  
 print(len(rows))  
 for row in rows:  
 try:  
 tds = row.find\_elements\_by\_tag\_name('td')  
 data[tds[0].text] = tds[1].text  
 except:  
 print('Could not find the data')  
 continue  
  
 except:  
 print('No product details found for {}'.format(product\_link))  
  
 return data

A Sample raw json data -

{"Name": "Samsung Galaxy S21 FE 5G", "Brand": "samsung",   
 "Link": "https://www.mobiledokan.com/samsung/samsung-galaxy-s21-fe-5g/",   
 "Official \u272d": "\u09f369,999 8/128 GB", "First Release": "January 7, 2022",   
 "Colors": "White, Graphite, Lavender, Olive", "Network": "2G, 3G, 4G, 5G",   
 "SIM": "Hybrid Dual Nano SIM", "WLAN": "\u2705 dual-band, Wi-Fi direct, Wi-Fi hotspot",   
 "Bluetooth": "\u2705 v5.0, A2DP, LE", "GPS": "\u2705 A-GPS, GLONASS, BDS, GALILEO",   
 "Radio": "Unspecified", "USB": "v3.2", "OTG": "\u2705", "USB Type-C": "\u2705",   
 "NFC": "\u2705", "Style": "Punch-hole", "Material": "Gorilla Glass Victus front, plastic back, aluminum frame",   
 "Water Resistance": "\u2705 IP68 dust / waterproof (up to 1.5m for 30 mins)",   
 "Dimensions": "155.7 x 74.5 x 7.9 millimeters", "Weight": "177 grams", "Size": "6.4 inches",   
 "Resolution": "32 Megapixel", "Technology": "Dynamic AMOLED 2X Touchscreen",   
 "Protection": "\u2705 Corning Gorilla Glass Victus", "Features": "Loudspeaker (stereo speakers), 32-bit/384kHz audio",   
 "Video Recording": "4K (2160p), gyro-EIS", "Type and Capacity": "Lithium-polymer 4500 mAh (non-removable)",   
 "Fast Charging": "\u2705 25W Fast Charging (50% in 30 minutes)\n15W Fast Wireless Charging\nUSB Power Delivery 3.0",   
 "Reverse Charge": "\u2705 4.5W Reverse Wireless Charging", "Operating System": "Android 12 (One UI 4)",   
 "Chipset": "Exynos 2100 (5 nm)", "RAM": "8 GB", "Processor": "Octa core, up to 2.9 GHz", "GPU": "Mali-G78 MP14",   
 "ROM": "128 GB", "MicroSD Slot": "\u2716", "3.5mm Jack": "\u2716", "Fingerprint": "\u2705 In-display (optical)",   
 "Face Unlock": "\u2705", "Notification Light": "", "Sensors": "Fingerprint, Accelerometer, Gyro, Proximity, Compass",   
 "More Features": "\u2013 Samsung Pay (Visa, MasterCard certified)\n\u2013 Bixby",   
 "Manufactured by": "Samsung", "Made in": "Bangladesh", "Sar Value": ""}

### 

### Preprocess data

We examined each column of our data thoroughly and kept only the best suited column for training purpose. There are almost 45 columns in the raw dataset and we discarded most of them because they are redundant or unnecessary for our classification. Full preprocessing code is available in **data\_preprocess.ipynb** file.

**All columns after preprocessing -**

1. PriceRange  
2. ReleaseYear  
3. Camera  
4. AvailableColors  
5. Ram\_GB  
6. NetworkSupport  
7. BluetoothVersion  
8. Rom\_GB  
9. BatteryCapacity  
10. VideoRes  
11. NumOfSensors  
12. FastCharging

13. WR\_version  
14. OTG  
15. USB\_Version  
16. TypeC  
17. NFC  
18. Display  
19. Protection  
20. Chipset  
21. Processor  
22. GPU  
23. Fingerprint  
24. FaceLock

Few important features such as Camera Megapixel, ROM, Network support, Display protection, Bluetooth version etc that highly influence the Classification attribute which is Price Range are shown below. These plots show us the correlation between the features and the label.

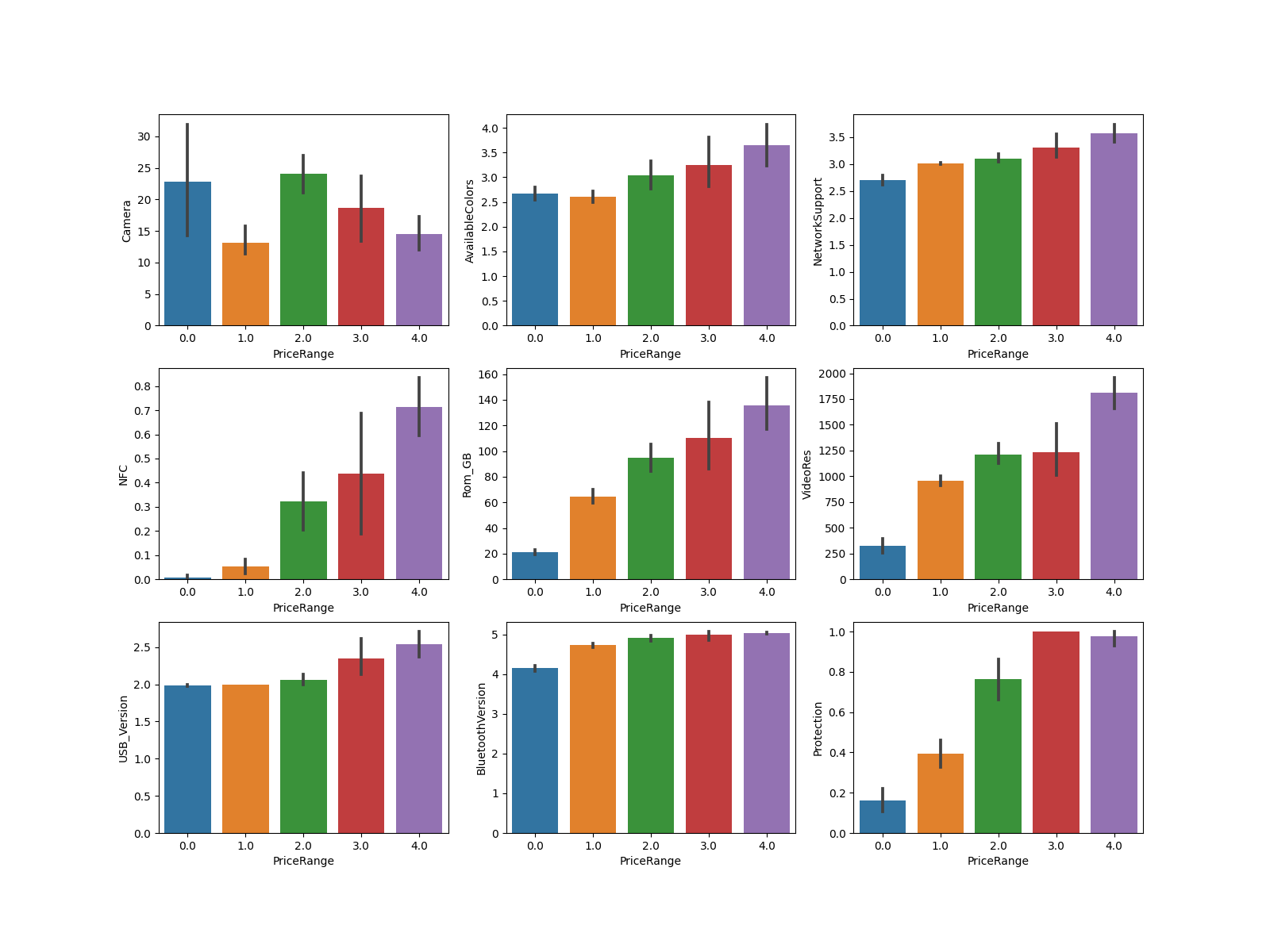


Figure 1: Features versus PriceRange curves

The cleansed and usable dataset after preprocessing is available in **Dataset/final\_dataset.csv**

## 

## Training models

We have around 500 columns, which obviously is not good enough to train a classification model. But to overcome our limitation by getting the maximum result, we preprocessed our dataset further.

### 

### Further processing

As still are few columns those are not numeric and we can not directly use them to train our model.

Those columns are - 1. Display 2. Chipset 3. Processor 4. GPU

As these columns have direct influence in the price range, we turned them into One hot encoded Columns by treating their value as categorical. Also we normalized our numeric columns for better result. Therefore, our dataset became usable for training purpose.

### 

### Test and train split

Though we have limited amount of dataset, we still have to split our dataset for training and testing purpose. We extracted **20%** of our rows randomly for testing purpose and rest of the rows will be using to train our models.

### 

### Train and Evaluate models

We will be training seven classification models. Which are -

1. **Random Forest Classifier:** A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Number of trees used to train our model is 100
2. **Logistic Regression Classifier:** A statistical model that in its basic form uses a logistic function to model a binary dependent variable.
3. **KNN Classifier:** Classifier implementing the k-nearest neighbors vote. We used 10 neighbors.
4. **SVM Classifier:** A supervised learning method effective in high dimensional spaces.
5. **Naive Bayes Classifier:** A set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of conditional independence between every pair of features.
6. **Decision Tree Classifier:** A non-parametric supervised learning method used for classification.
7. **XGBoost:** XGBoost stands for e**X**treme **G**radient **B**oosting. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost is fast and performs well most of the time.

### 

### Evaluation metrics

We will be using four performance metric to evaluate our models. They are calculated from confusion matrix which is the following.

|  | Actually Positive | Actually Negative |
| --- | --- | --- |
| Predicted Positive | True Positive, TP | False Positive, FP |
| Predicted Negative | False Negative, FN | True Negative, TP |

1. **Accuracy:** Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. *Accuracy = (TP+TN)/(TP+FP+FN+TN)*
2. **Recall or Sensitivity:** Recall is the ratio of correctly predicted positive observations to the all observations in actual true class. *Recall = TP/(TP+FN)*
3. **Precision:** Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. *Precision = TP/(TP+FP)*
4. **F1 Score:** F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. *F1 Score = 2*(Recall \* Precision) / (Recall + Precision)\*

### 

### Final Result

| Model Name | Accuracy | Recall | Precision | F1 Score |
| --- | --- | --- | --- | --- |
| Random Forest | 0.808 | 0.685 | 0.685 | 0.649 |
| Logistic Regression | 0.747 | 0.653 | 0.574 | 0.608 |
| KNN | 0.777 | 0.705 | 0.829 | 0.706 |
| SVM | 0.787 | 0.688 | 0.619 | 0.650 |
| Naive Bayes | 0.696 | 0.648 | 0.598 | 0.611 |
| Decision Tree | 0.747 | 0.687 | 0.697 | 0.676 |
| XGBoost | 0.818 | 0.727 | 0.739 | 0.711 |

## 

## Discussion

After training seven machine learning model for classification, we came to know that XGBoost performs better than all other models with almost 82% accuracy. The second best model is Random Forest with 81% accuracy. So we can observe a fact that, Ensemble Learning, which is using multiple model together, results in better performance. The possible reason could be, as our training dataset is limited, there is a high chance of overfit. So Ensembling models or decision trees ameliorate overfitting and results in higher accuracy. The model which performs worst among all of them is Naive Bayes, which is understandable. As we know, Naive Bayes classifier is suitable for categorical input variables where our dataset are highly numerical.

But still, there are still some rooms for improvement. For example if we could feed more data to our model, it would performs well. There are also some factors that affects the price range of a phone, for example ‘Brand Value’ or the Design aesthetics, materials used, which we were unable to incorporate in our dataset.