Capstone Project

Supervised ML – Classification
Mobile Price Range Prediction

Subham Choudhary

Contents

Introduction

Mobile phones have become such an intrinsic part of our lifestyle that we can’t imagine our life without it. The reliance on mobile phones have increased significantly due to the comfort it provides in our day-to-day life.

Keeping in mind with the ever more evolving technology and to keep pace with it, mobile manufacturers employ software and hardware in their mobile phones, so that it can cater to the needs of customers.

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(e.g.: - 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.

Further based on different features of mobile phones, develop a model that would classify each mobile into different categories of price range, i.e., into categorical value of 0 (low), 1 (Medium), 2 (High) and 3 (Very High).

Data Overview

- The dataset contains information about mobile phones features which is used to estimate the price range.
- The dataset contains 2000 non-null observations and 21 columns.

Data Attributes:

- **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

Data Overview (continued)

- **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)

Exploratory Data Analysis

Data Cleaning: -

- ❑ Columns having values = 0, are assigned with their mean values.
- ❑ Any Duplicate values are dropped.
- ❑ The column features were binned into 3 categories: - Continuous features, Discrete features and Binary features.

```
[ ] # Total phones with sc_w = 0
    len(mobile_df[mobile_df.sc_w == 0])

    180
```

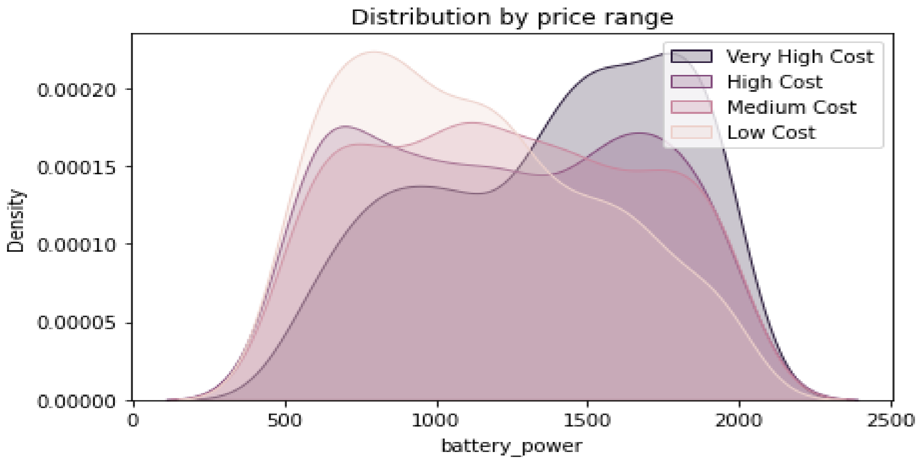
```
[ ] # Total phones with px_height = 0
    len(mobile_df[mobile_df.px_height == 0])

    2
```

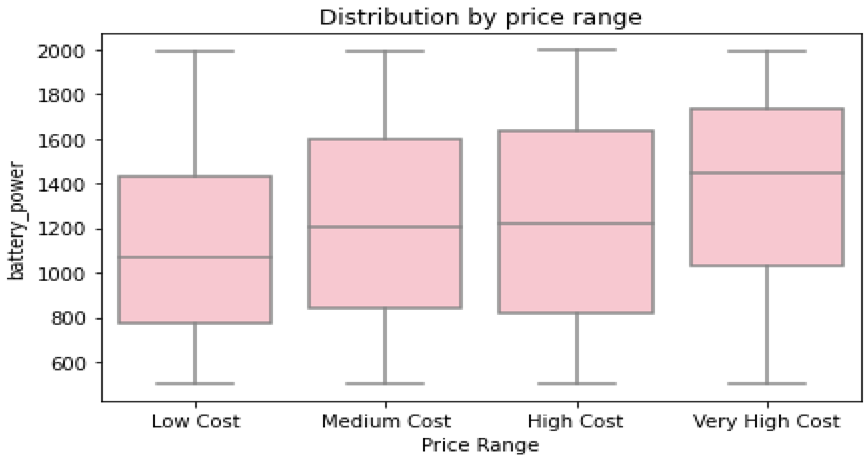
```
# Categorizing binary, discrete and continuous features

binary_features = ['blue', 'dual_sim', 'four_g', 'three_g', 'touch_screen', 'wifi']
discrete_features = ['clock_speed', 'fc', 'int_memory', 'm_dep', 'mobile_wt', 'n_cores', 'pc', 'sc_h', 'sc_w', 'talk_time']
continuous_features = ['battery_power', 'px_height', 'px_width', 'ram']
```

Data Visualization

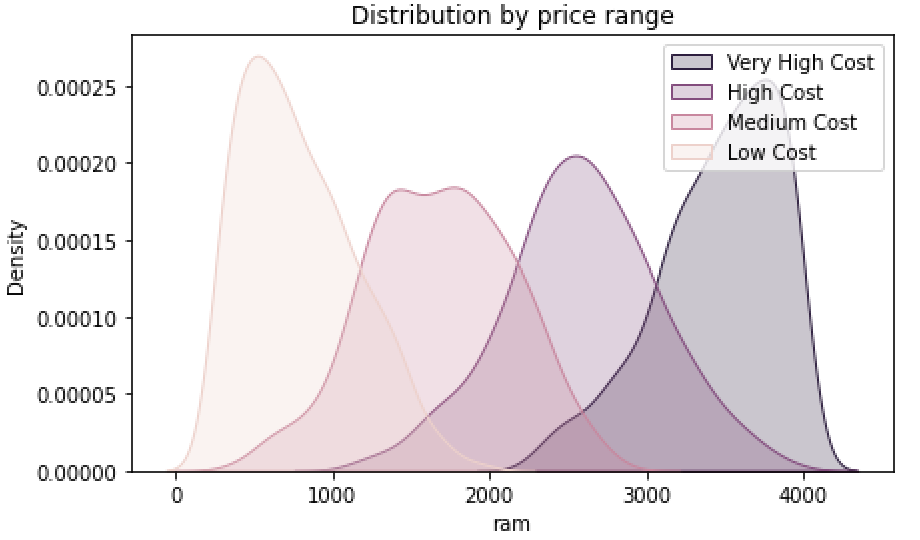
Continuous Features

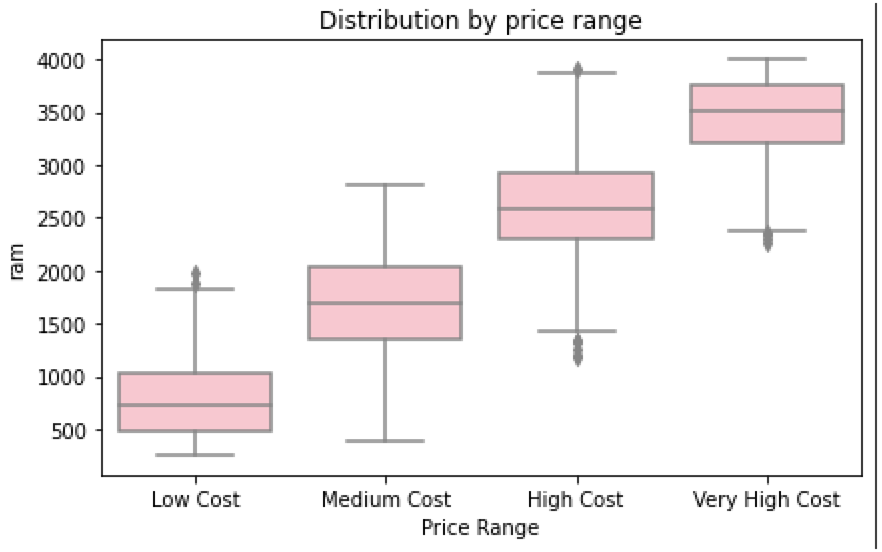
Battery Power





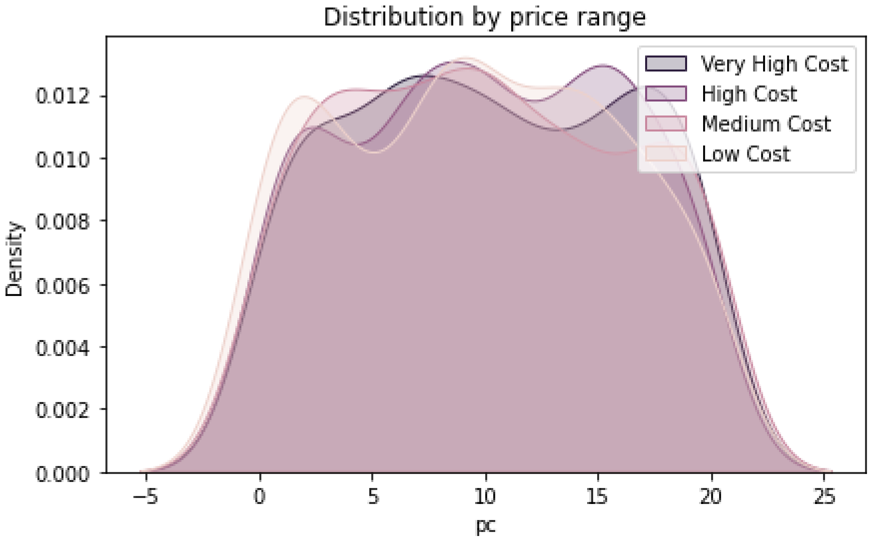
RAM

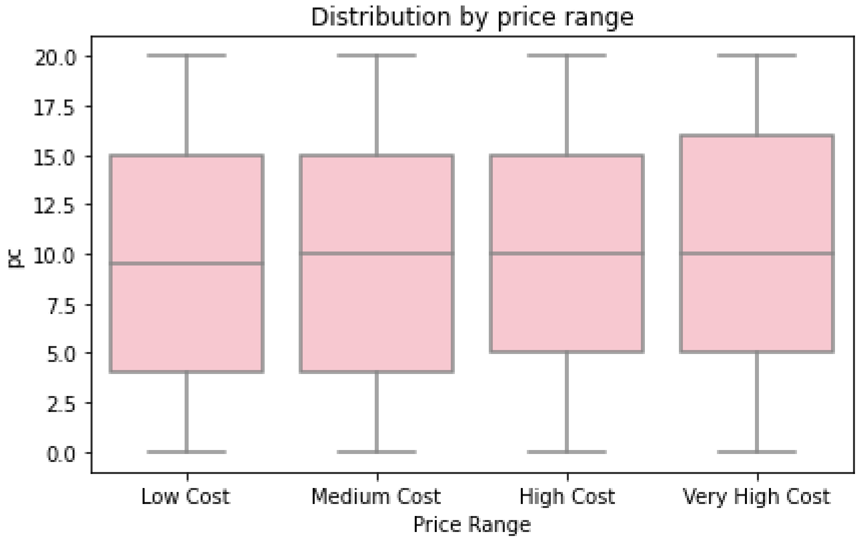




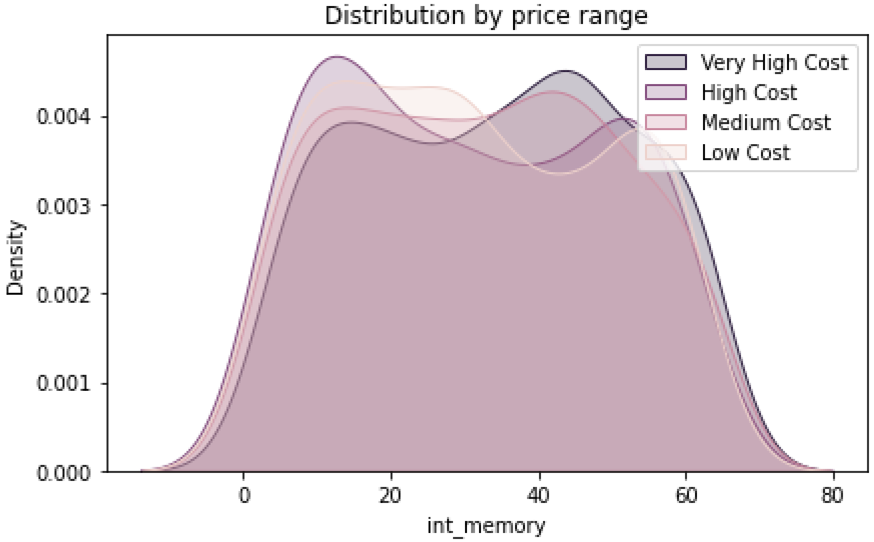
Discrete Features

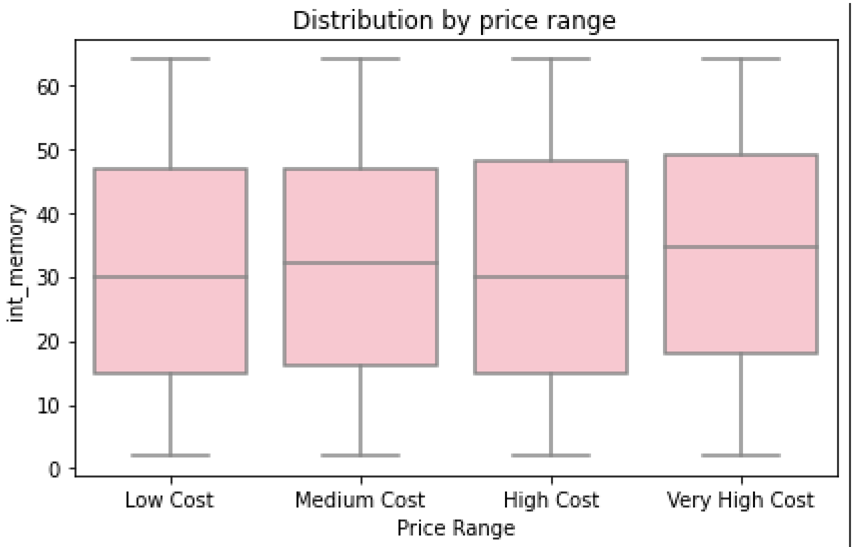
Primary Camera





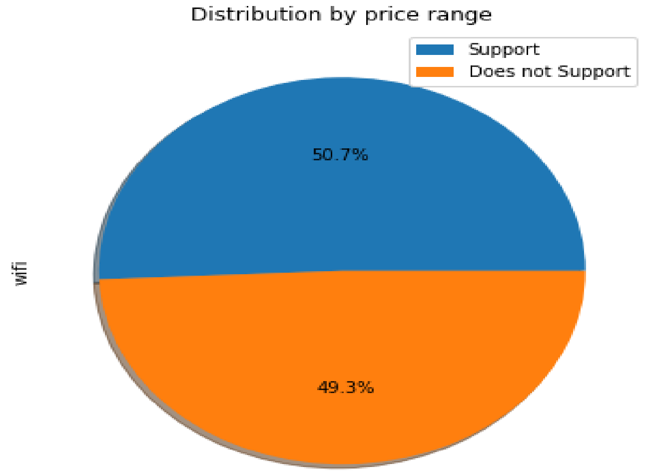
Internal Memory

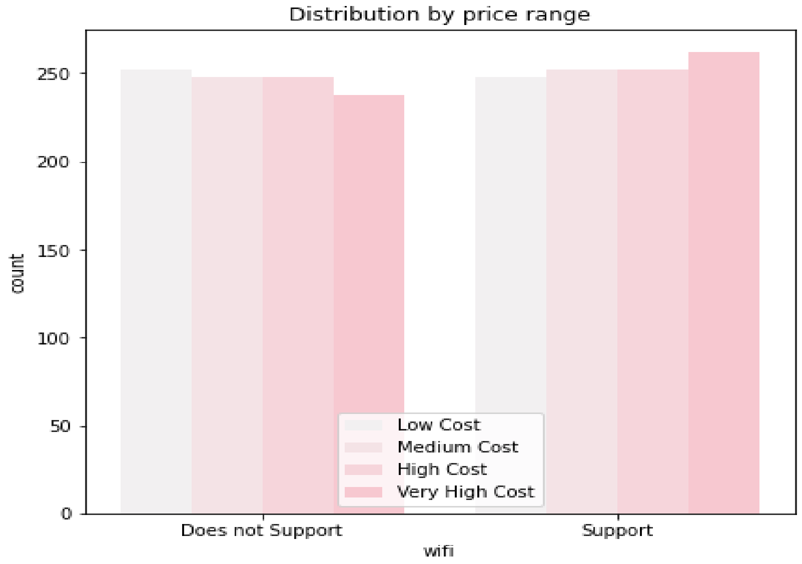




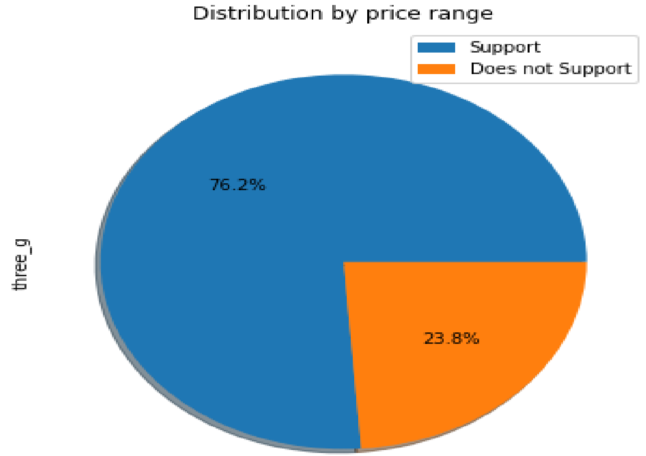
Binary Variable

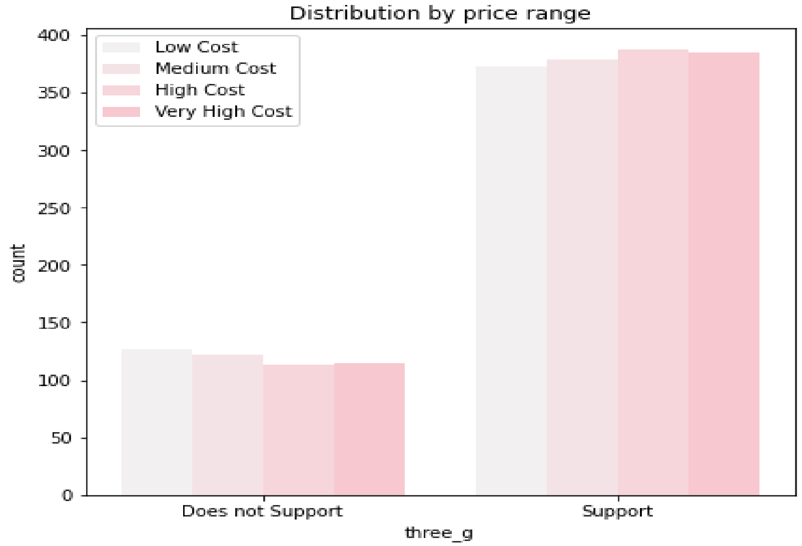
WIFI





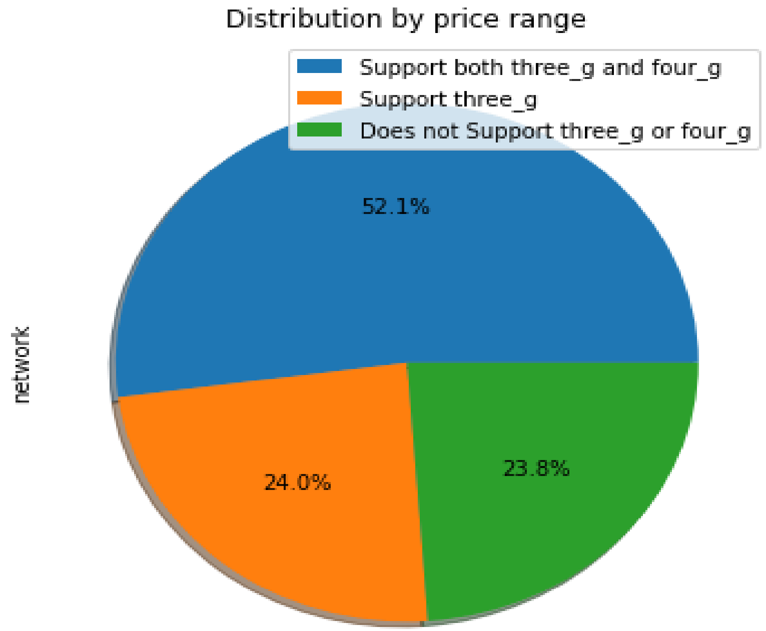
3G

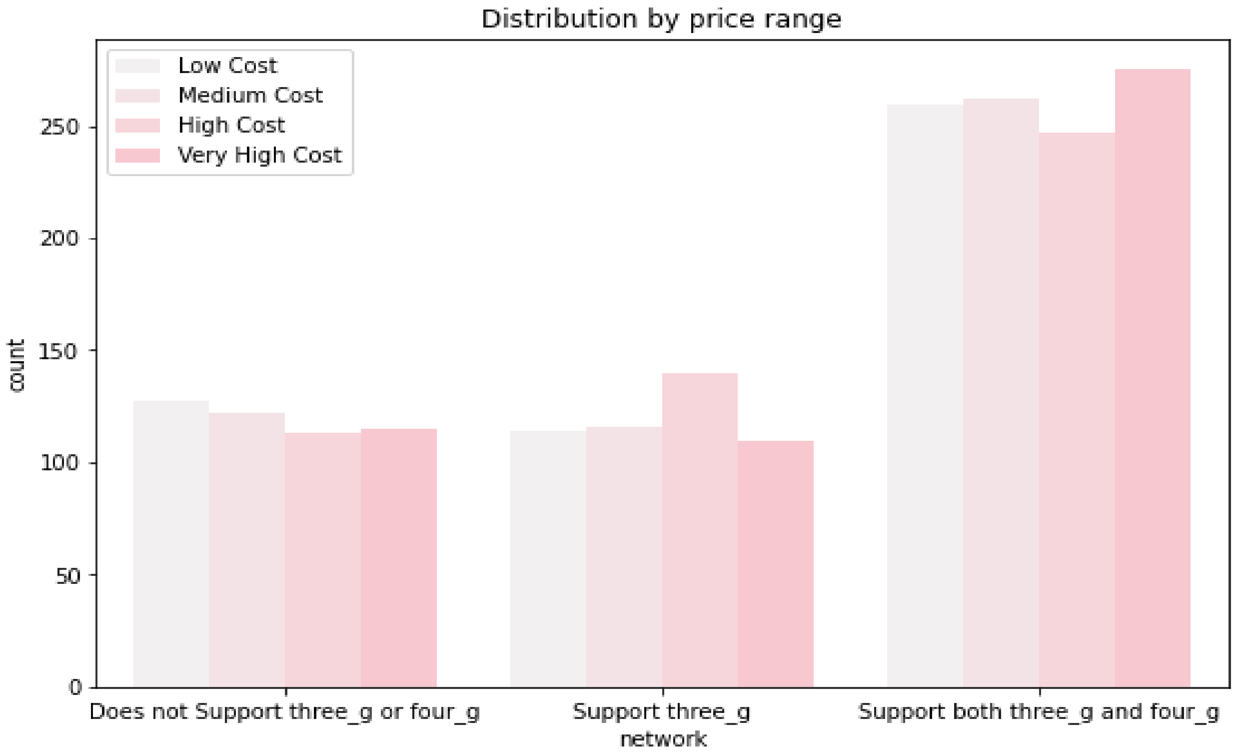




Data Preparation

- Screen width and screen height converted into one variable as “screen size”
- 3G and 4G features are merged into one variable as “network”
- Pixel height and width merged into one variable “pixels”





Predictive Modelling

Logistic Regression

- ❖ Model is able to fit the data very well.
- ❖ No overfitting can be observed

Classification report for Logistic Regression (Train set)=

	precision	recall	f1-score	support
0	0.97	0.95	0.96	403
1	0.88	0.90	0.89	402
2	0.86	0.88	0.87	400
3	0.95	0.93	0.94	395
accuracy			0.92	1600
macro avg	0.92	0.92	0.92	1600
weighted avg	0.92	0.92	0.92	1600

Classification report for Logistic Regression (Test set)=

	precision	recall	f1-score	support
0	0.97	0.95	0.96	107
1	0.86	0.88	0.87	89
2	0.82	0.82	0.82	91
3	0.93	0.92	0.92	113
accuracy			0.90	400
macro avg	0.89	0.89	0.89	400
weighted avg	0.90	0.90	0.90	400

Decision Tree

- Hyperparameters are tuned with following values.

```
param_grid = {'max_depth': (5, 30), 'max_leaf_nodes': (10, 100)}
```

- The performance of the model is not at par.

```
Classification Report for Decision Tree (Train set)=
              precision    recall  f1-score   support

           0       0.95      0.97      0.96       395
           1       0.90      0.88      0.89       409
           2       0.88      0.86      0.87       408
           3       0.93      0.95      0.94       388

    accuracy                           0.92      1600
   macro avg       0.92      0.92      0.92      1600
weighted avg       0.91      0.92      0.91      1600
```

```
Classification Report for Decision Tree (Test set)=
              precision    recall  f1-score   support

           0       0.96      0.89      0.92       105
           1       0.75      0.86      0.80        91
           2       0.78      0.72      0.75        92
           3       0.89      0.91      0.90       112

    accuracy                           0.85       400
   macro avg       0.84      0.84      0.84       400
weighted avg       0.85      0.85      0.85       400
```

Random Forest

- Hyperparameters are tuned with following values.

```
param_grid = {'n_estimators': (10, 50),
              'max_depth': (3, 20),
              'min_samples_leaf': (10, 50)},
```

- The model seems to be overfitting the training set.

Classification Report for tuned Random Forest(Train set)=

	precision	recall	f1-score	support
0	0.98	0.98	0.98	395
1	0.92	0.94	0.93	409
2	0.92	0.92	0.92	408
3	0.98	0.96	0.97	388
accuracy			0.95	1600
macro avg	0.95	0.95	0.95	1600
weighted avg	0.95	0.95	0.95	1600

Classification Report for tuned Random Forest(Test set)=

	precision	recall	f1-score	support
0	0.93	0.94	0.94	105
1	0.83	0.82	0.83	91
2	0.77	0.83	0.80	92
3	0.93	0.88	0.90	112
accuracy			0.87	400
macro avg	0.87	0.87	0.87	400
weighted avg	0.87	0.87	0.87	400

XGBoost

- Model's hyperparameters are tuned to following values.

```
param_grid={'n_estimators': (10, 200),
           'learning_rate': [1, 0.5, 0.1, 0.01, 0.001],
           'max_depth': (5, 10),
           'gamma': [1.5, 1.8], 'subsample': [0.3, 0.5, 0.8]},
```

- The performance is better than other tree-based models but it is overfitting the training set.

Classification Report for tuned XGBoost(Train set)=

	precision	recall	f1-score	support
0	0.99	0.99	0.99	395
1	0.98	0.98	0.98	409
2	0.98	0.99	0.98	408
3	1.00	0.99	0.99	388
accuracy			0.99	1600
macro avg	0.99	0.99	0.99	1600
weighted avg	0.99	0.99	0.99	1600

Classification Report for tuned XGBoost(Test set)=

	precision	recall	f1-score	support
0	0.94	0.94	0.94	105
1	0.87	0.89	0.88	91
2	0.84	0.87	0.86	92
3	0.94	0.90	0.92	112
accuracy			0.90	400
macro avg	0.90	0.90	0.90	400
weighted avg	0.90	0.90	0.90	400

SVM

- ➢ Following hyperparameters are tuned.

```
param_grid = {'C':(1, 10),
              'kernel': ['rbf','linear', 'poly', 'sigmoid']}
```

- ➢ The model performance gives the best result among every other models and the model has successfully avoided overfitting.

Classification Report for tuned SVM (Train set) =

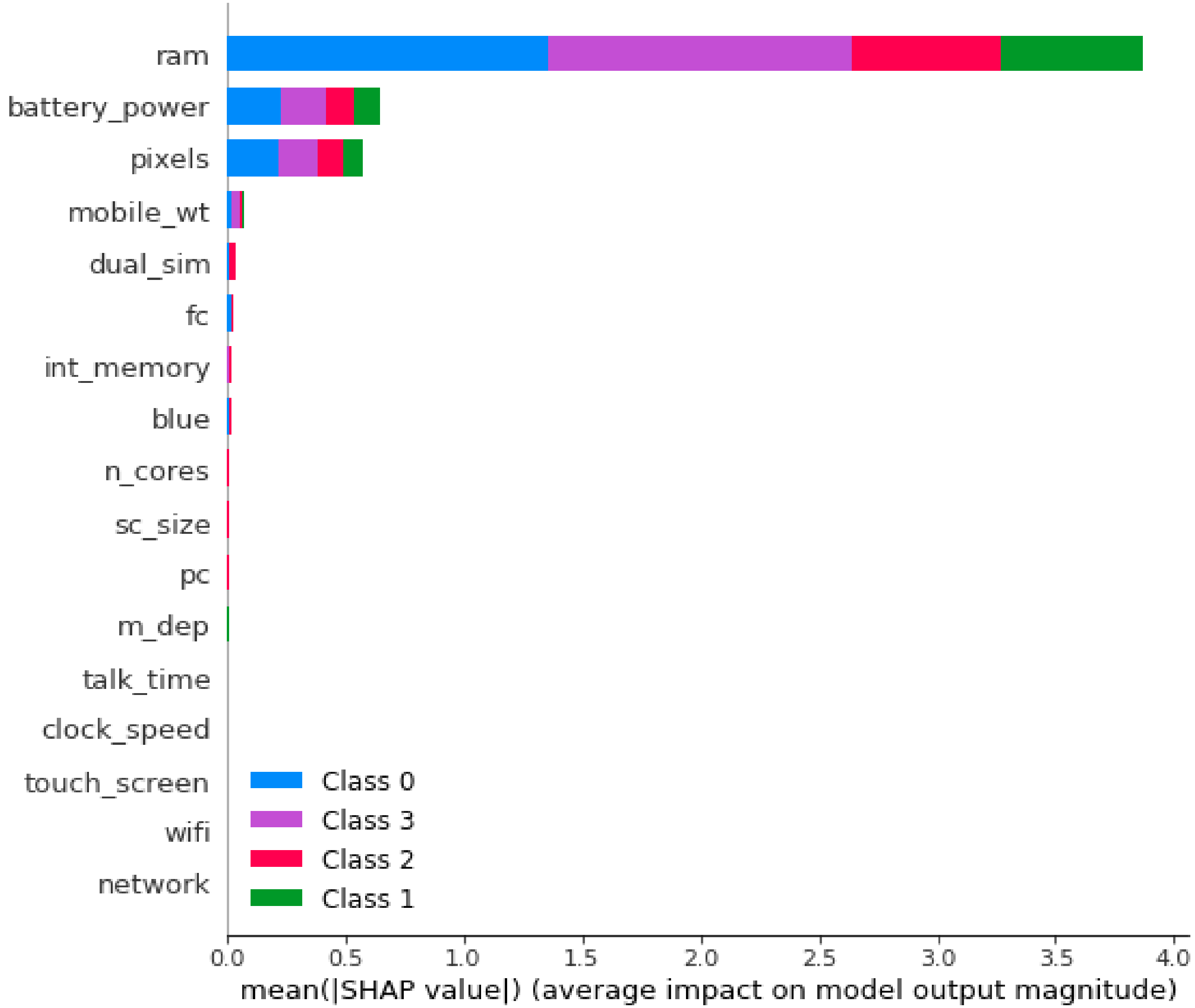
	precision	recall	f1-score	support
0	0.98	0.98	0.98	395
1	0.93	0.94	0.93	409
2	0.92	0.90	0.91	408
3	0.96	0.96	0.96	388
accuracy			0.94	1600
macro avg	0.94	0.94	0.94	1600
weighted avg	0.94	0.94	0.94	1600

Classification Report for tuned SVM (Test set) =

	precision	recall	f1-score	support
0	0.97	0.96	0.97	105
1	0.90	0.93	0.92	91
2	0.88	0.86	0.87	92
3	0.93	0.93	0.93	112
accuracy			0.92	400
macro avg	0.92	0.92	0.92	400
weighted avg	0.92	0.92	0.92	400

Model Explainability

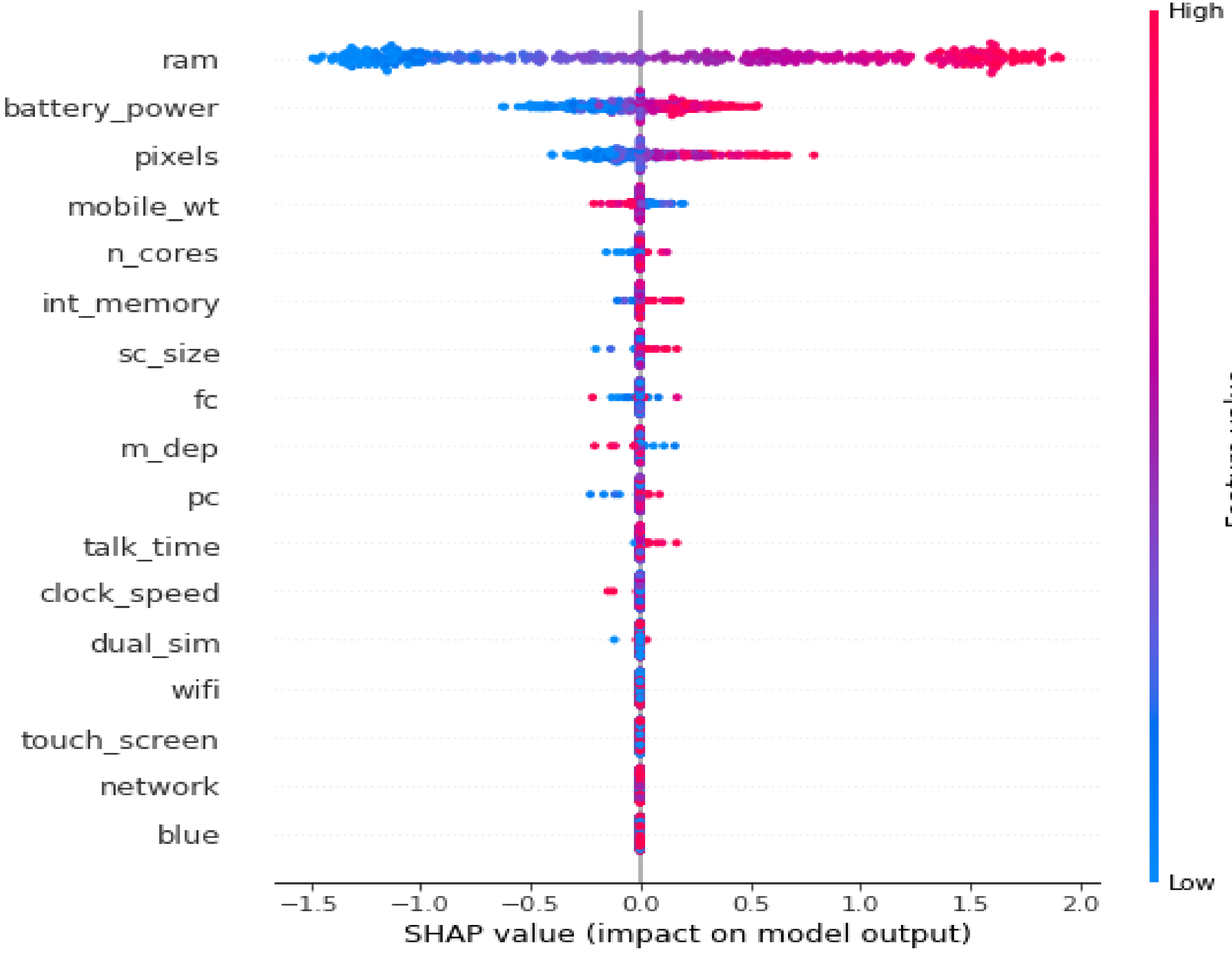
- SVM model is used to explain the features important in prediction of the price ranges.
- The following figure shows features (such as, RAM, battery, power) are more important than the rest of the features.
- Class 0 and 3 contributed more in model predictability than class 1 and 2.



Model Explainability (cont.)

❖ The red part shows how high an impact have on the positive or negative side of the feature whereas blue part shows how low the feature had an impact.

❖ RAM has a huge positive impact on model predictability followed by battery power and pixels.



Challenges

- To find the best sets of hyperparameter to train the model was most challenging part. Trying various combinations did took us few hours.

- SVM didn’t work on regular shap explainer.

- The shap instance for SVM model using KernelShap was taking ages to produce. Going over various websites and finally reducing the sample helped me.

Conclusion

- ✓ Features like RAM, battery power and pixels are most important in model predictability. This can also be seen while doing data visualization.

- ✓ Logistic Regression performs better than tree-based models because important features changes linearly with price changes and logistic regression has linear decision boundary, so it gives better results.

- ✓ XGBoost performs better than other tree-based models because while generating tree, the leaves are penalized which doesn't improve the model predictability.

- ✓ SVM performed much better than all other models because it has more accurate results and also able to generalize the features much better than others. The classification report also shows good prediction score for each class (0, 1, 2 and 3).