Mobile Price Range Prediction Using XGBClassifier

Tongzhao Liu

Professor: Andras Zsom

DATA 1030

5 December 2022

Github : https://github.com/tliu121/Final_project

INTRODUCTION:

As a new area of technology has arrived, mobile has become a basic need for everyone. So a lot of money is spent each year on purchasing mobile phones, but people need to be made aware of the features and their cost correlation. It is important to get the best value for their money. Therefore this project will benefit customers by helping them understand the relationship between features and price and help them make better predictions. The dataset for this project is downloaded from the Kaggle website, and it contains two parts, one is for training, and one is for testing. The testing dataset does not contain a target variable. The training dataset contain target variable "price_range" and other 20 features(fc, n_cores, pc, sc_h, sc_w, talk_time, blue, dual_sim, four_g, three_g, touch_screen, wifi, battery_power, clock_speed, int_memory, m_dep, mobile_wt, px_height, px_width, ram), each feature comprises 1000 data points. For my project, the goal is to predict the price range for mobile in the test dataset. The target value is a range, which is considered a classification problem.

Exploratory Data Analysis:

For the EDA part, check the missing value, datatype, data distribution, and correlation between features and target variables. After processing this part, I found there is no missing value and the data is perfect balance.

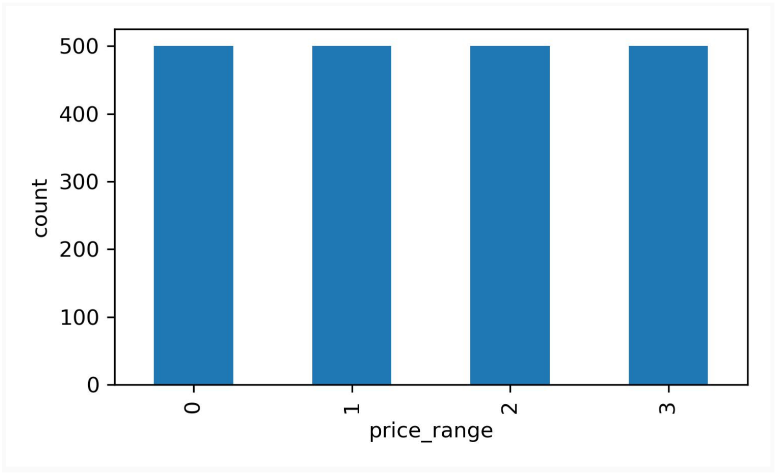


Figure 1 This bar plot shows that for every price range we have the same points, which indicates the dataset is perfectly balanced.

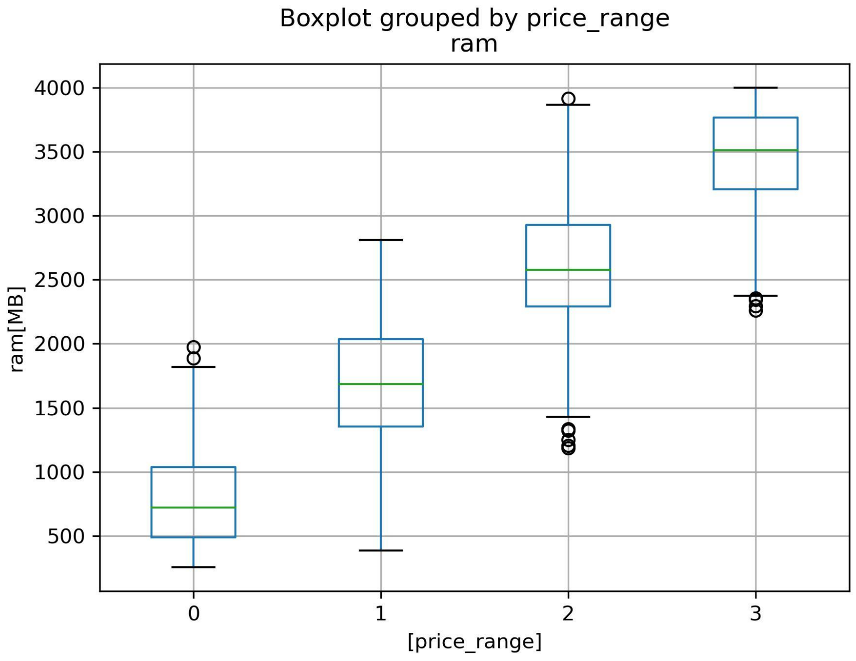


Figure 2 This boxplot shows the relation between ram and price_range. The figure shows that ram and price_range have a very strong positive relationship. A phone with more ram will have a higher price.

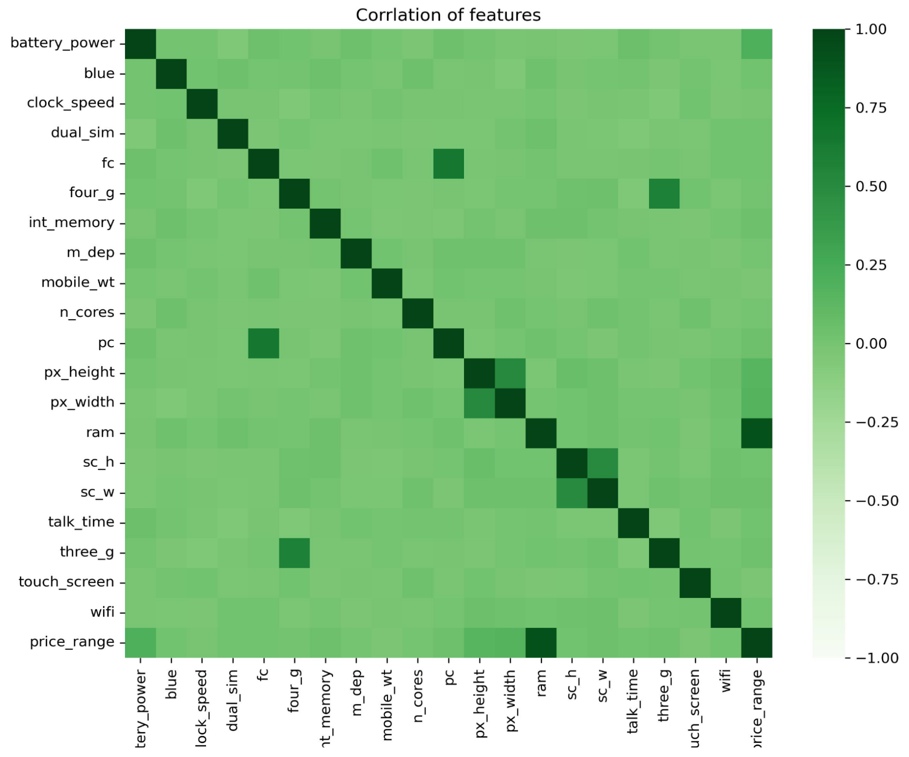


Figure 3 This heatmap shows the correlation between each feature to the target variable(price_range). Ram is most strongly positively correlated with price_range, where the ram is Random Access Memory. It improves the conclusion of figure2.

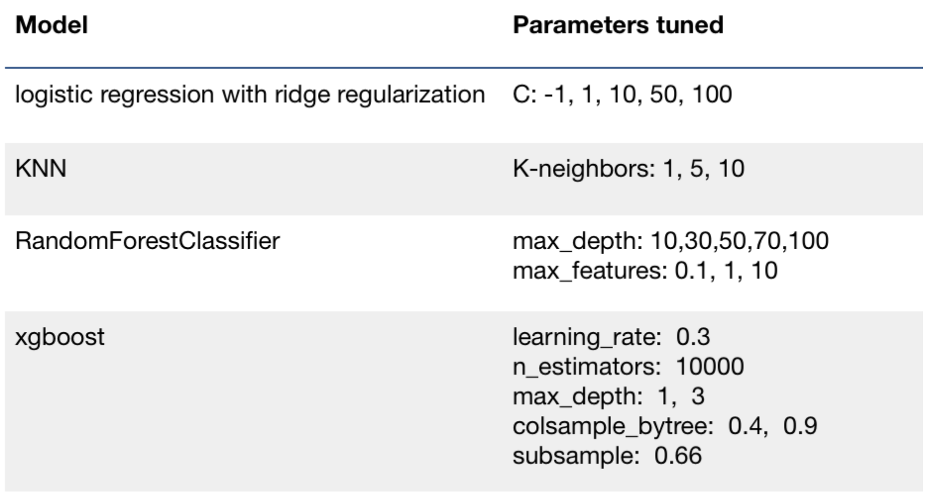
Method

- **Data splitting and preprocessing:**

After the EDA part, it shows the dataset was perfectly balanced and without a missing value. The dataset is iid since previous observation is independent from the next one.. In addition, the dataset does not have a group structure or time series data. For the preprocessing, first, categorize the various features into continuous, categrical, ordinal feature, apply OrdinalEncoder on ordinal features(fc, n_cores, pc, sc_h, sc_w, talk_time) and OnehotEncoder on categorical features(blue, dual_sim, four_g, three_g, touch_screen, wifi), lastly choose StandardScaler for the continuous features(battery_power, clock_speed, int_memory, m_dep, mobile_wt, px_height, px_width, ram). After preprocessing, there are 26 features in the preprocessed data. After preprocessing, I define an MLpipe_KFold_Accu function, including splitting strategy: 20% data in validation, 20% data in test, and rest 60% data for training purpose.

- **ML algorithms**

After the steps above, before adding algorithms to the pipeline, preparing the grid search first is necessary. It will find the best parameters of each model, and K-ford is used for cross-validation. When considering choosing which metric to perform the model, we know it is a classification problem, and with perfect balance data, accuracy would be a good choice for general cases. This project applies four different ML algorithms based on the classification problem: logistic regression with ridge regularization, K-nearest neighbor, RandomForestClassifier, and XGBoostClassfier. The below figure shows the parameters that were tuned.



Model	Parameters tuned
logistic regression with ridge regularization	C: -1, 1, 10, 50, 100
KNN	K-neighbors: 1, 5, 10
RandomForestClassifier	max_depth: 10,30,50,70,100 max_features: 0.1, 1, 10
xgboost	learning_rate: 0.3 n_estimators: 10000 max_depth: 1, 3 colsample_bytree: 0.4, 0.9 subsample: 0.66

Figure 5 List the model and parameter tuning.

Results

- **accuracy**

This dataset is balanced with 4 classes. Hence the baseline accuracy is 0.25. According to the XGB mean score of 0.89875 and standard derivation 0.009585, after calculating, we find the XGBoost model is 67.68 standard deviations above the baseline. The accuracy score below for each model can use to compare the most predictive one. According to the comparison, XGBoost have the highest accuracy score, it is the best model of these four models.

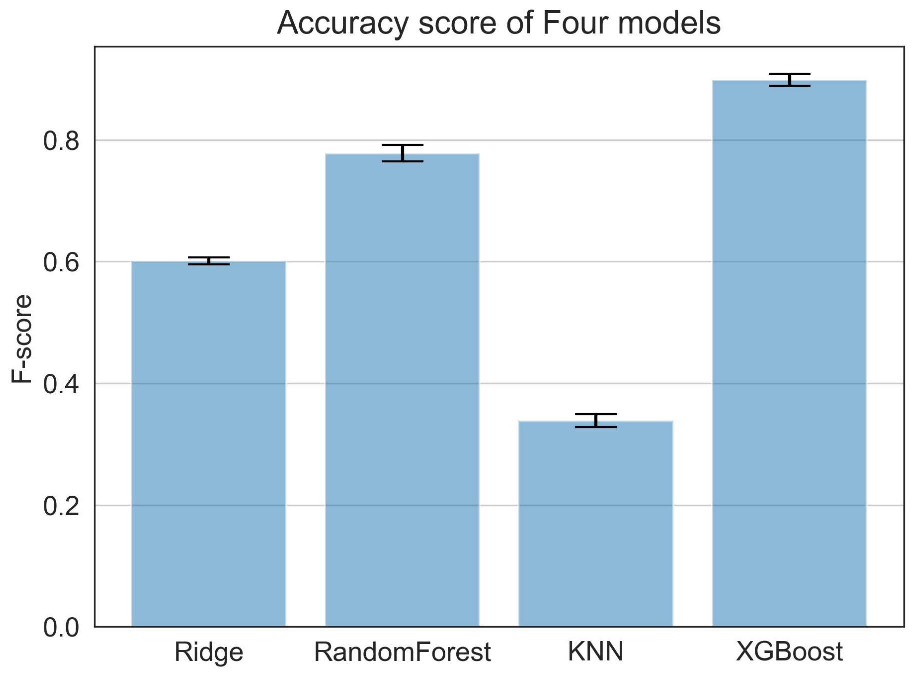


Figure 6 This figure shows that XGBoost is the best model for this dataset with highest accuracy score. And KNN is the worst score with the lowest accuracy score.

- **global feature importance**

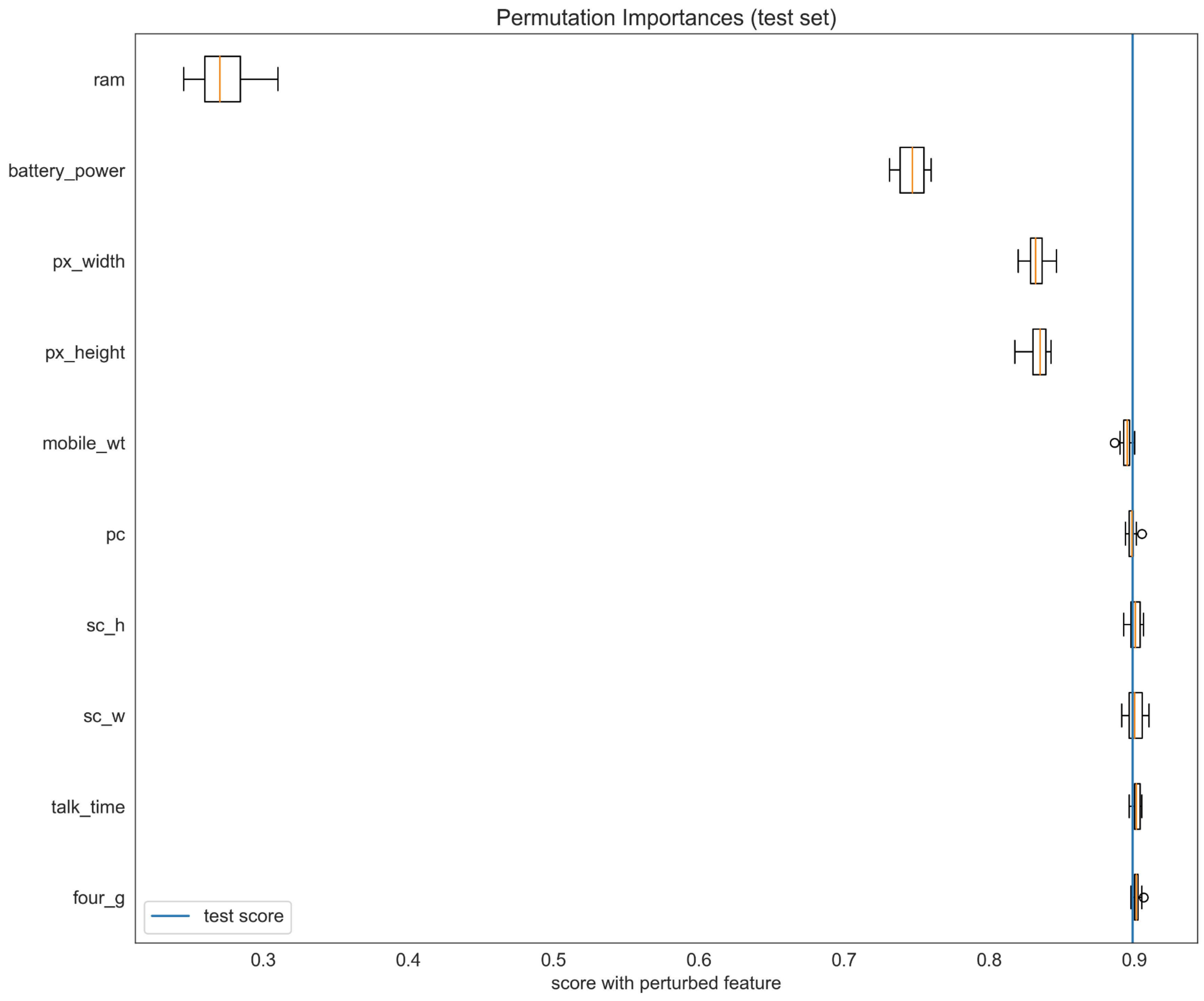


Figure 7 From this figure, we can see that the essential feature is the ram and then is battery _power, px_width, px_height, which improves our result in figure 3.

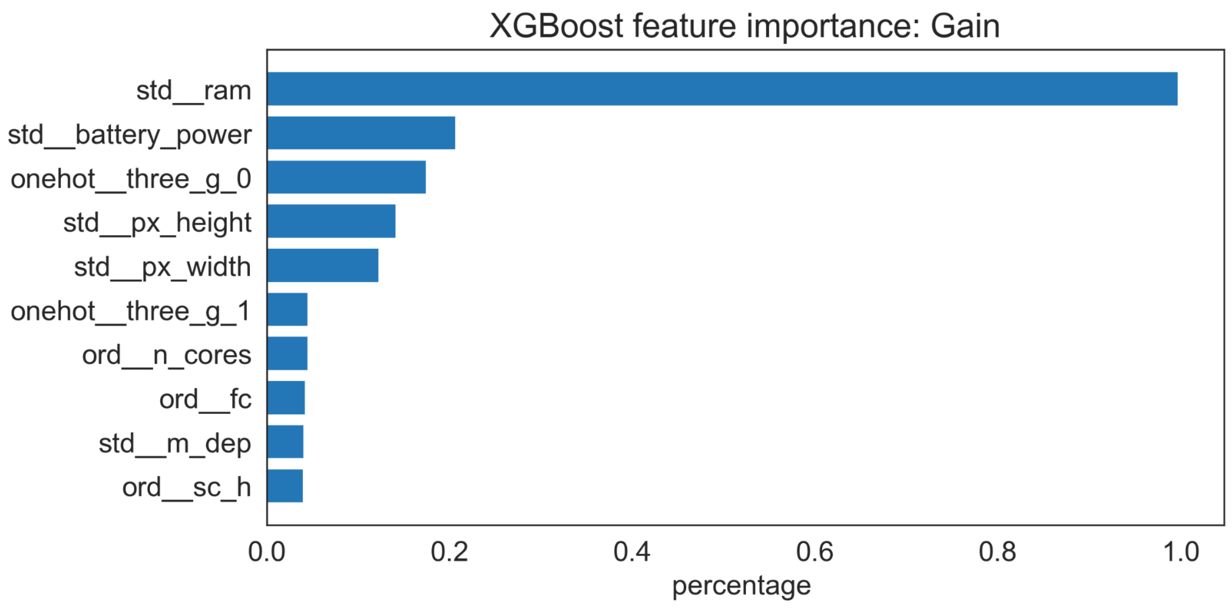


Figure 8 The Gain implies the relative contribution of the corresponding feature to the model calculated by taking each feature's contribution for each tree in the model. From this figure, we can observe that using gain to evaluate XGBoost feature importance, the most important three features are ram, battery_power, and three_g.

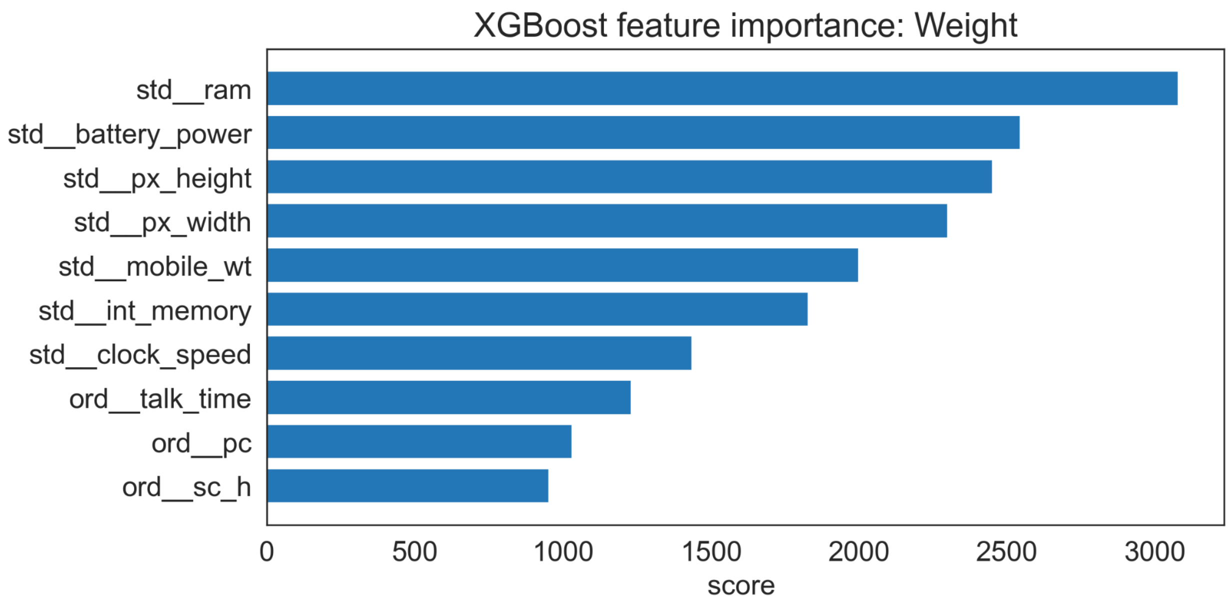


Figure 9 The weight shows the number of times the feature is used to split data. From this figure, we can observe that using weight to evaluate XGBoost feature importance, the most important three features are ram, battery_power, and px_height.

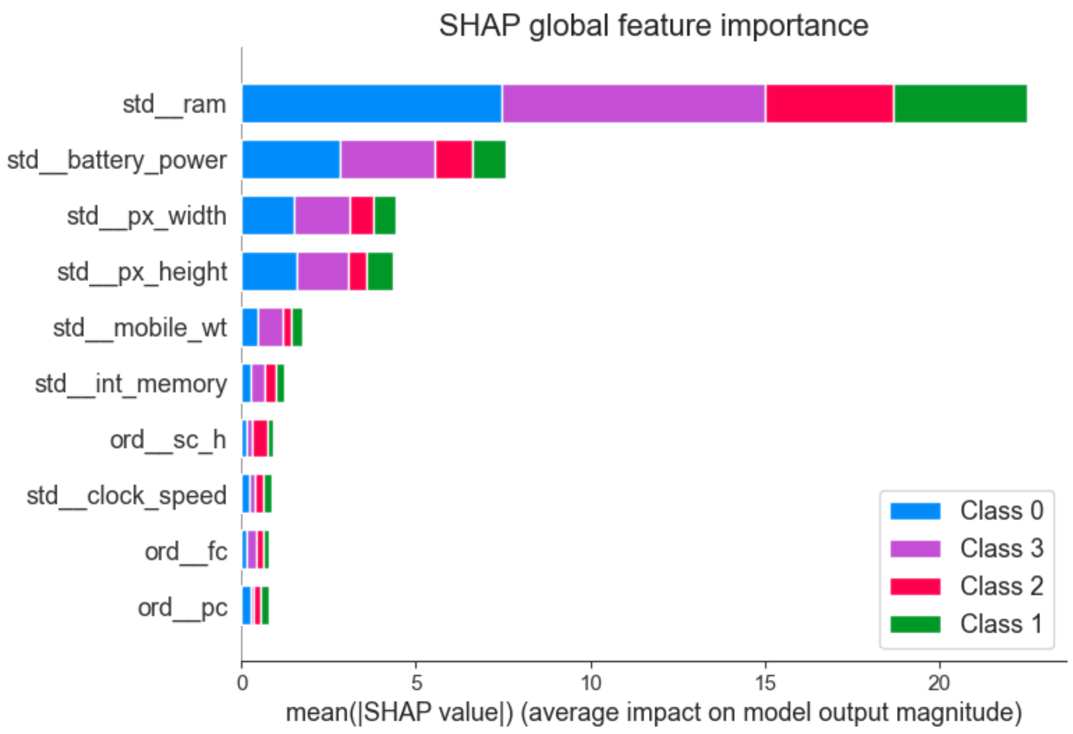


Figure 10 From the shap figure, the most important three features are ram, battery_power, and px_weight. It got the same result as the permutation feature importance method.

- **Local feature importance**

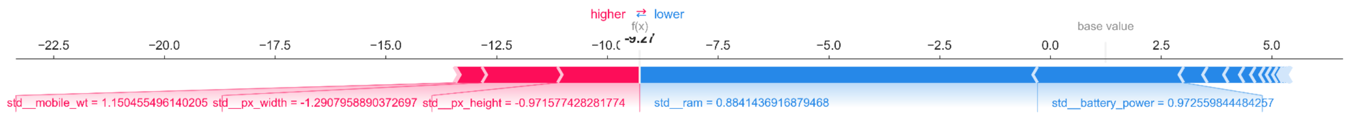


Figure 11 local feature importance of class 0

According to the global feature importance, ram is the most important feature for predicting price range. Which is improved by the correction heatmap.

For the local feature importance, the predicting output value is 242473.24, std_px_height, std_px_width...(feattures in red) are contribute positively, std_ram, std_battery_power...(features in blue) are negative contribute.

Outlook

Around 80% of features do not attribute well to the model, and their correlation with target variable(price_range) is not strong either, which is not expected. This will reduce the model's accuracy. One way to improve it is to make more feature selections and choose stronger relation features to build our model. Another way is to put more parameters in XGBoost, which can also improve the result accuracy; of course, trying as many algorithms as possible is also a helpful way to compare the best model. Like SVC, gradient-boosted tree and Naive Bayes. Also, since the data is collected five years ago, and the technology is developing very fast, so collecting data in recent years should be more helpful to predict scientific and technological products related questions.

REFERENCES:

[1] https://www.kaggle.com/code/engnoura2/phone-price-classification

[2] “Use of the different classifiers to predict Mobile price range.”, Available: https://www.kaggle.com/code/paulbacher/mobile-price-classification