Mobile Price Range Prediction

Ajith R

Abstract:

Different feature selection algorithms are used to identify and remove less important and redundant features and have minimum computational complexity. Different classifiers are used to achieve as higher accuracy as possible. Results are compared in terms of highest accuracy achieved and minimum features selected. Conclusion is made on the base of best feature selection algorithm and best classifier for the given dataset. This work can be used in any type of marketing and business to find optimal product (with minimum cost and maximum features). Future work is suggested to extend this research and find more sophisticated solution to the given problem and more accurate tool for price estimation.

Introduction

Mobile now a days is one of the most selling and purchasing device. Every day new mobiles with new version and more features are launched. Hundreds and thousands of mobiles are sold and purchased on daily basis. So here the mobile price_class prediction is a case study for the given type of problem i.e finding optimal product. The same work can be done to estimate real price of all products like cars, bikes, generators, motors, food items, medicine etc. Many features are very important to be considered to estimate price of mobile. For example, Processor of the mobile. Battery timing is also very important in today's busy schedule of human being. Size and thickness of the mobile are also important decision factors. Internal memory, Camera pixels, and video quality must be under consideration. Internet browsing is also one of the most important constraints in this technological era of 21st century. And so is the list of many features based upon those, mobile price is decided. So, we will use many of above-mentioned features to classify whether the mobile would be very economical, economical, expensive or very expensive.

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.

- **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
- **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).

Steps involved:

Data Pre-processing:

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. It involves the following:

- **Check Missing Data:** NaN values found in the dataset are changed to 0.
- **Create new variables:** New variables were created like PPI, screen_size, camera variables and binned categorical variables.
- **Encoding Categorical Data:** The categorical variables were encoded with binary digits 0 and 1.
- **Reducing data correlation:** This involves a correlation heatmap and based on that some features are removed in order to reduce collinearity to minimum.
- **Splitting dataset into training and test set:** The data is split into 80-20 format i.e. 80% for training and 20 % for testing purpose.

Exploratory Data Analysis:

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions

with the help of summary statistics and graphical representations. This includes univariate and multivariate variate analysis.

- **Univariate Analysis:** Univariate analysis is the simplest form of analysing data. Each variable is analyzed separately.
- **Multivariate Analysis:** Multivariate analysis is stated to be an analysis of any concurrent relation between the dependent variable and other independent variables.

Model Implementation:

The models implemented for this dataset are:

- Logistic Regression
- K Nearest Neighbours.
- Support Vector Machine
- Bagging Classifier
- XGB Classifier

Cross Validation and Hyperparameter Tuning:

The cross validation and hyperparameter tuning are done using Grid Search CV. This is to reduce overfitting of the models.

Algorithms:

Multinomial Logistic Regression:

Multinomial logistic regression is a simple extension of binary logistic regression that allows for more than two categories of the dependent or outcome variable. Like binary logistic regression, multinomial logistic regression uses maximum likelihood estimation to evaluate the probability of categorical membership.

Data Preparation for Logistic Regression:

- The Dependent variable should be either nominal or ordinal variable. Nominal variable is a variable that has two or more categories but it does not have any meaningful ordering in them. Ordinal variable are variables that also can have two or more categories but they can be ordered or ranked among themselves.
- Set of one or more independent variables can be continuous, ordinal or nominal. Continuous variables are numeric variables that can have infinite number of values within the specified range values. Ordinal variables should be treated as either continuous or nominal.
- The Observations and dependent variables must be mutually exclusive and exhaustive. Mutually exclusive means when there are two or more categories, no observation falls into more than one category of dependent variable. The categories are exhaustive means that every observation must fall into some category of dependent variable.
- No Multicollinearity between Independent variables.
- There should be no Outliers in the data points.

K Nearest Neighbors

K-Nearest Neighbors (kNN) is a supervised machine learning algorithm that can be used to solve both classification and regression tasks. kNN classifier determines the class of a data point by majority voting principle. If k is set to 5, the classes of 5 closest points are checked. Prediction is done according to the majority class.

The kNN working can be explained on the basis of the below algorithm:

- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbors is maximum.
- Step-6: Our model is ready.

How to select the value of K in the K-NN Algorithm?

- Below are some points to remember while selecting the value of K in the K-NN algorithm:
- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
- A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties.

Support Vector Machine

Support Vector Machines (SVMs in short) are machine learning algorithms that are used for classification and regression purposes. SVMs are one of the powerful machine learning algorithms for classification, regression and outlier detection purposes. An SVM classifier builds a model that assigns new data points to one of the given categories. Thus, it can be viewed as a non-probabilistic binary linear classifier.

SVMs can be used for linear classification purposes. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using the kernel trick. It enables us to implicitly map the inputs into high dimensional feature spaces.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}[y^{(i)}(-log(\hat{p}^{(i)})) + (1-y^{(i)})(-log(1-\hat{p}^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

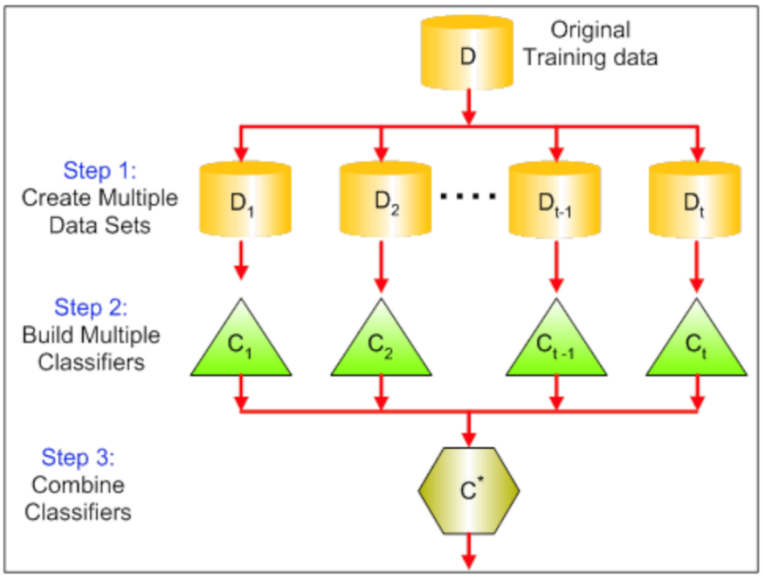
Bagging Classifier

Decision trees are prone to overfitting. ensemble learning is one way to tackle bias-variance trade-off. There are various ways to ensemble weak learners to come up with strong learners:

- Bagging
- Boosting
- Stacking

We are focusing on Bagging since it is about Bagging Classifier.

Bagging: Bagging is an ensemble technique used to reduce the variance of our predictions by combining the result of multiple classifiers modelled on different sub-samples of the same data set. The following figure will make it clearer:



The steps followed in bagging are:

1. **Create Multiple DataSets:** Sampling is done with replacement on the original data and new datasets are formed.
2. **Build Multiple Classifiers:** Classifiers are built on each data set. Generally, the same classifier is modeled on each data set and predictions are made.
3. **Combine Classifiers:** The predictions of all the classifiers are combined using a mean, median or mode value depending on the problem at hand. The combined values are generally more robust than a single model.

XGB Classifier: XGBoost is a popular implementation of gradient boosting.

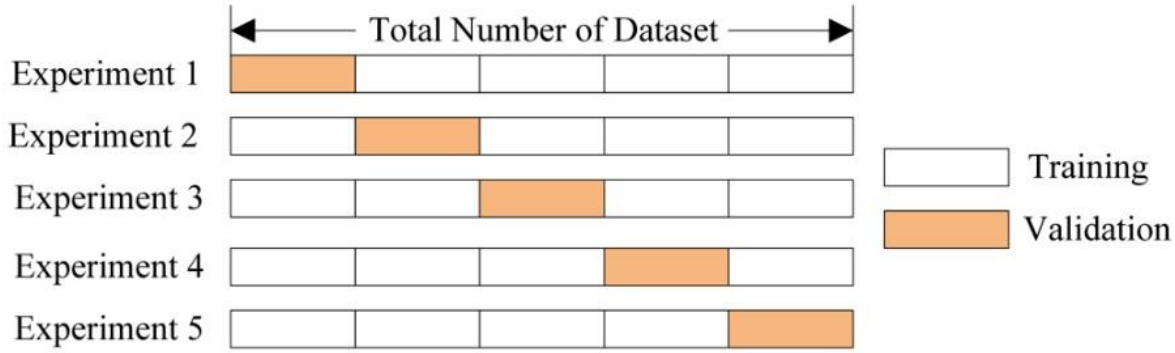
- **Regularization:** XGBoost has an option to penalize complex models through both L1 and L2 regularization. Regularization helps in preventing overfitting
- **Handling sparse data:** Missing values or data processing steps like one-hot encoding make data sparse. XGBoost incorporates a sparsity-aware split finding algorithm to handle different types of sparsity patterns in the data
- **Weighted quantile sketch:** Most existing tree-based algorithms can find the split points when the data points are of equal weights (using quantile sketch algorithm). However, they are not equipped to handle weighted data. XGBoost has a distributed weighted quantile sketch algorithm to effectively handle weighted data
- **Block structure for parallel learning:** For faster computing, XGBoost can make use of multiple cores on the CPU. This is possible because of a block structure in its system design. Data is sorted and stored in in-memory units called blocks. Unlike other algorithms, this enables the data layout to be reused by subsequent iterations, instead of

computing it again. This feature also serves useful for steps like split finding and column sub-sampling

- **Cache awareness:** In XGBoost, non-continuous memory access is required to get the gradient statistics by row index. Hence, XGBoost has been designed to make optimal use of hardware. This is done by allocating internal buffers in each thread, where the gradient statistics can be stored
- **Out-of-core computing:** This feature optimizes the available disk space and maximizes its usage when handling huge datasets that do not fit into memory

Cross Validation and Hyperparameter tuning

In cross-validation, we run our modelling process on different subsets of the data to get multiple measures of model quality. For example, we could have 5 folds or experiments. We divide the data into 5 pieces, each being 20% of the full dataset.



Hyperparameters are sets of information that are used to control the way of learning an algorithm. Each algorithm requires a specific hyperparameters grid that can be adjusted according to the business problem.

We have used GridSearchCV which is available in sci-kit learn. Grid Search combines a selection of hyperparameters and runs through all of them to evaluate the model's performance. Its advantage is that it is a simple technique that will go through all the programmed combinations. The biggest disadvantage is that it traverses a specific region of the parameter space and cannot understand which movement or which region of the space is important to optimize the model.

Model Performance

A classification model performance can be evaluated using the following metrics.

Confusion Matrix: Confusion Matrix is used to know the performance of a Machine learning classification. It is represented in a matrix form. Confusion Matrix gives a comparison between Actual and predicted values. The confusion matrix is a N x N matrix, where N is the number of classes or outputs.

True Positive (TP)

- The predicted value matches the actual value.
- The actual value was positive and the model predicted a positive value.

True Negative (TN)

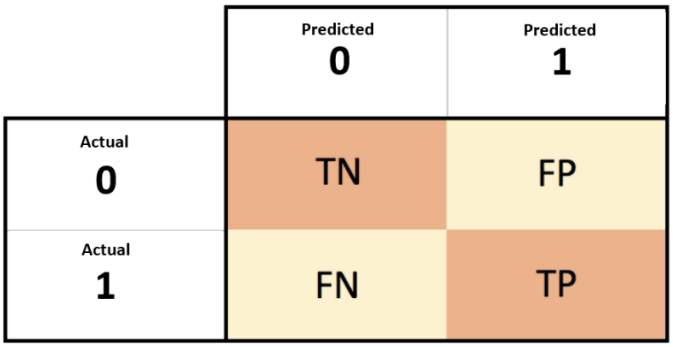
- The predicted value matches the actual value.
- The actual value was negative and the model predicted a negative value.

False Positive (FP) – Type 1 error

- The predicted value was falsely predicted.
- The actual value was negative but the model predicted a positive value.
- Also known as the **Type 1 error.**

False Negative (FN) – Type 2 error

- The predicted value was falsely predicted.
- The actual value was positive but the model predicted a negative value.
- Also known as the **Type 2 error.**



Accuracy: Accuracy will require two inputs (i) actual class labels (ii)predicted class labels. To get the class labels from probabilities (these probabilities will be probabilities of getting a HIT), you can take a threshold of 0.5. Any probability above 0.5 will be labelled as class 1 and anything less than 0.5 will be labelled as class 0.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: Precision for a label is defined as the number of true positives divided by the number of predicted positives. Report precision in percentages.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

Recall: Recall for a label is defined as the number of true positives divided by the total number of actual positives. Report recalls in percentages.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

F1-Score: This is defined as the harmonic mean of precision and recall.

$$F1 = 2.\frac{Precision \times Recall}{Precision + Recall}$$

AUC-ROC: The Receiver Operator Characteristic (ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise'. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve.

Model Analysis of Train and Test data using above metrics

Train Data:

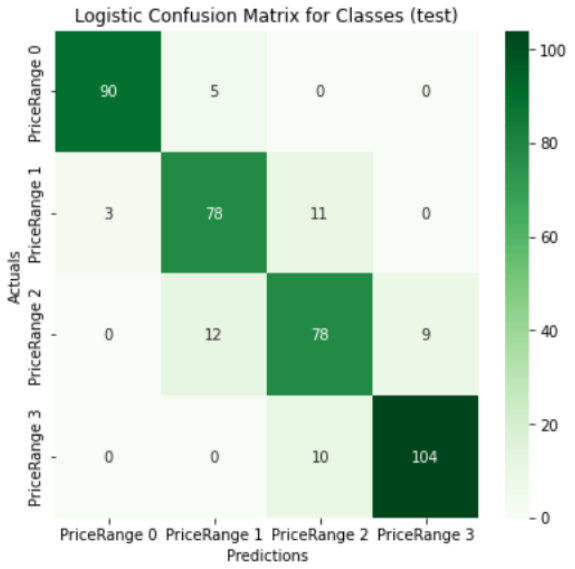
	Accuracy	Recall	Precision	f1_score	ROC_AUC
Logistic regression	0.900000	0.900215	0.899908	0.900049	0.989057
K Nearest Neighbours	0.856875	0.856531	0.858431	0.856785	0.979551
Support Vector Machine	0.905625	0.905724	0.905262	0.905434	0.988104
XGboost Classifier	0.841429	0.842442	0.840351	0.840928	0.969338
Bagging Classifier	0.852143	0.852922	0.852352	0.852561	0.974157

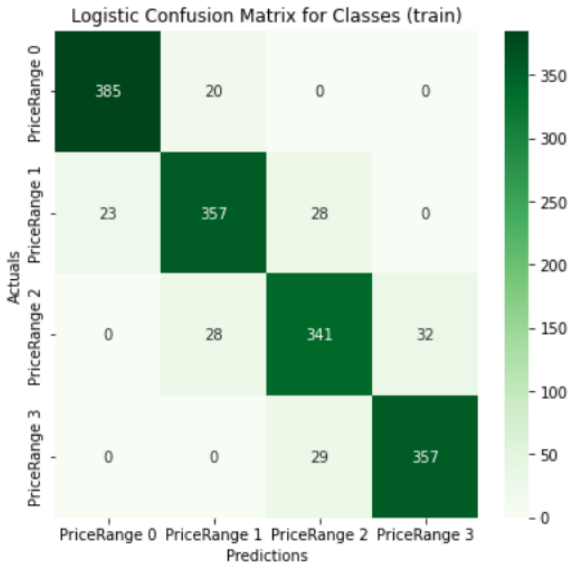
Test Data:

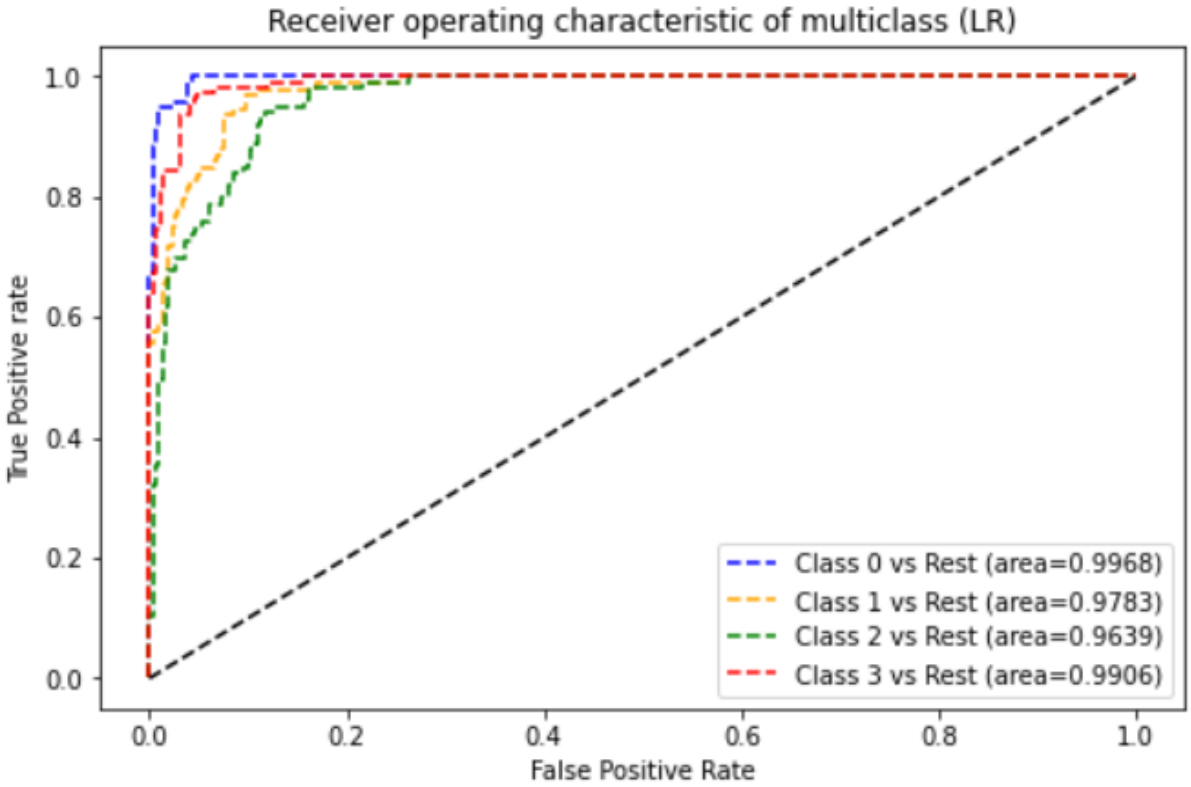
	Accuracy	Recall	Precision	f1_score	ROC_AUC
Logistic regression	0.875000	0.873838	0.874257	0.873962	0.982398
K Nearest Neighbours	0.835000	0.831843	0.830389	0.830085	0.966605
Support Vector Machine	0.867500	0.865824	0.866540	0.865877	0.979645
XGboost Classifier	0.821667	0.819229	0.816520	0.816794	0.963954
Bagging Classifier	0.853333	0.850077	0.849867	0.849398	0.970593

Logistic Regression is the best performing model having a very good accuracy, recall, precision, f1 score and ROC_AUC score.

Confusion Matrix and ROC AUC plots of Logistic Regression







Hyperparameter Tuning for Logistic Regression

It is performed using GridSearchCV. The following are tuned hyperparameters:

- C = 1
- max_iter= 140
- penalty = l2

Conclusion

- Logistic Regression is the best model.
- SVM also performed well, might have performed better with more computation time.
- Small dataset
- Missing features.

References

Almabetter, sklearn, GeeksforGeeks and Analytics Vidhya.