**Mobile Price Range Prediction**

**Sujit Musale**

**Data science trainees,**

**AlmaBetter, Bangalore**

**Abstract:**

We have provided with a data set about mobile. Having various label which describe the mobile feature. By using that data we have to develop a model in what range the respective mobile will have a price

Our experiment can help understand what could be the reason for the classification of such labels by feature selection, data analysis and prediction with machine learning algorithms taking into account previous trends to determine the correct classification.

***Keywords: machine learning ,Price range ,ram ,Battery power***

**1. 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 (eg:- 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 available

Data Description -

* 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).

**2. Introduction**

### The mobile data set is having feature like Battery power, blue tooth ,wifi and many more our target is to find out the exact range of the mobile phone based on this data. For this we check the data given data by doing EDA and treat outlier also used various model to and checked score for each .

Our aim to develop the model which is helpful to mobile company to set the price of mobile,

**6. Steps involved:**

* **Null values Treatment**

Our dataset contains a large number of null values which might tend to disturb our accuracy hence we dropped the null value for pixel height and set mean value for null value in sc\_w column in order to beginning of our project in order to get a better result.

* **Exploratory Data Analysis**

After loading the dataset we performed this method by comparing our target variable that is Price range with other independent variables. This process helped us figuring out various aspects and relationships among the target and the independent variables. It gave us a better idea of which feature behaves in which manner compared to the target variable.

* **Encoding of categorical columns**

We don’t use any encoding because all variable are in good form.

* **Feature Selection**

We convert the co-linear feature in to single feature like sc\_h and sc\_w to sc\_size and px\_height and px\_width to pixel.

* **Outlier treatment-**

### We removed outlier present in fc and px height columns to have normally distributed data. By removing upper quartile data.

* **Standardization of features**

Our main motive through this step was to scale our data into a uniform format that would allow us to utilize the data in a better way while performing fitting and applying different algorithms to it.

We use MinMaxScaler for Standardization.

The basic goal was to enforce a level of consistency or uniformity to certain practices or operations within the selected environment.

* **Fitting different models**

For modeling we tried various classification algorithms like:

1. **Decision Tree Classifier**
2. **Random Forest Classifier**
3. **Gradient Boosting classifier**
4. **XGboost Classifier**
5. **SVM Classifier**

* **Tuning the hyperparameters for better accuracy**

Tuning the hyperparameters of respective algorithms is necessary for getting better accuracy and to avoid overfitting in case of tree based models

like Random Forest Classifier and XGBoost classifier.

* **Important feature**

We have used the graph to shows the important feature extracted from respective model.

**3. Algorithms:**

1. **Decision Tree:**

It is a tool that has applications spanning several different areas. Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.

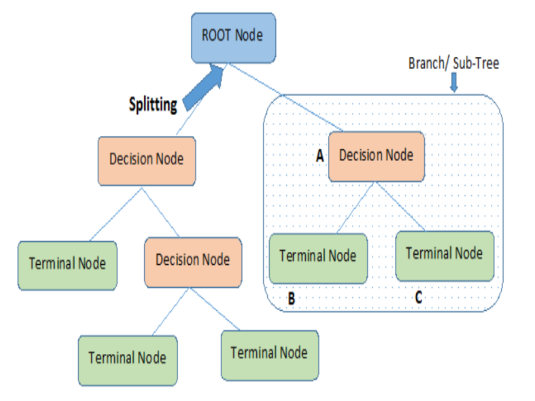
Before learning more about decision trees let’s get familiar with some of the terminologies.

Root Nodes – It is the node present at the beginning of a decision tree from this node the population starts dividing according to various features.

Decision Nodes – the nodes we get after splitting the root nodes are called Decision Node

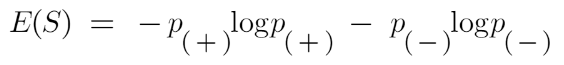
Leaf Nodes – the nodes where further splitting is not possible are called leaf nodes or terminal nodes

Sub-tree – just like a small portion of a graph is called sub-graph similarly a sub-section of this decision tree is called sub-tree.



We use following to terms in decision tree.

1.Entrophy-



2.Information gairn-

information gain Decision tree algorithm

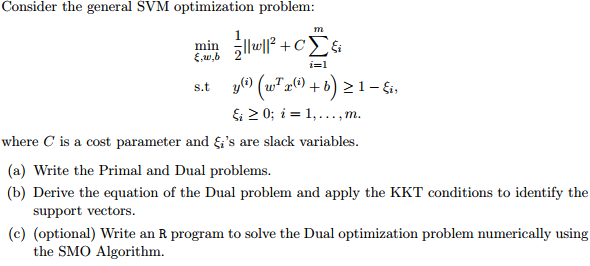
We get accuracy up to 0.87 and 0.84 for train and test data set respectively

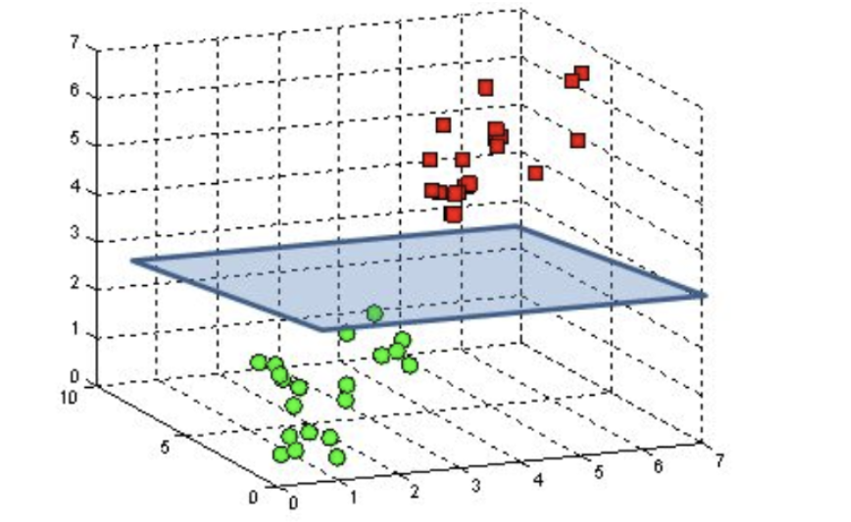
Which is nice for clasification.

1. **Support Vector Machine Classifier:**

SVM is used mostly when the data cannot be linearly separated by logistic regression and the data has noise. This can be done by separating the data with a hyperplane at a higher order dimension.

In SVM we use the optimization algorithm as:





We use hinge loss to deal with the noise when the data isn’t linearly separable.

Kernel functions can be used to map data to higher dimensions when there is inherent non linearity.

We got accuracy score 0.94 and 0.93 for train and test data set respectively

Which is higher than other model that we used to perform the classification.

1. **Random Forest Classifier:**

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.



We got accuracy score up to 0.79 and 0.74 for train and test data set respectively.

1. **XGBoost-**

To understand XGBoost we have to know gradient boosting beforehand.

* **Gradient Boosting-**

Gradient boosted trees consider the special case where the simple model is a decision tree

****

In this case, there are going to be 2 kinds of parameters P: the weights at each leaf, w, and the number of leaves T in each tree (so that in the above example, T=3 and w=[2, 0.1, -1]).

When building a decision tree, a challenge is to decide how to split a current leaf. For instance, in the above image, how could I add another layer to the (age > 15) leaf? A ‘greedy’ way to do this is to consider every possible split on the remaining features (so, gender and occupation), and calculate the new loss for each split; you could then pick the tree which most reduces your loss.

**XGBoost**

Is one of the fastest implementations of gradient boosting. trees. It does this by tackling one of the major inefficiencies of gradient boosted trees: considering the potential loss for all possible splits to create a new branch (especially if you consider the case where there are thousands of features, and therefore thousands of possible splits). XGBoost tackles this inefficiency by looking at the distribution of features across all data points in a leaf and using this information to reduce the search space of possible feature splits.

**4. Model performance:**

Model can be evaluated by various metrics such as:

**1. Confusion Matrix**-

The confusion matrix is a table that summarizes how successful the classification modelis at predicting examples belonging to various classes. One axis of the confusion matrix is the label that the model predicted, and the other axis is the actual label.

1. **Precision/Recall**-

Precision is the ratio of correct positive predictions to the overall number of positive predictions : TP/TP+FP

Recall is the ratio of correct positive predictions to the overall number of positive examples in the set: TP/FN+TP

1. **Accuracy**-

Accuracy is given by the number of correctly classified examples divided by the total number

of classified examples. In terms of the confusion matrix, it is given by: TP+TN/TP+TN+FP+FN

1. **Area under ROC Curve(AUC)**-

ROC curves use a combination of the true positive rate (the proportion of positive examples predicted correctly, defined exactly as recall) and false positive rate (the proportion of negative examples predicted incorrectly) to build up a summary picture of the classification performance.

**5. Hyper parameter tuning:**

Hyperparameters are sets of information that are used to control the way of learning an algorithm. Their definitions impact parameters of the models, seen as a way of learning, change from the new hyperparameters. This set of values affects performance, stability and interpretation of a model. Each algorithm requires a specific hyperparameters grid that can be adjusted according to the business problem. Hyperparameters alter the way a model learns to trigger this training algorithm after parameters to generate outputs.

We used Grid Search CV, Randomized Search CV and Bayesian Optimization for hyperparameter tuning. This also results in cross validation and in our case we divided the dataset into different folds. The best performance improvement among the three was by Bayesian Optimization.

1. **Grid Search CV-**Grid Search combines a selection of hyperparameters established by the scientist 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.
2. **Randomized Search CV-** In Random Search, the hyperparameters are chosen at random within a range of values that it can assume. The advantage of this method is that there is a greater chance of finding regions of the cost minimization space with more suitable hyperparameters, since the choice for each iteration is random. The disadvantage of this method is that the combination of hyperparameters is beyond the scientist’s control

**6. Conclusion:**

as we can see from all above model are having good accuracy score and AUC score too. But the top most model is SVM model giving accuracy 94.42 and 93.51 for train and test data respectively after hyper parameter tuning . in each and every model top feature is ram so this feature is most important feature.

**References-**

1. MachineLearningMastery
2. GeeksforGeeks
3. Analytics Vidhya