Mobile Price Range Prediction

Vivek Kumar Soni
Data science trainee,
almabetter

Abstract:

The main goal of this study is to see "whether a mobile phone with specified capabilities falls into a certain price range." This dataset is provided by Almabetter. To find and delete less significant and redundant features with the least amount of computational cost, different feature selection techniques are used. To attain the highest level of accuracy, many classifiers are used. The maximum accuracy achieved and the minimum features picked are compared. The best feature selection algorithm and best classifier for the given dataset are used to get a conclusion. This activity can be applied to any sort of marketing and business in order to discover the best product (with minimum cost and maximum features). To forecast the mobile price range's precision.

General Terms
Machine Learning

Keywords- Machine Learning, XG Boost, KNN

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.

Several aspects determine the cost of a mobile phone. The brand name, as well as specs such as internal memory, camera, ram, sizes, connectivity, and so on, are essential considerations in determining the pricing. From a commercial standpoint, it becomes critical to assess these elements on a regular basis and come up with the ideal set of specs and pricing ranges so that consumers purchase their mobile phones.

As a result, through this exercise and our forecasts, we will attempt to assist firms in estimating the price of mobiles in order to compete with other mobile manufacturers, as well as to assist customers in ensuring that they are paying the best possible price for mobile.

1.1 Data Set:

The Input Variables: These factors were classified as input variables. The output variable represents the predicted price range based on those inputs.

- 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).

1.2. The Output Variable

The output variable is the price range, and its domain is:

- 0: (low cost),
- 1: (medium cost),
- 2: (high cost), and
- 3: (very high cost).

2. Introduction:

Price is the most important part of marketing and company. The first query a buyer has is about the price of the items. "Will he be able to buy something with the specified specs or not?" all of the consumers question at first. Artificial Intelligence (AI) is a hot topic in engineering right now, with robots capable of intelligently answering queries. Classification, regression, supervised learning, unsupervised learning, and other artificial intelligence techniques are all available through machine learning.

MATLAB, Python, Cygwin, WEKA, and other technologies are available for machine learning applications. We can use a variety of classifiers, including Linear Regression, KNN, and others. There are a variety of feature selection algorithms that may be used to select only the best features and reduce the size of the dataset. The problem's computational complexity will be reduced as a result of this. Because this is an optimization problem, several optimization approaches are utilised to minimise the dataset's dimensionality. The mobile phone is becoming one of the most popular selling and purchasing devices. Every day, new mobile phones are released, each with a new version and more capabilities.

While evaluating the cost of a mobile phone, many aspects must be taken into account. Consider the mobile processor as an example. In today's demanding human activity, battery timing is crucial. The phone's size and thickness are also important factors to consider. Internal memory, camera pixels, and video quality are all factors that must be considered. Internet browsing is also one of the most major technological limitations of the twenty-first century. The list of several features on which mobile pricing is based is also extensive. As a result, we'll use a combination of the aforementioned characteristics to assess whether a phone's price is low, medium, high, or extremely high.

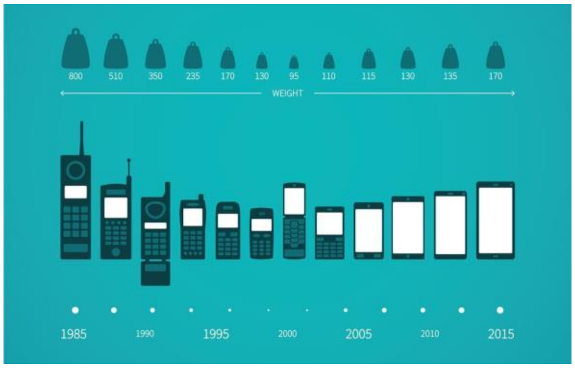
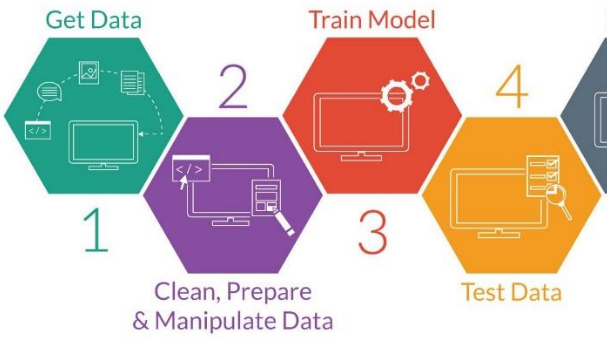


Fig. 1.1 Illustrations of mobile phones as per weight and year

3. Methodology

We must apply various models/algorithms to anticipate the cellphone price range. If we just use one model to make predictions, our results must be skewed. As a result, numerous models are required to solve this problem. We can't generate better forecasts if we only use one model.

The outcomes of each strategy described in this study must be compared to those of other classic machine learning algorithms to highlight the significance of the positives and disadvantages of each approach. The methods employed in this investigation will be detailed in this paper. Furthermore, there is no one-size-fits-all machine-learning technique that should be utilised in every circumstance (Wolpert & Macready, 1997). As a result, seven prediction algorithms were tested in this study to assess how well they performed.



4. Steps involved:

- **Exploratory Data Analysis**

  We used this strategy after importing the dataset by comparing our objective variable, Bike rented count, with other independent variables. This method assisted us in determining numerous characteristics and correlations between the target and independent variables. It helped us understand which features behave in which ways in relation to the goal variable.

- **Null values Treatment**

  We used the IsNull() function to see if our dataset contained any null values, but there were none. Having no null values in our dataset is advantageous to us. We either eliminate null numbers or replace them with mean, median, and mode, depending on the situation.

- **Outliers Treatment**

  Outliers are exceptional results that differ from other data observations; they can suggest measurement variability, experimental errors, or novelty. To put it another way, an outlier is an observation that deviates from a sample's main pattern. However, the outliers in our sample are only found in the variable front camera. We can overlook the few outliers in the front camera because they don't cause any problems.

- **Standardization of features**

When features of an input data set have considerable discrepancies between their ranges, or simply when they are measured in multiple measurement units, standardisation is required. Our main goal in this step was to scale our data into a standard format so that we could better utilise it while fitting and applying different algorithms to it. The major goal was to make certain that specific behaviours or processes in the specified environment were consistent.

- **Selecting Features**

  There are numerous approaches to choosing the greatest feature from a dataset. The SelectKBest method is used in this dataset to select 12 features based on their score. There are numerous features that are irrelevant or less important, and the model's accuracy has suffered as a result. As a result, the best feature selection strategy is employed to eliminate this problem.

- **<u>Fitting different models</u>**
  For modelling we tried various classification algorithms like:

1. **Decision Tree**
2. **Random Forest**
3. **Gradient Boosting**
4. **XGBOOST**
5. **K-Nearest Neighbors (KNN)**

- **<u>Tuning the hyperparameters for better accuracy</u>**
  Hyperparameter optimization or tuning in machine learning refers to the process of finding an acceptable set of hyperparameters for a learning algorithm.
  In the case of tree-based models, tuning the hyperparameters of respective algorithms is required to improve accuracy and minimise overfitting.
  Gradient Boost and XGBoost are two examples.

5. Algorithms

1. K-Nearest Neighbour(KNN)

The K-nearest neighbours (KNN) algorithm is a supervised machine learning technique that can address both classification and regression problems. KNN appears to be based on real-life experiences. The individuals surrounding them have an impact on people. The friends we grew up with shape our behaviour. In certain respects, our parents influence our personality. It's very likely that if you grow up among folks who enjoy sports, you'll enjoy sports as well. Exceptions do exist, of course. In a similar vein, KNN operates.

A data point's value is defined by the data points that surround it. If you have one close friend with whom you spend the most of your time, you will find that you share similar interests and like comparable activities. With k=1, this is KNN.

The majority voting principle is used by the KNN classifier to identify the class of a data item. When the value of k is set to 5, the classes of the five nearest points are examined. The majority class is used to make predictions. Similarly, in KNN regression, the mean value of the five closest points is used.

We seek out people who are similar, but how do we determine which data points are similar? The distance between the data points is determined. Distance can be calculated in a number of different ways. Euclidean distance (minkowski distance with p=2) is one of the most widely used distance metrics. The following diagram shows how to calculate the Euclidean distance between two points in a two-dimensional space. It is calculated using the square of the difference between x and y coordinates of the points.

How does KNN work?

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

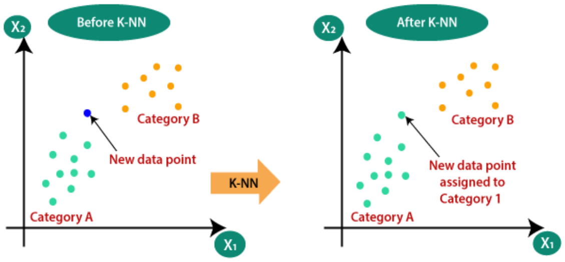
Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

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 neighbor is maximum.

Step-6: Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image



2. Decision Trees

Decision Tree is a supervised learning technique that may be used to solve both classification and regression problems, however it is most commonly employed to solve classification issues. Internal nodes represent dataset attributes, branches represent decision rules, and each leaf node provides the conclusion in this tree-structured classifier.

The procedure for determining the class of a given dataset in a decision tree starts at the root node of the tree. This algorithm checks the values of the root attribute with the values of the record (actual dataset) attribute and then follows the branch and jumps to the next node based on the comparison.

The algorithm compares the attribute value with the other sub-nodes and moves on to the next node. It repeats the process until it reaches the tree's leaf node. The following algorithm can help you understand the entire process:

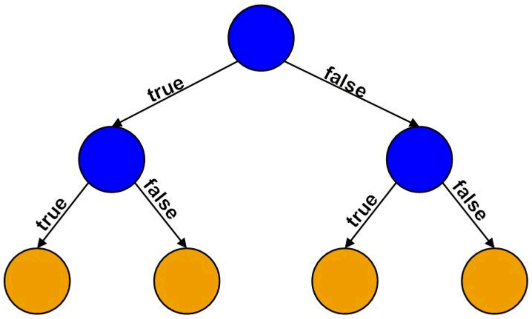
Step 1: Start with the root node, which holds the entire dataset, explains S.

Step 2: Using the Attribute Selection Measure, find the best attribute in the dataset (ASM).

Step 3: Subdivide the S into subsets that contain the best attribute's possible values.

Step 4: Create the node of the decision tree that has the best attribute.

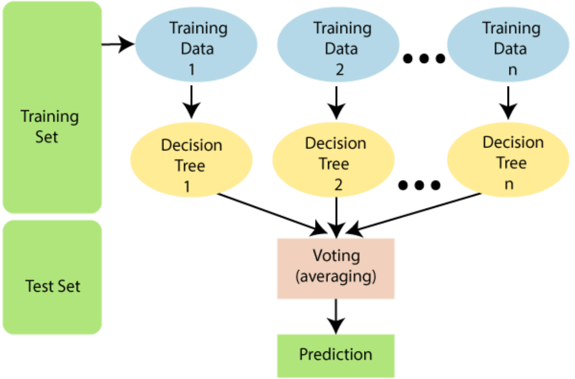
Step 5: Create additional decision trees in a recursive manner using the subsets of the dataset obtained in step 3. Continue this process until the nodes can no longer be classified, at which point the final node is referred to as a leaf node.



3. Random Forest

Random Forest is a well-known machine learning algorithm that uses the supervised learning method. In machine learning, it can be utilised for both classification and regression issues. It is based on ensemble learning, which is a method of integrating several classifiers to solve a complex problem and increase the model's performance. Random Forest is a classifier that contains a number of decision trees on various subsets of a given dataset and takes the average to enhance the predicted accuracy of that dataset, according to the name. Instead than relying on a single decision tree, the random forest collects the forecasts from each tree and predicts the final output based on the majority votes of predictions.

The bigger the number of trees in the forest, the more accurate it is and the problem of overfitting is avoided.



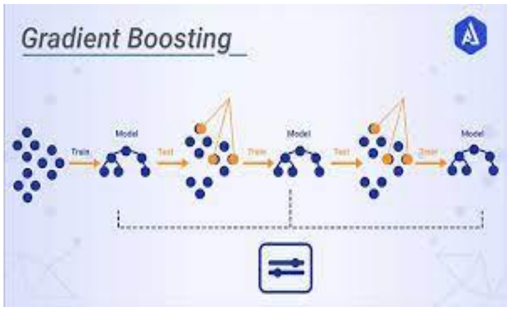
Because the random forest combines numerous trees to forecast the dataset's class, some decision trees may correctly predict the output while others may not. However, when all of the trees are combined, the proper result is predicted. As a result, two assumptions for a better Random forest classifier are that the dataset's feature variable should have some actual values so that the classifier can predict accurate results rather than guesses.
Each tree's predictions must have very low correlations.

3. Gradient Boost

Gradient boosting is a type of ensemble machine learning algorithm that can be applied to classification or regression predictive modelling issues.
Ensembles are built from decision tree models. Trees are added to the ensemble one at a time and fitted to correct the prediction errors caused by preceding models. This is a sort of ensemble machine learning model known as boosting.
One of the most powerful machine learning techniques is the gradient boosting algorithm. As we know, machine learning algorithm errors are widely categorized into two types: bias errors and variance errors. As one of the boosting techniques, gradient boosting is used to reduce model bias error.

The base estimator in the gradient boosting algorithm cannot be mentioned by us. The base estimator for the Gradient Boost algorithm is fixed and i.e. *Decision Stump*. Like, AdaBoost, we can tune the n_estimator of the gradient boosting algorithm. However, if we do not mention the value of n_estimator, the default value of n_estimator for this algorithm is 100.

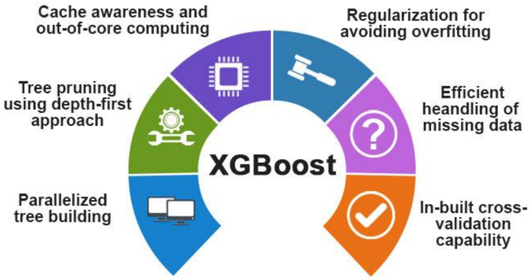
The gradient boosting approach can be used to forecast both continuous (as a Regressor) and categorical target variables (as a Classifier). When used as a regressor, the cost function is Mean Square Error (MSE), while when used as a classifier, the cost function is Log loss.



4. XGBOOST

XGBoost is an optimized distributed gradient boosting library designed to be highly **efficient**, **flexible** and *portable*. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way.

In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. Weight of variables predicted wrong by the tree is increased and these the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.


Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient heandling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

6. Evaluation Metrics:

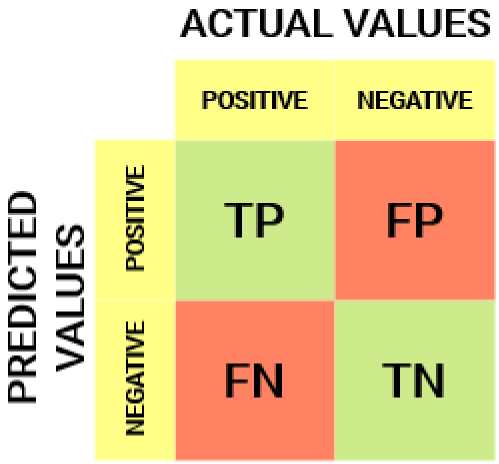
While data preparation and training a machine learning model are critical steps in the machine learning pipeline, evaluating the performance of this trained model is critically important. The ability of the model to generalize on previously unknown data is what distinguishes adaptive from non-adaptive machine learning models.

After implementing models and getting output, the next step is to find out the effectiveness of the model based on some metric using test datasets. Different performance metrics used to evaluate different models. For evaluating we will choose precision, recall and accuracy.

1. Confusion Matrix

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

For a binary classification problem, we would have a 2 x 2 matrix as shown below with 4 values:



		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	TP	FP
	NEGATIVE	FN	TN

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**

2. Accuracy

Accuracy is the ratio of correct prediction respective to all data. It is a faster way to evaluate a set of predictions in a classification problem.

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

3. Precision

Precision is the ratio of correct prediction respected to correct and incorrect prediction. It defines that the correct prediction it makes is actually correct or incorrect based on data. Suppose, algorithm identify a number of people who has cancer and actually how many of them has cancer, the ration between these terms can be said precision.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

4. Recall / Sensitivity

It means that a model exactly predicts correctly from true value.

Suppose, algorithm correctly predict a number of people who actually has cancer and actually how many of them has cancer, the ratio between two terms can be said recall.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

5. F1 Score

In practice, when we try to increase the precision of our model, the recall goes down, and vice-versa. The F1-score captures both the trends in a single value:

$$F1 - score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

F1-score is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. It is maximum when Precision is equal to Recall.

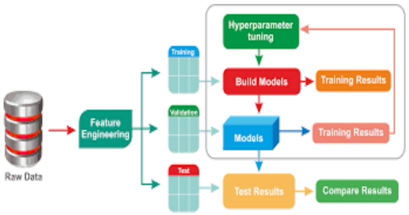
But there is a catch here. The interpretability of the F1-score is poor. This means that we don’t know what our classifier is maximizing – precision or recall? So, we use it in combination with other evaluation metrics which gives us a complete picture of the result.

7. Hyper parameter tuning:

The process of selecting the appropriate hyperparameters for a learning algorithm is known as hyperparameter tuning. A hyperparameter is a model argument whose value is determined prior to the start of the learning process. Hyperparameter tweaking is the cornerstone to machine learning algorithms.

We can adjust or change the frequency of the model parameters using hyperparameters, which are analogous to radio knobs.

We cannot adjust the model parameters directly; instead, we must change or set the hyperparameters. Hyperparameters are essentially algorithm properties.



- n_estimators = number of trees in the forest
- max_features = max number of features considered for splitting a node
- max_depth = max number of levels in each decision tree
- min_samples_split = min number of data points placed in a node before the node is split
- min_samples_leaf = min number of data points allowed in a leaf node
- bootstrap = method for sampling data points (with or without replacement)

Above are some examples of hyperparameters we use in Tree based algorithm. We can add cross validation as well in hyperparamters.

A) GridSearchCV:

The GridSearchCV technique evaluates a machine learning model for a variety of hyperparameter settings. GridSearchCV refers to this technique, which seeks the best set of hyperparameters from a grid of hyperparameter values.

B) RandomizedSearchCV:

Because it only runs through a predetermined number of hyperparameter adjustments, RandomizedSearchCV overcomes the shortcomings of GridSearchCV. It moves randomly throughout the grid in order to identify the optimal collection of hyperparameters. This method eliminates the need for extra computation.

8. Conclusion:

We tried a variety of models, and the table above summarises the results of one set of models.

- K-Nearest Neighbours has the best overall accuracy of 95 percent.
- The optimal accuracy for Random Forest, Decision Tree, Gradient Boosting, and XG boost was 84 percent, 90 percent, 90 percent, and 91 percent, respectively.

However, we'll make XG boost our best model because it provides good overall and individual class accuracy.

- With hyperparameter adjustment, overfitting in XG Boost is slightly reduced, although accuracy is lower than in KNN.

- According to this correlation matrix, the most essential attributes in terms of mobile pricing range forecasts are Ram, Battery Power, Pixel height, Pixel width, Mobile weight, Internal memory, Front Camera megapixels, Number of cores, Primary Camera megapixels, Screen height, Screen width and Talk time

References-

1. MachineLearningMastery
2. GeeksforGeeks
3. Analytics Vidhya
4. Towards DataScience
5. Data Science from scratch