Classifying Price Range of Mobile Phone Based on Specifications

Daniel Lee 1122326184, Haoyu Lu 111308297 **CSE351**

Introduction:

This paper will explore the methodologies that have been used to classify the price range of mobile phones given their specifications and determine which model has the best performance given the dataset using visualization and numerical indicators.

Methodologies:

We obtained the mobile phone data from a CSV file which included 2000 distinctive mobile phones and numerical price ranges from 0 to 4 where 0 represents the lowest and 4 represents the highest range. For each mobile phone, there are 20 features ranging from physical to technological specifications such as ram size and battery power.

We have analyzed the data and determined that there are errors in some of the values. After identifying the errors, we removed the rows that have corrupted columns. First, we identified that the dimensions of the screen, width and height, cannot be zero centimeters. Therefore, we marked these as errors and excluded the corrupted data. Furthermore, we used box plots to visually identify which fields have outliers. We found out that outliers existed in Front Camera megapixels (fc) and Pixel Resolution Height (px_height). We then used the interquartile range method to eliminate these data points.

After cleaning the given dataset, we have broken down the training and testing set into a 80-20 ratio to reduce overfitting and underfitting; the data was then randomly assigned to their set. Afterwards, we used 3 classification algorithms to predict the price range of mobile phones given their specifications which are: Support Vector Machine, Random Forest, and Naive Bayes.

Support Vector Machine (SVM) is a classification algorithm that finds the boundaries between data given the number of classes. For a data with two features and two classes, it can be plotted on a two-dimensional graph which only requires a line to split the data into 2 categories. The SVM algorithm finds the most optimal placement of the line between these two clusters by finding two points from different clusters that have the greatest margin from the line. The sum of greatest margin from both points to the same line is called maximum margin and these selected points are called support vectors since these are only points that contribute for the placement of the line or determine the boundary between the classes. The reason why the points are referred to as a vector is because these points can be expressed as vectors. Furthermore, when the number of features increases, vectors are able to express higher dimensional datasets just like our phone specification dataset. In higher dimensional dataset, the algorithm doesn't find an optimal placement of a line, but instead a hyperplane.

Furthermore, different types of kernels for SVM are used in specific situations in given datasets. For example, given a dataset where a line cannot be drawn to split the classes into two since one class is surrounded by the other, the solution is to go up a dimension in order for the boundary to be placed. This is where kernels come into play. Kernels modify the data up a dimension for a boundary to be placed. The most commonly used kernel is the gaussian radial

basis function (RBF) which is used when there is no prior knowledge about the data that is obtained.

The next classification algorithm that we used is called the Random Forest classification which uses ensemble learning. Ensemble learning combines multiple machine learning algorithms, specifically Decision Trees. First, the decision tree classification algorithm finds the boundary of classes by splitting the data into two where the majority of the classes are grouped based on the given features. For each split, the most significant feature (a feature that is able to identify differences of individual classes) will be used to segregate them. Lastly, the number of splits done by the algorithm is determined by the number of classes.

Back to Random Forest classification, the algorithm uses N decision trees which are randomly given K data points from the training set. Then each decision tree finds boundaries for the given number of classes. Each tree gives a classification of a certain boundary which is referred to as a vote. The forest will choose the class that has the most votes on that specific boundary. This is repeated for the boundaries that overlap with other decision trees. The increase in the number of trees allows for better performance because the sampling method adds randomness and diversity which is known as Bootstrap sampling.

Lastly we used Naive Bayes which is a classification algorithm that takes advantage of probability and Bayes' Theorem: $P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$. Naive Bayes classifier work by correlating features of class, and then using Bayes' Theorem to calculate a probability that the class the target object belongs to. Here is an example to understand how it works. To predict if a mobile phone is in the price range of 0 or 1, the classifier will first infer an initial probability for mobile phones in the range of 0 and the range of 1 based on their presence in the training sample taken from the training set.

Let the size of the training sample be 10000. Since 8900 phones out of the sample is in the price range 0, then the initial probability for mobile in 0 would be $P(0) = \frac{8900}{10000} = 0.89$, and the initial probability for mobile in price range 1 would be $P(1) = \frac{1100}{10000} = 0.11$. Then the classifier calculates the probabilities of seeing each feature of mobiles in the range of 1 as well as mobiles in the range of 0 (for the algorithm in our project, the probabilities is taken from the values on the y-axis of Gaussian[Normal] distributions for each feature). For example:

$$P(250 < ram < 500|0) = 0.5 \qquad P(250 < ram < 500|1) = 0.4$$
$$P(\ 500 < battery\_power < 1000|0) = 0.6 \qquad P(\ 500 < battery\_power < 1000|1) = 0.3$$
$$P(4G = 1|0) = 0.05 \qquad P(4G = 1|1) = 0.2$$

Now, to classify a mobile whose ram is 300, battery power is 750, and has 4G. The classier plugs the value into Bayes' Theorem, then gets the output:

$$P(price\ range = 0) * P(250 < ram < 500\ |0) * P(500 < battery\_power < 1000|0) * P(4G = 1|0) = (0.89)(0.5)(0.6)(0.05) = 0.01335$$

$$P(price\ range = 1) * P(250 < ram < 500\ |1) * P(500 < battery\_power < 1000|1) * P(4G = 1|1) = 0.11(0.4)(0.3)(0.2) = 0.00264$$

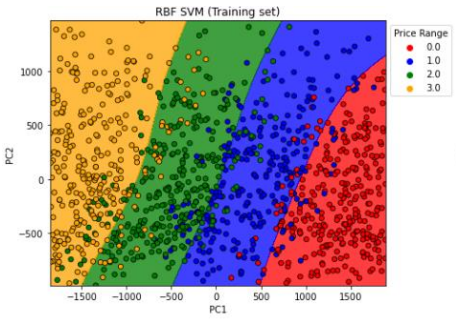
Since the "score" of mobile in the range of 0 is greater than the range of 1, the classier will classify the mobile in the range of 0.

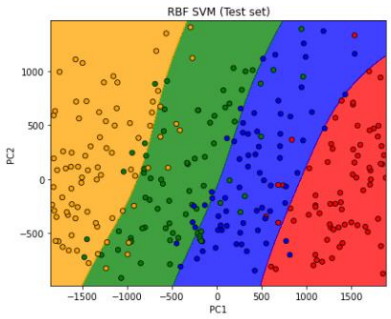
To visualize the performance of the multidimensional model, we used Principal Component Analysis (PCA) to reduce the dimensionality and plotted the data into a two dimensional graph.

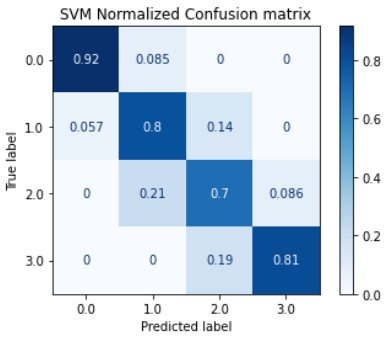
Observations:

After training all three models, we used visualization to show how the models created the boundaries using the training data and how accurate the boundaries are by using the testing data. Furthermore, a confusion matrix was created to support the conclusion we made from the visualizations. For each graph, the price ranges are color-coded for the points and the boundaries which are red, blue, green, and yellow from lowest to highest.

We first used the **SVM** classification algorithm with a gaussian radial basis function kernel.



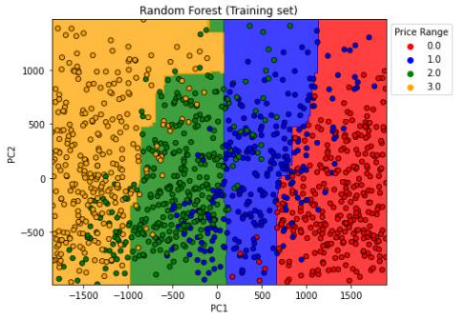


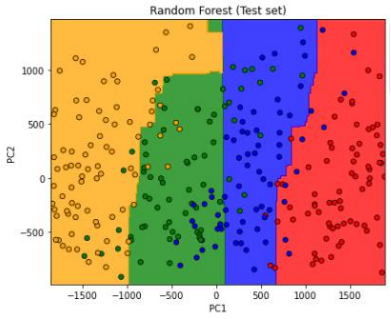


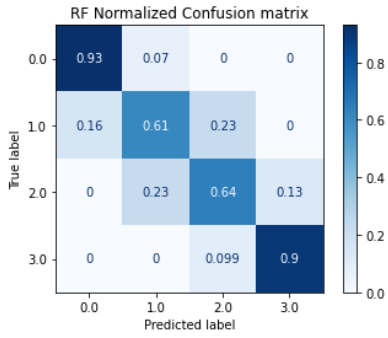
Observing the training set graph, the boundary SVM model has been created for four classes all having a nonlinear smooth curve. The boundary between two classes seems to be placed between the area where both data collides. Furthermore, it can be seen that the SVM model struggled to find the boundary between price ranges 1 and 2 as their data values may have many similar features.

The model's prediction seems to have captured the majority of the testing data as the most of the points are plotted on a boundary corresponding to their colors in the test set graph. However, few misclassifications can be seen as such as green points which are located in the blue boundary. According to the confusion matrix, most of the price ranges were predicted by the model, but struggled with predicting price range of 2, with only 70% predicted correctly. Most of the errors are in predicting the price range of 1 and 2 as 20% of price range 2 was misclassified as 1 (numerous blue points located in the green boundary as indicated in the above images). This model is able to predict the price range of 0 very well, with nearly 90% of the prices predicted correctly. In conclusion, the accuracy of the model is about 80% which performs significantly well.

Next we used the **Random forest** classification algorithm.

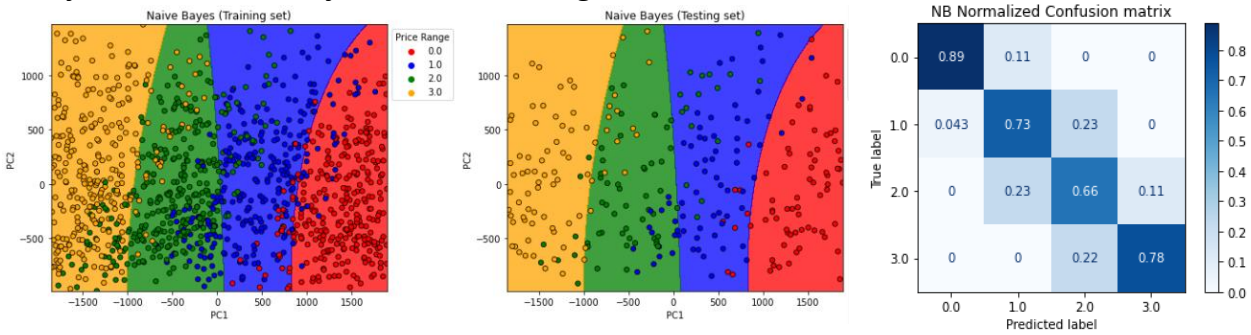






From the visualizations, one can identify that the boundaries being drawn by the Random Forest Trees (RFT) are constructed of many lines (jagged), which can be explained by the "splits" drawn by the decision tree algorithm. An interesting observation can be made about the boundary of price range 2. There are two separate areas that are classified as price range 2 where there is a tiny fragment of area in between yellow and blue. Similarly to SVM, Random Forest Tree algorithm struggles with classifying between price ranges 1 and 2. However, RFT has struggled more as most of the blue points have been misclassified as green shown in the training set graph, which may be explained by the similarity between these two classes; a straight line has been drawn as boundary of price range 1 and 2. Using the confusion matrix, the struggle of the model can be seen when only 61% of price range 1 and 64% of price range 2 were predicted correctly while SVM had 80% and 70% respectively. Surprisingly, the RFT model was able to predict price ranges 0 and 3 very well, up to 90% accuracy. However, because of lack of accuracy in the mid-range prices, the RFT model only has a total accuracy of 77%. Since Random Forest lacks confidence in the prediction for the mid-range prices, SVM is better.

Lastly we used **Naive Bayes** classification algorithm.



Similarly to SVM, the boundary between classes all have nonlinear smooth curves. Collisions of points with a different color show that the classifier struggles to find the boundary of price range (1~3), which is more visually intense than the previous two models. Compared to the previous two confusion matrices, GNB has a decent performance on the prediction to the price range of 1, which is 89%, slightly lower than the RF(93%) and SVM(92%). However, the classifier predicted poorly on the price range from 1~3. Only 78% of price range 3 were predicted, which has the lowest accuracy among the three.

Conclusion:

In conclusion, the SVM model with RBF kernel is the best classifier algorithm compared to the models that we explored with an accuracy of 80%. SVM had the least misclassification errors while the others had much more. All of the algorithms had a conundrum of placing an accurate boundary between price ranges 1 and 2 as their points were clustered together. Most of them did however perform well for the extremes price ranges 0 and 3. Our experiment concluded that SVM model performed the best.