ISTANBUL TECHNICAL UNIVERSITY

FACULTY OF MANAGEMNT

DEPARTMENT OF INDUSTRIAL ENGINEERING

DATA ANALYTICS FOR BUSINESS

PROJECT

Yakup Emre GÜLHAN	*070180147*
Salim KAPLAN	*070180708*
Batıkan SOYSAL	*070180114*

Contents

1- Exploration of the Data and Feature Explaination

2- First Model : Logistic Regression

3- Second Model : KNN

4- Third Model : Decision Trees

5- Fourth Model : Random Forest

6- Conclusions

1- Exploration of the Data and Feature Explanation

First thing we should have done was to find a good dataset. We have searched for a dataset from Kaggle and found one we liked, linked on: https://www.kaggle.com/iabhishekofficial/mobile-price-classification

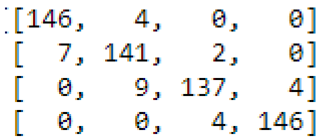
After finding the data, we explored it by using plots and the information given in Kaggle. The target and the features given is explained below.

1- ‘price_range’ is our target and defined as : This is the target variable with value of 0(low cost), 1(medium cost), 2(high cost) and 3(very high cost).
2- ‘battery_power’ : Total energy a battery can store in one time measured in mAh
3- ‘blue’ : Has bluetooth or not
4- ‘clock_speed’ : Speed at which microprocessor executes instructions
5- ‘dual_sim’ : Has dual sim support or not
6- ‘fc’ : Front Camera mega pixels
7- ‘four_g’ : Has 4G or not
8- ‘int_memory’ : Internal Memory in Gigabytes
9- ‘m_dep’ : Mobile Depth in cm
10- ‘mobile_wt’ : Weight of mobile phone
11- ‘n_cores’ : Number of cores of processor
12- ‘pc’ : Primary Camera mega pixels
13- ‘px_height’ : Pixel Resolution Height
14- ‘px_width’ : Pixel Resolution Width
15- ‘ram’ : Random Access Memory in Megabytes
16- ‘sc_h’ : Screen Height of mobile in cm
17- ‘sc_w’ : Screen Width of mobile in cm
18- ‘talk_time’ : Longest time that a single battery charge will last when you are
19- ‘three_g’ : Has 3G or not
20- ‘touch_screen’ : Has touch screen or not
21- ‘wifi’ : Has wifi or not

By looking at all the information given on columns. We can say that they are all important on finding the price range of a phone. It is not required to do any data cleaning since there are no Na or outlier values. Also, the dataset is well prepared so we did not have to add any dummy variables for qualitative features. After looking at distribution plots, target is distributed equally and most qualitative features are distributed half to half on 1 and 0 too. More explanation feature by feature and distribution plots are given in the python code.

2- First Model : Logistic Regression Model

First algorithm and model we will be using is Logistic Regression. It is quite easy to use since it is like a classification version of Linear Regression. Firstly, we should split our data in train and test sets to overcome some bias and overfitting. Secondly we imported the LogisticRegression from sklearn.linear_model. Then predicted the y_test values by fitting the model with our train datasets. We have reached an error where it says total number of f and g evaluations exceeds limit. It is recommended to do scaling or increase the number of iterations. After increasing the max_iter value to one hundred thousand, we still got this error so we decided to do scaling. We used preprocessing from sklearn, and used StandardScaler. After fitting and testing the model, we had a very good test accuracy score of 0.945. Then we did cross validation to see if this was a lucky split, and got a mean cross validation score of 0.96, which is even better. After that we decided to do Feature Selection so we used SFS from mlxtend.feature_selection. We used forward feature selection and came up with a subgroup of features that had an average score of 0.971, higher than our cross validation score we had on our scaled logistic regression model. After using this subset of features to fit a new logistic regression model, we had a test accuracy score of 0.95. Which is better than the base model but lower than the cross validation score we expected. After doing cross validation, we realized that we were making a mistake somewhere but couldn't find what was the problem. The confusion matrix of the model with FFS is given below.



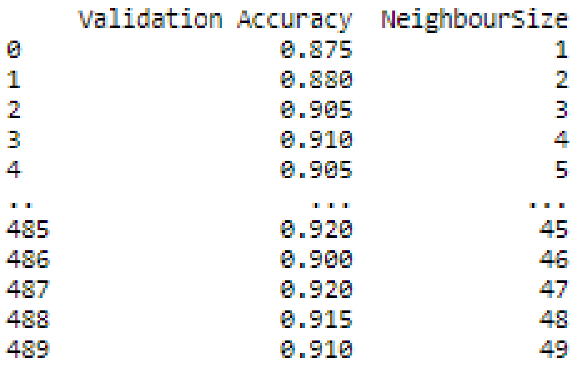
```
[146,   4,   0,   0]
[  7, 141,   2,   0]
[  0,   9, 137,   4]
[  0,   0,   4, 146]
```

Figure 1.

3- Second Model : K-Nearest-Neighbours

The second model we applied using the mobile price dataset is K-Nearest-Neighbours. After seperating traning and test set, an KNN classifier with the KNN value of 3 is set as a parameter using KneighboursClassifier from sklearn.neighbors library. As we fit the model using X_train and y_train, the accuracies that are yielded from predicting the y test values resulted in 0.91. In order to prevent the problems that may arise from selecting random training and test set, we proceeded by implementing 10 fold cross validation in our model. Besides, instead of just looking couple of KNN values manually, we made an iteration to examine different value scores and showed in a line graph..

The changes in a Dataframe



```
     Validation Accuracy  NeighbourSize
0                  0.875              1
1                  0.880              2
2                  0.905              3
3                  0.910              4
4                  0.905              5
..                   ...            ...
485                0.920             45
486                0.900             46
487                0.920             47
488                0.915             48
489                0.910             49
```

Figure 2.

The changes in accuracy with different KNN values

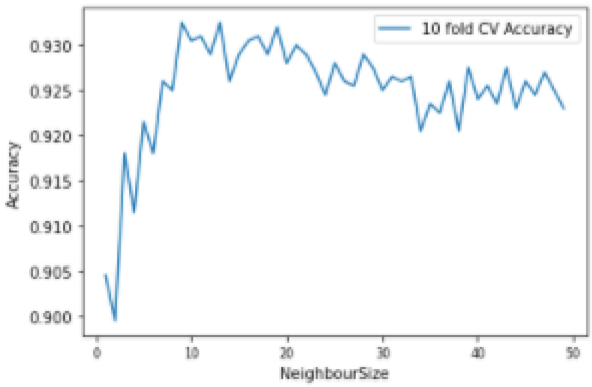


Figure 3.

It is seen that implementing 10 fold cv with neighbour size 9 yielded the best result which is 0.9325 as cross validation accuracy for our model.

4- Third Model : Decision Trees

Decision trees are known as being less accurate but more interpretable compared to the other methods that can be applied to a classification problem. It may be the case that making a compromise from the accuracy of the model worths in some situations. Therefore, we continue to try different techniques to our mobile price dataset with decisiontree library from sklearn. As always, after splitting our dataset in a training and a test set, we fit a classifier and set nothing to the model as a parameter at the beginning so as to see a fully grown tree.

A fully grown decision tree

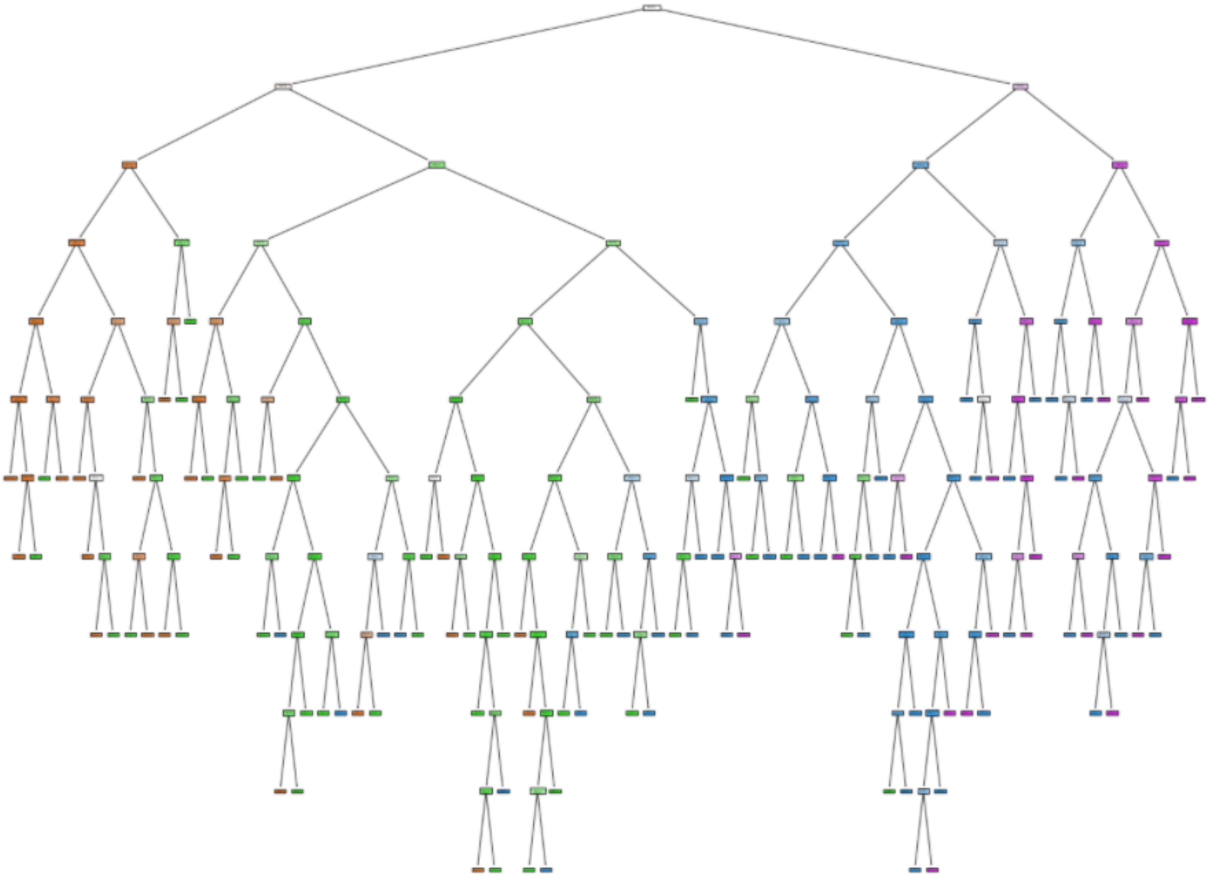


Figure 4.

It is obvious that this tree has no usage, in other words, we do not get any insight from that. Adding "ccp_alpha" value 0.01 as an argument gives a penalty for each node added to the tree. That helps us prune the tree in order to have both interpretable and accurate model.

The decision tree with ccp alpha value 0.01

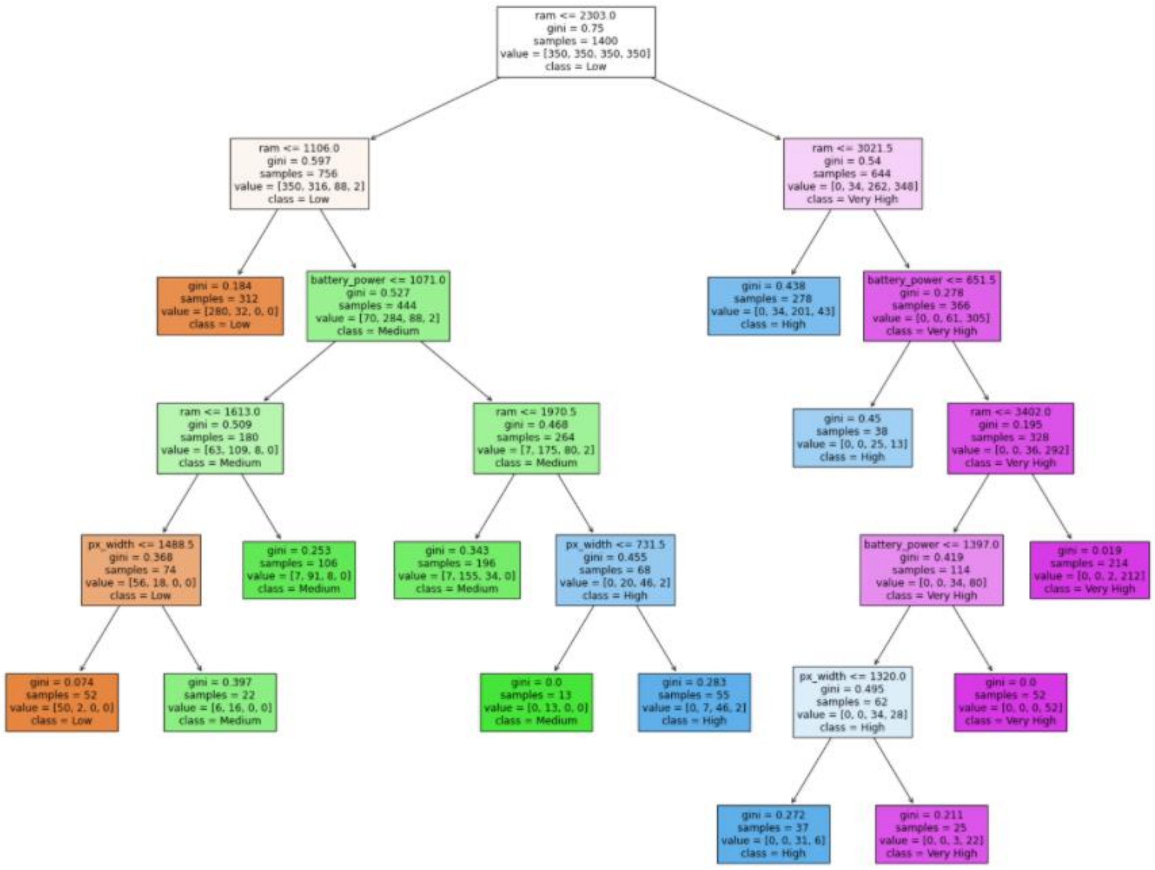


Figure 5.

That is pretty much changed the way we observe the decision tree. It is now better to have an idea what features has an impact on detecting the response variable which is the price range in our case. The decision tree gave an accuracy of 0.7833 after comparing training the model on the y test set. However, instead of assigning the penalty for pruning the tree, we need a more systematic way to find the optimal accuracy result. That is why we again made an iteration including different ccp alpha values as penalties for pruning the decision tree. We also applied 10 fold cross validation to prevent randomness in the dataset. As a result of this, our model came across with an accuracy value of 0.8233 , with the alpha value 0.002381. But still, this alpha value did not prune our tree enough for us to have a good interpretation.

The pruned decision tree

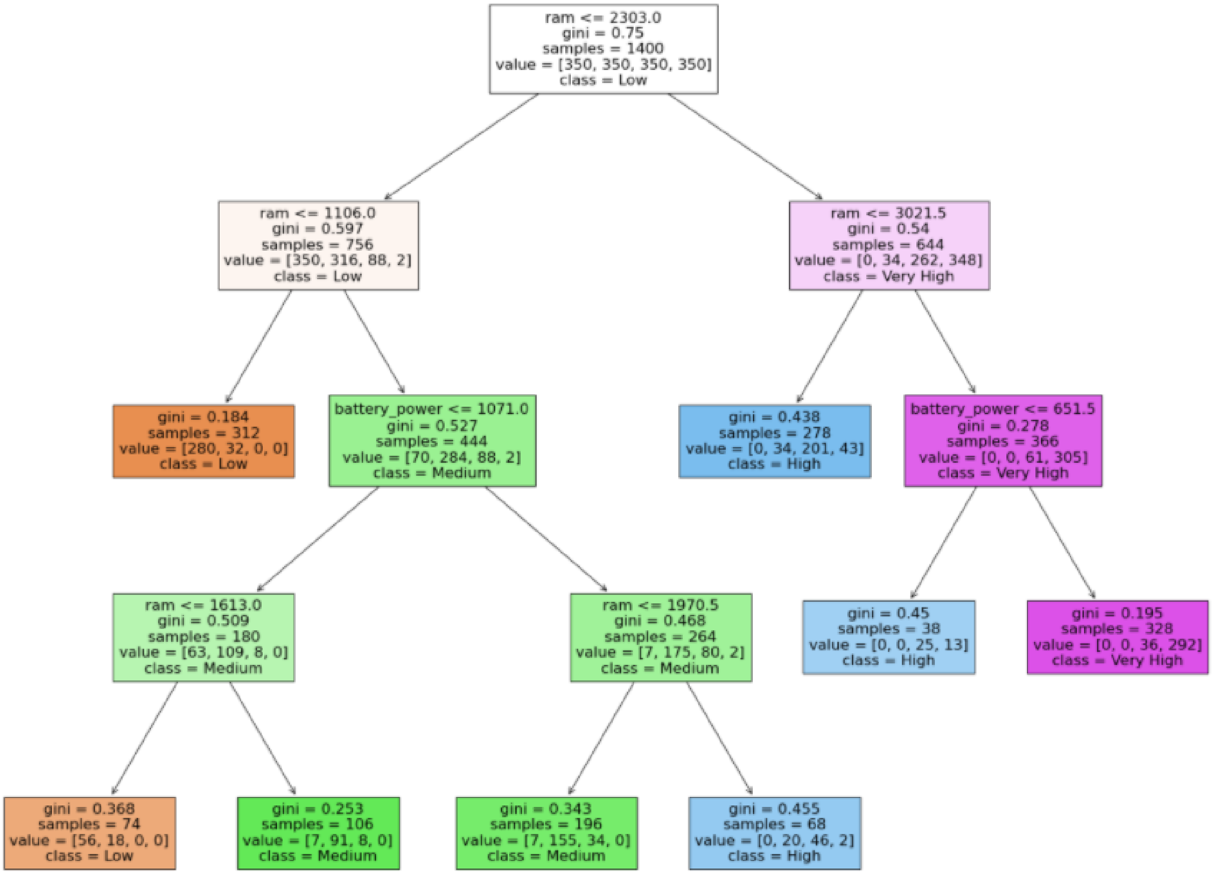


Figure 6.

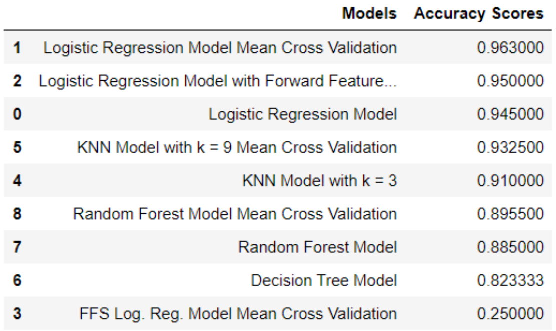
At the end, the decision tree above with ccp_alpha value of 0.01185, is yielded as the most reliable model for predicting the price range of mobile phones. Its accuracy score for the test data is calculated as 0.78. It is readable, interpretable and most importantly gave a robust accuracy rate, even if the other models(logistic regression, KNN) have better accuracy results, for a decision tree model.

5- Fourth Model : Random Forest Model

After the decision tree models, we are going to use random forest to determine a better tree. We first used bagging to find a test accuracy of 0.876, then used a random forest to find the same test accuracy. After that we used different number of trees and plotted the OOB Accuracy with the different numbers of trees. Then we used GridSearch to find 'max_features': 8, 'n_estimators': 100 to fit our model. After fitting, we found a test accuracy of 0.885. Later, we used cross validation (we already used on GridSearch so this is a nested cross validation) to find a mean cross validation score of 0.8955, highest for the decision tree models. Since nested cross validation and GridSearch takes a long time to run, it is not recommended to run those lines.

6- Conclusions

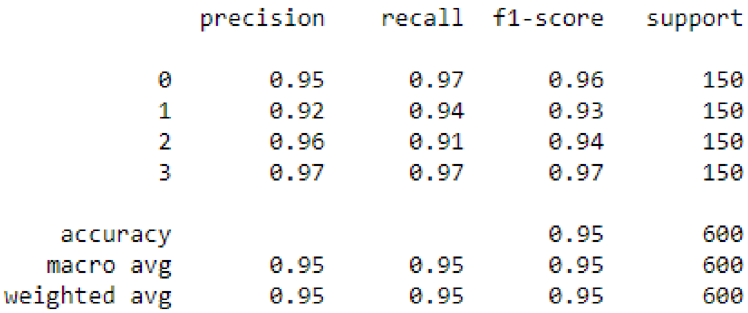
Below is the dataframe for our different models and their respective accuracy scores.After finding the best model as Logistic Regression, we decided to select Feature Selected Logistic Regression model as our final model since it is easier to interpret.



	Models	Accuracy Scores
1	Logistic Regression Model Mean Cross Validation	0.963000
2	Logistic Regression Model with Forward Feature...	0.950000
0	Logistic Regression Model	0.945000
5	KNN Model with k = 9 Mean Cross Validation	0.932500
4	KNN Model with k = 3	0.910000
8	Random Forest Model Mean Cross Validation	0.895500
7	Random Forest Model	0.885000
6	Decision Tree Model	0.823333
3	FFS Log. Reg. Model Mean Cross Validation	0.250000

Figure 7.

First thing we are going to analyze is the classification report which gives the precision, recall, f1-score and accuracy scores.



```
              precision    recall  f1-score   support

           0       0.95      0.97      0.96       150
           1       0.92      0.94      0.93       150
           2       0.96      0.91      0.94       150
           3       0.97      0.97      0.97       150

    accuracy                           0.95       600
   macro avg       0.95      0.95      0.95       600
weighted avg       0.95      0.95      0.95       600
```

Figure 8.

Then we examined the coeffiecents of the features, plotted them by importance and had the results below.

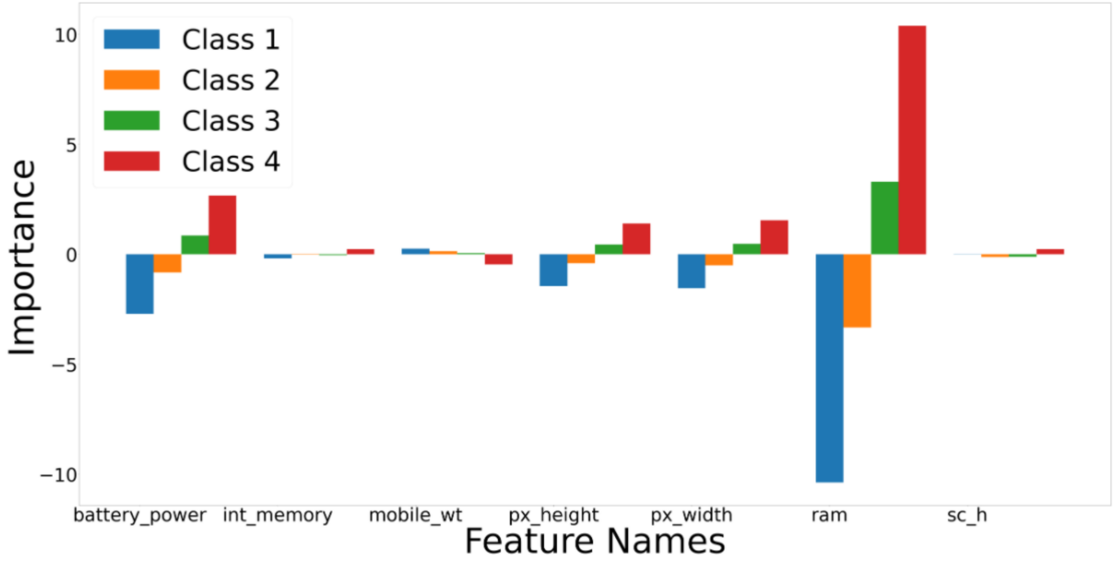


Figure 9.

We can say that as importance reaches high - or high + values, that feature is very important to determine the price range. From the plot above, we can see that the ram of the phone is very important to find the price range of that phone. High ram increases the price range, same with batter_power, int_memory, px_height and px_width. But mobile_wt has an opposite affect on the price range. Features named sc_h and int_memory has a very small effect on the price_range. Since we know what does columns represent, we can conclude that we have analyzed and found out which features affect the price of a phone and how big of an effect that feature has. We have not used the test dataset given in the site we took our train data (used as a full dataset) from since they have no price_range columns and we are not going to be doing unsupervised learning.