Project 11: Finding Best Perfoming Model for Mobile Price Prediction

Name:	**Sugandh Mittal, Sumouli Chakraborty**
Registration No./Roll No.:	20278, 20279
Institute/University Name:	IISER Bhopal
Program/Stream:	EECS
Problem Release date:	February 02, 2022
Date of Submission:	16/04/2023

1 Introduction

Aim: Predicting the price range of mobile phones based on their specifications.
Task: Classify the price range into four categories. We will discuss detailed data engineering,model selection,and performance evaluation.
Data Description: The data was provided by our professor.It contains 2000 data points, 20 features ranging from blue(bluetooth),fc(front camera),pc(primary camera)etc. and 4 classes - 0(cheap),1(moderate),2 (economical) and 3(expensive).

2 Methods

We used a dataset of mobile phones with their features such as battery power, RAM, screen size, etc. We preprocessed the dataset by handling missing values, categorical variables, scaling the features, statistical analysis, and exploratory data analysis. We split the dataset into training and testing sets. We trained the following classification models on the training set: Support Vector Machine(SVM), Random Forest, Gaussian Naive Bayes , Logistic Regression, K-Nearest Neighbor(KNN) and finally ensemble learning (AdaBoost).
We applied hyperparameter tuning for each model using grid search and cross-validation. For hyperparametertuning we used parameters declared as default in scikit-learn for best performance of the model according to the scoring method of macro f1 score. Specifically for the kNN, we also made a graph for error rate vs number of neighbours which can be seen below 9
The parameters for our models have been mentioned below in our table.1 We have also mentioned pseudo-code for the hyperparameter tuning method we have used. 1

```
//parameter_grid={**parameters}
    grid_search = GridSearchCV(model(),param_grid=parameter_grid, scoring='*')
    * refers to the scoring measure eg: f1_macro(in our case)
//We fit the predictor and target variables from the training set
    grid_search.fit(X_train, y_train)
//The parameters which gives the best result according the the scoring mentioned
//will be printed.
    print(grid_search.best_estimator_)
```

Figure 1: Pseudo-code for hyperparameter tuning

The detailed overview of the method, inferences and visulaizations can be viewed on my Github repostory for this project : Sumouli Chakraborty, Sugandh Mittal

Table 1: Performance Of All Classifiers Using All Parameters

Classifier	Main Parameters	Accuracy	F-measure
Logistic Regression	solver=newton-cg	0.9925	0.9924
Support Vector Machine	'C': 0.0001, 'kernel': 'linear'	0.9900	0.9900
Support Vector Machine	'C': 10000.0, 'gamma': 1e-08, 'kernel': 'rbf'	0.9875	0.9875
Support Vector Machine	'C': 1000.0, 'degree': 2, 'kernel': 'poly'	0.9800	0.9800
k Nearest Neighbour	'n neighbors': 24, 'leaf size':5,'metric'=minkowski	0.9450	0.9449
Gaussian Naive Bayes	'var smoothing: 2.848035869e-07', 'average:'weighted'	0.80	0.8092
AdaBoost	Best SVM(Learning Rate = 3)	0.98(varying)	0.9775
AdaBoost	Decision tree(Learning Rate = 1)	0.91(varying)	0.91

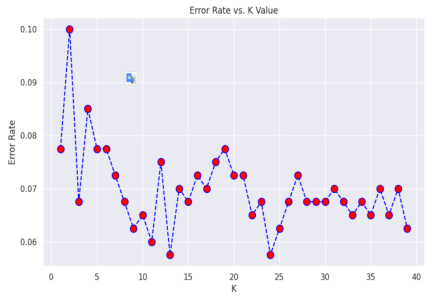

Figure 2: K-value vs Error rate graph

3 Evaluation Criteria

Accuracy is used to determine the overall performance of each model in predicting the correct mobile phone prices.

Precision gives us an idea of how often the model correctly predicts high-priced mobile phones when it predicts that a mobile phone is high-priced.

Recall gives us an idea of how often the model correctly identifies high-priced mobile phones out of all the high-priced mobile phones in the dataset.

F1-score is the harmonic mean of precision and recall,used to evaluate the overall performance of the models.

The evaluation criteria is decided on the nature of the task and the data available.

4 Analysis of Result

In this section we compare the values of accuracy and f1 score for all the classification models and using confusion matrices of all the classifiers used after performing hyperparameter tuning on them. We have also evaluated accuracy, macro average, weighted average etc. for all the classes(for us 4 classes) separately as well.1table

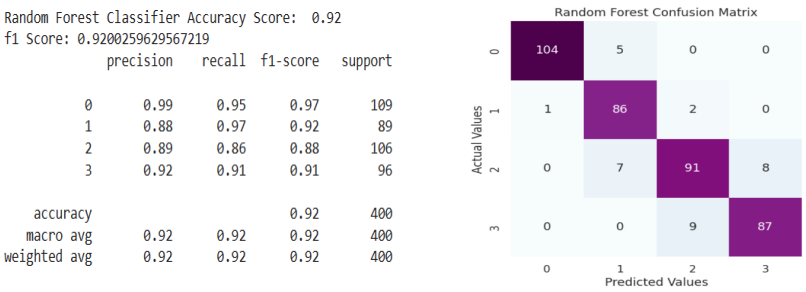

Figure 3: Output of Random Forest Classifier

KNN Classifier Accuracy Score: 0.945
KNN Classifier f1 Score: 0.9449371400821219

	precision	recall	f1-score	support
0	0.99	0.96	0.98	109
1	0.91	0.99	0.95	89
2	0.94	0.90	0.92	106
3	0.94	0.94	0.94	96
accuracy			0.94	400
macro avg	0.94	0.95	0.94	400
weighted avg	0.95	0.94	0.94	400

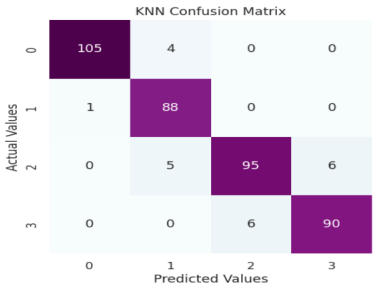

Figure 4: Output of k Nearest Neighbours

Accuracy Score: 0.805
f1 Score: 0.8092761260866478

	precision	recall	f1-score	support
0	0.96	0.88	0.92	109
1	0.64	0.75	0.69	89
2	0.71	0.72	0.71	106
3	0.93	0.86	0.90	96
accuracy			0.81	400
macro avg	0.81	0.80	0.81	400
weighted avg	0.82	0.81	0.81	400

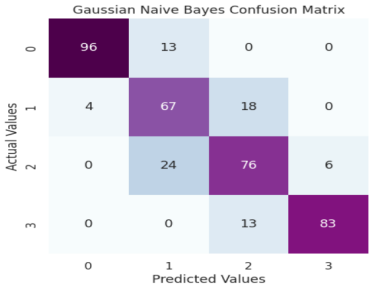

Figure 5: Output of Gaussian Naive Bayes

SVM (Linear) Classifier Accuracy Score: 0.99
SVM (Linear) Classifier f1 Score: 0.9900183198810535

	precision	recall	f1-score	support
0	1.00	1.00	1.00	109
1	1.00	0.98	0.99	89
2	0.97	0.99	0.98	106
3	0.99	0.99	0.99	96
accuracy			0.99	400
macro avg	0.99	0.99	0.99	400
weighted avg	0.99	0.99	0.99	400

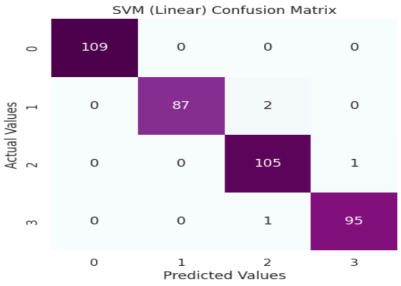

Figure 6: Output of SVM (Linear)

SVM (rbf) Classifier Accuracy Score: 0.9875
SVM (rbf) Classifier f1 Score: 0.9875151893520068

	precision	recall	f1-score	support
0	1.00	0.99	1.00	109
1	0.98	0.99	0.98	89
2	0.98	0.98	0.98	106
3	0.99	0.99	0.99	96
accuracy			0.99	400
macro avg	0.99	0.99	0.99	400
weighted avg	0.99	0.99	0.99	400

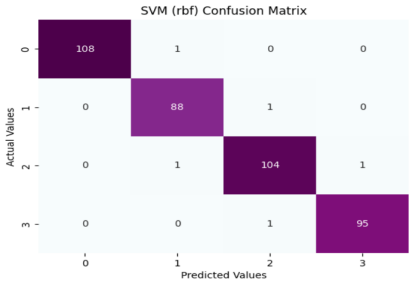

Figure 7: Output of SVM (rbf)

SVM (Polynomial) Classifier Accuracy Score: 0.98
SVM (Polynomial) Classifier f1 Score: 0.9800284444924577

	precision	recall	f1-score	support
0	1.00	0.98	0.99	109
1	0.96	0.99	0.97	89
2	0.98	0.96	0.97	106
3	0.98	0.99	0.98	96
accuracy			0.98	400
macro avg	0.98	0.98	0.98	400
weighted avg	0.98	0.98	0.98	400

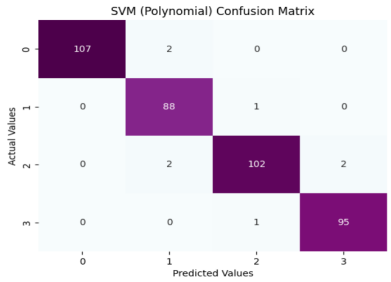

Figure 8: Output of SVM (Polynomial)

Accuracy: 0.9825
f1 Score: 0.9825374844358759

	precision	recall	f1-score	support
0	1.00	0.97	0.99	109
1	0.96	0.98	0.97	89
2	0.98	0.98	0.98	106
3	0.99	1.00	0.99	96
accuracy			0.98	400
macro avg	0.98	0.98	0.98	400
weighted avg	0.98	0.98	0.98	400

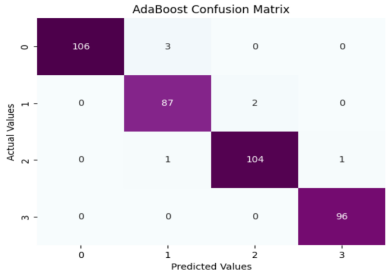

Figure 9: Output of AdaBoost

5 Discussions and Conclusion

We evaluated the performance of each model based on accuracy and F1-score (as we have a balanced dataset our data will produce equal macro and micro averaged score) and present those in the table below. Alongside we applied Ensemble learning technique through AdaBoost. We applied Ensemble learning over SVM(linear) and Decision tree classifier) which showed accuracy of 0.98 and 0.91 respectively. From the above observation we can say for our best performing model is Logistic Regression with tuning over solver parameter with an accuracy of 0.9925 and f1 score 0.9924. Also SVM (specially tuned over linear kernel) showed significance accuracy for our model.

6 Future goals

In this project we also tried implementing Artificial Neural Network model but due to lack of knowledge we couldn't improve much. So ANN model can be improved by tuning number of layers in ANN of increasing number neurons per layer and also can use further models like neural network models, decision trees, and gradient boosting models.In this study, we used a set of predefined features. However, there may be other features that could be useful and by exploring those techniques, we may be able to improve the accuracy of the models. By applying these strategies, we can create more accurate and robust models that can be used in real-world applications.

7 Roles Played

1.Sugandh Mittal- Code-Half the data analysis and first three models, Report- section 1,3,4,6
2.Sumouli Chakraborty - Code-Rest half of the data analysis,next 3 models, adaboost, Report- section 2,4,5,7.