Mobile Price Class Prediction

Name:	**Mohammad Anish**
Registration No./Roll No.:	19195
Institute/University Name:	IISER Bhopal
Program/Stream:	Physics
Problem Release date:	August 17, 2023
Date of Submission:	November 19, 2023

1 Introduction

In the modern world, mobile technology is a rapidly evolving sector which continually introduces new devices with different specification and features. Due to this vast range of availability of choices, it's becomes necessary for both manufacturers and buyers to understand the pricing dynamics of phones. The aim of the project is to address this challenge by predicting the price range for mobile phones using different features.

The dataset contains 20 different features and a class label for the mobile price range with 2000 training and 1000 test instances. The class label has four values - 0, 1, 2 and 3 for four price range categories - cheap, moderate, economical and expensive respectively. Using these 20 features, we are trying to train a classification model.

Feature	**Count**	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
battery_power	2000	1238.519	439.418	501	851.75	1226	1615.25	1998
blue	2000	0.495	0.500	0	0	0	1	1
clock_speed	2000	1.522	0.816	0.5	0.7	1.5	2.2	3
dual_sim	2000	0.510	0.500	0	0	1	1	1
fc	2000	4.310	4.341	0	1	3	7	19
four_g	2000	0.522	0.500	0	0	1	1	1
int_memory	2000	32.047	18.146	2	16	32	48	64
m_dep	2000	0.502	0.288	0.1	0.2	0.5	0.8	1
mobile_wt	2000	140.249	35.400	80	109	141	170	200
n_cores	2000	4.521	2.288	1	3	4	7	8
pc	2000	9.917	6.064	0	5	10	15	20
px_height	2000	645.108	443.781	0	282.75	564	947.25	1960
px_width	2000	1251.516	432.199	500	874.75	1247	1633	1998
ram	2000	2124.213	1084.732	256	1207.5	2146.5	3064.5	3998
sc_h	2000	12.307	4.213	5	9	12	16	19
sc_w	2000	5.767	4.356	0	2	5	9	18
talk_time	2000	11.011	5.464	2	6	11	16	20
three_g	2000	0.762	0.426	0	1	1	1	1
touch_screen	2000	0.503	0.500	0	0	1	1	1
wifi	2000	0.507	0.500	0	0	1	1	1

Table 1: Description of Data

2 Methods

2.1 Data Preprocessing

Looking at the data we get to know that it doesn't have any missing value. Rather some features like pixel height and screen width have a minimum value assigned as '0'. In real life scenario it's not possible to have phones with these particular features. Thus, we had replaced those values with there respective categorical mean, i.e. 645.108 and 5.767 respectively.

Our data contains categorical and numerical features. Categorical features are Bluetooth, dual sim, 4g, 3g, touch screen and WiFi. And the remaining ones are numerical features. So, to standardizing the numerical features, we have used Standard Scalar. Standard Scalar is a prepossessing technique which standardizes features by scaling the mean to zero and variance scaled to one unit. For categorical features, we have used another prepossessing technique known as One Hot Encoding. It is a technique used for categorical features. It involves creating binary columns for each category within the feature which represents their presence or absence with 1s and 0s. For example, the 'Bluetooth' feature might become two columns: 'Bluetooth_Yes' and 'Bluetooth_No'. It's ensures that our model is correctly interpreting categorical data without assuming any order of hierarchy. This contributes to accurate predictions.

2.2 Feature Selection

For feature selection technique, we have used Select K-Best with the '*mutual_info_classif*' scoring method. Mutual Information computes the mutual information between each feature and target variable, measuring the dependency between them. It is equal to zero, only if two random variables are independent and higher mutual information values indicate that a feature provides more information about the target variable. And, this score is used by the Select K-Best to choose top k numbers of features with the highest scores.

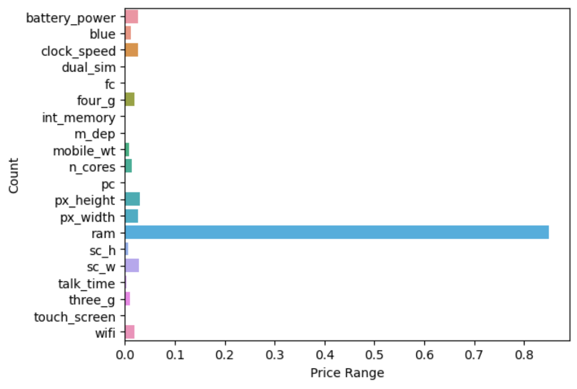

Figure 1: Mutual Information Score of different features

2.3 Model Training

Initially, we began by splitting the dataset into 80% for training and 20% for testing and class labels as an input for ensuring stratification for balanced representation. Then, we applied five distinct techniques for model training: Decision Tree, Random Forest, Support Vector Machine, K Nearest Neighbours, and Logistic Regression. To optimize performance, we used *GridSearchCV* for hyper-parameter tuning which tests various parameter combinations and selecting those that give the best f1-macro score. This process involves training classifiers with different parameter values and identifying the optimal configuration for enhanced model performance. Link to code - Github

1. Decision Tree: It is a machine learning algorithm that organizes data like a tree and makes decision at each step based on specific features. It breaks down the data by choosing the most useful features, creating branches until it reaches final predictions at the leaves. The tree is built

by optimizing decisions to improve accuracy. When predicting, the input follows the tree's path, and the final leaf provides the outcome.

S.No.	Parameter Name	Parameter values	
1.	criterion	entropy	gini
2.	min_samples_split	2	5
3.	min_samples_leaf	2	4

Table 2: Hyper-parameter tuning for Decision Tree

2. Random Forest: It is a powerful machine learning method for tasks like classification and regression. It builds multiple decision trees during training, combining their results to improve accuracy. Each tree uses a random subset of features and data, reducing over-fitting. In predictions, the ensemble of trees provides robust and accurate results.

S.No.	Parameter Name	Parameter values	
1.	criterion	entropy	gini
2.	min_samples_split	2	5
3.	min_samples_leaf	2	4

Table 3: Hyper-parameter tuning for Random Forest

3. Support Vector Machine (SVM): It is a machine learning algorithm used for both classification and regression tasks. SVM works by finding an optimal hyperplane that separates different classes in a high-dimensional feature space. The objective is to maximize the margin, which is the distance between the hyperplane and the nearest data points of each class.

S. No.	Parameter Name	Parameter values		
1.	C	0.1	1	100
2.	gamma	scale	auto	
3.	shrinking	True	False	

Table 4: Hyper-parameter tuning for SVM

4. K Nearest Neighbours (KNN): It is an intuitive instance-based learning algorithm, classifies new instances by identifying the majority class among their k nearest neighbors, fostering a reliance on local patterns within the data.

S. No.	Parameter Name	Parameter values		
1.	n_neighbors	800	900	
2.	weights	uniform	distance	
3.	algorithm	auto	kd_tree	
4.	leaf_size	10	15	20

Table 5: Hyper-parameter tuning for KNN

5. Logistic Regression: It is a linear model commonly employed for binary classification. It estimates the probability of an instance belonging to a particular class, employing the logistic function to transform the raw output into a probability, facilitating an informed and robust decision-making process.

S. No.	Parameter Name	Parameter values		
1.	C	0.5	1	
2.	solver	lbfgs	liblinear	saga
3.	penalty	l1	l2	None

Table 6: Hyper-parameter tuning for Logistic Regression

3 Evaluation Criteria

1. Precision is the ratio of true positives to the sum of true positives and false positives and it assesses the accuracy of positive predictions.

$$Precision = \frac{True\,Positive}{True\,Positive + False\,Positive}$$

2. Recall is the ratio of True positives to the sum of true positives and false negatives.

$$Recall = \frac{True\,Positive}{True\,Positive + False\,Negative}$$

3. f1-score combines both of these as it is the harmonic mean of precision and recall, and thus offers a balanced measure which consider both false positives and false negatives.

$$f1 - score = \frac{2 \times Precision \times Recall}{Prescision + Recall}$$

   f1-measure can have a value ranging from 0 to 1, with 1 being perfect score and 0 being the worst possible score. f1-macro average is the unweighted average of the f1-measure scores for each class. Since the evaluation considers the performance across all classes equally, it is useful in the cases when a certain class has fewer members. For most of our decisions, we will be looking at f1-measure and f1-macro averages to determine the performance of our classifiers.

where each of the above term is defined as:

- True Positives: It is the instances that are correctly predicted as positive by the model.
- True Negatives: It is the instances that are correctly predicted as negative by the model.
- False Positives: It is the instances that are incorrectly predicted as positive by the model when they are actually negative.
- False Negatives: It is the instances that are incorrectly predicted as negative by the model when they are actually positive.

4 Analysis of Results

As we can see that Logistic Regression is accurate in predicting the price range for the mobile features. And this are the best hyper-parameter for it are C as 1, penalty as l1 and solver as saga with selecting all the parameters.

1.	Decision Tree	87.74%	87.75%	87.73%
2.	Random Forest	92.25%	92.25%	92.24%
3.	SVM	90.68%	90.75%	90.71%
4.	KNN	77.50%	77.0%	76.90%
5.	Logistic Regression	98.04%	98.00%	**98.00%**

Table 7: F1 Macro Score Comparison

5 Discussions

For further work, we can add more parameters to hyper parameter tuning which might provide better predictions.