Exploratory anaysis and Predictive analytics on mobile phone price data

Name:	**Alli Khadga Jyoth**
Registration No./Roll No.:	19024
Institute/University Name:	IISER Bhopal
Program/Stream:	DSE
Problem Release date:	February 02, 2022
Date of Submission:	April 24, 2022

1 Introduction

The project aimed at developing a robust classical machine learning model to accurately classify different categories of mobile phones, categorized with respect to their price. This led to an exploration of various classical ML methods and a variety of feature selection and feature engineering techniques. The dataset we used contained 20 different feature attributes, including battery power, CPU clock speed, dual sim support or not, Front Camera mega pixels, has 4G or not, has Wi-Fi or not, etc. Logistic Regression turned out to be the best model we got, with 99% accuracy and a 0.99 f-score on the validation data.

2 Methods

In this project, we used 11 different classifiers, including KNN, Logistic Regression, Decision Tree, SVC, NuSVC, Random Forest, AdaBoost, Gradient Boosting Classifier, SGDClassifier, Gaussian Naive Bayes, and MultinomialNB. The code file of the project can be found at `https://github.com/KhadgaA/DSML-Project` The following steps were followed throughout the process:

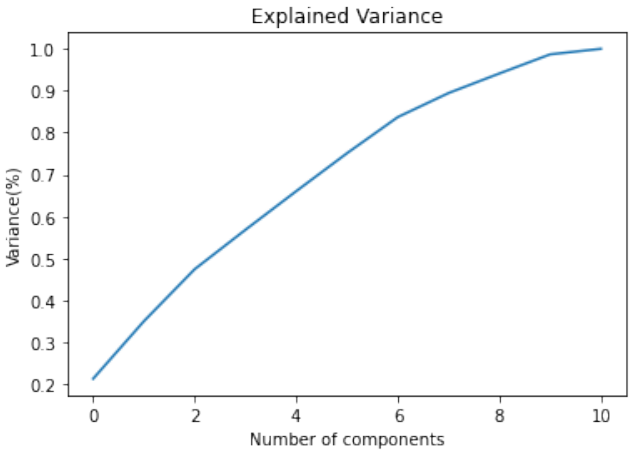

Figure 1: Explained Variance vs no. of Components

2.1 Loading and Preprocessing

The mobile phone dataset was provided to us by the instructor. The dataset was already split into training and testing sets, with testing class labels removed. The training set contained 2000 samples with 500 samples from each of the four classes labeled 0 to 3. The data includes 20 feature attributes.

The data didn't contain any Null values, and it was balanced well with 500 samples for each of the 4 classes. Therefore the preprocessing step was not necessary

2.2 Feature Engineering and Feature Selection

Since the data contained only 20 attributes, the feature selection was unnecessary, but it helped gain insight into the data. We used Correlation Matrix with the Class Labels, SelectKBest algorithm with K = 10, and Reverse Feature Elimination for feature selection.

Table 1: First 5 rows of the training data

battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores
842	0	2.2	0	1	0	7	0.6	188	2
1021	1	0.5	1	0	1	53	0.7	136	3
563	1	0.5	1	2	1	41	0.9	145	5
615	1	2.5	0	0	0	10	0.8	131	6
1821	1	1.2	0	13	1	44	0.6	141	2

First 10 attributes

Table 2: Next 10 attributes of Data

pc	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi
2	20	756	2549	9	7	19	0	0	1
6	905	1988	2631	17	3	7	1	1	0
6	1263	1716	2603	11	2	9	1	1	0
9	1216	1786	2769	16	8	11	1	0	0
14	1208	1212	1411	8	2	15	1	1	0

2.2.1 Feature Engineering

By combining two or more features of the raw data 15 more features were engineered thus resulting in a total of 35 features. Some of the engineered features are, $sc\_diag = \sqrt{sc\_h^2 + sc\_w^2}$, $sc\_area = sc\_h \times sc\_w$, and $ram\_pre\_core = \frac{ram}{n\_cores}$

2.2.2 Selecting Features with Correlation Matrix

The correlation Matrix was calculated for all the 35 features with the output variable. And from the figure 2, most of the features are uncorrelated with the *price_range* attribute, which is the class label of the data. So the feature selection was made using a correlation threshold of 0.01. Any feature which correlated below 0.01 was removed. This resulted in a dropout of 6 features.

2.2.3 Variation Inflation Factor

Variance Inflation Factor(VIF) quantifies the severity of multicollinearity between the predictor variables. As a rule of thumb, any feature with a VIF greater than 5 is considered to have high multicollinearity and should be discarded. So we did the same. This resulted in 18 discarded features. So we are left with 11 features.

2.2.4 Reverse Feature Elimination

This is the method of selecting K best features according to a performance measure; here, the features are discarded iteratively till the performance drops drastically. Its also known as Backward Feature Elimination, it was fitted on the 35 features after feature engineering, and 10 best features were selected.

2.2.5 Principle Component Analysis

Principle Component Analysis (PCA) is the technique used to reduce the dimensionality of the data. We fitted PCA on the 11 features obtained after the VIF stage. From Graph 1 we see that most of the explained variance is explained by only 10 features, so we select 10 components from PCA.

2.2.6 SelectKBest

SelectKBest selects K number of features from the dataset according to a particular score. We use ANOVA F-value to select 10 best features from the 35 features after features engineering. The top 5 among the 10 selected features were $battery\_power, px\_height, px\_width, ram, px\_diag$.

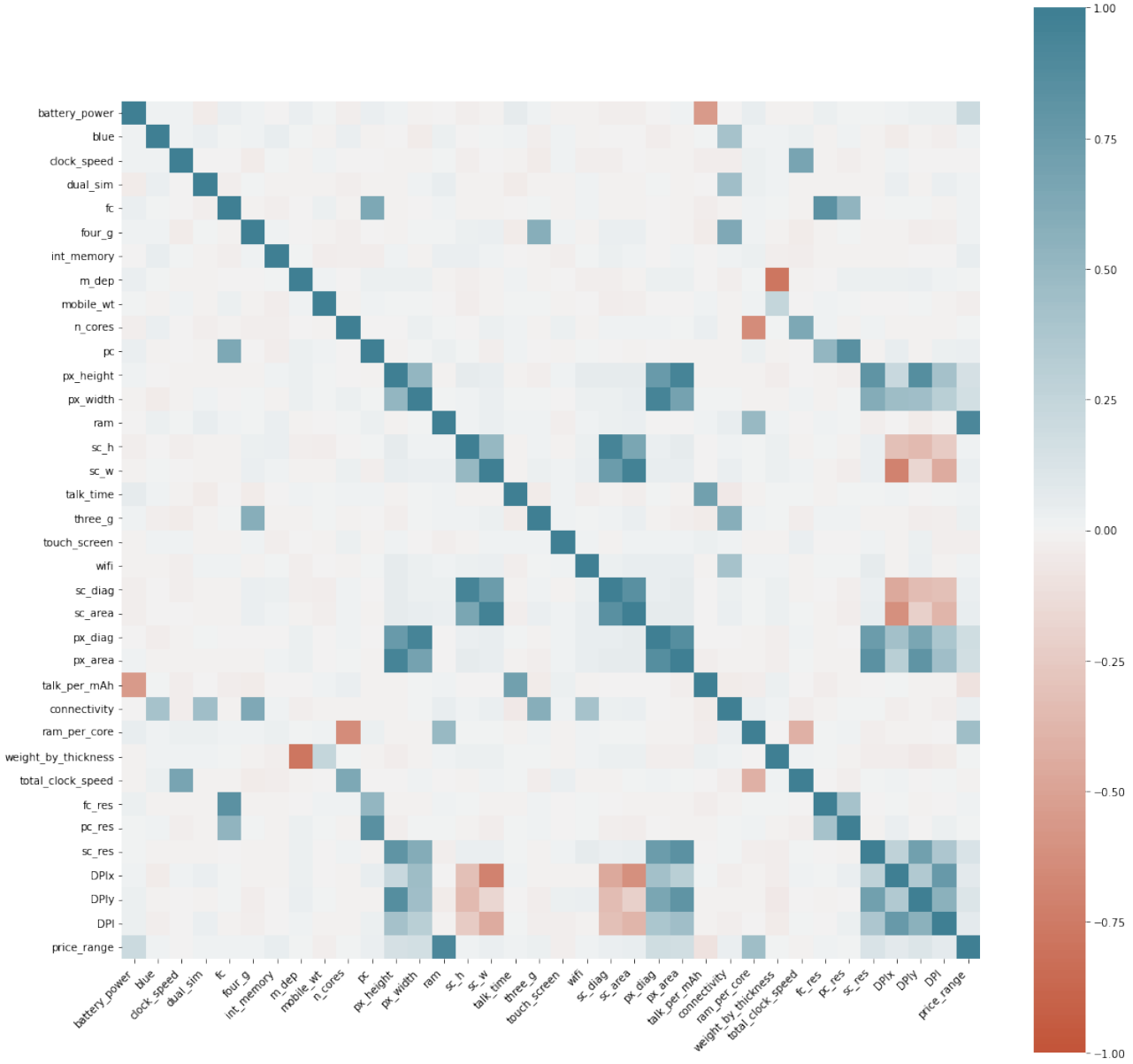

Figure 2: Correlation Matrix Plot

3 Experimental Analysis

Each of the 11 models' performance was measured and recorded before and after every feature selection and feature engineering technique. The performance of each model can be viewed in the excel file attached here `https://docs.google.com/spreadsheets/d/1S3PVeLqoQCGK7-esGmvBzUlu6_f4Va0UjSkRdqZEBx8/edit?usp=sharing`. The models were also passed to the GridSearch of sklearn to learn the best parameters for a given model.

Out of all the models, Logistic Regression achieved the highest performance with an accuracy 0.99 and a micro-avg score 0.99 as tested on validation data containing 400 samples. Surprisingly, this was achieved on raw data, i.e., the data given by the instructor, which hasn't been processed with feature selection or feature engineering. The following best models were from SVC and NuSVC, as seen from table 3.

Model	Processing Step	Accuracy	Precision	Recall	macro averaged
Logistic Regression	Raw Data	0.99	0.99	0.99	0.99
SVC	Raw Data	0.98	0.98	0.98	0.98
NuSVC	Feature Engineering	0.98	0.98	0.98	0.98
SVC	Feature Engineering + Correlation	0.98	0.98	0.98	0.98
NuSVC	Feature Engineering + Correlation	0.98	0.98	0.98	0.98

Table 3: Best 5 Models

4 Discussions

The dataset only contained 2000 training samples, which are undoubtedly at the low end compared to other dataset sizes. But, this doesn't mean that a classifier can't learn the general patterns to classify the data correctly. As a surprise, we see that the Logistic regression performed the best of all the models used and that too on raw data, which was not featured nor feature selected. This shows that Logistic regression can learn a linear discriminant function given a set of best parameters learned through GridSearch. We also notice that any model is very susceptible to changes in its parameters, i.e., for example, Logistic regression had the best performance at C value = 0.2807; even a small change to 0.3 resulted in terrible performance. This showed us the importance of hyperparameter tuning and how important a role it plays in model performance. In the future, we might be able to use unsupervised techniques to classify the data points without ever needing the true train labels.