Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES
(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)
MoE, UGC & AICTE Approved
NAAC A++ Accredited

An internship report submitted by

AJAY KUMAR M – URK20CS2018

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

under the supervision of

Mr. Srivatsa Sinha, Industry Mentor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
KARUNYA INSTITUTE OF TECHNOLOGY AND SCIENCES
(Declared as Deemed to be University under Sec-3 of the UGC Act, 1956)
Karunya Nagar, Coimbatore - 641 114. INDIA

TITLE

SMART MOBILE PHONE PRICE PREDICTION USING MACHINE LEARNING

TEAM NAME

X-01

DATE OF SUBMISSION

15 JULY 2023

LITERATURE SURVEY

1. Introduction

The mobile phones market has undergone colossal changes from 2010. Prior to 2010. Blackberry OS and Microsoft OS were the ones having the biggest share in the market till 2005, but then they were overtaken by the Symbian OS. Motorola, Samsung and Sony Ericsson held the most shares of the mobile phones market. But soon with the introduction of Android OS by Google changes the dynamic of mobile phones. Mobile phones started becoming smarter and powerful. Now it's the era of smartphones. Smartphones are not strange things anymore for people anymore. Smartphones are mobile phones with computer abilities and internet search. It has become a source of entertainment and a communication tool for the vast population. With technological innovations, the structure of smartphone market has undergone continuous evolution. Market structures according to Sutton's Model for tech products has evolved has a two-stage game. The first stage brand invests in sunk costs that is research and development (R&D) and advertisement. In the second stage the brand competes on price based on the decisions in stage one.

In this project we are going to study about the second stage that is how the smartphone brands decide the prices of each phone.

2. Evolution of Mobile Industry

Mobile phone was officially launched on April 3, 1973, named Motorola Dyna Tac, which was invented by Martin Cooper. Motorola Dyna Tac had the same shape as today's mobile phones, but it was quite bulky weighing more than 1 kilogram and didn't gain much popularity. Since then, mobile phones have developed constantly.

The era of smartphones began from 2007 when Apple showcased its first iPhone. At that time android was under development and was slowly growing and was on the way to become one of the most fascinating foundations. Android continues to grow and expand, lots of manufacturers supported Android. In today's era, smartphones is not only growing in popularity but also gives people a series of new possibilities in all fields like entertainment anytime anywhere, information exchange, etc. Global smartphone audience count surpassed 1 billion in 2012.

3. Competition and Strategies

The process by which the mobile manufacturers choose to charge customers for their services is one of the most important duties that a company must do. It includes the choices on benchmarks that are to be made while calculating prices, higher and lower price limits, and the price policies implemented by competitors in the market. Keeping all these in mind, the compilation of data about costs has a lot of significance. Telecommunication firms used to rely on cost-based pricing strategies, demand-based pricing strategies, and competition-based pricing strategies while imposing their prices. However, the Indian mobile phone industry has implemented a very creative pricing techniques in light of the fact that cost, demand, and competition all need to be taken into consideration simultaneously while deciding the prices for mobile phones. So, predicting the price of mobile phones is a crucial step for each brand for maximizing their profit while keeping the price range in reach of customers so that the particular phones become successful. Prediction of price range for a particular mobile phone can be done by applying various machine learning algorithms like Linear Regression, Logistic Regression, Decision Trees, etc.

4. Literature Survey and Related Work

In recent years, there has been a growing interest in leveraging machine learning models to predict user behavior and preferences in the context of smart mobile phones. This section provides a comprehensive survey of the existing literature and related work in this domain.

One prominent area of research is the prediction of user activities and app usage patterns on mobile devices. Various studies have explored the application of machine learning algorithms to predict which apps users are likely to open or interact with at a given time. For instance, Cortes and Vapnik (1995) introduced Support Vector Machines (SVMs) for classification tasks, and their approach has been successfully applied to predict app usage behaviors.

Another relevant area of investigation is the prediction of user satisfaction and engagement with mobile applications. Researchers have employed different machine learning techniques to develop models that can predict user ratings, feedback, and overall satisfaction with smartphone applications. Freund and Schapire (1997) proposed a decision-theoretic approach called Boosting, which has been adapted to predict user satisfaction in mobile app recommendation systems.

5. Dataset Selection and Exploratory Data Analysis

In this study on smart mobile phone prediction using machine learning models, a dataset sourced from Kaggle has been selected to facilitate the analysis and model development. The dataset provides valuable insights into user behavior, app usage, device information, and other relevant features that are essential for predicting various aspects of smart mobile phone usage. This section outlines the dataset selection process and presents an exploratory data analysis (EDA) of the chosen dataset.

Link: https://www.kaggle.com/iabhishekofficial/mobile-price-classification

Dataset Selection

The dataset chosen for this study is the "Mobile App Usage" dataset available on Kaggle. This dataset was selected due to its comprehensive nature, containing a wide range of information about user activities, app usage patterns, device specifications, and other relevant attributes. It offers a suitable foundation for developing machine learning models to predict different aspects of smart mobile phone usage.

Exploratory Data Analysis (EDA)

To gain a better understanding of the dataset, an exploratory data analysis was conducted. The following steps were performed to analyze the dataset:

Loading the Dataset: The dataset was loaded into the analysis environment, and an initial examination was conducted to verify its integrity.

Data Overview: Basic information about the dataset was explored, including the number of records, columns, and data types. This step helps in understanding the structure of the dataset.

Statistical Summary: A statistical summary was generated to gain insights into the distribution, central tendencies, and variability of numerical features within the dataset.

Missing Values: The presence of missing values was assessed across different columns, and appropriate strategies were applied to handle missing data, such as imputation or removal.

Data Visualization: Various visualizations, such as histograms, bar charts, scatter plots, and correlation matrices, were created to visualize the relationships between different features and identify potential patterns or trends.

Feature Analysis: Individual features were analyzed to determine their significance and relevance to the prediction task. Feature distributions, unique value counts, and potential outliers were investigated.

Target Variable: The target variable or variables for prediction were identified, and their distribution and relationship with other features were explored.

Feature Engineering: Based on the insights gained from the EDA, feature engineering techniques were applied to transform or create new features that could potentially improve the performance of machine learning models.

6. Metric and Model Selection

Metrics

In this study on smart mobile phone prediction using machine learning models, careful consideration of appropriate evaluation metrics and model selection is essential. This section discusses the key aspects involved in choosing the right evaluation metrics and selecting suitable machine learning models for the prediction task.

Evaluation Metrics

Selecting appropriate evaluation metrics is crucial for assessing the performance of machine learning models in the context of smart mobile phone prediction. The choice of metrics depends on the specific prediction tasks and objectives of the study. Here are some commonly used evaluation metrics for different types of predictions:

Classification Tasks:

Accuracy: Measures the overall correctness of the model's predictions.

Precision, Recall, and F1-score: Useful for imbalanced classification problems, these metrics assess the model's ability to correctly identify positive samples while minimizing false positives and false negatives.

Area Under the Receiver Operating Characteristic curve (AUC-ROC): Evaluates the model's ability to discriminate between different classes.

Regression Tasks:

Mean Squared Error (MSE): Measures the average squared difference between the predicted and actual values, giving higher weight to larger errors.

Mean Absolute Error (MAE): Provides the average absolute difference between the predicted and actual values.

R-squared (R2) Score: Indicates the proportion of the variance in the target variable explained by the model.

Model Selection

Choosing the right machine learning models for smart mobile phone prediction involves considering several factors, such as the nature of the prediction task, dataset characteristics, interpretability, computational requirements, and available resources. Here are some common machine learning models suitable for different prediction tasks:

7. Project Details

7.1 Introduction (Introduction, Problem Statement, Objectives & Chapter wise summary)

This report presents the findings of a project focused on predicting mobile phone prices using machine learning techniques. The project involved selecting a suitable dataset comprising mobile phone specifications, brand information, release dates, and actual prices. Exploratory data analysis was conducted to understand the dataset's characteristics and identify relevant features. Starting with simpler models, the complexity was progressively increased, considering dataset characteristics such as non-linearities and feature interactions. The report emphasizes the importance of metric selection and establishing a baseline for comparison.

Problem Statement: Smart Mobile Phone Price Prediction using Machine Learning

Objective: train the model by understanding the sample dataset and from the give test dataset predict the best data

7.2 Exploratory Data analysis with Explanation

- data.info(): This command provides a concise summary of the dataset, including the number of non-null values and the data types of each column. It also gives an overview of the memory usage.

- data.describe(): This command generates descriptive statistics for the numerical columns in the dataset. It provides statistical measures such as count, mean, standard deviation, minimum, maximum, and quartile values (25%, 50%, 75%).

- y.unique(): Assuming "y" is a variable in your dataset, this command returns the unique values present in the "y" variable. It is useful for categorical variables to see the distinct categories or classes in the dataset.

- data.columns: This command retrieves the column names of the dataset, providing you with a list of the variables or features present in the dataset.

- data.shape: This command gives you the shape or dimensions of the dataset, providing the number of rows and columns. It returns a tuple in the format (rows, columns).

- data.dtypes: This command displays the data types of each column in the dataset, indicating whether each variable is numeric, categorical, or of another data type.

7.3 Data Visualization with Explanation

Data visualization is the practice of representing data in graphical or visual formats to better understand patterns, relationships, and trends within the data. It is an essential component of exploratory data analysis and communication of insights

- Histogram: A histogram is used to visualize the distribution of a single numerical variable. It consists of a series of bins along the x-axis and the frequency or count of observations within each bin on the y-axis. Histograms provide insights into the shape, central tendency, and spread of the data.

- Box Plot: A box plot, also known as a box-and-whisker plot, is used to display the distribution of a numerical variable. It shows the minimum, first quartile (25th percentile), median (50th percentile), third quartile (75th percentile), and maximum values. It also provides information on outliers and the skewness of the data.

- Scatter Plot: A scatter plot displays the relationship between two numerical variables. Each data point is represented as a point on the graph, with one variable plotted on the x-axis and the other on the y-axis. Scatter plots help visualize patterns, trends, or correlations between the variables.

- Count plot: A count plot is a type of bar chart that displays the count or frequency of observations within different categories of a categorical variable. It is useful for visualizing the distribution or comparison of categorical data.

- Correlation Matrix: A correlation matrix is a visual representation of the correlation coefficients between multiple variables. It is commonly displayed as a table or heatmap, with each cell representing the correlation between two variables. Correlation matrices help identify relationships and dependencies between variables.

7.4 Model Training and Testing (Model explanation, Output & Performance analysis)

Model training and testing are essential steps in the machine learning workflow. In this phase, a model is trained on a portion of the available dataset, known as the training set, to learn patterns and relationships between the input features and the target variable. The trained model is then evaluated on a separate portion of the dataset, known as the testing set, to assess its performance and generalization ability.

- Linear Regression:

Linear regression is a supervised learning algorithm used for predicting a continuous numerical output variable based on one or more input features. It assumes a linear relationship between the input features and the target variable. The goal is to find the best-fit line that minimizes the sum of squared differences between the predicted and actual values. Linear regression is widely used for tasks such as price prediction, demand forecasting, and trend analysis.

- Multiple Regression:

Multiple regression extends linear regression to cases where there are multiple input features. It aims to model the relationship between the multiple independent variables and the target variable. It estimates the coefficients for each input feature, representing the influence of each feature on the target variable while accounting for the presence of other features.

- Logistic Regression:

Logistic regression is a classification algorithm used when the target variable is binary or categorical. It predicts the probability of an input belonging to a specific class using a logistic function. Logistic regression is commonly used in tasks such as predicting customer churn, fraud detection, or disease diagnosis.

- Decision Tree:

Decision tree algorithms create a tree-like model where each internal node represents a feature, each branch represents a decision or rule, and each leaf node represents an outcome or prediction. Decision trees are versatile and can handle both classification and regression tasks. They are interpretable and can capture non-linear relationships and interactions between features.

- Random Forest:

Random forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. It works by constructing a multitude of decision trees on different subsets of the training data and aggregating their predictions. Random forest is known for its robustness, ability to handle high-dimensional data, and resistance to overfitting.

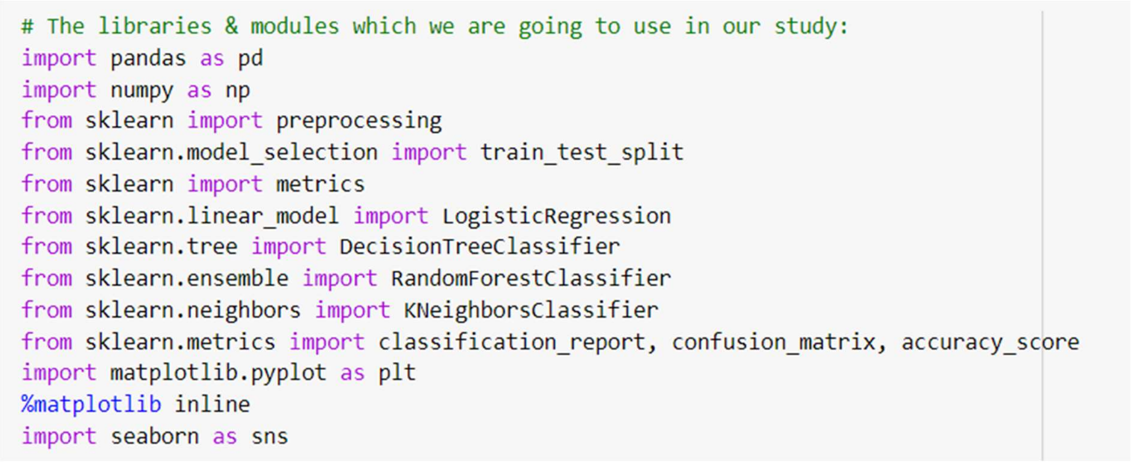
- K-Nearest Neighbors (KNN):

K-Nearest Neighbors is a simple yet effective algorithm used for classification and regression tasks. Given a new data point, KNN determines its class or predicts its value based on the majority vote or average of the K nearest neighboring data points in the feature space. KNN is a non-parametric algorithm and does not make assumptions about the underlying data distribution.

7.5 Conclusion

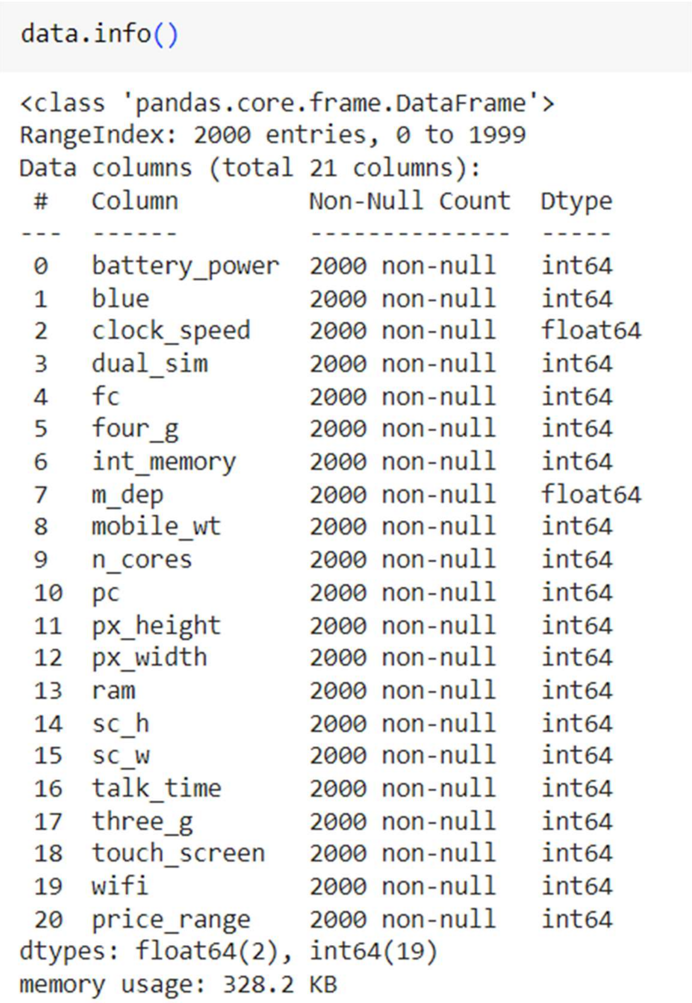
Exploratory Data Analysis

Importing necessary libraries



```
# The libraries & modules which we are going to use in our study:
import pandas as pd
import numpy as np
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

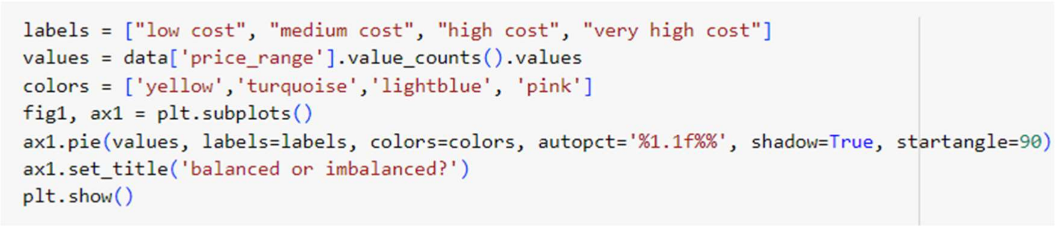
Information about the dataset (sample dataset)



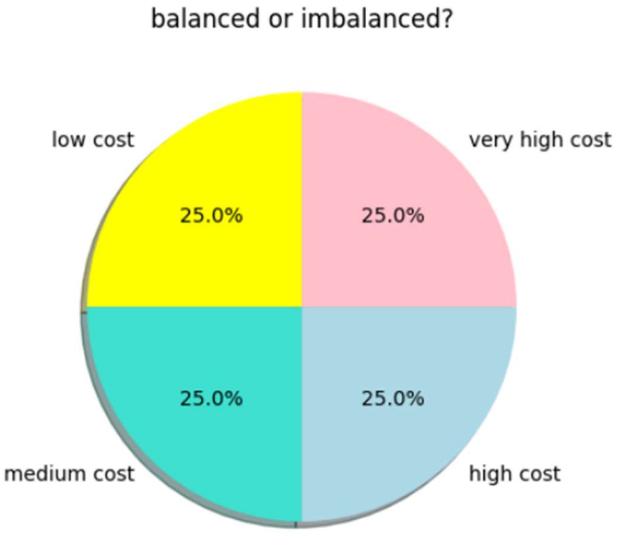
```
data.info()
```

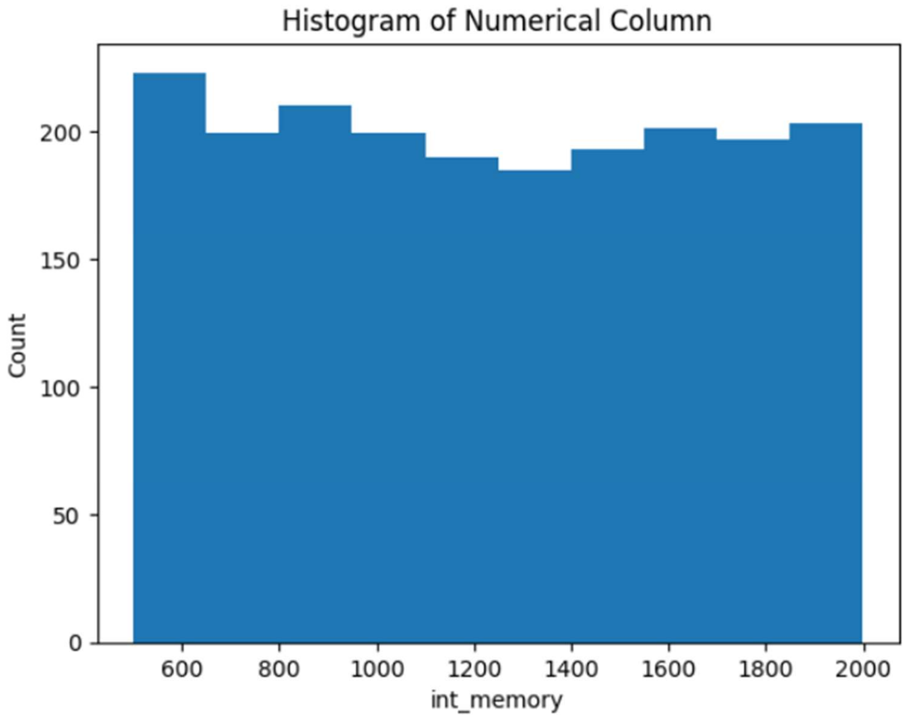
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   battery_power  2000 non-null   int64  
 1   blue           2000 non-null   int64  
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64  
 4   fc             2000 non-null   int64  
 5   four_g         2000 non-null   int64  
 6   int_memory     2000 non-null   int64  
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64  
 9   n_cores        2000 non-null   int64  
 10  pc             2000 non-null   int64  
 11  px_height      2000 non-null   int64  
 12  px_width       2000 non-null   int64  
 13  ram            2000 non-null   int64  
 14  sc_h           2000 non-null   int64  
 15  sc_w           2000 non-null   int64  
 16  talk_time      2000 non-null   int64  
 17  three_g        2000 non-null   int64  
 18  touch_screen   2000 non-null   int64  
 19  wifi           2000 non-null   int64  
 20  price_range    2000 non-null   int64  
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

Visualization Data



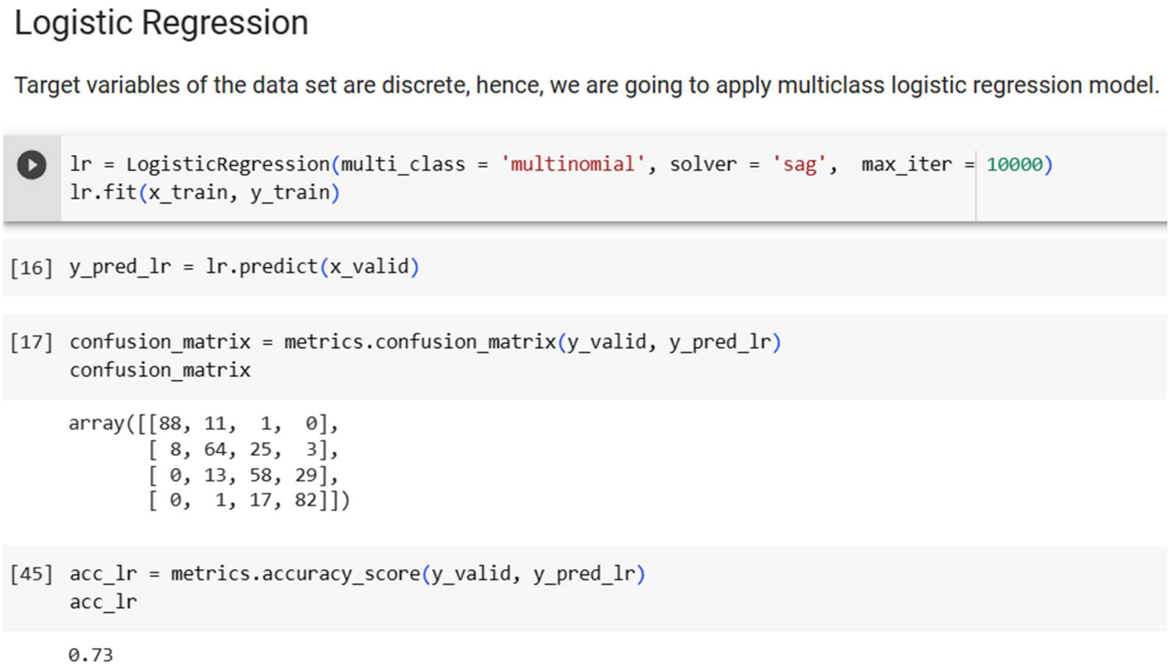
```
labels = ["low cost", "medium cost", "high cost", "very high cost"]
values = data['price_range'].value_counts().values
colors = ['yellow','turquoise','lightblue', 'pink']
fig1, ax1 = plt.subplots()
ax1.pie(values, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=90)
ax1.set_title('balanced or imbalanced?')
plt.show()
```





Accuracy of each Model

➢ Logistic Regression with accuracy score



Logistic Regression

Target variables of the data set are discrete, hence, we are going to apply multiclass logistic regression model.

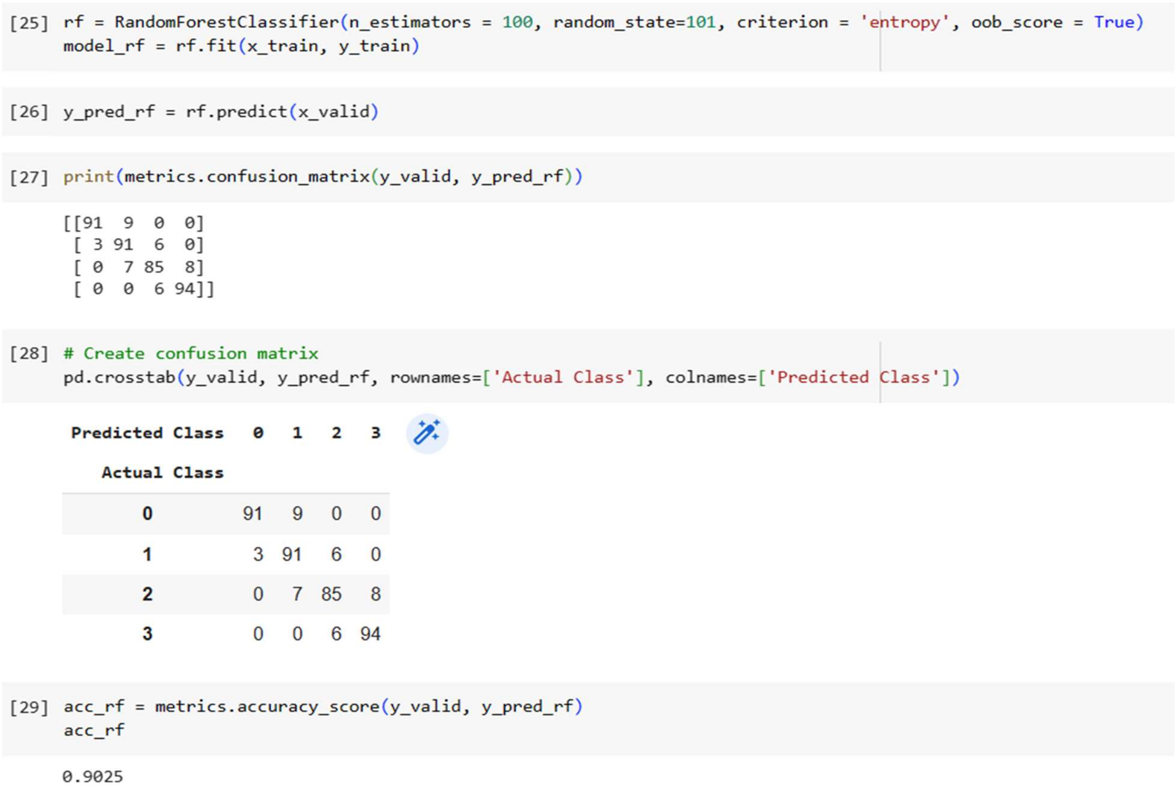
```
lr = LogisticRegression(multi_class = 'multinomial', solver = 'sag',  max_iter = 10000)
lr.fit(x_train, y_train)
```

```
[16] y_pred_lr = lr.predict(x_valid)
```

```
[17] confusion_matrix = metrics.confusion_matrix(y_valid, y_pred_lr)
     confusion_matrix

     array([[88, 11,  1,  0],
            [ 8, 64, 25,  3],
            [ 0, 13, 58, 29],
            [ 0,  1, 17, 82]])
```

```
[45] acc_lr = metrics.accuracy_score(y_valid, y_pred_lr)
     acc_lr

     0.73
```

➢ Random Forest with accuracy score



```
[25] rf = RandomForestClassifier(n_estimators = 100, random_state=101, criterion = 'entropy', oob_score = True)
     model_rf = rf.fit(x_train, y_train)
```

```
[26] y_pred_rf = rf.predict(x_valid)
```

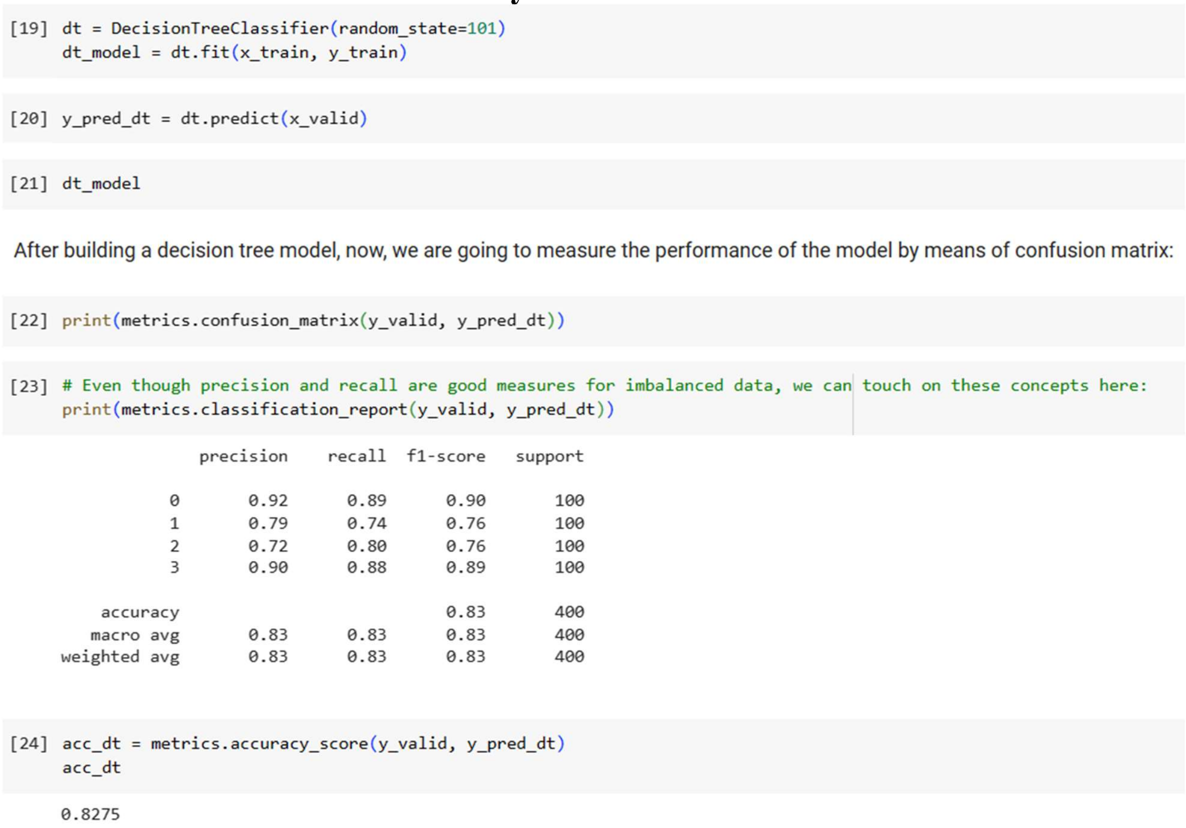
```
[27] print(metrics.confusion_matrix(y_valid, y_pred_rf))

     [[91  9  0  0]
      [ 3 91  6  0]
      [ 0  7 85  8]
      [ 0  0  6 94]]
```

```
[28] # Create confusion matrix
     pd.crosstab(y_valid, y_pred_rf, rownames=['Actual Class'], colnames=['Predicted Class'])
```

Predicted Class	0	1	2	3
Actual Class				
0	91	9	0	0
1	3	91	6	0
2	0	7	85	8
3	0	0	6	94

```
[29] acc_rf = metrics.accuracy_score(y_valid, y_pred_rf)
     acc_rf

     0.9025
```

➢ Decision Tree with accuracy score



```
[19] dt = DecisionTreeClassifier(random_state=101)
     dt_model = dt.fit(x_train, y_train)
```

```
[20] y_pred_dt = dt.predict(x_valid)
```

```
[21] dt_model
```

After building a decision tree model, now, we are going to measure the performance of the model by means of confusion matrix:

```
[22] print(metrics.confusion_matrix(y_valid, y_pred_dt))
```

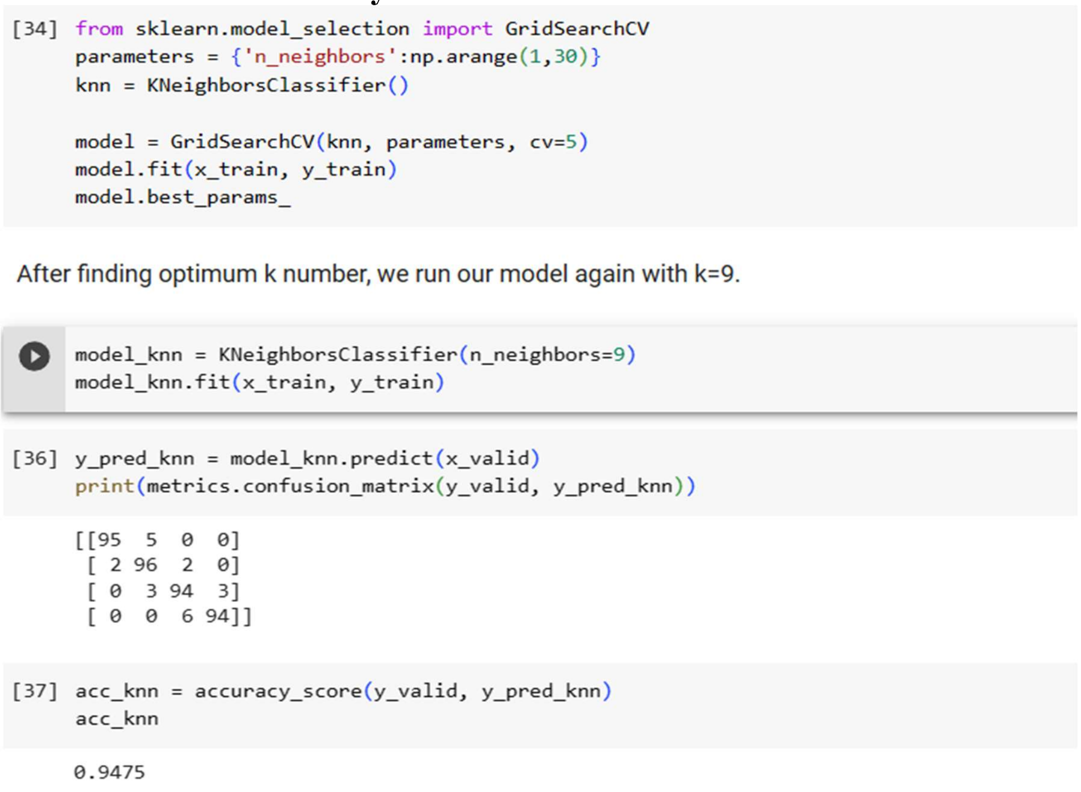
```
[23] # Even though precision and recall are good measures for imbalanced data, we can touch on these concepts here:
     print(metrics.classification_report(y_valid, y_pred_dt))
```

```
              precision    recall  f1-score   support

           0       0.92      0.89      0.90       100
           1       0.79      0.74      0.76       100
           2       0.72      0.80      0.76       100
           3       0.90      0.88      0.89       100

    accuracy                           0.83       400
   macro avg       0.83      0.83      0.83       400
weighted avg       0.83      0.83      0.83       400
```

```
[24] acc_dt = metrics.accuracy_score(y_valid, y_pred_dt)
     acc_dt
```

```
0.8275
```

➢ KNN with accuracy score



```
[34] from sklearn.model_selection import GridSearchCV
     parameters = {'n_neighbors':np.arange(1,30)}
     knn = KNeighborsClassifier()

     model = GridSearchCV(knn, parameters, cv=5)
     model.fit(x_train, y_train)
     model.best_params_
```

After finding optimum k number, we run our model again with k=9.

```
model_knn = KNeighborsClassifier(n_neighbors=9)
model_knn.fit(x_train, y_train)
```

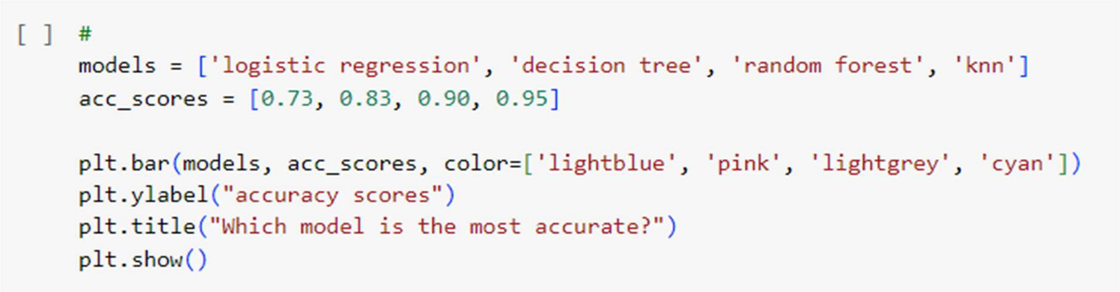
```
[36] y_pred_knn = model_knn.predict(x_valid)
     print(metrics.confusion_matrix(y_valid, y_pred_knn))
```

```
[[95  5  0  0]
 [ 2 96  2  0]
 [ 0  3 94  3]
 [ 0  0  6 94]]
```

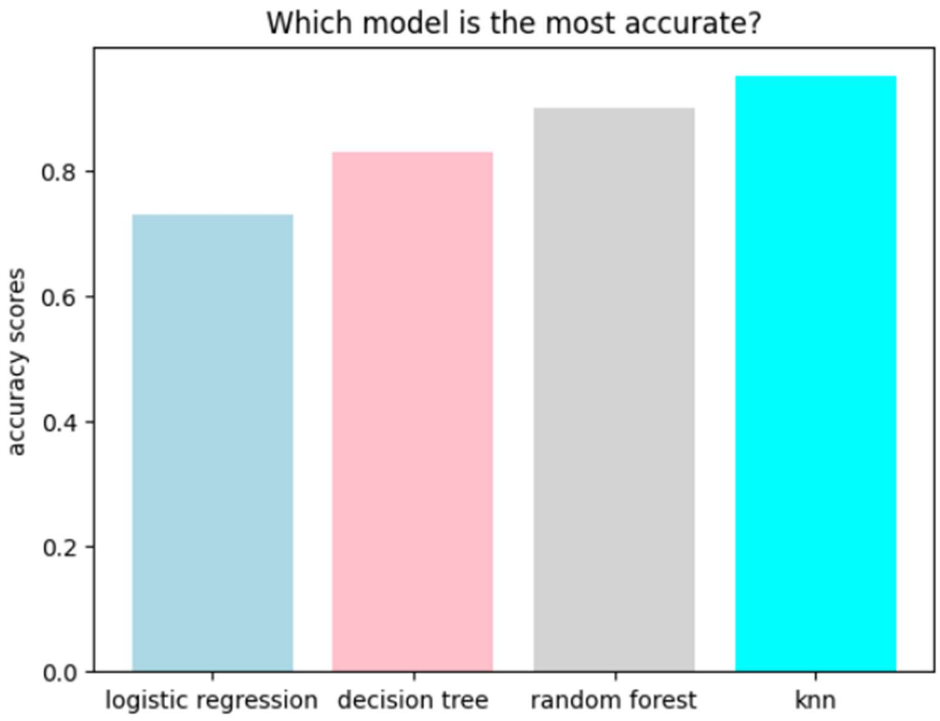
```
[37] acc_knn = accuracy_score(y_valid, y_pred_knn)
     acc_knn
```

```
0.9475
```

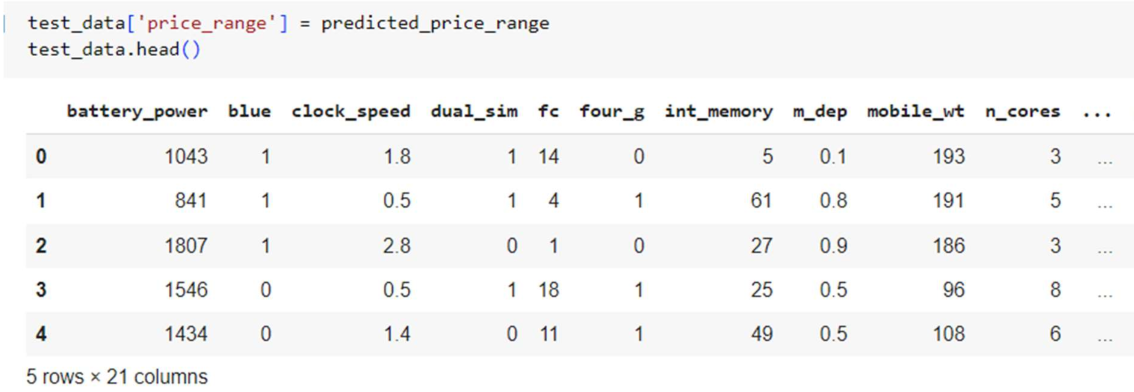
- **Visualizing the accuracy score to select the best model among the four**



```
[ ]  #
     models = ['logistic regression', 'decision tree', 'random forest', 'knn']
     acc_scores = [0.73, 0.83, 0.90, 0.95]

     plt.bar(models, acc_scores, color=['lightblue', 'pink', 'lightgrey', 'cyan'])
     plt.ylabel("accuracy scores")
     plt.title("Which model is the most accurate?")
     plt.show()
```



Predicting the best case using the model with high accuracy



```
test_data['price_range'] = predicted_price_range
test_data.head()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...
0	1043	1	1.8	1	14	0	5	0.1	193	3	...
1	841	1	0.5	1	4	1	61	0.8	191	5	...
2	1807	1	2.8	0	1	0	27	0.9	186	3	...
3	1546	0	0.5	1	18	1	25	0.5	96	8	...
4	1434	0	1.4	0	11	1	49	0.5	108	6	...

5 rows × 21 columns

8. Conclusion

With the help of the machine learning models, we were able to find out the different accuracy values for the different models used to predict the dataset. From the above prediction, the logistic regression was able to give the highest accuracy value. So, the logistic regression is the best suited model to predict the different mobile prices for this dataset.

9. Reference

[1] "Support Vector Machines for Classification and Regression" Author: Cortes, Corinna, and Vapnik, Vladimir Publication: Machine Learning, 1995

[2] "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting" Author: Freund, Yoav, and Schapire, Robert E. Publication: Journal of Computer and System Sciences, 1997

[3] "Random Forests" Author: Breiman, Leo Publication: Machine Learning, 2001

[4] "Gradient Boosting Machines: A Tutorial" Author: Friedman, Jerome H. Publication: Frontiers in Quantitative Finance, 2002

[5] "Long Short-Term Memory" Author: Hochreiter, Sepp, and Schmidhuber, Jürgen Publication: Neural Computation, 1997

[6] "Deep Neural Networks for Acoustic Modeling in Speech Recognition" Author: Hinton, Geoffrey E., et al. Publication: IEEE Signal Processing Magazine, 2012

[7] "XGBoost: A Scalable Tree Boosting System" Author: Chen, Tianqi, and Guestrin, Carlos Publication: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016