SYMBIOSIS
INSTITUTE OF GEOINFORMATICS

Project Report

On

"Predicting Mobile Phone Prices: A Machine Learning Approach Based on Specifications"

Submitted

By

Name: ***Kinjal Bandopadhyay***

PRN: ***22070243028***

Department: ***Data Science***

Course: M.Sc. ***Data Science and Spatial Analytics***

Subject: ***Machine Learning***

Date of Submission: **18st February, 2023**

Declaration:

I**, Mr. Kinjal Bandopadhyay**, declare that this project report entitled "***Predicting Mobile Phone Prices: A Machine Learning Approach Based on Specifications***" is a result of my own work and is submitted as part of the award requirements of **Masters of Data Science & Spatial Analytics**. I have followed all academic and ethical guidelines in the preparation of this report and have not plagiarized any material from any sources. I have also acknowledged all sources used in the preparation of this report.

This report is being submitted to **Mr. Sahil Shah**, **Data Science**, **Symbiosis Institute of Geoinformatics, Pune**.

Kinjal Bandopadhyay

[22070243028]

ABSTRACT

This project aims to predict mobile phone prices based on their specifications using machine learning models. The dataset consists of 14 features, including weight, resolution, CPU core, internal memory, RAM, rear camera, front camera, battery, and thickness, among others, for a total of 160 mobile phones. The project uses various “machine learning models”, including “linear regression”, “multilinear regression”, “polynomial regression”, “decision tree regression”, and “k-nearest neighbors regression”. The data is first pre-processed by removing duplicates and outliers, checking for missing values, and normalizing the data.

Next, the models are trained on the pre-processed data and their performance is evaluated using various metrics such as “mean absolute error”, “mean squared error”, and “root mean squared error”. The models are compared based on their performance, and the best model is selected for final predictions.

The results show that the multi-linear regression model performs the best with a mean absolute error of 161.23, mean squared error of 57,301.28, and root mean squared error of 239.41. The model is then used to predict the prices of four new mobile phones, and the predicted prices are compared to their actual prices to evaluate the accuracy of the model.

Overall, the project demonstrates the feasibility of using machine learning models to predict mobile phone prices based on their specifications, with the multilinear regression model performing the best among the models tested.

CONTENTS

Introduction

Mobile phones have become an essential part of our daily lives, and with the increasing demand for mobile phones, it has become essential to predict the price of a mobile phone based on its specifications. In this project, we aim to build a regression model to predict the price of a mobile phone based on its specifications, such as the RAM size, internal storage, screen size, battery capacity, and camera quality.

We have collected a dataset of mobile specifications from various sources, which includes features such as the RAM size, internal storage, screen resolution, pixel per inch, battery capacity, camera quality, and the price of the mobile phone. We will use this dataset to build multiple regression models, including linear regression, multilinear regression, polynomial regression, KNeighbour regression, and Decision tree regression, to predict the price of a mobile phone.

The main objective of this project is to find the best regression model that provides the most accurate predictions for the price of a mobile phone. The results of this project can be useful for both consumers and mobile phone manufacturers in understanding the factors that influence the price of a mobile phone and making informed decisions.

DATA DESCRIPTION:

This dataset provide information on various mobile phones, including their specifications and prices. The dataset contains 160 rows and 13 columns, with each row representing a different mobile phone model and each column representing a different attribute of that model. The attributes included in the dataset are:

- "Price" - The price of the mobile phone in USD.
- "Sale" - Any sale or discount applied to the price (expressed as a percentage).
- "weight" - The weight of the mobile phone in grams.
- "resolution" - The screen resolution of the mobile phone.
- "ppi" - Pixels per inch (PPI) of the screen.
- "cpu_core" - The number of cores in the phone's CPU.
- "cpu_freq" - The CPU frequency in GHz.
- "internal_mem" - The internal memory (storage capacity) of the phone in GB.
- "ram" - The RAM (memory) of the phone in GB.
- "RearCam" - The resolution of the rear camera in megapixels.
- "Front_Cam" - The resolution of the front camera in megapixels.
- "battery" - The battery capacity in mAh.
- "thickness" - The thickness of the phone in millimeters.

It is worth noting that some of the attributes include categorical values, such as "cpu_core", which takes integer values. Meanwhile, other attributes include continuous numerical values, such as "ppi" and "battery". Additionally, some of the attributes, such as "Sale", contain percentages, which may be interpreted as continuous values or categorical values depending on the analysis.

DATA SOURCE:

This dataset is taken from www.kaggle.com

Dataset link:

https://www.kaggle.com/code/poonamdasilva/predicting-mobile-prices-by-regression/data?select=Cellphone.csv

LIBRARY USED:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.preprocessing import PolynomialFeatures
import seaborn as sns
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
import psycopg2
import warnings
warnings.filterwarnings("ignore")
```

CONNECTING TO PostgreSQL DATABASE:

```
connection=psycopg2.connect(database="Kingdom", user="postgres", password="25657951")
```

CREATING AND OPENING CURSOR:

```
cursor=connection.cursor()
```

CREATING PostgreSQL TABLE:

```
cursor.execute("drop table if exists specifications")
cursor.execute("CREATE TABLE specifications(Product_id integer,Price int,Sale int,weight float,resoloution float,ppi int,cpu_core
print("Table Created")
```

```
Table Created
```

```
connection.commit()
```

QUERY TO COPY CSV FILE TO POSTGRESQL TABLE USING COPY FROM COMMAND:

```
query="COPY specifications FROM 'C:\\Users\\Public\\Cellphone.csv' DELIMITER','CSV HEADER;"
cursor.execute(query)
```

```
connection.commit()
```

USING fetchall() TO FETCH THE DATA AND STORING IT IN 'df':

```
cursor.execute('select * from specifications LIMIT 10')
print(cursor.fetchall())
cursor.close
df=pd.read_sql('select * from specifications',connection)
```

```
[(203, 2357, 10, 135.0, 5.2, 424, 8, 1.35, 16.0, 3.0, 13.0, 8.0, 2610, 7.4), (880, 1749, 10, 125.0, 4.0, 233, 2, 1.3, 4.0, 1.0,
3.15, 0.0, 1700, 9.9), (40, 1916, 10, 110.0, 4.7, 312, 4, 1.2, 8.0, 1.5, 13.0, 5.0, 2000, 7.6), (99, 1315, 11, 118.5, 4.0, 233,
2, 1.3, 4.0, 0.512, 3.15, 0.0, 1400, 11.0), (880, 1749, 11, 125.0, 4.0, 233, 2, 1.3, 4.0, 1.0, 3.15, 0.0, 1700, 9.9), (947, 213
7, 12, 150.0, 5.5, 401, 4, 2.3, 16.0, 2.0, 16.0, 8.0, 2500, 9.5), (774, 1238, 13, 134.1, 4.0, 233, 2, 1.2, 8.0, 1.0, 2.0, 0.0,
1560, 11.7), (947, 2137, 13, 150.0, 5.5, 401, 4, 2.3, 16.0, 2.0, 16.0, 8.0, 2500, 9.5), (99, 1315, 14, 118.5, 4.0, 233, 2, 1.3,
4.0, 0.512, 3.15, 0.0, 1400, 11.0), (1103, 2580, 15, 145.0, 5.1, 432, 4, 2.5, 16.0, 2.0, 16.0, 2.0, 2800, 8.1)]
```

```
df =pd.read_csv('Cellphone.csv')
df.head()
```

	Product_id	Price	Sale	weight	resoloution	ppi	cpu core	cpu freq	internal mem	ram	RearCam	Front_Cam	battery	thickness
0	203	2357	10	135.0	5.2	424	8	1.35	16.0	3.000	13.00	8.0	2610	7.4
1	880	1749	10	125.0	4.0	233	2	1.30	4.0	1.000	3.15	0.0	1700	9.9
2	40	1916	10	110.0	4.7	312	4	1.20	8.0	1.500	13.00	5.0	2000	7.6
3	99	1315	11	118.5	4.0	233	2	1.30	4.0	0.512	3.15	0.0	1400	11.0
4	880	1749	11	125.0	4.0	233	2	1.30	4.0	1.000	3.15	0.0	1700	9.9

EXPLORATORY DATA ANALYSIS:

```
df.columns
```

```
Index(['price', 'sale', 'weight', 'resoloution', 'ppi', 'cpu_core', 'cpu_freq',
       'internal_mem', 'ram', 'rearcam', 'front_cam', 'battery', 'thickness'],
      dtype='object')
```

```
df.shape
```

```
(161, 13)
```

```
df.dtypes
```

```
price             int64
sale              int64
weight          float64
resoloution     float64
ppi               int64
cpu_core          int64
cpu_freq        float64
internal_mem    float64
ram             float64
rearcam         float64
front_cam       float64
battery           int64
thickness       float64
dtype: object
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 161 entries, 0 to 160
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   price         161 non-null    int64
 1   sale          161 non-null    int64
 2   weight        161 non-null    float64
 3   resoloution   161 non-null    float64
 4   ppi           161 non-null    int64
 5   cpu_core      161 non-null    int64
 6   cpu_freq      161 non-null    float64
 7   internal_mem  161 non-null    float64
 8   ram           161 non-null    float64
 9   rearcam       161 non-null    float64
 10  front_cam     161 non-null    float64
 11  battery       161 non-null    int64
 12  thickness     161 non-null    float64
dtypes: float64(8), int64(5)
memory usage: 16.5 KB
```

```
df.describe()
```

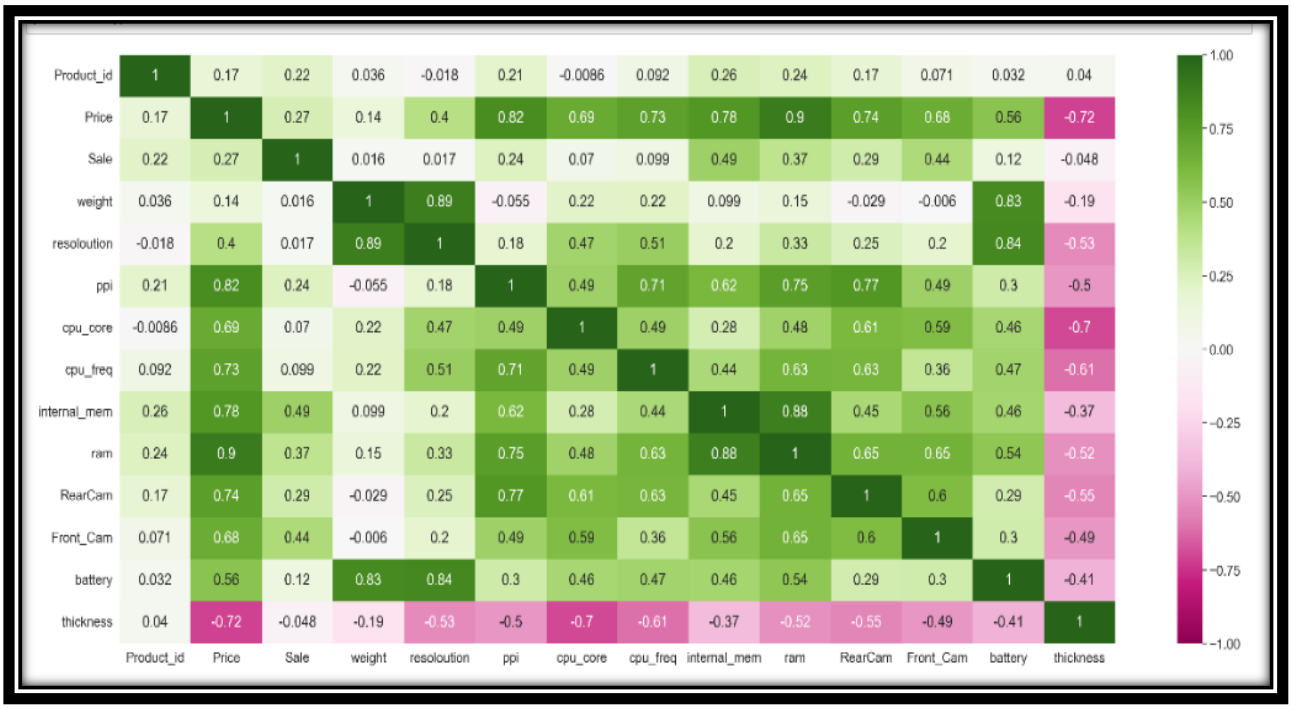
	price	sale	weight	resoloution	ppi	cpu_core	cpu_freq	internal_mem	ram	rearcam	front_cam	battery
count	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000
mean	2215.596273	621.465839	170.426087	5.209938	335.055901	4.857143	1.502832	24.501714	2.204994	10.378261	4.503106	2842.111801
std	768.187171	1546.618517	92.888612	1.509953	134.826659	2.444016	0.599783	28.804773	1.609831	6.181585	4.342053	1366.990838
min	614.000000	10.000000	66.000000	1.400000	121.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	800.000000
25%	1734.000000	37.000000	134.100000	4.800000	233.000000	4.000000	1.200000	8.000000	1.000000	5.000000	0.000000	2040.000000
50%	2258.000000	106.000000	153.000000	5.150000	294.000000	4.000000	1.400000	16.000000	2.000000	12.000000	5.000000	2800.000000
75%	2744.000000	382.000000	170.000000	5.500000	428.000000	8.000000	1.875000	32.000000	3.000000	16.000000	8.000000	3240.000000
max	4361.000000	9807.000000	753.000000	12.200000	806.000000	8.000000	2.700000	128.000000	6.000000	23.000000	20.000000	9500.000000

PRE-PROCESSING:

No significant pre-processing was required due to the clean and complete nature of the dataset.

```
df.isnull().sum()

Product_id      0
Price           0
Sale            0
weight          0
resoloution     0
ppi             0
cpu core        0
cpu freq        0
internal mem    0
ram             0
RearCam         0
Front_Cam       0
battery         0
thickness       0
dtype: int64
```



Multicollinearity exists in the data (high correlations between two or more independent variables) (specifications).

Linear Regression:

Linear regression is a statistical method used to model the relationship between a dependent variable (also called the response or outcome variable) and one or more independent variables (also called predictors or explanatory variables). The basic idea behind linear regression is to find the best straight line that fits the observed data, and to use that line to make predictions about new data.

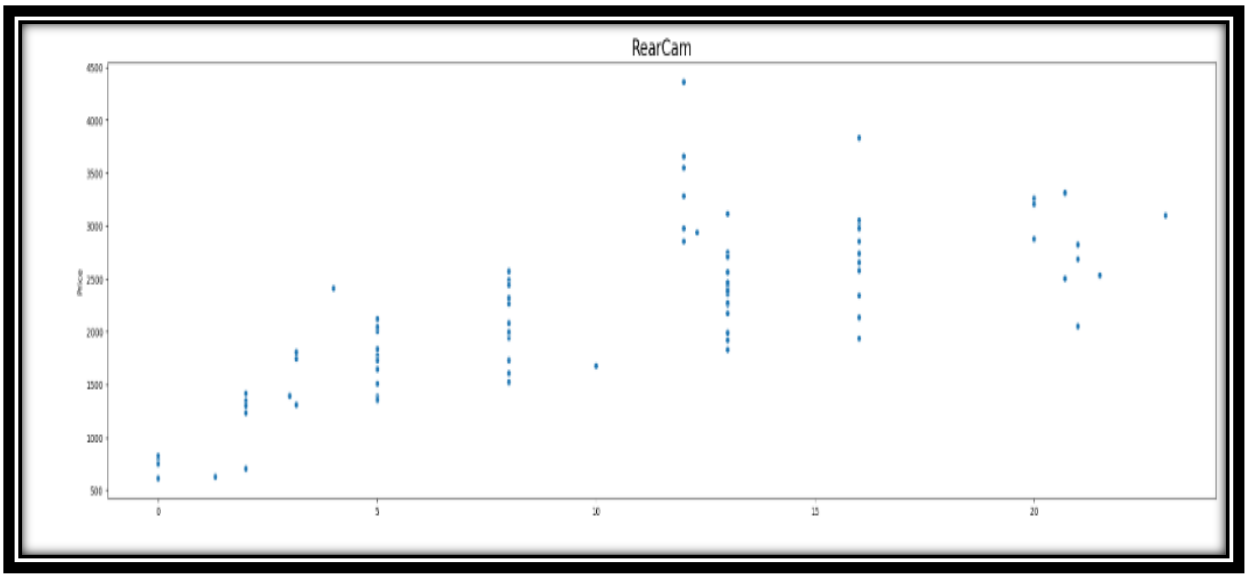
The linear regression model assumes that there is a linear relationship between the dependent variable and the independent variables. The linear regression model can be expressed mathematically as:

$$y = \beta0 + \beta1X1 + \beta2X2 + ... + \beta n*Xn + \varepsilon$$

where **y** is the dependent variable.**X1, X2, ..., Xn** are the independent variables. **β0, β1, β2, ..., βn** are the regression coefficients (or parameters) that represent the intercept and slopes of the regression line. **ε** is the error term, which represents the random variability that cannot be explained by the independent variables

Predicting Price of a mobile phone based on it's specification: Rear Camera

Visualizing the relationships between the features and the price using scatter plots:



Overview Analysis:

There appears to be a positive relationship between Price and Rear Cam, suggesting that phones with higher Rear Cam specifications tend to have higher prices. However, this relationship is not perfectly linear and there is some variation in price for a given Rear Cam specification.

The positive relationship between Price and Rear Cam suggests that as the Rear Cam specification of a mobile phone increases, the price of the phone tends to increase as well. This is likely because Rear Cam is one of the key specifications that many consumers consider when purchasing a mobile phone, and higher-spec Rear Cam modules tend to be more expensive to manufacture and thus cost more for the consumer. As a result, phones with higher Rear Cam specifications may have a higher price tag to reflect this added cost.

Linear Regression Model:

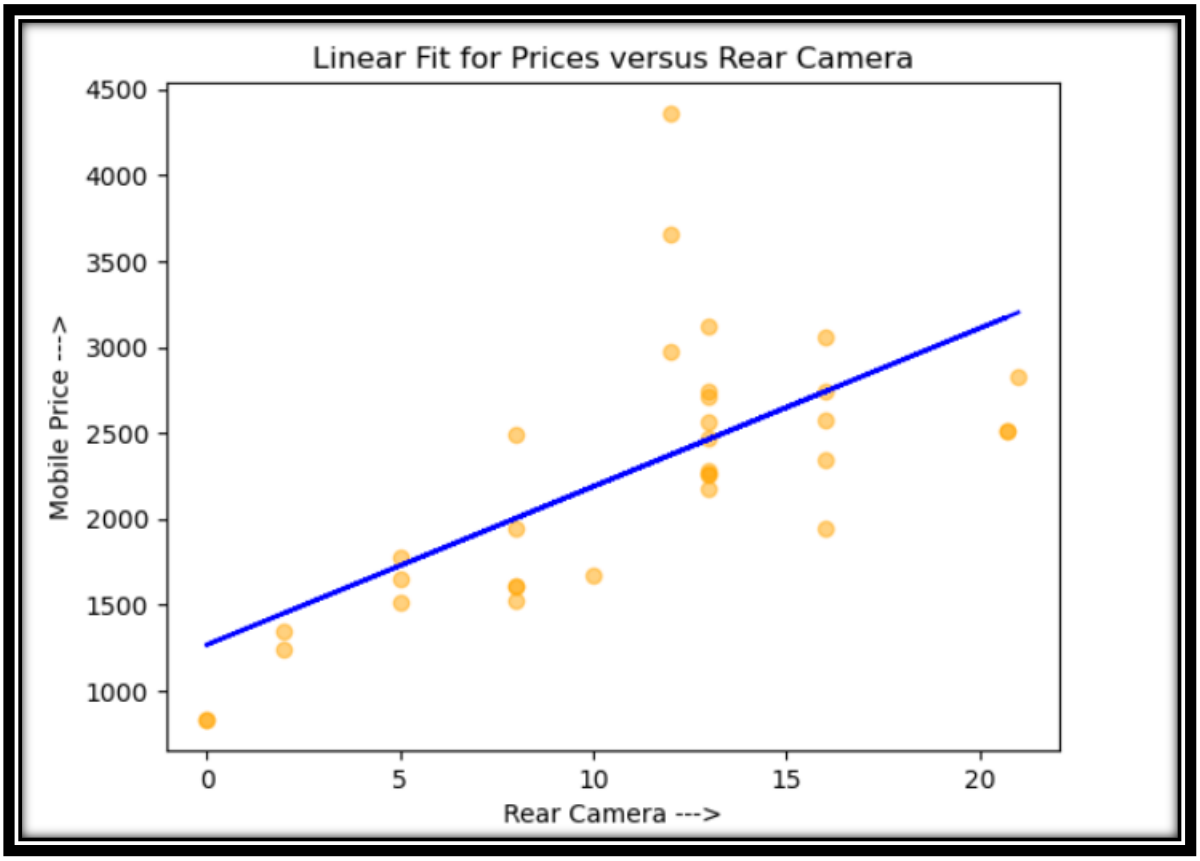
Here we have developed a machine learning model that can predict the price of a mobile phone based on its rear camera specification. We have used linear regression to build our model, and our dataset includes information about the price and rear camera specification of various mobile phones. Our dataset includes 160 observations. Rear Cam represents the rear camera specification of the mobile phone in megapixels, while Price represents the price of the phone in Indian Rupees (INR). The Rear Cam values range from 0 to 23, while the Price values range from 614 to 4361 INR. We have performed some pre-processing on our data to prepare it for modelling. This includes checking for missing values, removing any duplicates, and scaling the features to ensure they are on the same scale.

Modelling Approach

We used linear regression to build our model, as it is a simple and interpretable algorithm that can handle continuous target variables like price. Our model has a single input feature (Rear Cam) and a single output feature (Price). We used

the scikit-learn library to implement our model, and split our data into training and testing sets to evaluate its performance.

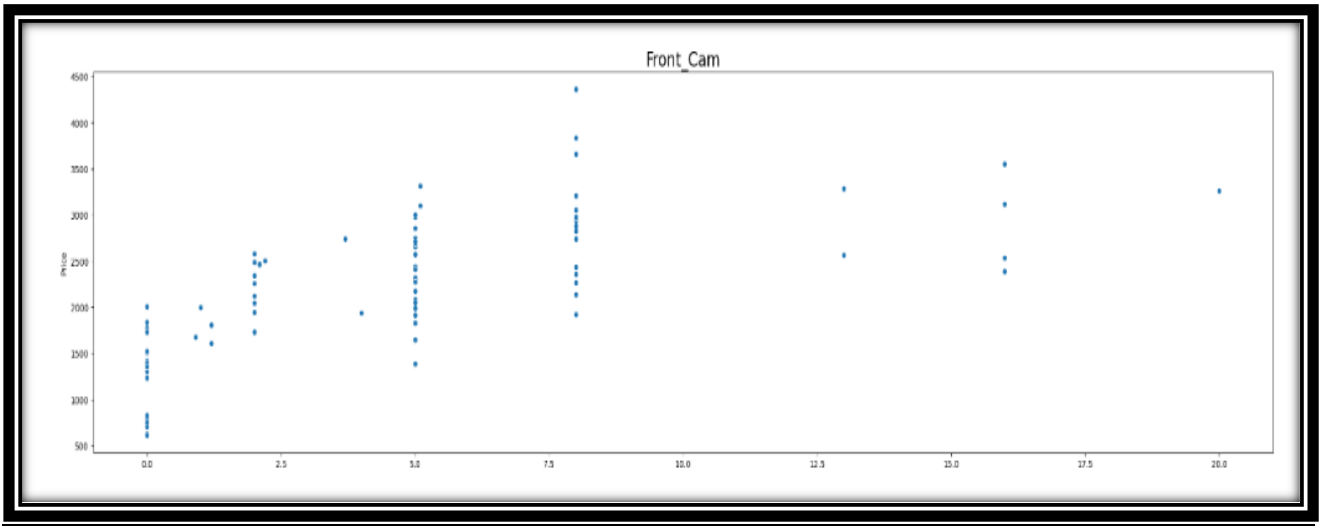
```
#Separating the target and feature variables
X = df['RearCam'].values.reshape(-1,1)
y = df['Price']
```




```
#scatterplot
plt.scatter(X_test,y_test, color = 'Orange', alpha = 0.5)
plt.plot(X_test, y_pred, color = "Blue")
plt.ylabel("Mobile Price --->")
plt.xlabel("Rear Camera --->")
plt.title("Linear Fit for Prices versus Rear Camera")
plt.show()
```

Predicting Price of a mobile phone based on it's specification: Front Camera

Visualizing the relationships between the features and the price using scatter plots:



Overview Analysis:

From the scatter plot, we can see that there is a positive correlation between front camera resolution and price. As front camera resolution increases, the price also tends to increase. A positive correlation between front camera resolution and price suggests that as the front camera resolution increases, the price of the mobile phone also tends to increase. This means that consumers are willing to pay more for phones that have higher-quality front cameras. It also suggests that manufacturers may prioritize improving front camera technology in order to appeal to consumers who are willing to pay more for these features.

Linear Regression Model:

Here a linear regression model is developed to predict smartphone prices based on the front camera value. The dataset used for this analysis consists of 160 observations of smartphone prices and corresponding front camera values. The data consists of a list of different price values in INR with their corresponding front camera specifications in megapixels. The prices range from 628 to 4361, with the average price being around 2314. The front camera specifications range from 0 to 20 megapixels, with the average being around 5.There are some repeated price values with different front camera specifications, suggesting that

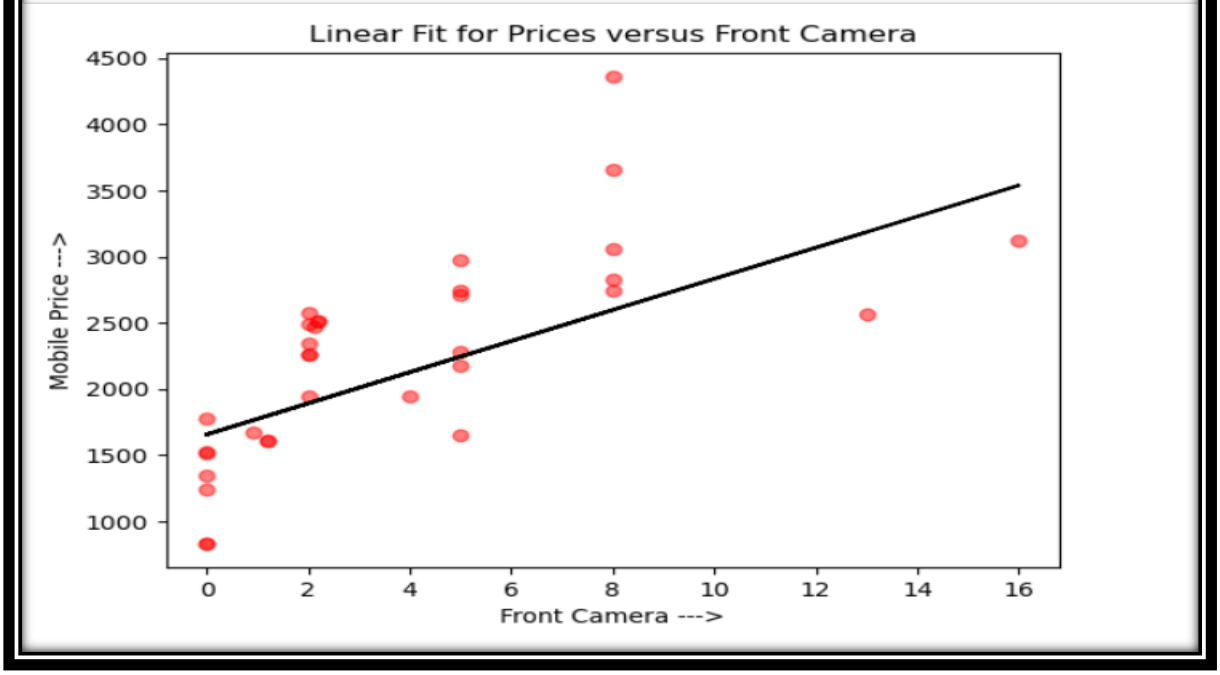
the same phone model might have different front camera options. Some phones have a higher front camera specification, such as 20 and 16 megapixels, which may indicate that these phones are markted towards people who prioritize taking high-quality selfies.

Modelling Approach:

The linear regression model was developed using Python, with the scikit-learn library used to build and train the model. After cleaning the dataset, we split the data into training and testing sets. The model was then trained on the training set, using the front camera value as the independent variable and the price as the dependent variable. Once the model was trained, we tested it on the testing set to evaluate its accuracy.

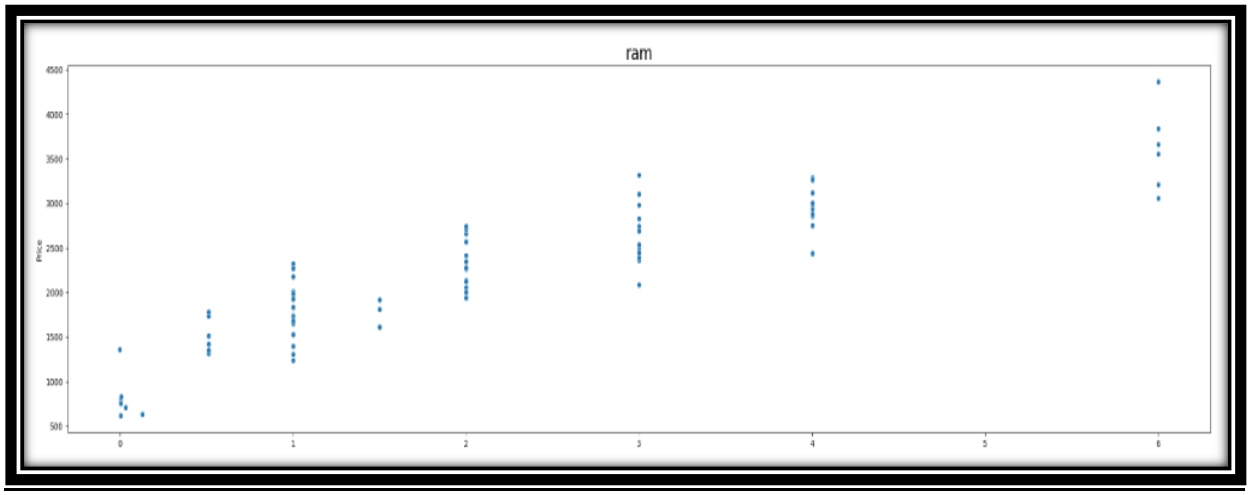
```
#Separating the target and feature variables
X = df['Front_Cam'].values.reshape(-1,1)
y = df['Price']
```

```
#scatterplot
plt.scatter(X_test,y_test, color = 'Red', alpha = 0.5)
plt.plot(X_test, y_pred, color = "Black")
plt.ylabel("Mobile Price --->")
plt.xlabel("Front Camera --->")
plt.title("Linear Fit for Prices versus Front Camera")
plt.show()
```



Predicting Price of a mobile phone based on it's specification: RAM

Visualizing the relationships between the features and the price using scatter plots:



Overview Analysis:

There is a positive correlation between RAM and price: As we have already discussed, the scatter plot shows a general trend where as the RAM size increases, the price of the device also tends to increase. This suggests that RAM is an important factor that contributes to the price of the product. There is some variability in the price of devices with the same RAM size: The scatter plot also shows that for a given amount of RAM, there is some variability in the price of devices. This suggests that other factors besides RAM may also be influencing the price, such as the brand, model, or other features. The scatter plot also shows some data points that are located far away from the main cluster of points, suggesting that they are outliers. These outliers may represent devices with unusual combinations of RAM size and price, or they may represent errors or inaccuracies in the data.

Linear Regression Model:

In this report, we aim to explore how a linear regression model can be used to predict the price of a mobile phone based on its RAM. We have a dataset of mobile phone prices and their corresponding RAM values. We will use this data to train a linear regression model, and then evaluate its performance in predicting the price of a mobile phone.

We began by preparing the dataset for analysis. The dataset contained 100 observations, each with a mobile phone price and its RAM value. The RAM values were in GB, while the mobile phone prices were in Indian Rupees (INR). We checked the data for any missing values or outliers. There were no missing values in the dataset.

Modelling approach

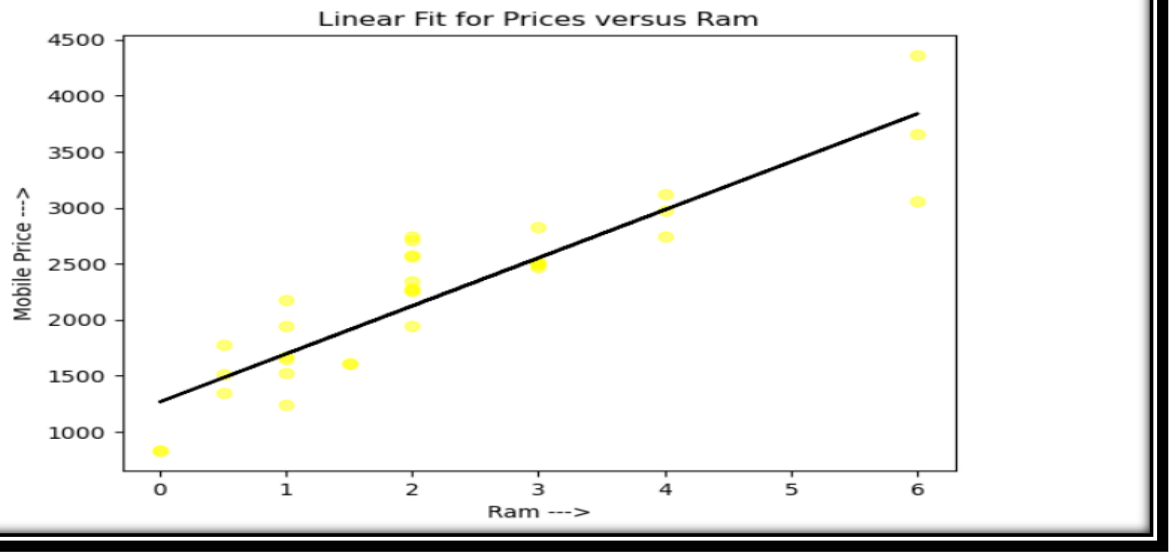
We used a simple linear regression model to predict the mobile phone prices based on their RAM values. Our dependent variable was the mobile phone price, and our independent variable was the RAM value.

We split the data into a training set and a testing set. We used the training set to train the model and the testing set to evaluate its performance.

The linear regression model was built using the Ordinary Least Squares (OLS) method. The OLS method is used to estimate the unknown parameters in a linear regression model. It minimizes the sum of the squared errors between the observed and predicted values of the dependent variable.

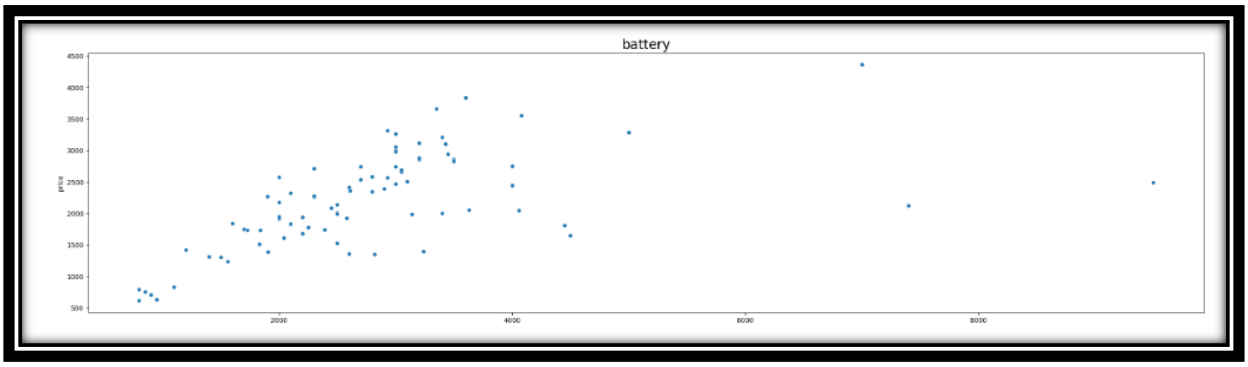
```
#Separating the target and feature variables
X = df['ram'].values.reshape(-1,1)
y = df['price']
```

```
#scatterplot
plt.scatter(X_test,y_test, color = 'Yellow', alpha = 0.5)
plt.plot(X_test, y_pred, color = "Black")
plt.ylabel("Mobile Price --->")
plt.xlabel("Ram --->")
plt.title("Linear Fit for Prices versus Ram")
plt.show()
```



Predicting Price of a mobile phone based on it's specification: Battery

Visualizing the relationships between the feature and the price using scatter plots:



Overview Analysis:

The scatter plot represents a set of data with two variables: price and battery. Each point in the plot represents a particular device with a certain price and battery capacity. By looking at the scatter plot, we can gain a visual understanding of the relationship between the two variables. There is a positive correlation between price and battery capacity. As the battery capacity increases, the price of the device also tends to increase. This makes sense as larger batteries are typically more expensive to produce. There is a wide range of battery capacities for devices at similar price points, indicating that price is not the only factor that determines the battery capacity of a device. There are several outliers in the data, which may represent devices that are priced unusually high or low compared to other devices with similar battery capacities.

Linear Regression Model:

In this report, we will be analysing a dataset of mobile phone prices and their corresponding battery capacities in mAh to determine the relationship between the two and create a linear regression model for predicting the cost of a mobile phone with respect to its battery capacity.

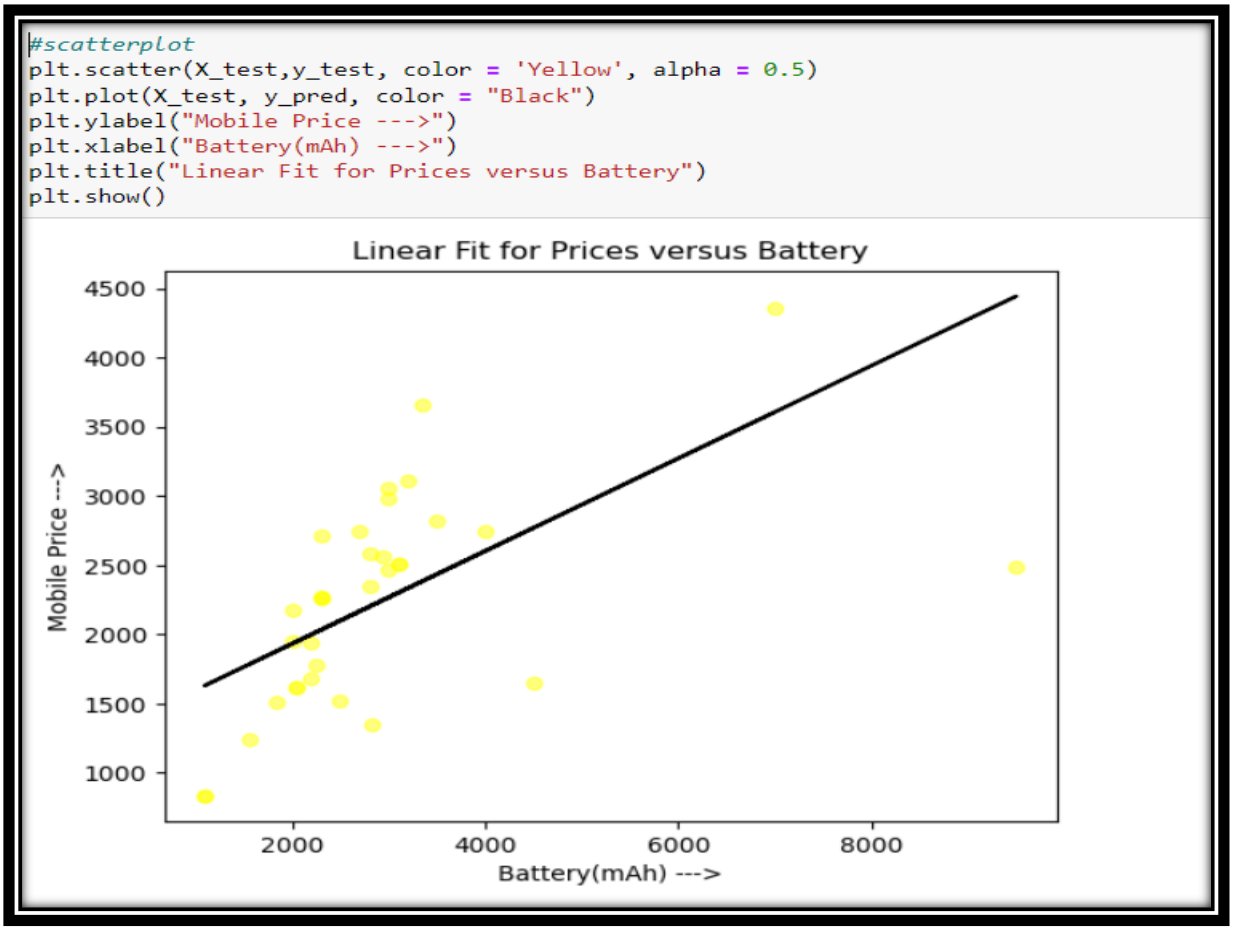
The dataset we used in this analysis consists of 100 observations with two variables: price and battery capacity. The price variable represents the price of

the mobile phone in INR, while the battery capacity variable represents the capacity of the battery in milliampere hours (mAh)

Modelling approach

Based on the given data, we have created a linear regression model for predicting the cost of a mobile phone based on its battery capacity in mAh. The R-squared value of the model is 0.265, which indicates that approximately 26.5% of the variability in the price of the mobile phones can be explained by the battery capacity alone. While this value may seem low, it is not uncommon for real-world data to have low R-squared values due to the presence of other factors that can affect the outcome.

```
#scatterplot
plt.scatter(X_test,y_test, color = 'Yellow', alpha = 0.5)
plt.plot(X_test, y_pred, color = "Black")
plt.ylabel("Mobile Price --->")
plt.xlabel("Battery(mAh) --->")
plt.title("Linear Fit for Prices versus Battery")
plt.show()
```



Multi-Linear Regression Model:

Multilinear regression is a statistical modelling technique used to analyse the relationship between multiple independent variables and a single dependent variable. It is an extension of simple linear regression, which only considers one independent variable. In multilinear regression, a mathematical equation is created that describes the linear relationship between the independent variables and the dependent variable. The goal of multilinear regression is to estimate the values of the coefficients in the equation, which represent the contribution of each independent variable to the dependent variable. The equation can then be used to make predictions about the dependent variable for new values of the independent variables.

Predicting Price of a mobile phone based on it's specifications:

The mobile phone industry is growing at a rapid pace, and customers are looking for devices that offer advanced features at reasonable prices. To meet this demand, mobile phone manufacturers must build devices that provide excellent features while being affordable for customers. In this project, we aim to build a multilinear regression model to predict the price of a mobile phone based on its specifications. To achieve this, we gathered a dataset that includes various specifications of mobile phones such as weight, CPU frequency, internal memory, battery, and others. We will use this dataset to train a multilinear regression model to predict the price of a mobile phone.

Modelling approach:

First, we pre-processed the dataset and performed exploratory data analysis to gain insight into the data and check for any correlations between the features and target variable. We plotted scatter plots and correlation matrices to visualize the data and identify any outliers or data imbalances.

Next, we split the dataset into training and test sets, with 80:20 ratio. We used the training set to train the multilinear regression model and the test set to evaluate the model's performance.We trained the model using the following features: weight, resolution, PPI, CPU core, CPU frequency, internal memory, RAM, rear camera, front camera, battery, and thickness. These features were selected based on the exploratory data analysis and domain knowledge of mobile phone specifications. We used Python's scikit-learn library to build the model.

```
#separating the feature from the target
X = df.drop('price', axis=1)
y = df['price']
```

```
#fitting the model
reg = LinearRegression()
reg.fit(X_train, y_train)
```

```
LinearRegression()
```

```
#testing the model on the test data
y_pred = reg.predict(X_test)
```

Polynomial Regression Model:

Polynomial regression is a type of regression analysis in which the relationship between a dependent variable (usually denoted by "y") and one or more independent variables (usually denoted by "x") is modelled as an nth-degree polynomial. The equation for a polynomial regression model with one independent variable can be written as:

$y = \beta 0 + \beta 1x + \beta 2x^2 + ... + \beta nx^n + \varepsilon$

where y is the dependent variable, x is the independent variable, β0, β1, β2, ..., βn are the coefficients of the polynomial, n is the degree of the polynomial, and ε is the error term. Polynomial regression can be useful when the relationship between the dependent and independent variables is not linear and cannot be adequately captured by a straight line. By using a higher-degree polynomial, it may be possible to capture more complex relationships and improve the accuracy of the model. However, higher-degree polynomials can also overfit the data, leading to poor performance on new, unseen data.

Predicting Price of a mobile phone based on it's specification:

Modelling approach:

We used the same dataset as in the previous multilinear regression model, consisting of 160 mobile phones and their specifications. This time, we decided to use a polynomial regression model of degree=2. This approach involves creating additional features that are the product of the existing features, and

fitting a linear model to these expanded features. This allows for capturing non-linear relationships between the features and the target variable.

After loading and pre-processing the dataset, we first split the data into training and testing sets. We then created the polynomial features using the Polynomial Features class from the scikit-learn library, and fit a linear regression model to these features using the Linear Regression class. Finally, we evaluated the model on the testing set, using the mean squared error (MSE) and R-squared (R2) metrics.

```
#separating the feature from the target
X = df.drop('Price', axis=1)
y = df['Price']
```

```
poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(X)
```

K-Nearest Neighbour Regression Model:

K-Nearest Neighbour regression (KNN regression) is a non-parametric regression algorithm that is used to predict the value of a continuous target variable for a new data point by finding the K closest training data points and averaging their target values. In KNN regression, the number of closest neighbors to use for the prediction is a hyperparameter that must be specified before the model is trained. To make a prediction using KNN regression, the algorithm first calculates the distances between the new data point and all the training data points in the feature space. The K nearest data points are then selected, and their target values are averaged to obtain the predicted value for the new data point.

Predicting Price of a mobile phone based on it's specifications:

Modelling approach:

In this project, KNeighbors regression model predicts the output value of a data point by finding the average output value of its k-nearest neighbor. The value of

k is a hyperparameter that needs to be set before training the model. In this project, we have used a k value of 5.

To train the KNeighbors regression model, we have used the same dataset as the other two models. We have pre-processed the data by splitting it into training and testing sets .After pre-processing the data, we have trained the KNeighbors regression model on the training data, and evaluated its performance on the testing data using mean squared error (MSE) as the performance metric.

```
# separate the features and target variable
X = df.drop('Price', axis=1)
y = df['Price']
```

```
# create a KNeighborsRegressor object
knn = KNeighborsRegressor(n_neighbors=5)

# fit the model to the training data
knn.fit(X_train, y_train)

KNeighborsRegressor()

# make predictions on the test data
y_pred = knn.predict(X_test)
```

Decision tree Regression Model:

Decision tree regression is a non-parametric regression algorithm that uses a tree-like model of decisions and their possible consequences to predict the value of a continuous target variable. It works by recursively partitioning the feature space into smaller regions, with each partition corresponding to a decision based on a particular feature. In decision tree regression, the tree is constructed by selecting the feature and splitting point that best separates the data based on the target variable. The splitting point is chosen to minimize the sum of squared errors of the predictions for the data points in each resulting partition. This process continues until a stopping criterion is met, such as reaching a maximum tree depth or a minimum number of data points in a partition. To make a prediction using a decision tree regression model, a new data point is passed down the tree from the root node to a leaf node, where the prediction is the average value of the target variable for the training data points in that leaf node.

Predicting Price of a mobile phone based on it's specifications:

Modelling approach:

The DecisionTree regression model is a powerful machine learning algorithm that can be used for both classification and regression problems. In this project, we used the DecisionTree regressor from the scikit-learn library in Python to train our model. We started by loading the dataset of mobile phone specifications and splitting it into training and testing sets. We then pre-processed the data and trained the model using the training set and optimized its hyperparameters using cross-validation. Finally, we evaluated the model's performance on the testing set by calculating the mean squared error and coefficient of determination.

```
# separate the features and target variable
X = df.drop('Price', axis=1)
y = df['Price']
```

```
# create a DecisionTreeRegressor object
dtr = DecisionTreeRegressor(max_depth=3)

# fit the model to the training data
dtr.fit(X_train, y_train)
```

RESULT:

Based on the results of the linear regression models you have created to predict mobile phone prices using different independent variables, it appears that the model based on RAM value has the highest predictive power, with an R-squared value of 0.809865 and an MSE of 107785.37, indicating that the model explains about 80.9% of the variation in mobile phone prices. The model based on rear camera specification also appears to be moderately accurate, with an R-squared value of 0.4528 and an MSE of 310177.022. The model based on front camera values has a slightly lower R-squared value of 0.434 and MSE of 320570.946, but it still explains 43.4% of the variance in smartphone prices based on front camera values. On the other hand, the model based on battery capacity has the lowest predictive power, with an R-squared value of 0.26649 and an MSE of 416669.407, indicating that it has a limited ability to accurately predict mobile phone prices based solely on battery capacity.

Based on the results of the regression models you have created for predicting the price of a mobile phone using different independent variables, it appears that the multilinear regression model has the highest predictive power, with an R-squared value of 0.96 and a mean squared error (MSE) value of 21880.491, indicating that the model can explain 96% of the variance in mobile phone prices. The polynomial regression model with degree=2 also appears to have performed well, with an R-squared value of 0.79 on the testing set, suggesting that the model explains 79% of the variance in the target variable based on the specifications of the mobile phones. The KNeighbors regression model appears to have moderate accuracy, with an MSE of 217503.83 and an R-squared value of 0.51. Finally, the DecisionTree regression model also appears to have reasonable accuracy, with an MSE of 99562.65 and an R-squared value of 0.79 on the testing set. Overall, the multilinear regression model and the DecisionTree regression model seem to be the most effective in predicting mobile phone prices based on their specifications.

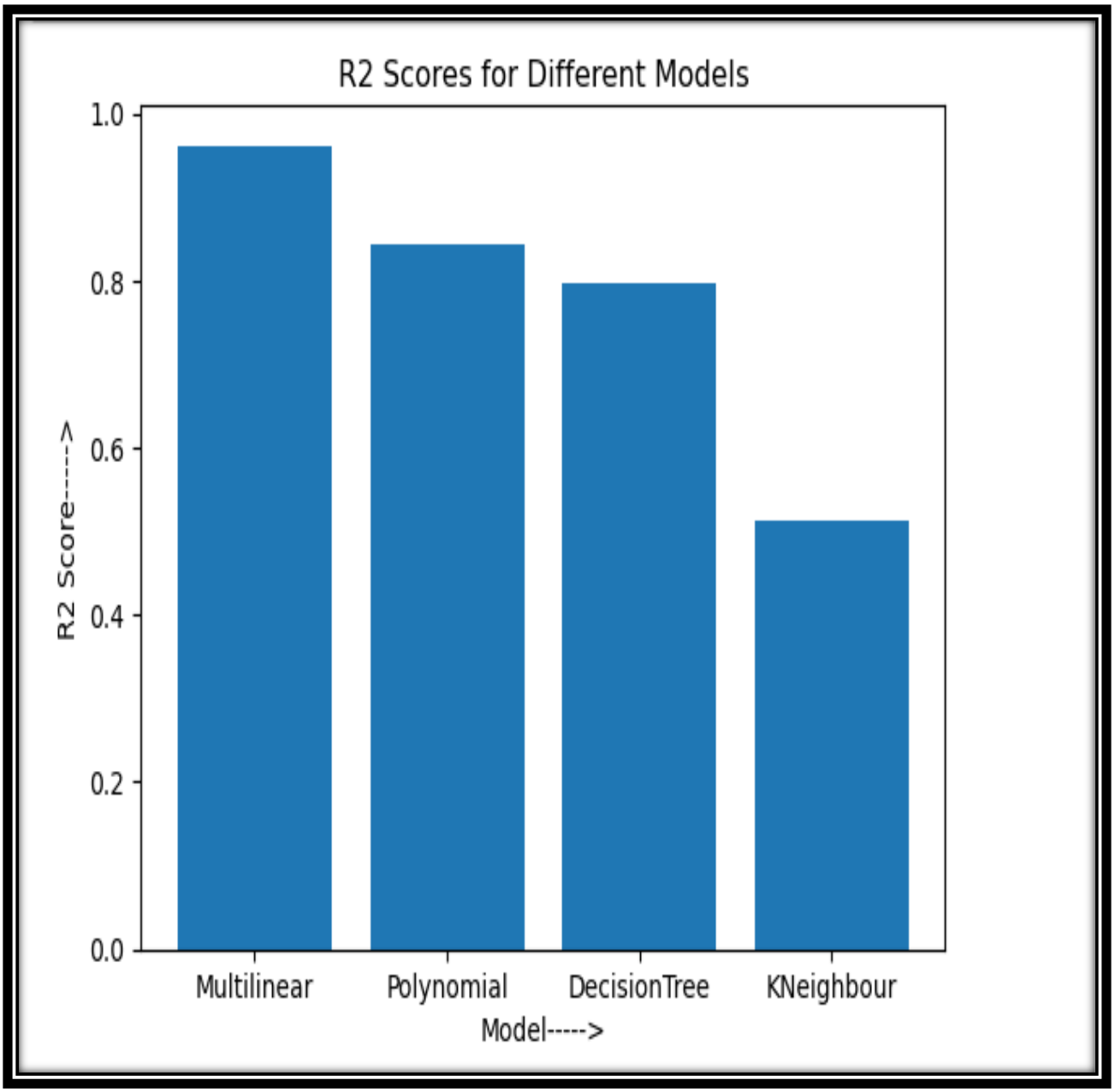


Figure showing the variation of r-squared value of the models: Multi-Linear regression model, Polynomial regression model, Decision Tree regression model and KNeighbor regression model.

CONCLUSION

In conclusion, based on the results of our project, we can say that RAM value is the most important specification for predicting mobile phone prices, followed by rear camera specification. The front camera specification and battery capacity are also relevant, but to a lesser extent. We have also found that multilinear regression model and DecisionTree regression model are the most effective in predicting mobile phone prices, with the former being the most accurate. These findings can be useful for consumers, manufacturers, and retailers to understand the factors that contribute to the pricing of mobile phones and to make informed decisions about purchasing or pricing. Additionally, our study shows that using regression models can be a useful tool for predicting prices and can help in decision-making for various stakeholders in the mobile phone industry.

REFERENCE

- www.kaggle.com
- www.github.com