Faculty of Engineering & Technology			
Ramaiah University of Applied Sciences			
Department	Computer Science and Engineering	**Programme**	B. Tech.
Semester/Batch	6th/2020		
Course Code	20AIC204A	**Course Title**	**Principles of Artificial Intelligence**
Course Leader	P.PadmaPriyaDharishini, Dr.Jyothi, Nithin Rao R.		

Assignment			
Name of Student		Register No	

Sections		Marking Scheme	Max Marks	First Examiner Marks	Second Examiner Marks
B	1.1	Data pre-processing and exploration	5		
	1.2	Feature engineering and selection	5		
	1.3	Model selection and justification	5		
	1.4	Model training and evaluation	5		
	1.5	Model optimization	5		
		B.1 Max Marks	**25**		
		Total Assignment Marks	**25**		

Course Marks Tabulation				
Component- CET B Assignment	**First Examiner**	**Remarks**	**Second Examiner**	**Remarks**
B				
Marks (out of 25)				
Signature of First Examiner				Signature of Second Examiner

Please note:

1. Documental evidence for all the components/parts of the assessment such as the reports, photographs, laboratory exam / tool tests are required to be attached to the assignment report in a proper order.

RAMAIAH UNIVERSITY OF APPLIED SCIENCES

2. The First Examiner is required to mark the comments in RED ink and the Second Examiner's comments should be in GREEN ink.
3. The marks for all the questions of the assignment have to be written only in the **Component – CET B: Assignment** table.
4. If the variation between the marks awarded by the first examiner and the second examiner lies within +/- 3 marks, then the marks allotted by the first examiner is considered to be final. If the variation is more than +/- 3 marks then both the examiners should resolve the issue in consultation with the Chairman BoE.

Assignment– 1

Instructions to students:

1. The assignment consists of 1 question.
2. Maximum marks is 25.
3. The assignment has to be neatly word processed as per the prescribed format.
4. The maximum number of pages should be restricted to 8.
5. Restrict your report to a maximum of 6 pages.
6. The printed assignment must be submitted to the subject leader.
7. **Submission Date: 17th April 2023**
8. **Submission after the due date is not permitted.**
9. **IMPORTANT**: It is essential that all the sources used in preparation of the assignment must be suitably referenced in the text.
10. Marks will be awarded only to the sections and subsections clearly indicated as per the problem statement/exercise/question

Assignment **25 Marks**

In this Assignment, students are required to design and develop a project on Machine Learning. The project can be focused on either a regression or classification problem, where you will be predicting a target variable based on various features. You are free to use any appropriate machine learning algorithm for this problem, but you must justify your choice of algorithm. You must use a publicly available dataset for this assignment.

You must perform the following tasks:

- Data exploration and pre-processing: Perform data cleaning, visualization, and pre-processing.
- Feature engineering: Create new features if needed and perform feature selection if necessary.
- Model selection: Choose an appropriate algorithm for the problem and justify your choice.
- Model training and evaluation: Train the model on the dataset and evaluate its performance using appropriate metrics.
- Model optimization: Optimize the model by tuning its hyper-parameters and/or using regularization techniques.

You must submit the following:

- Code, documentation, and explanations.
- A report (IEEE format) summarizing your approach, results, and conclusions.

LAPTOP PRICE PREDICTION USING AI

DEEPTHI KOLHAR
dept of computer science
Ramaiah University
Benglore,India
deepthibk462@gmail.com

ADHYA GANESH
dept of computer science
Ramaiah University
Benglore,India
adhyaganeshgk@gmail.com

KEERTHANA R J
dept of computer science
Ramaiah University
Benglore,India
rjkeerthana23@gmail.com

***Abstract*—This paper explores the use of machine learning algorithms for analyzing and predicting the price of the laptop. We discuss the advantages and limitations of various algorithms and apply them to laptop data to identify key factors that contribute to the price and pricing of the laptop by companies. Additionally, we use machine learning algorithms such as random forest and support vector machines to predict match outcomes and provide recommendations for the buys based on the analysis.**

The results of our study suggest that machine learning can provide valuable insights into laptop performance and offer a powerful tool for decision-making in buying or pricing the laptops. However, the accuracy of the predictions is highly dependent on the quality of the data used, and further research is needed to improve the models' performance.

.

I. INTRODUCTION

Several different factors can affect laptop computer prices. These factors include the brand of computer and the number of options and add-ons included in the computer package. In addition, the amount of memory and the speed of the processor can also affect pricing. Though less common, some consumers spend additional money to purchase a computer based on the overall “look” and design of the system.
In many cases, name brand computers are more expensive than generic versions. This price increase often has more to do with name recognition than any actual superiority of the product. One major difference between name brand and generic systems is that in most cases, name brand computers offer better warranties than generic versions. Having the option of returning a computer that is malfunctioning is often enough of an incentive to encourage many consumers to spend more money.
Functionality is an important factor in determining laptop computer prices. A computer with more memory often performs better for a longer time than a computer with less memory. In addition, hard drive space is also crucial, and the size of the hard drive usually affects pricing. Many consumers may also look for digital video drivers and other types of recording devices that may affect the laptop computer prices. Most computers come with some software pre-installed. In most cases, the more software that is installed on a computer, the more expensive it is. This is especially true if the installed programs are from well-established and recognizable software publishers. Those considering purchasing a new laptop computer should be aware that many of the pre-installed programs may be trial versions only, and will expire within a

certain time period. In order to keep the programs, a code will need to be purchased, and then a permanent version of the software can be downloaded.

Many consumers who are purchasing a new computer are buying an entire package. In addition to the computer itself, these systems typically include a monitor, keyboard, and mouse. Some packages may even include a printer or digital camera. The number of extras included in a computer package usually affects laptop computer prices.
Some industry leaders in computer manufacturing make it a selling point to offer computers in sleek styling and in a variety of colors. They may also offer unusual or contemporary system design. Though this is less important to many consumers, for LAthose who do value “looks,” this type of system may be well worth the extra cost.

I-A: APPLICATIONS

There are several potential applications of laptop price prediction, including:

- Online marketplaces: E-commerce platforms that sell laptops can use price prediction models to forecast prices and adjust them in real-time to maximize profits and competitiveness.
- Inventory management: Retailers can use laptop price prediction models to optimize their inventory management and ensure they have the right amount of stock on hand at all times.
- Consumer research: Researchers can use laptop price prediction models to better understand consumer behavior and preferences, including which features and specifications are most important to them and how they impact price.
- Financial analysis: Financial analysts can use laptop price prediction models to evaluate the performance of companies that manufacture or sell laptops, as well as forecast future earnings and growth potential.
- Price comparison websites: Websites that compare laptop prices across different retailers can use price prediction models to provide more accurate and up-to-date information to their users.

Overall, laptop price prediction has the potential to provide valuable insights and benefits for a range of industries and applications.

I-B : APPLICATION IDEA

Laptop price prediction is a useful application of artificial intelligence (AI), and there are several ways in which it can be used in AI. Some of these uses include:

- **Machine learning**: Laptop price prediction models can be developed using machine learning algorithms that analyze large datasets of laptop prices and related features. These models can then be used to make predictions about future prices based on new data.
- **Natural language processing:** Some laptop price prediction models may use natural language processing (NLP) techniques to analyze text data such as product descriptions and reviews. This can help to identify important features and attributes that impact laptop prices.
- **Data mining:** Laptop price prediction models may use data mining techniques to identify patterns and relationships in large datasets of laptop prices and related features. This can help to uncover hidden insights and make more accurate predictions.
- **Deep learning:** Deep learning techniques such as neural networks can be used to develop more sophisticated laptop price prediction models that can learn from complex data and make more accurate predictions.
- **Recommendation systems**: Laptop price prediction models can also be used as part of recommendation systems that suggest laptops to users based on their budget and other preferences. This can help to improve the overall user experience and increase sales for retailers.

Overall, the use of AI in laptop price prediction can help to improve accuracy, efficiency, and effectiveness in a range of applications and industries.

II- BACKGROUND

The work for laptop price prediction typically involves the following steps:

- **Data Collection:** Collecting data on laptop prices from various sources such as e-commerce websites, retail stores, and market reports. This data may include information such as brand, model, processor type, RAM, storage capacity, graphics card, display size, and price.
- **Data Cleaning and Preprocessing:** Cleaning and preprocessing the collected data by removing duplicates, filling in missing values, and converting categorical variables to numerical ones.
- **Exploratory Data Analysis:** Exploring the data to gain insights into the relationships between different variables and identifying patterns in the data.
- **Feature Selection:** Selecting the most relevant features (i.e., variables) that have the most impact on laptop prices. This may involve using techniques such as correlation analysis and feature importance analysis.
- **Model Selection:** Selecting an appropriate machine learning algorithm to build a predictive model. This may involve using techniques such as linear regression, decision trees, random forests, or neural networks.
- **Model Training:** Training the selected machine learning algorithm on the preprocessed data to create a predictive model that can accurately predict laptop prices.
- **Model Evaluation:** Evaluating the performance of the predictive model by using metrics such as mean squared error, root mean squared error, and R-squared.
- **Deployment:** Deploying the predictive model in a real-world setting, such as an e-commerce website, where it can be used to provide accurate laptop price predictions to customers.

Overall, the background work for laptop price prediction involves collecting and cleaning data, selecting relevant features, choosing an appropriate machine learning algorithm, training and evaluating the model, and deploying it in a real-world setting.

II-A PROBLEM AREA:

While predicting laptop prices with AI can be highly beneficial, there are some common problems and challenges that can arise. Here are some of the main ones:

- **Lack of data:** One of the biggest challenges in laptop price prediction is the availability of high-quality data. Without enough data or with data that is inaccurate or incomplete, the accuracy of the predictions can be compromised.
- **Feature selection:** Choosing the right features that affect laptop price is crucial in developing an accurate model. However, determining which features are most important can be difficult, and selecting too few or too many features can lead to inaccurate predictions.
- **Overfitting:** Overfitting occurs when a model is too complex and trained on too few data points, leading to it memorizing the training data instead of generalizing. This can result in poor performance when predicting laptop prices on new, unseen data.
- **Changing market conditions**: The laptop market is constantly evolving, with new models and technologies being introduced regularly. This can make it challenging to keep laptop price prediction models up-to-date and accurate over time.
- **Bias:** Bias can occur if the training data used to develop the laptop price prediction model is not representative of the real-world data. This can result in predictions that are skewed towards certain brands, regions, or demographics.
- **Interpretability:** Some AI models used for laptop price prediction can be complex and difficult to interpret. This can make it challenging to understand how the model arrived at its predictions, making it harder to refine the model and correct any inaccuracies.

Addressing these challenges is crucial to ensure that laptop price prediction models are accurate, reliable, and effective. This can involve collecting high-quality data, selecting the right features, regular model updates, and developing models that are both accurate and interpretable.

III- OUR WORK

Data optimization, which refers to the process of improving the quality and usability of data for a specific task or application. There are several processes involved in data optimization apart from above work, which may include:

- **Data integration:**

Combining data from multiple sources or formats into a single dataset that can be analyzed or modeled. Data integration may also involve resolving conflicts or inconsistencies between different datasets.

- **Data validation:**

Verifying the accuracy, completeness, and consistency of the data. Data validation may involve using statistical or machine learning techniques to identify outliers or anomalies in the data

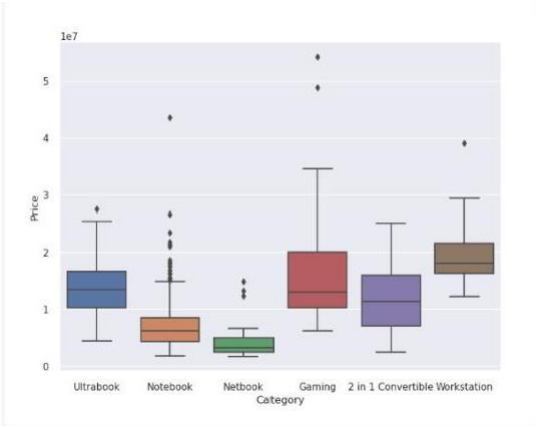
- **Data modeling:**

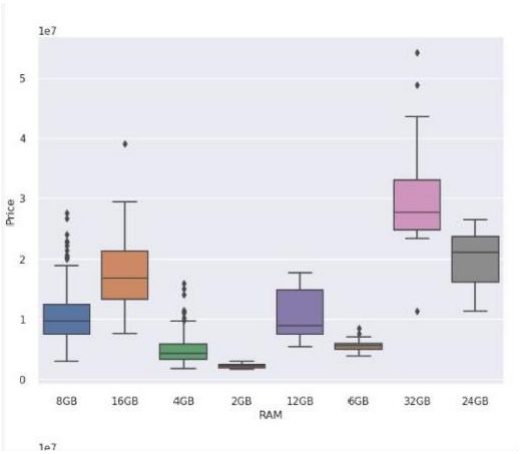
Building models or algorithms that can make predictions or decisions based on the data. Data modeling may involve using techniques such as regression, classification, clustering etc

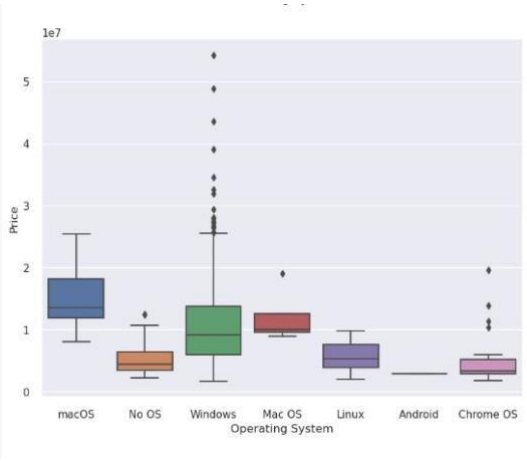
- **Data visualization:**

Using charts, graphs, or other visual aids to explore and communicate insights from the data. Data visualization may help identify patterns, trends, or relationships in the data that are not immediately apparent from raw data.

III-B VISUALISATION







These are box plots that represent the category ,RAM,OS with respect to Price.

III-C SELECTION OF MODEL

We have used 2 models here , mainly Random Forest Regressor and Linear Regression Model

Random Forest Regressor is a type of algorithm used for regression tasks. It is based on the concept of decision trees and combines multiple decision trees to make a final prediction. In a random forest regressor, multiple decision trees are created using random subsets of the training data and a random subset of the features for each split in the decision tree. This creates a diverse set of decision trees that are less prone to over fitting. When making a prediction, each decision tree in the forest independently predicts a value, and the final prediction is the average of all the individual tree predictions. Random Forest Regressor is a powerful and popular machine learning algorithm that can handle complex regression problems and is known for its robustness, scalability, and ability to handle missing data. It is widely used in various fields such as finance, healthcare, and marketing.

Linear regression is a popular machine learning algorithm used for predicting a continuous target variable based on one or more predictor variables. It assumes a linear relationship between the predictors and the target variable.

Linear regression model and random forest regressor are both machine learning algorithms used for regression tasks, but they have some key differences in terms of their approach and performance.

- Linear regression is a simple yet powerful algorithm that models the linear relationship between the input features and the output variable.
- Random forest regressor, on the other hand, is an ensemble learning algorithm that combines multiple decision trees to make a final prediction. It creates a diverse set of decision trees using random subsets of the training data and features, which reduces the risk of over fitting and improves the accuracy of the model.
- Random forest regressor is more flexible and can handle complex datasets with a large number of input features

Here are our test results
→Models
1. Linear Regression
   - Training accuracy : 0.8387595187152533
   - Testing accuracy : 0.7935316574237211
   - Mean Absolute Error : 1953117.081047944
   - Root Mean Square Error : 2696368.1093766172
   - r2_score 0.7935316574237211

2. Random Forest Regressor
   - Training accuracy : 0.957658607401638
   - Testing accuracy : 0.8180228352184167
   - Mean Absolute Error : 1816206.923974398
   - Root Mean Square Error : 2531400.68590522
   - r2_score 0.8105261561676067

In the Random Forest Regressor model we had 81.80% accuracy and in the Linear Regression model we had 79.35% accuracy
This clearly means that Random Forest Regressor is best and is hence chosen.

IV – TECHNOLOGY

IV-A: Languages-

Python

IV-B :Software and services-

Jupyter notebook and dataset from Kaggle

Model implementation

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.metrics import accuracy_score, classification_report
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.model_selection import train_test_split
```

importing libraries

filling the empty places with the mode value

```
df['Operating System Version']=df['Operating System Version'].fillna(df['Operating System Version'].mode()[
df_test['Operating System Version']=df_test['Operating System Version'].fillna(df_test['Operating System Ve
```

checking for null values in each column

```
(df.isnull().sum()/len(df))
```

```
Manufacturer              0.0
Model Name                0.0
Category                  0.0
Screen Size               0.0
Screen                    0.0
CPU                       0.0
RAM                       0.0
 Storage                  0.0
GPU                       0.0
Operating System          0.0
Operating System Version  0.0
Weight                    0.0
Price                     0.0
dtype: float64
```

removing the string 'kg' and setting the datatype of the weight variable as float

```
df['Weight'].replace(to_replace='kg', value='', regex=True, inplace=True)
df_test['Weight'].replace(to_replace='kg', value='', regex=True, inplace=True)
df_test['Weight'].replace(to_replace='s', value='', regex=True, inplace=True)
```

+ Code + Markdown

```
df['Weight'] = df['Weight'].astype("float64")
df_test['Weight'] = df_test['Weight'].astype("float64")
```

Transforming the data and converting into necessary condition

```
df['Model_name_avg'] = df.groupby(['Model Name'])['Price'].transform('median').round(2)
df['CPU_avg'] = df.groupby(['CPU'])['Price'].transform('median').round(2)
df['GPU_avg'] = df.groupby(['GPU'])['Price'].transform('median').round(2)

df_test['Model_name_avg'] = df_test.groupby(['Model Name'])['Price'].transform('median').round(2)
df_test['CPU_avg'] = df_test.groupby(['CPU'])['Price'].transform('median').round(2)
df_test['GPU_avg'] = df_test.groupby(['GPU'])['Price'].transform('median').round(2)
```

```
def group(value):
    if value >= 25000000:
        return 0
    elif value >= 20000000  and value < 25000000:
        return 1
    elif value >= 15000000  and value < 20000000:
        return 2
    elif value >= 10000000  and value < 15000000:
        return 3
    else:
        return 4
```

```
df['model_name_group'] = df.apply(lambda x: group(x['Model_name_avg']),axis=1)
df['cpu_group'] = df.apply(lambda x: group(x['CPU_avg']),axis=1)
df['gpu_group'] = df.apply(lambda x: group(x['GPU_avg']),axis=1)

df_test['model_name_group'] = df_test.apply(lambda x: group(x['Model_name_avg']),axis=1)
df_test['cpu_group'] = df_test.apply(lambda x: group(x['CPU_avg']),axis=1)
df_test['gpu_group'] = df_test.apply(lambda x: group(x['GPU_avg']),axis=1)
```

Dropping the unnecessary

```
df = df.drop(['Model Name', 'CPU', 'GPU', 'Model_name_avg', 'CPU_avg', 'GPU_avg'], axis = 1)
df_test = df_test.drop(['Model Name', 'CPU', 'GPU', 'Model_name_avg', 'CPU_avg', 'GPU_avg'], axis = 1)
```

Converting categorical data to numeric data

```
from sklearn.preprocessing import LabelEncoder

label_encoder_Manufacturer = LabelEncoder()
label_encoder_Category = LabelEncoder()
label_encoder_Screen_Size = LabelEncoder()
label_encoder_Screen = LabelEncoder()
label_encoder_RAM = LabelEncoder()
label_encoder_Storage = LabelEncoder()
label_encoder_Operating_System = LabelEncoder()
label_encoder_Operating_System_Version = LabelEncoder()

df['Manufacturer'] = label_encoder_Manufacturer.fit_transform(df['Manufacturer'])
df['Category'] = label_encoder_Category.fit_transform(df['Category'])
df['Screen Size'] = label_encoder_Screen_Size.fit_transform(df['Screen Size'])
df['Screen'] = label_encoder_Screen.fit_transform(df['Screen'])
df['RAM'] = label_encoder_RAM.fit_transform(df['RAM'])
df[' Storage'] = label_encoder_Storage.fit_transform(df[' Storage'])
df['Operating System'] = label_encoder_Operating_System.fit_transform(df['Operating System'])
df['Operating System Version'] = label_encoder_Operating_System_Version.fit_transform(df['Operating System

df_test['Manufacturer'] = label_encoder_Manufacturer.fit_transform(df_test['Manufacturer'])
df_test['Category'] = label_encoder_Category.fit_transform(df_test['Category'])
df_test['Screen Size'] = label_encoder_Screen_Size.fit_transform(df_test['Screen Size'])
df_test['Screen'] = label_encoder_Screen.fit_transform(df_test['Screen'])
df_test['RAM'] = label_encoder_RAM.fit_transform(df_test['RAM'])
df_test[' Storage'] = label_encoder_Storage.fit_transform(df_test[' Storage'])
df_test['Operating System'] = label_encoder_Operating_System.fit_transform(df_test['Operating System'])
df_test['Operating System Version'] = label_encoder_Operating_System_Version.fit_transform(df_test['Operat
```

divide into training and testing: dependent variable - 'price'

```
X_train = df.drop('Price', axis = 1)
X_train = X_train.values
y_train = df['Price']
X_test = df_test.drop('Price', axis = 1)
X_test = X_test.values
y_test = df_test['Price']
```

standard scaling

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_standard_train = scaler.fit_transform(X_train)
y_standard_train = scaler.fit_transform(y_train.values.reshape(-1,1))
X_standard_test = scaler.fit_transform(X_test)
y_standard_test = scaler.fit_transform(y_test.values.reshape(-1,1))
```

```
from sklearn.ensemble import RandomForestRegressor
regressor_rf = RandomForestRegressor(n_estimators = 100, min_samples_split = 2,
max_depth= 7,  criterion = 'squared_error', random_state = 0)
regressor_rf.fit(X_train, y_train)
lr_normal_rf = regressor_rf.score(X_train, y_train)
lr_normal_rf_test = regressor_rf.score(X_test, y_test)
previsoes = regressor_rf.predict(X_test)
mae_lr_normal_rf  = mean_absolute_error(y_test, previsoes)
rmse_lr_normal_rf = np.sqrt(mean_squared_error(y_test, previsoes))

print('Train :', lr_normal_rf)
print('Test :', lr_normal_rf_test)
print('Mean Absolute Error :', mae_lr_normal_rf)
print('Root Mean Square Error :', rmse_lr_normal_rf)
```

```
Train : 0.957658607401638
Test : 0.8180228352184167
Mean Absolute Error : 1816206.923974398
Root Mean Square Error : 2531400.68590522
```

```
from sklearn.linear_model import LinearRegression
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
lr_normal_score_train = lr_model.score(X_train, y_train)
lr_normal_score_test = lr_model.score(X_test, y_test)
previsoes = lr_model.predict(X_test)
mae_lr_normal = mean_absolute_error(y_test, previsoes)
rmse_lr_normal = np.sqrt(mean_squared_error(y_test, previsoes))
prediction = lr_model.predict(X_test)
from sklearn.metrics import r2_score
a=r2_score(y_test, prediction)

print('Train :', lr_normal_score_train)
print('Test :', lr_normal_score_test)
print('Mean Absolute Error :', mae_lr_normal)
print('Root Mean Square Error :', rmse_lr_normal)
print('r2_score',a)
```

```
Train : 0.8387595187152533
Test : 0.7935316574237211
Mean Absolute Error : 1953117.081047944
Root Mean Square Error : 2696368.1093766172
r2_score 0.7935316574237211
```

V-CONCLUSION

We can see that we have numerical and continuous data, in our database, we don't have a good amount of data, which makes our work more difficult, when we look at our data we can see that we have null values, which makes treatment necessary, we can also verify that we have columns that have many values, which makes it necessary to group these columns so that it is possible to carry out the analysis.
Looking at our exploratory analysis we can see that we have some predominant values in our database, they are practically just notebooks with 8 or 4 gigs of ram with Windows operating system in version 10, when we look at the brand we can see that we have predominance in 3 -4 marks, when we look at our continuous variables we can see that the higher the price, the less amount of data we have.
In our analysis of the target variable, we can see some behaviors such as computers with a higher amount of Ram generally have a higher price, when we analyze the brand variable, we can also see that Mac computers are generally more expensive than normal computers.

Now talking about our machine learning models we achieved a satisfactory result after grouping the variables, without grouping it would be impossible to run the machine learning models, making the grouping mandatory, our best model was the Random Forest with 81.80% accuracy.
When we look at the most important variables of the model, we can see that the grouping we made of the variable Brand of the Model is the one that best explains our target variable.

VII- REFERENCES:

1. https://www.kaggle.com/code/danielbethell/laptop- prices-prediction
2. Laptop Price Prediction using Machine Learning, Vaishali Surjuse; Sankalp Lohakare, et. al, DOI: 10.47760/ijcsmc.2022.v11i01.021
3. https://scikit-learn.org/stable/