
# CAPSTONE PROJECT

## PGA-20

## Module: MACHINE LEARNING

## Title: Price Prediction of Laptops

Submitted by: C Chandramouli

# Abstract

The goal of this project is to predict the price of the product in euros based on the specifications and features of the new product. The dataset is carefully analysed and all the necessary steps are followed like data cleaning, feature selection, feature engineering and feature scaling. The dataset is divided into training and the test dataset and Regression Models were developed based on the training dataset and applied to the test dataset to find out the accuracy of each model based on the predicted values generated. Based on these values we can determine how good a particular model is predicting the prices of the new products.

# Table Of Contents

# Introduction

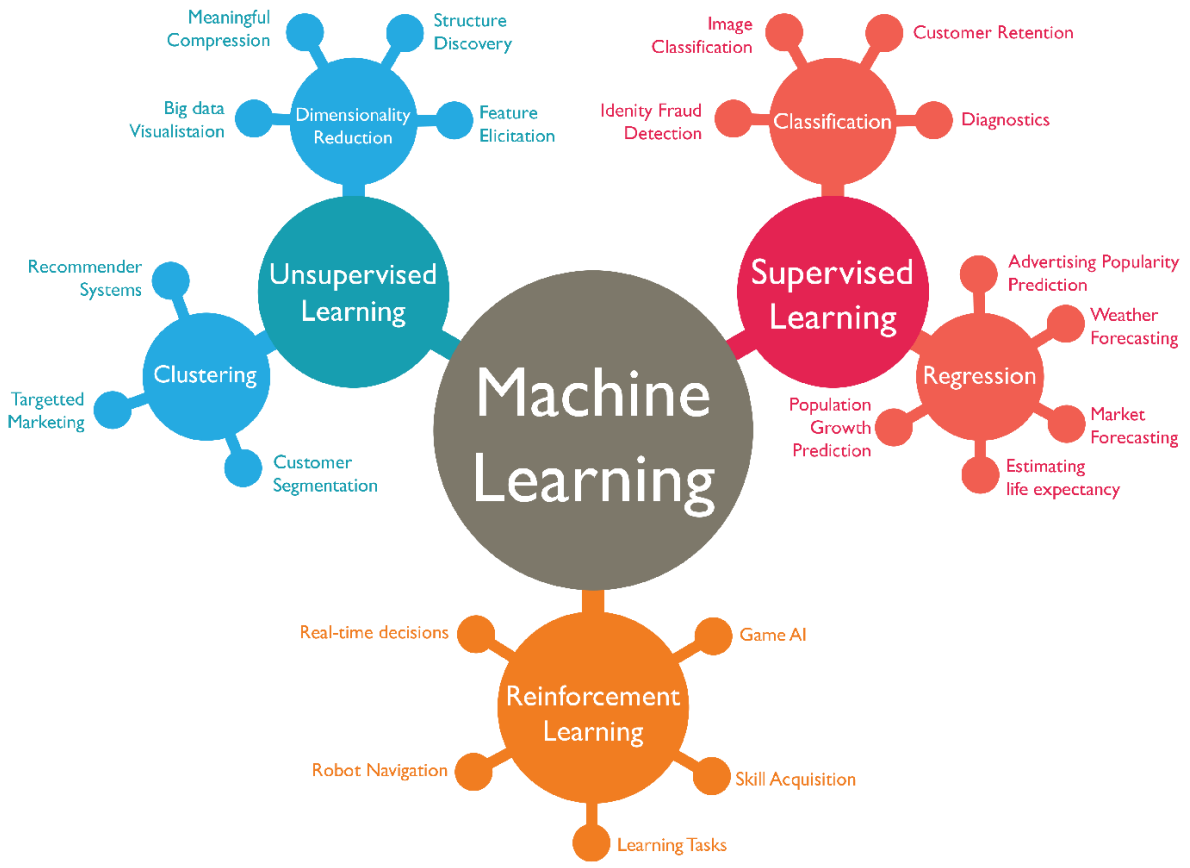
Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs.

Machine learning is a continuously developing field. Because of this, there are some considerations to keep in mind as you work with machine learning methodologies, or analyse the impact of machine learning processes.

There are three types of the machine learnings:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning



For our project, the prediction of price for the Laptops and Notebooks we will proceed with the Regressions models. As our output variable /dependent variable is Continuous values. The models planned for this project are Linear Regression, Decision Tree and Random Forest models. The metrics for the respective model are documented into a superate data frame. The final conclusion is drawn from the metrics and a best model is chosen.

# Description of Data:

The data LAPTOPS is collected from the Kaggle datasets.

The information of data is filed as

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Manufacturer              1303 non-null   object
 1   Model Name                1303 non-null   object
 2   Category                  1303 non-null   object
 3   Screen Size               1303 non-null   object
 4   Screen                    1303 non-null   object
 5   CPU                       1303 non-null   object
 6   RAM                       1303 non-null   object
 7    Storage                  1303 non-null   object
 8   GPU                       1303 non-null   object
 9   Operating System          1303 non-null   object
 10  Operating System Version  1133 non-null   object
 11  Weight                    1303 non-null   object
 12  Price (Euros)             1303 non-null   object
dtypes: object(13)
memory usage: 132.5+ KB
```

# Variables:

The dataset consists of 13 variables

1. **Manufacturer:** A manufacturer is a person or company that produces finished goods from raw materials by using various tools, equipment, and processes, and then sells the goods to consumers, wholesalers, distributors, retailers, or to other manufacturers for the production of more complex goods.

2. **Model Name:** it is the name given to the companies product for the sake of identification and reorganization. a good name is well settled in customers mind easily than unknown names .
3. **Category**: based on the characteristics of the product the product is categorised into different groups. Categories help the customers choose the ideal product based on their requirement and work.
4. **Screen Size**: it gives information about the display size of the screen
5. **Screen**: it consists the information about the technology used in the products display and the resolution of the display. The more the resolution the sharper the display and clearer the display is.
6. **CPU**: the CPU is heart of the laptops, there is two companies that creates the cpu's one is Intel and Ryzen. The other companies are that succussed in making their own cpu is Apple that is M-series chips
7. **RAM**: The random acess memory is the memory that's is used during the processing of the applications in the system,
8. **Storage**: the storage is used to store the data and information.
9. **GPU**: it is responsible to process the graphics that is displayed in the system display. It is much more important in gaming category
10. **Operating System**: it is the operating system that is used in the system. It acts as the interface between the user and the computer.
11. **Operating System Version:** it the version of the operating system. Latest version has the latest features and optimised software's.
12. **Weight:** It gives the information on the weight of the product
13. **Price (Euros):** the price of the product at which the product is sold.

# Assumptions:

- The data is collected from real time, each specification have its effects on to the market.
- Only the extreme values are considered as outliers

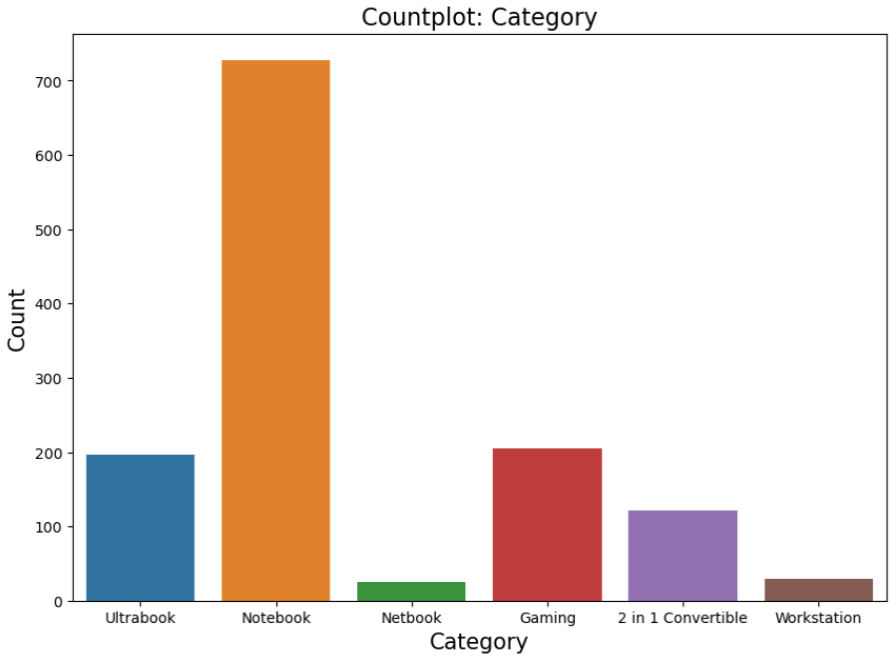
## Imported libraries:

The library's used in the project are,

- Numpy
- Pandas
- Matplotlib
- Seaborn
- Math
- Statsmodels
- Sklearn

# EDA:

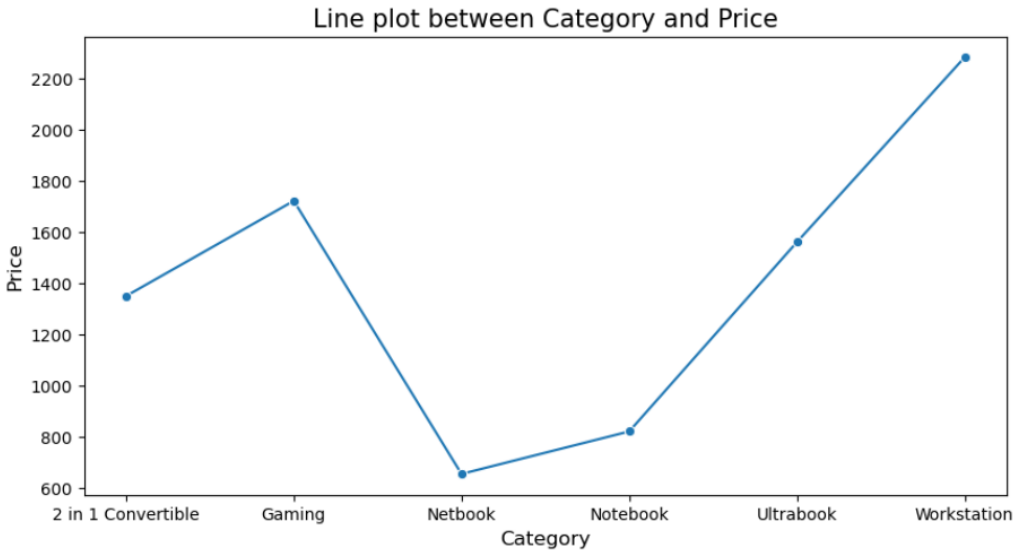
**PLOT**: The plot is plotted by taking the Category on X-axis and Count in the y axis. The plot is plotted as an count plot from the seaborn library.




insights:

- Notebooks are most produced products followed by gaming laptops and ultrabooks
- Netbooks are least produced products

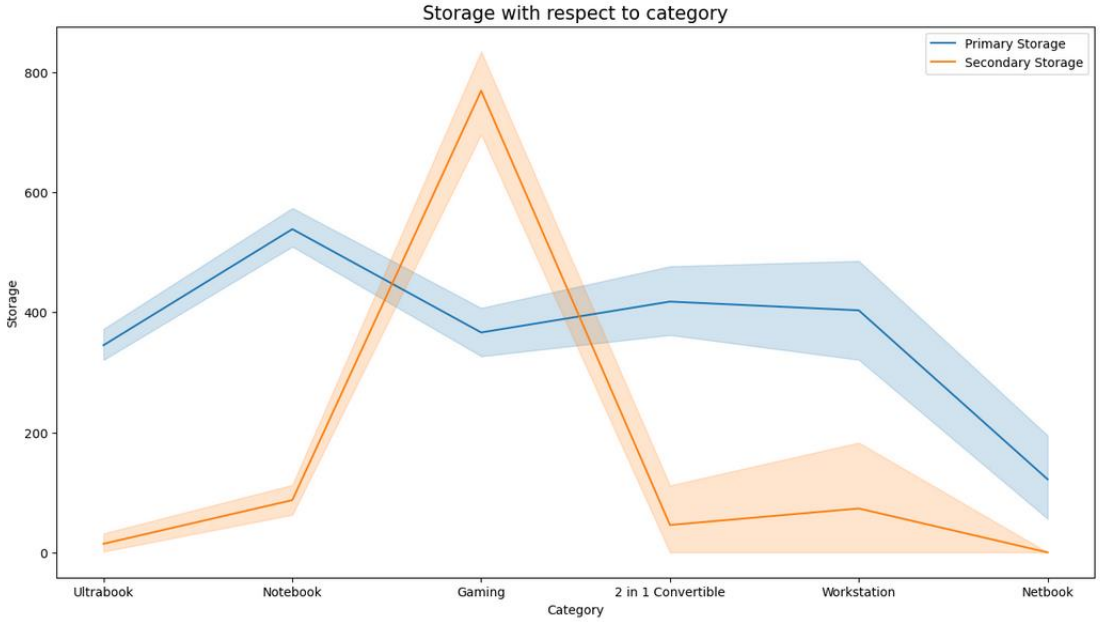
**PLOT**: the line plot is plotted by taking the average of prices in Y-axis and Category in X-axis. The plot is plotted from the seaborn library




insights:

- The Netbooks are costs less when compared to all other laptops
- Workstations are the most expensive among the laptops

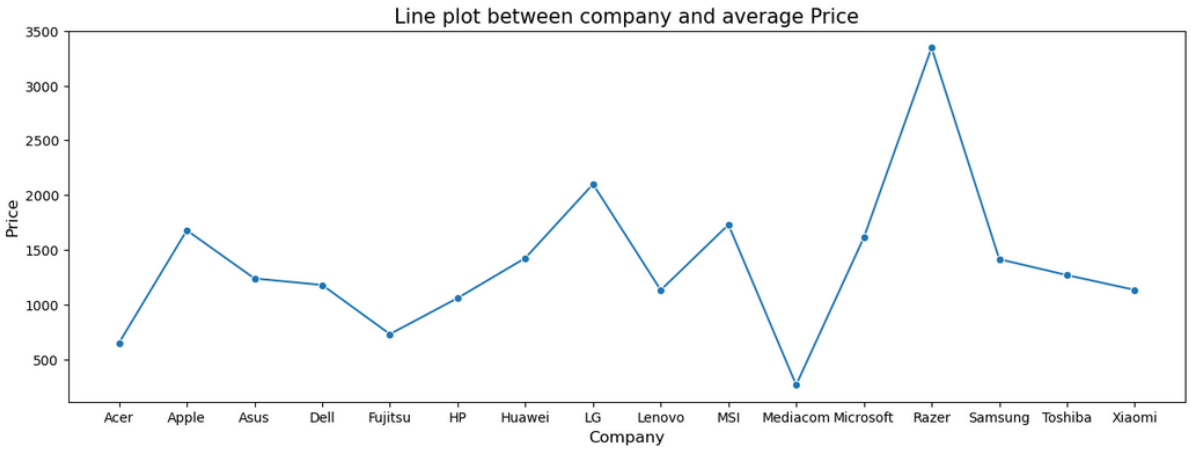
**PLOT**: the line plot is plotted by taking the category in X-axis and storage capacity in Y-axis. the plot is plotted for both primary and secondary storages




Point insights s:

- Notebooks have the highest primary storage capacity and net books at the lowest
- Gaming laptops have highest secondary storage capacity when compared to any other laptops category.

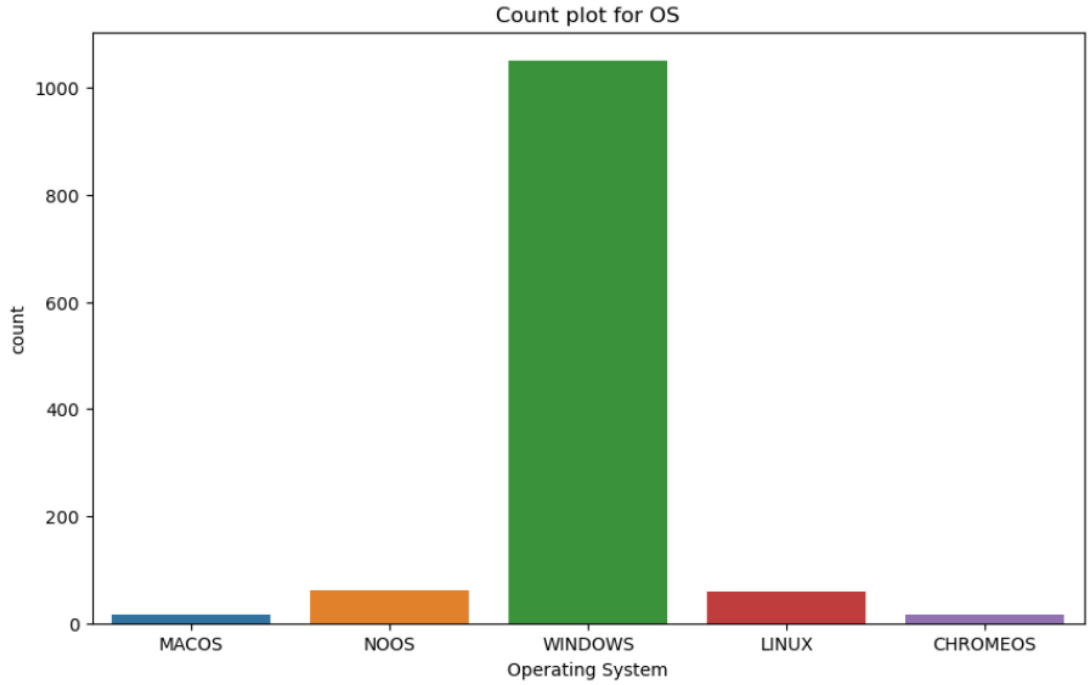
**PLOT**: the line plot is plotted by taking the category in X-axis and storage capacity in Y-axis. the plot is plotted for both primary and secondary storages




Insights:

- The price of mediacon laptops is the lowest and razor laptops is the highest.

PLOT: a count plot is plotted by taking the type of operating system on X-axis and the count in the Y-axis




Insights:

- Windows is the most popular os that is used in many laptops
- A least number of laptops dosnt have any os denoted by NOOS and it gives the options to go with the windows or any other OS.

# DATA PREPROCESSING

The data is undergone to transform the data into a suitable form to build a model.

# Feature Engineering:

| Variable name | Description | Derived from | dtype |
|---|---|---|---|
| Model Name | Conversion of the model names into upcase of lowcase to avoid the misleading | Model name | Object |
| Price_in_Euros | it is created to replace the main price variable which is object type | Price (euros) | Int |
| RAM_in_GB | It is the converted version of RAM which is object type | RAM | Int |
| Weight_in_kgs | It is the numeric form of the original variable Weight | Weight | Int |
| Storage_Type | it describes the what kind of storage is used in the product (SSD,HDD) | Storage | Object |
| Primary_storage | It describes the primary storage capacity of the device. It's the local storage and used as admin storage | Storage | Int |
| Secondary_storage | Its is the external storage and storage extension | Storage | Int |
| Display | It descrives the technology used in the device display. The missing values display technology is filled as condiering default display as LCD Panel | Display | Object |
| Resolution | it is the resolution of the display. higher the number higher the contrast and resolution | Display | Object |
| Processor | It is the new variable to that is been processed from the CPU, this is created to reduce the variance of the variable | CPU | Object |

| | | | |
|---|---|---|---|
| Clock_speed_in_GHZ | It is also derived from the already existing variable CPU and it is a numeric variable that describes the speed at which the data is processed | CPU | Int |
| Genaration | It describes the version and Gen of the processor. It is drived from the pre existing column CPU | CPU | Int |
| Operating System | It is the interface in between the user and computer. Usually the mostly used os used in our data is Windows | OS | Object |
| Graphics | It is the simplified version of the graphics | GPU | Object |

## Outliers:

As the present of Outliers is justified by the features of the product.

Ex:

The high price of the product is justified by the Prescence of the highresolution display, better processor and other factors

Outliers is been treated for the presence of the extreme outliers in primary storage, ram, clock speed

## Elimination of features:

As the new variables is created from the preexisting variables the old variables are eliminated

The eliminated variables are:

Screen, CPU, GPU, Storage, Price (Euros), RAM, Weight

Features selection is done after division of the numeric and char type of variables, new data frames. With names numeric and chars.

Char variables is isolated by code,

```
char=laptops.select_dtypes(include='object')
char.head()
```

The object type of values which are collected into the char Data frame is then followed by dummy encoding or one hot encoding to transform the character to numeric variables which can be feeded to the algorithm to build an model

After the one hot encoding the shape of the char is (1200, 289)

Numeric variables is isolated by code,

```
num=laptops.select_dtypes(include=np.number)
num.head()
```

The shape of the numeric data is (1200, 7)

Due to the unevenness among the variables in the data there is difference in their weightage. In order to resolve and avoid such weightage error the data is undergone Scandalization.

# Feature Scaling:

**Scandalization:** Standardization is a scaling technique wherein it makes the data scale-free by converting the statistical distribution of the data into the below format:

mean - 0 (zero)

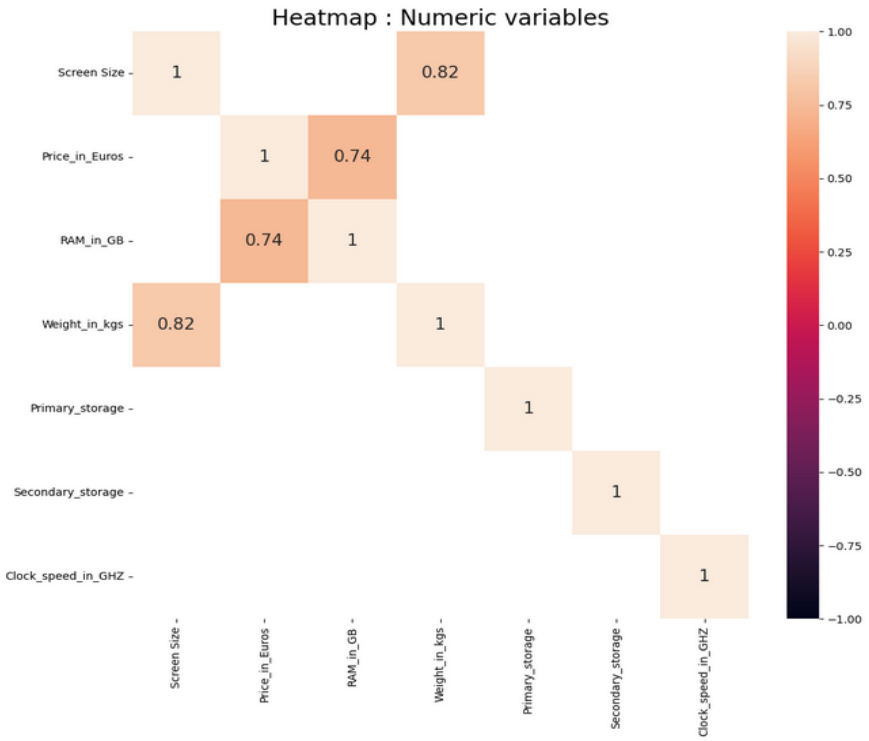
standard deviation – 1

```
# scaling of the numeric data to bring the balence between the variable vales
scaler=StandardScaler()
numeric=scaler.fit_transform(num[num.columns])
numeric=pd.DataFrame(numeric,columns=num.columns,index=None)
numeric.head()
```

| | Screen Size | Price_in_Euros | RAM_in_GB | Weight_in_kgs | Primary_storage | Secondary_storage | C |
|---|---|---|---|---|---|---|---|
| 0 | -1.366665 | 0.261702 | -0.120908 | -1.077234 | -0.907889 | -0.438948 | |
| 1 | -1.366665 | -0.378612 | -0.120908 | -1.122710 | -0.907889 | -0.438948 | |
| 2 | 0.352254 | -0.847594 | -0.120908 | -0.334471 | -0.558820 | -0.438948 | |
| 3 | 0.202783 | 2.001149 | 1.513905 | -0.379946 | 0.139319 | -0.438948 | |
| 4 | -1.366665 | 0.935411 | -0.120908 | -1.077234 | -0.558820 | -0.438948 | |

# Feature Selection:

The importance of the features in the dataset is checked in terms of Correlation by plotting an corelation matrix using heatmap




It's really helpful in filtering out unnecessary columns. Lower the correlation, lower is the attribute's importance.

- If the value is +1 or close to it then we say the variables are positively correlated. And they vary in the same direction simultaneously.
- If the value is -1 or close to it then we say the variables are negatively correlated. And they vary in the opposite direction simultaneously.
- If the value is 0 or close to it then we say the variables are not correlated.

The usual threshold is taken as 0.8 and the necessary features are selected. The final datasets consist of (1200, 7), (1200, 289).

The processed dataset is then combined to form a new final dataset named Laptop. the shape of the final dataset is (1200, 296)

# Machine Learning:

From the final dataset is divided into X, y variables where X consists of all independent variables and y consists of the dependent variable

## Model Building

```
laptop=sm.add_constant(laptop)
y=laptop['Price_in_Euros']
X=laptop.drop('Price_in_Euros',axis=1)
```

```
y.head()
```

```
0    0.261702
1   -0.378612
2   -0.847594
3    2.001149
4    0.935411
Name: Price_in_Euros, dtype: float64
```

```
X.head()
```

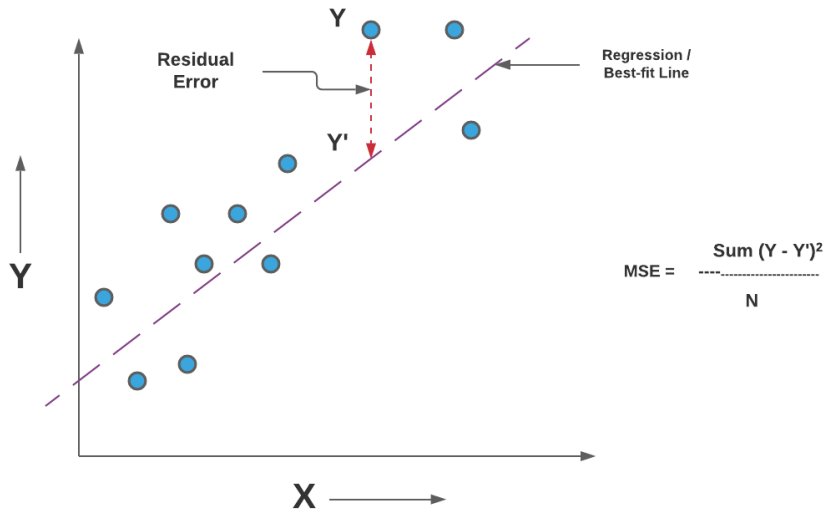
| const | index | Screen Size | RAM_in_GB | Primary_storage | Secondary_storage | Clock_speed_in_GHZ | inde |
|---|---|---|---|---|---|---|---|
| 00000 | 0 | -1.366665 | -0.120908 | -0.907889 | -0.438948 | -0.223267 | |
| 00000 | 1 | -1.366665 | -0.120908 | -0.907889 | -0.438948 | -1.453164 | |
| 00000 | 2 | 0.352254 | -0.120908 | -0.558820 | -0.438948 | 0.268691 | |
| 00000 | 3 | 0.202783 | 1.513905 | 0.139319 | -0.438948 | 0.760650 | |
| 00000 | 4 | -1.366665 | -0.120908 | -0.558820 | -0.438948 | 1.744567 | |

The each datasets are further divided test and train datasets namely

X train, X test - (840, 296), (360, 296)

Y train, Y test (840,), (360,)

# Metrics:




## MSE (Mean Squared Error):

In statistics, the mean squared error (MSE)[1] or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value. MSE is a risk function, corresponding to the expected value of the squared error loss.[2] The fact that MSE is almost always strictly positive (and not zero) is because of randomness or because the estimator does not account for information that could produce a more accurate estimate

## RMSE (Root Mean Squared Error):

One of the two main performance indicators for a regression model.

The Root Mean Squared Error (RMSE) is one of the two main performance indicators for a regression model. It measures the average difference between values predicted by a model and the actual values. It provides an estimation of how well the model is able to predict the target value (accuracy).

The lower the value of the Root Mean Squared Error, the better the model is. A perfect model (a hypothetic model that would always predict the exact expected value) would have a Root Mean Squared Error value of 0.

## R Squared:

R-Squared ($R^2$ or the coefficient of determination) is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable. In other words, r-squared shows how well the data fit the regression model (the goodness of fit).

R-squared can take any values between 0 to 1. Although the statistical measure provides some useful insights regarding the regression model, the user should not rely only on the measure in the assessment of a statistical model. The figure does not disclose information about the causation relationship between the independent and dependent variables.

### Adjusted R Squared:

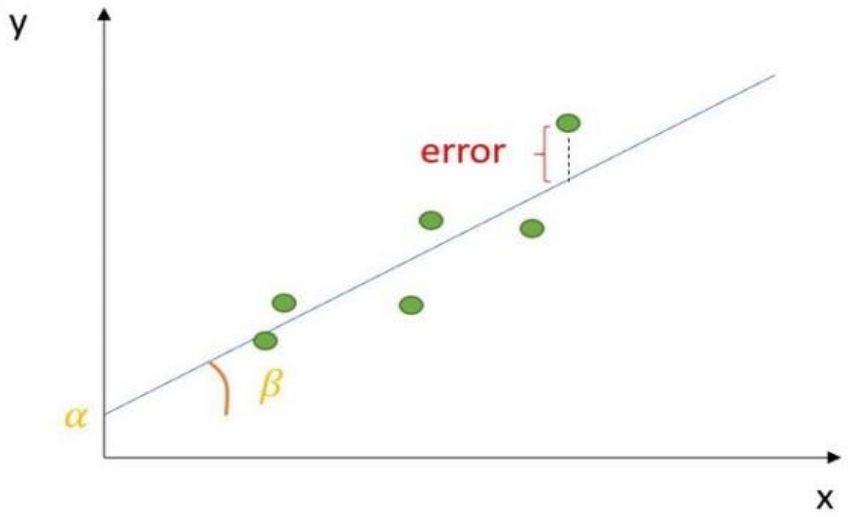
Adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model. The adjusted R-squared increases when the new term improves the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected. Typically, the adjusted R-squared is positive, not negative. It is always lower than the R-squared.

Adding more independent variables or predictors to a regression model tends to increase the R-squared value, which tempts makers of the model to add even more variables. This is called overfitting and can return an unwarranted high R-squared value. Adjusted R-squared is used to determine how reliable the correlation is and how much it is determined by the addition of independent variables.

# Model Building:

## Linear regression (OLS):

Ordinary least squares (OLS) regression is an optimization strategy that helps you find a straight line as close as possible to your data points in a linear regression model. OLS is considered the most useful optimization strategy for linear regression models as it can help you find unbiased real value estimates for your alpha and beta.




The OLS algorithm is performed for the given dataset and the Summary of the model is found to be

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | Price_in_Euros | R-squared: | 0.943 |
| Model: | OLS | Adj. R-squared: | 0.920 |
| Method: | Least Squares | F-statistic: | 40.25 |
| Date: | Fri, 23 Feb 2024 | Prob (F-statistic): | 1.34e-265 |
| Time: | 21:23:50 | Log-Likelihood: | -15.723 |
| No. Observations: | 840 | AIC: | 523.4 |
| Df Residuals: | 594 | BIC: | 1688. |
| Df Model: | 245 | | |
| Covariance Type: | nonrobust | | |

Metrics for the linear regression (OLS):

```
MSE: 0.19205318774696029
RMSE :0.43823873373648786
Squared : 0.9431851656990179
Adjucted R Squared :0.9197514377465926
```

# Decision Tree:

A decision tree is one of the most powerful tools of supervised learning algorithms used for both classification and regression tasks. It builds a flowchart-like tree structure where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. It is constructed by recursively splitting

the training data into subsets based on the values of the attributes until a stopping criterion is met, such as the maximum depth of the tree or the minimum number of samples required to split a node.

During training, the Decision Tree algorithm selects the best attribute to split the data based on a metric such as entropy or Gini impurity, which measures the level of impurity or randomness in the subsets. The goal is to find the attribute that maximizes the information gain or the reduction in impurity after the split.

Metrics for the decision tree model:

```
MAE : 0.3265738234227355
MSE : 0.2575876758012627
RMSE : 0.5075309604361715
R Squred : 0.6914117006074538
R Squared Adj : -0.7584634838400646
```

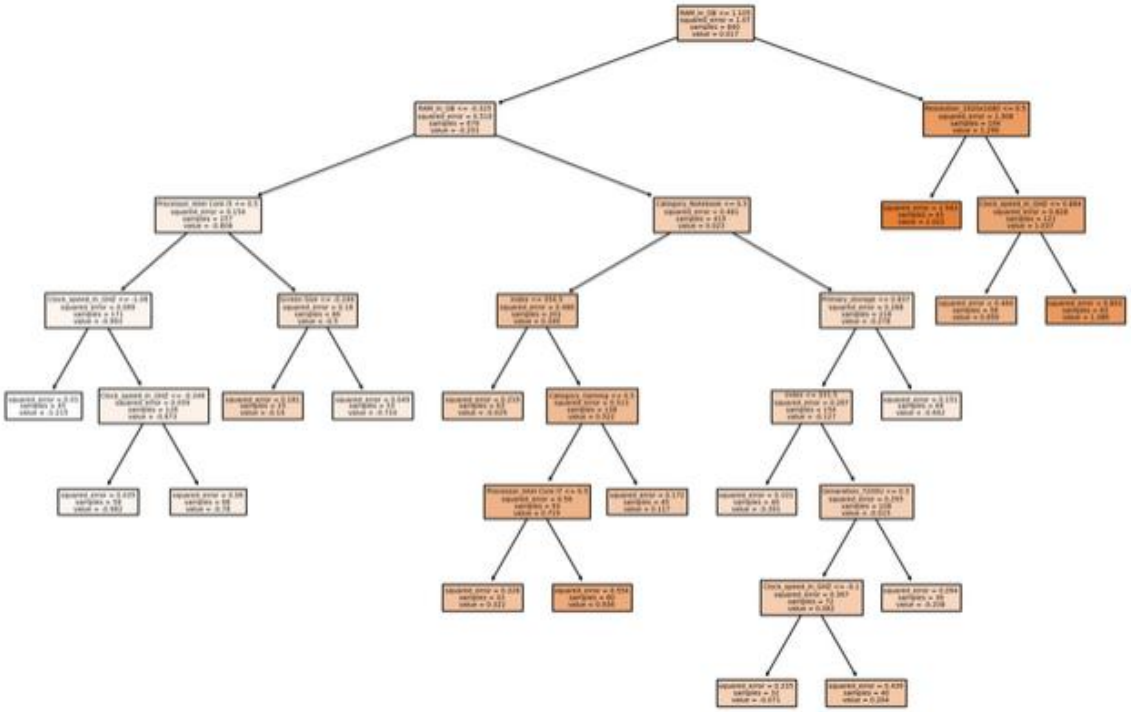
# Decision Tree with GridSearchCV:

## Grid Search:

Grid search is a technique for tuning hyperparameter that may facilitate build a model and evaluate a model for every combination of algorithms parameters per grid. We might use 10 fold cross-validation to search the best value for that tuning hyperparameter. Parameters like

in decision criterion, max_depth, min_sample_split, etc. These values are called hyperparameters. To get the simplest set of hyperparameters we will use the Grid Search method. In the Grid Search, all the mixtures of hyperparameters combinations will pass through one by one into the model and check the score on each model. It gives us the set of hyperparameters which gives the best score. Scikit-learn package as a means of automatically iterating over these hyperparameters using cross-validation. This method is called Grid Search.

- Grid Search takes the model or objects you'd prefer to train and different values of the hyperparameters. It then calculates the error for various hyperparameter values, permitting you to choose the best values.



Metrics for the Decision tree with Hyper parameters:

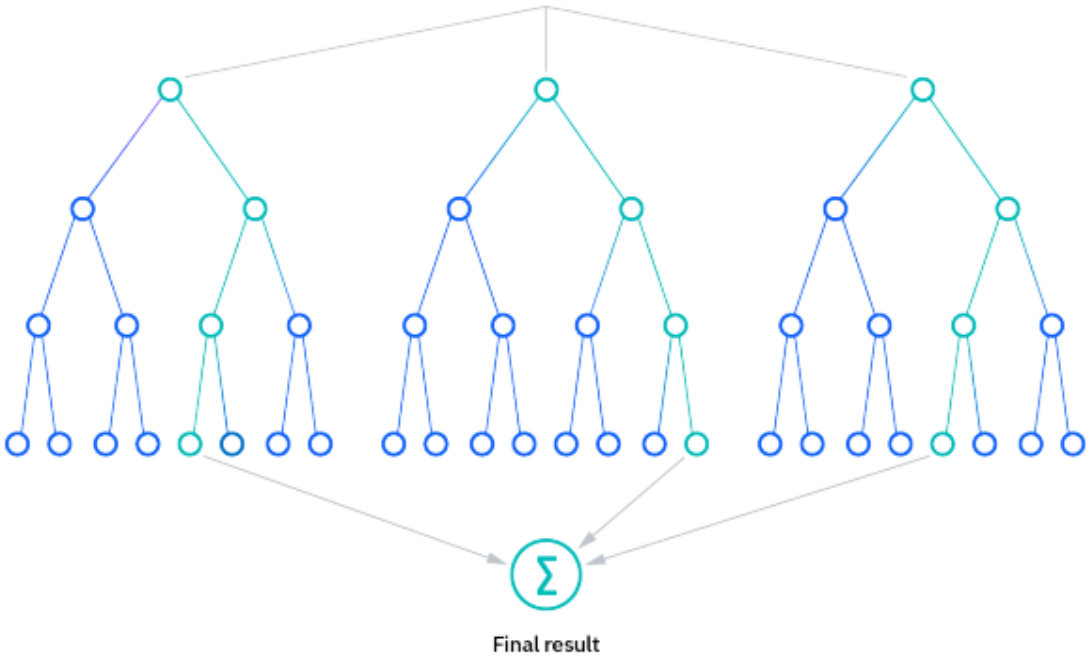
```
MAE : 0.37807128250757893
MSE : 0.29310403177912747
RMSE : 0.5413908308968
R Squred : 0.6488633455367494
R Squared Adj : -1.0009215706715393
```

# Random Forest:

Random forest algorithms have three main hyperparameters, which need to be set before training. These include node size, the number of trees, and the number of features sampled.

From there, the random forest classifier can be used to solve for regression or classification problems.

The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble is comprised of a data sample drawn from a training set with replacement, called the bootstrap sample. Of that training sample, one-third of it is set aside as test data, known as the out-of-bag (oob) sample, which we'll come back to later. Another instance of randomness is then injected through feature bagging, adding more diversity to the dataset and reducing the correlation among decision trees. Depending on the type of problem, the determination of the prediction will vary. For a regression task, the individual decision trees will be averaged, and for a classification task, a majority vote—i.e. the most frequent categorical variable—will yield the predicted class. Finally, the oob sample is then used for cross-validation, finalizing that prediction.




```
RandomForestRegressor
RandomForestRegressor(n_estimators=500, random_state=10)
```

## Metrics for Random Forest:

```
MAE : 0.24306649579792014
MSE : 0.1399720289279042
RMSE : 0.37412835889291285
R Squred : 0.8323144527973784
R Squared Adj : 0.044458548480298954
```

# Random Forest with GridSearchCV:

Before delving into the different kinds of processes available for hyperparameter tuning in Random Forest, let's take a look at the importance of hyperparameter tuning for random forest first.

Hyperparameter tuning in random forest is essential for the overall performance of the machine learning model. It is usually set before the learning process and occurs outside the model. So what happens when hyperparameter tuning random forest does not occur? Well, in such cases the model starts to produce errors and inaccurate results because the loss function does not get minimized. The ultimate goal of hyperparameter tuning random forest is to find a set of optimal hyperparameter values that will result in maximization of the model's performance, minimizing the loss and producing better output.

```
DecisionTreeRegressor
DecisionTreeRegressor(max_depth=25, max_leaf_nodes=30, min_samples_leaf=20,
                      min_samples_split=30, random_state=10)
```

## Metrics for the Random Forest with GridSearchCV:

```
MAE : 0.35798891601327243
MSE : 0.27039055744639906
RMSE : 0.5199909205422717
R Squred : 0.6760739346918017
R Squared Adj : -0.8458644038990986
```

The final metrics with respective to the models

| | Model | MSE | RMSE | R-Squared | Adj. R-Squared |
|---|---|---|---|---|---|
| 0 | Linear Regression | 0.192053 | 0.438239 | 0.943185 | 0.919751 |
| 1 | Decision Tree | 0.257588 | 0.507531 | 0.691412 | -0.758463 |
| 2 | Decision Tree with GridSearchCV | 0.293104 | 0.541391 | 0.648863 | -1.000922 |
| 3 | Random Forest | 0.139972 | 0.374128 | 0.832314 | 0.044459 |
| 4 | Random Forest with GridSearchCV | 0.270391 | 0.519991 | 0.676074 | -0.845864 |

# Conclusion

The laptops dataset is processed and the models are built, among the model LINEAR REGRESSION (OLS) has given the maximum accuracy of 94.3 %.

# References


https://www.analyticsvidhya.com/blog/2020/03/beginners-guide-random-forest-hyperparameter-tuning/

https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/

https://www.geeksforgeeks.org/regression-classification-supervised-machine-learning/

https://builtin.com/data-science/regression-machine-learning

https://www.amazon.in/b?ie=UTF8&node=16092374031