**Data Collection:**

For the project “Laptop Price Prediction”, the data has been collected from “Kaggle”. The author of the data specified that the data is “Uncleaned”. That’s why, the data is chosen to execute the entire pipeline from extensive data cleaning, preprocessing to finally real-time prediction.

**Data Review:**

First of all, the data has been downloaded in “CSV” format. The Data is uploaded to “Google Colab”.

Google Colab Workspace:

In Google colab, necessary libraries like “Pandas [For loading & data manipulation], Numpy[For Numerical Calculations], Seaborn, Matplotlib, Pylab [For Data Visualization] have been imported. Using the “df.read\_csv(“File Name”), data is stored in df “DataFrame”.

Basic overview of data consists of checking “Head” by df.head and “Tail” by df.tail. Furthermore, df.shape provides an overview of the number of columns and rows. And, df.info accumulates the rows of “null” values and respected columns data types.

Duplicate & Missing Values:

For our dataset, missing data are entirely row wise [sample]. That’s why we choose to delete the entire rows of missing values. For Duplicates items, we also choose this feature by utilizing df.dropna. In order to check the validity of the duplication and missing items, we used df.duplicated().sum() and df.isnull().sum().

Data Consistency & Entry-Related Errors:

In order to identify the data inconsistency and errors, we use the “df[“Column\_Name].unique()” feature to identify that. Once we have received any errors and inconsistencies, we either drop the entire column or else use the “Central Tendency” feature to replace the value. For the “Inches” column, we have deleted the rows consisting of error data. And, for the “Weight” column, we have made the data into mean value by assigning “NA” first and then taking the mean of the entire columns. The reason is that weight is an optimal choice for meaning as there is no significant diversion/outliers.

EDA for Pre-Processing & Feature Engineering:

For one of the feature columns named “Unnamed 0” has been dropped. It does not possess any information except the serial number.

CompanyName & TypeName:

For this column, all the companies have been plotted in “barplot” to see their effect on price”. Then, a unique company name[Categories] has been assigned to “Binary” for numerical computation in the Machine Learning Model. Because, categorical data is not a reliable way to feed into the ML model. Similar approach has been executed for the “TypeName” column as well.

RAM & Weight:

For Ram, data included GB in it. So, we removed the “GB” by replacing the String with “”. Similarly, for Weight, the “KG” is replaced with a similar approach. However, the following numerical values are actually stored in “String” format. That’s why, they have been converted to “Int” for Ram and Float for “Weight” as “Weight” comprises decimal values. This conversion has been done using df['Ram'].astype(int).

Then, we used a distplot to see the overall data distribution and we found most of the data falls within the 2-4 range. So, it’s expressing quite a normal distribution.

Price:

The price data is in Indian Rupee Format. The values are not normalized. In order to prove that, a distribution plot has been generated and we can see the price is “Rightly Skewed”. In such circumstances, we have transformed the data into natural log so normalize it. And, after the transformation, we have re-generated the distribution plot on logarithmic scale and we see the price distribution is normally distributed. The technique that has been used to normalize the data is np.log(df['Price']

Display Inches, Resolution, & PPI:

First, we have reviewed the inches data. We found that most of the data stays within a certain range. This has been done by creating scatter plots. Though, scatter plots are not an optimal visualization technique for this one.

For resolution, we have used “Python-Lambda expression” to separate the resolution for the string. However, the use of resolution by itself can not express any important features. Also, both the data was in “String” format. So, using “astype” converted them into integer values making it useful for PPI calculation.

That’s why, according to NPS(National Park Service) documentation, I went across the PPI. Which is a good measure about the picture quality. And, it is highly linked with display size and resolution. That’s why, PPI is calculated from display Inches and Resolution.

IPS, TouchScreen:

IPS, & Touchscreen has been separated by employing the “String Split” method. Then, we have assigned “binary encoding” to these individually to make it useful for machine learning models. Conventionally, laptops with touchscreen displays are highly correlated with higher price as well as “Display of IPS panel, OLED, AMOLED” panels have a good impact on price. Using the “Display”.valueCount() technique, we first reviewed the data and found the data are mostly divided into either touchscreen or IPS. That’s why the segmentation has been conducted. Furthermore, the impact of these types on price has been portrayed with “barplot” as well.

CPU:

CPU String is splitted into CPU Type with Brand Name and Clock Hertz. The limitation of this study is that we were unable to extract the “Generation” of the cpu due to missing “Generation” in multiple data samples. Clock Hertz data has been further splitted into only “Speed” by removing GHz and then converting the values into float. CPU type is stored as it is and barplot has been generated to see their individual relationship with price.

GPU:

For GPU, we have extracted the “Brand” wise data instead of “Model”. Models are multiple for each brand that would make the data more complex. For simplification, only brand names and their respective frequency has been calculated. However, “Intel” GPUs are mostly integrated GPUs whereas “AMD” & “NVidia: GPU" are external components. Then, GPU brands are encoded with binary encoding technique to ease further analysis.

Memory:

For the Memory column, as we have multiple data String, we created functions to create 4 columns on “SSD, HDD, Flash & Hybrid” and we have looped through the String and if any of a certain substring is found” we have placed its corresponding storage amount into that certain column”. This process has been done by creating another dataframe. After the process, the new dataframe has been merged with the old/main dataframe and the new dataframe has been dropped. After that, the values are converted into integers.

Operating System:

For OS, we have used value.count() and this resulted in multiple OS within Windows & MacOS. So, we used functions to accumulate all the versions of Windows OS and made the new column of Windows, MacOS or OtherOS category. Furthermore, this data has been encoded in binary to see the impact of each operating system.

After creating new suitable columns for further analysis, we have dropped some of the old columns to make the data look more eye-soothing. Furthermore, outliers have been identified while performing EDA and data table index has been reseted according to that.