**HDSC August ’22 Stage C Project Documentation: Web scraping and EDA using python**

By Sakshi Jha (Data Storytelling track)

**What is Web Scraping?**

Web scraping is basically used to describe the use of a program or algorithm to extract and process large amounts of data from the web. Whether you are in a profession of a data scientist, engineer, or anybody who analyses large amounts of datasets, the ability to scrape data from the web is a useful skill to have.

Scraped data extracted from data from a website, can information be collected and then exported into a format that is more useful for the user. Hence, helps collect these unstructured data and store it in a structured form. There are different ways to scrape websites such as online Services, APIs, or writing your own code.



**Applications Of Web Scraping:**

**Price Comparison:** Services such as Parse Hub use web scraping to collect data from online shopping websites and use it to compare the prices of products.

**Email address gathering:** Many companies that use email as a medium for marketing, use web scraping to collect email ID and then send bulk emails.

**Social Media Scraping:** Web scraping is used to collect data from Social Media websites such as Twitter to find out what’s trending.

**Research and Development:** Web scraping is used to collect a large set of data (Statistics, General Information, Temperature, etc.) from websites, which are analysed and used to carry out Surveys or for R&D.

**Job listings:** Details regarding job openings, interviews are collected from different websites and then listed in one place so that it is easily accessible to the user.

# How does Web Scraping work?

When you run the code for web scraping, a request is sent to the URL that you have mentioned. As a response to the request, the server sends the data and allows you to read the HTML or XML page. The code then, parses the HTML or XML page, finds the data, and extracts it.

To extract data using web scraping with python, you need to follow these basic steps:

1. Find the URL that you want to scrape
2. Inspecting the Page
3. Find the data you want to extract
4. Write the code
5. Run the code and extract the data
6. Store the data in the required format

**Aim of this project**: To extract data from the Flipkart website using Python. And then visualize it using various python libraries.

**Libraries used for Web Scraping:**

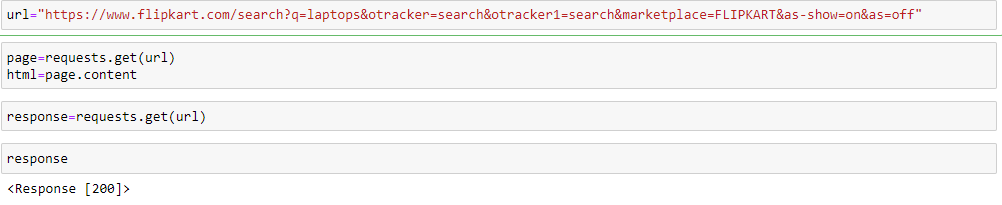
As we know, Python is used for various applications and there are different libraries for different purposes. In my further demonstration, the following libraries have been used:

* **BeautifulSoup**: Beautiful Soup is a Python package for parsing HTML and XML documents. It creates parse trees that are helpful to extract the data easily.
* **Pandas**: Pandas is a library used for data manipulation and analysis. It is used to extract the data and store it in the desired format.
* Use **matplotlib** and **seaborn** for data visualization

# Scrapped data from Flipkart website and here I have taken Laptop product

# Step 1: Find the URL that you want to scrape

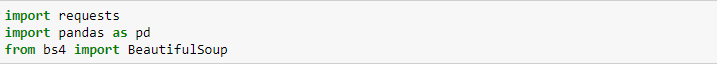
For this example, we are going to scrape the **Flipkart** website to extract the Name, Price, and Rating of Laptops. The URL for this page is <https://www.flipkart.com/search?q=laptops&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off&as-pos=1&as-type=HISTORY>.



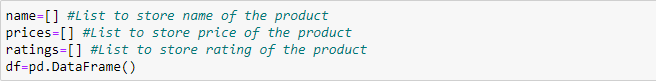
To check what our product’s optimal price should be, these prices can vary a lot. So, in this blog, I’m going to show how we can scrap data regarding a particular product (Laptop).

First, let us import all the necessary libraries:

open python file import libraries



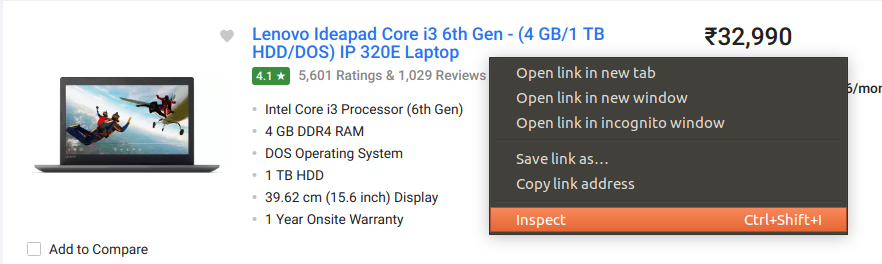
The data that we extract is unstructured data. So we’ll create empty lists to store them in a structured form,



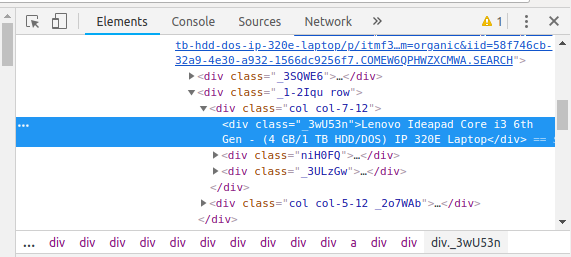
check the response of the URL using the above codes.

**Step 2: Inspecting the Page**

The data is usually nested in tags. So, we inspect the page to see, under which tag the data we want to scrape is nested. To inspect the page, just right click on the element and click on “Inspect”.



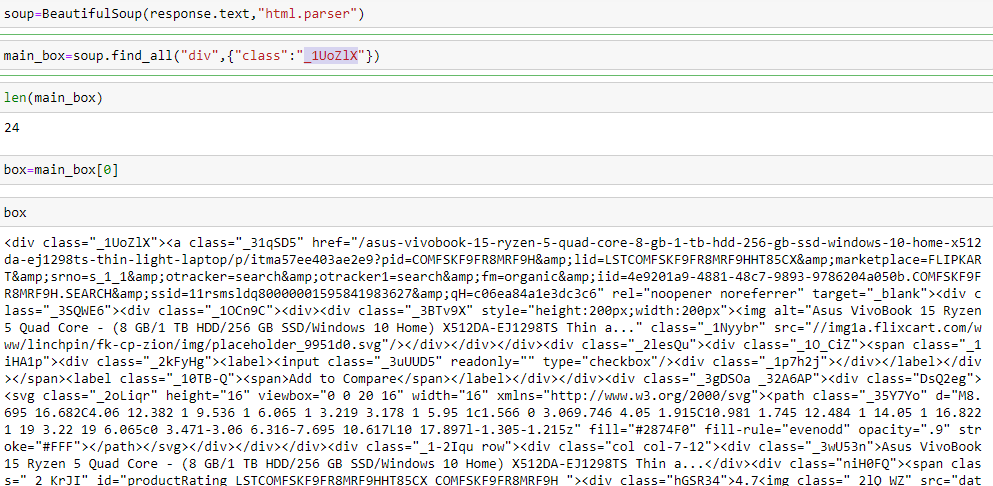
When you click on the “Inspect” tab, you will see a “Browser Inspector Box” open.



**Step 3: Find the data you want to extract**

Let’s extract the Name, Price, and Rating which is nested in the “div” tag respectively.

Now that we have written the code to open the URL, it’s time to extract the data from the website. As mentioned earlier, the data we want to extract is nested in <div> tags. So, I will find the div tags with those respective class-names, extract the data, and store the data in a variable. Refer to the code below:

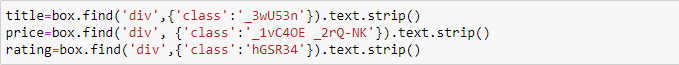


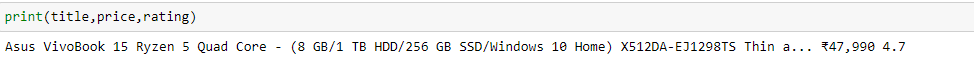
we need to use find all to extract all the details laptops and copy the tag code.

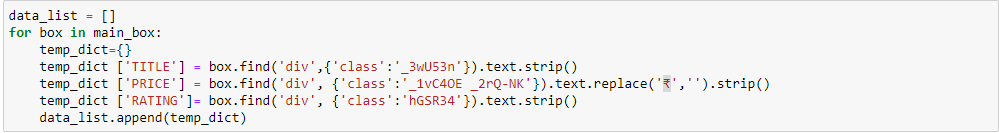
check the length of the code.

24 is no of items in that particular page.

# Step 5: Run the code and extract the data

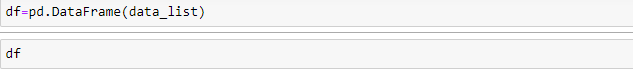




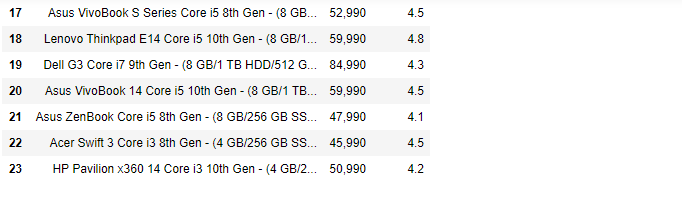


we need to remove ‘₹’ symbol from the price column we used text. replace command to remove that symbol from column

save the data\_list in data frame (df) format





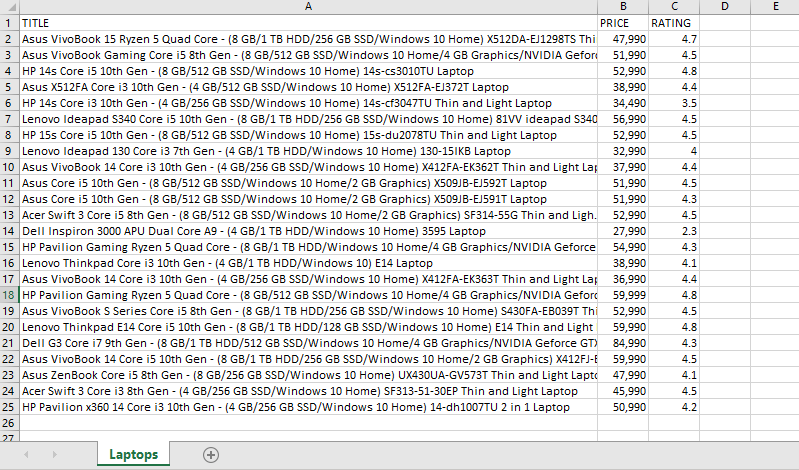


**Step 6: Store the data in a required format**

After extracting the data, you might want to store it in a format. This format varies depending on your requirement. For this example, we will store the extracted data in a CSV (Comma Separated Value) format. To do this, I will add the following lines to my code:

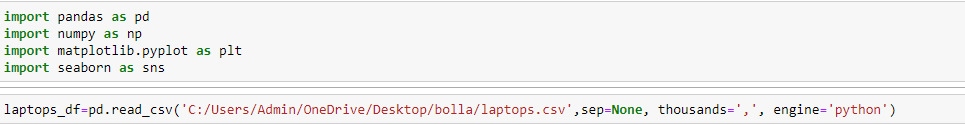


A file name “laptops.csv” is created and this file contains the extracted data.



**EDA and Visualization for this laptops.csv**

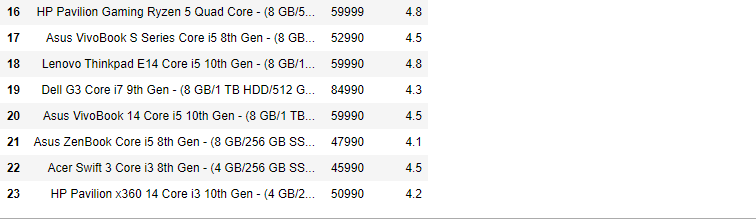
First, we need to open a new python file

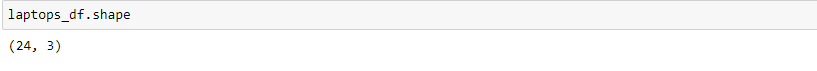


Here import the.CSV file from the location of the driveI mention thousands =’,’ to remove them, after the number in the price 47,990

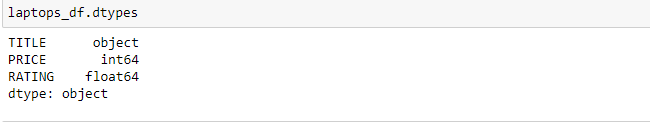






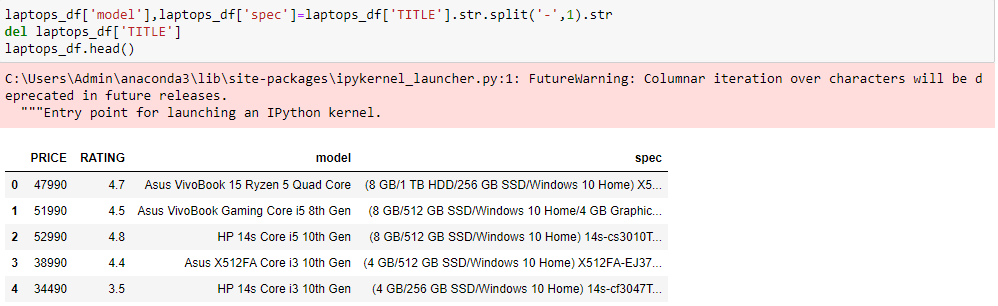


**Number of rows and columns**

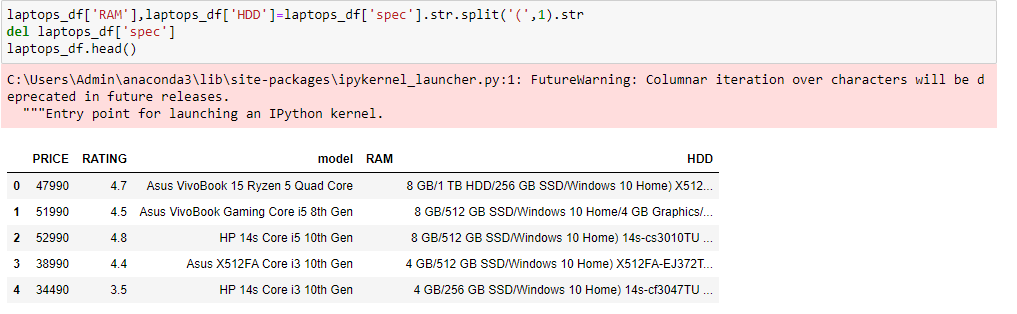


**Datatypes**

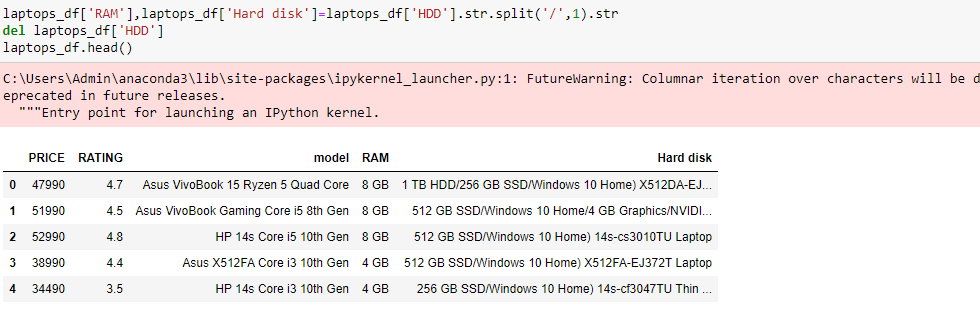
The title was split because of all specifications and laptop names in one row, split to different columns separately by using the split() option.



This split ‘-’ option to separate from the title and create a separate column as model and spec and del the title.



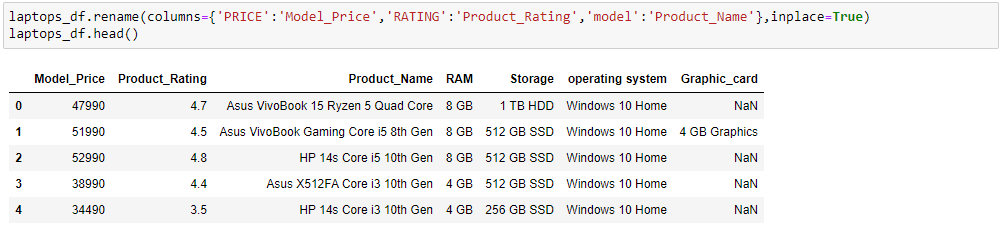
Also, here we use split ‘(‘option to remove the open bracket



here we use split ‘ /’ option to sperate from the hard disk

like this, a separate column was created for each model, ram, storage, operating system, graphic card.

we can rename the columns



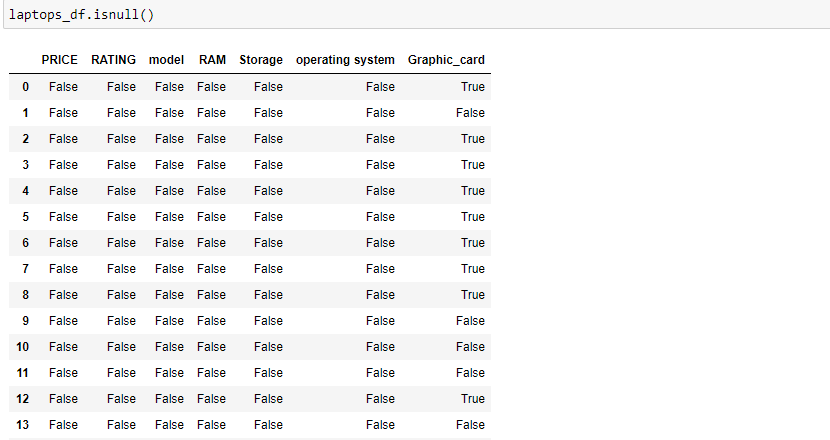
**Identifying Missing Values**

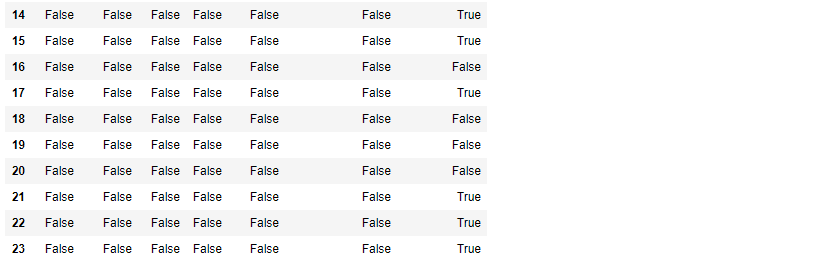
The methods isnull() and notnull() are the most common ways of identifying missing values.

While handling missing data, one first needs to identify the rows and columns containing missing values, count the number of missing values, and then decide how you want to treat them.

It is important that **you treat missing values in each column separately**, rather than implementing a single solution (e.g. replacing NaNs by the mean of a column) for all columns.

isnull() returns a boolean (True/False) which can then be used to find the rows or columns containing missing values.

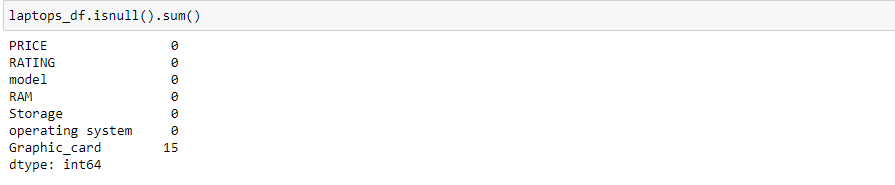




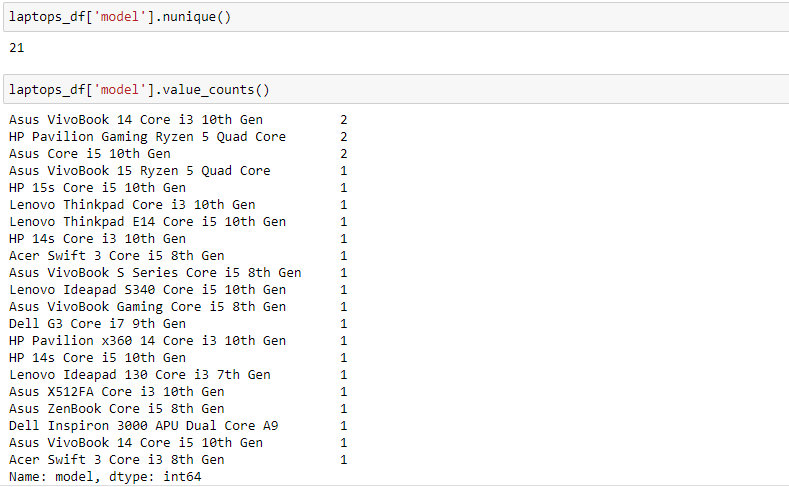
true means value is missing in that row

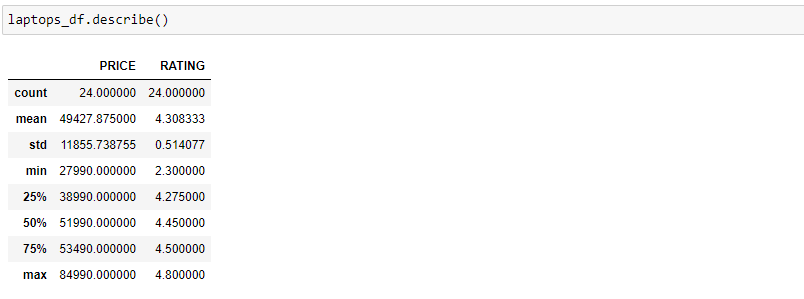
**Identifying Missing Values in Columns**

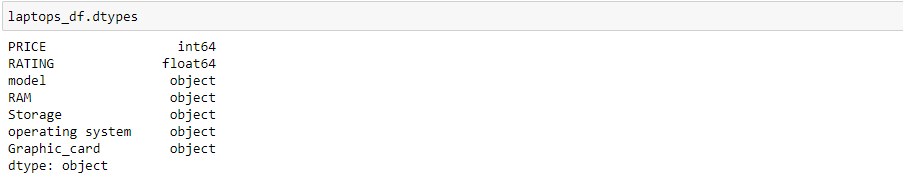
Let’s first compute the total number of missing values in the data frame. You can calculate the number of missing values in each column by df.isnull().sum()



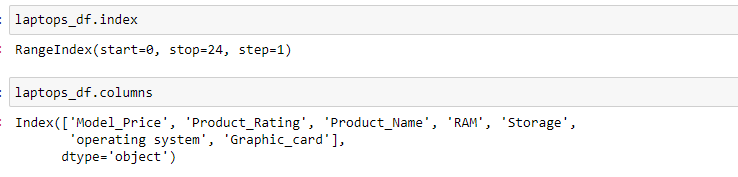
In this graphic card column, 15 items are missing



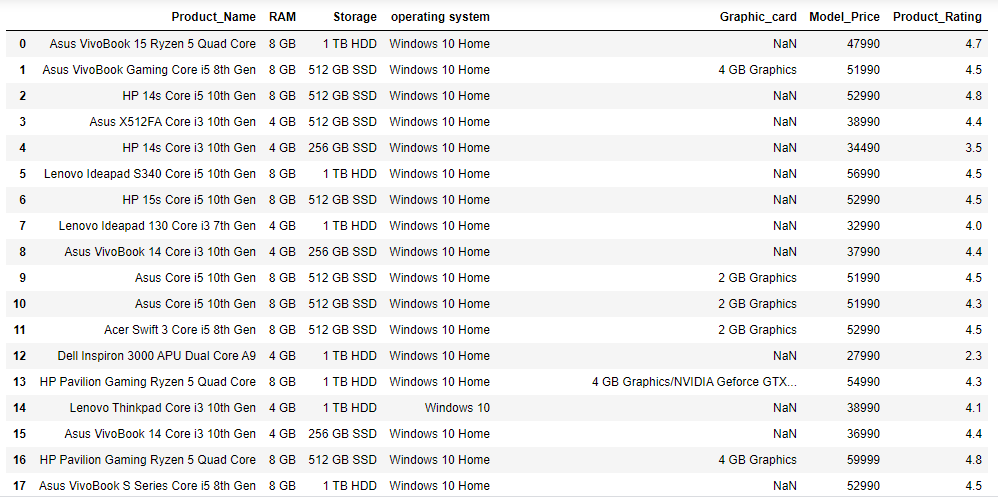


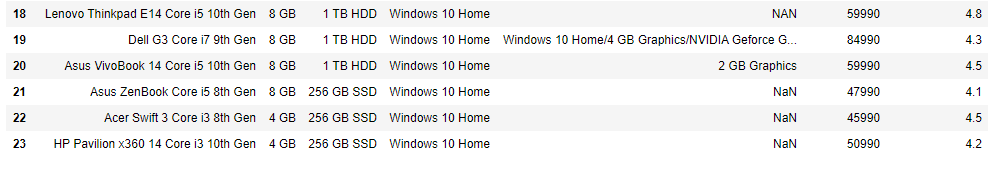


**Index of the columns:**

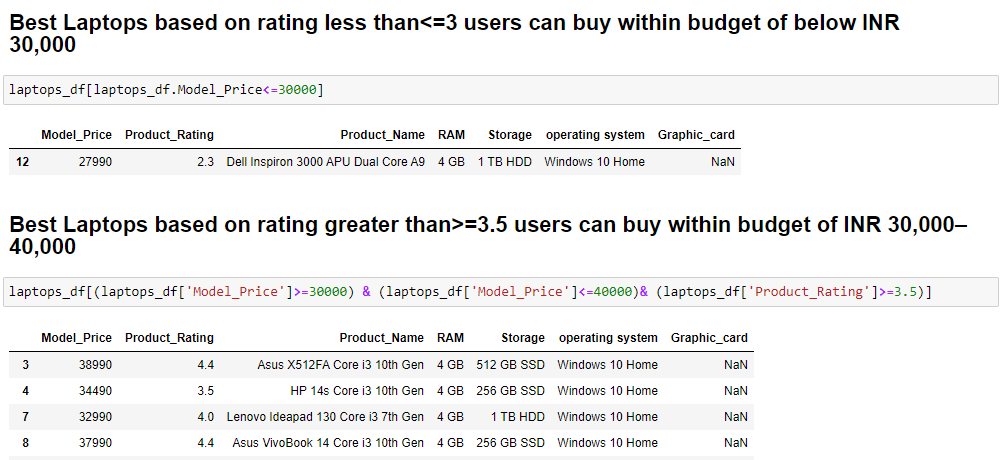


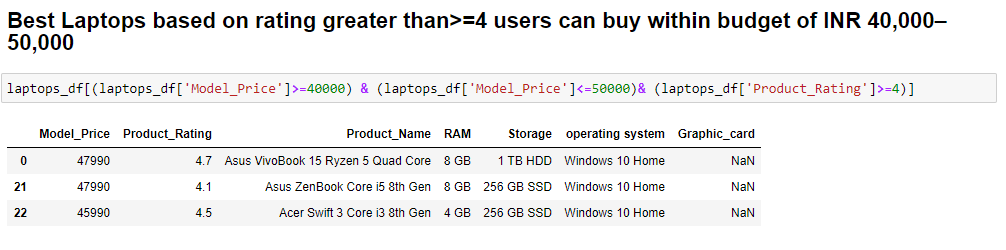
After data cleaning and processing the final list is

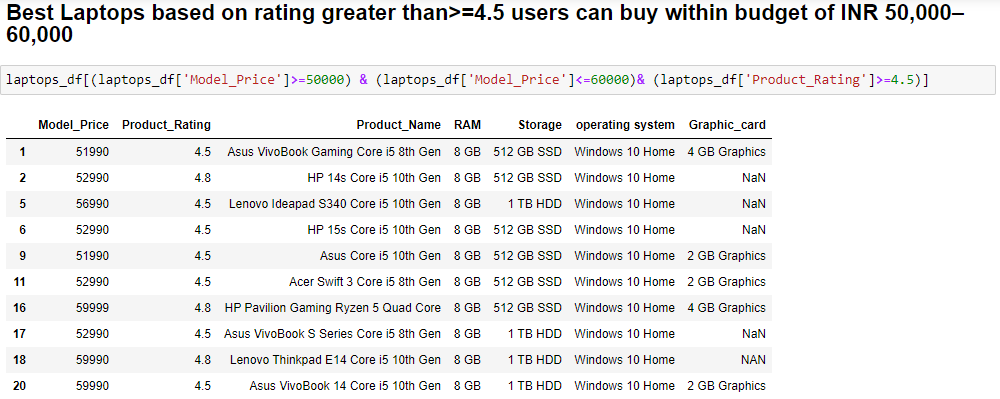


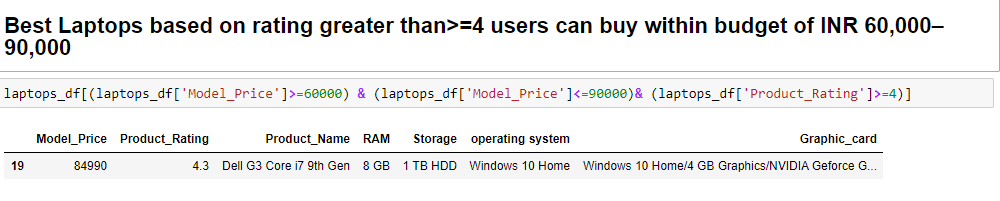


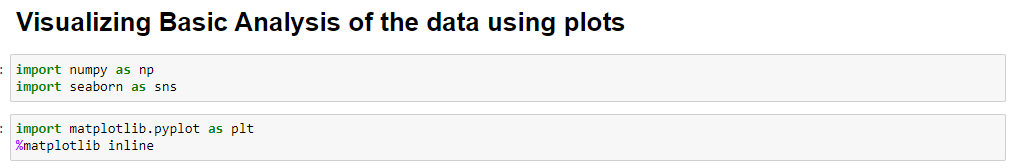
Filters for the best laptops with good specification, price, rating from the list:



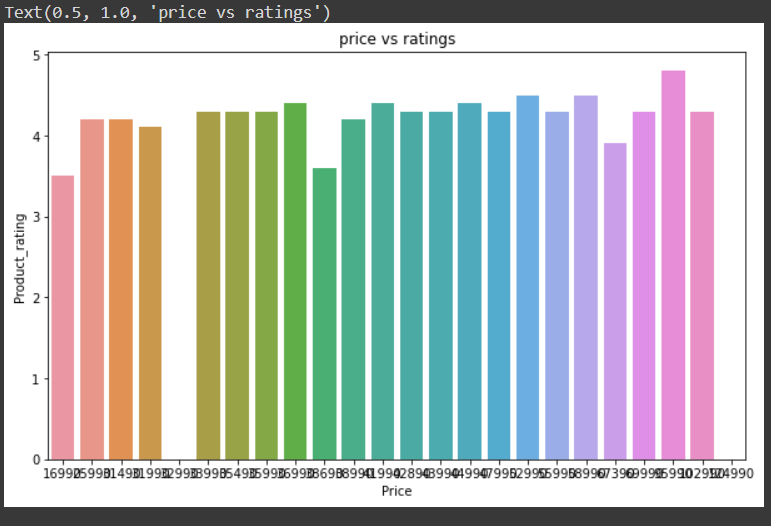


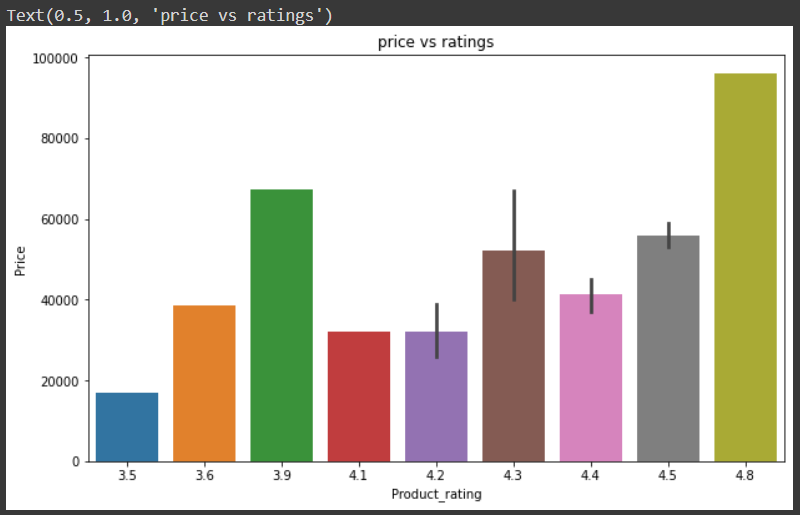






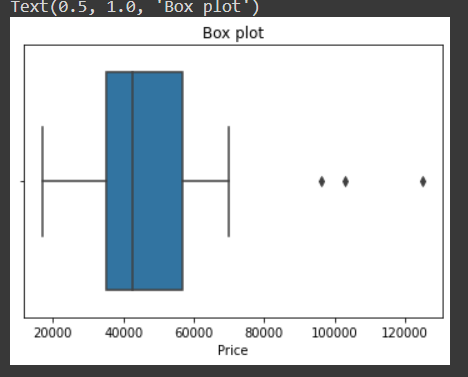
**Bar plot price vs rating:**





It can be concluded from here that products with higher prices have higher ratings to some extent.

**plotting a box plot for price**



As we can see there is one outlier where the price range is very high.

**Conclusion:**

From the above visualization, my analysis is that product available within 4.1,4.5,4.8 ratings with Price constraint 40,000 to 60,000.

**Project Outcome**

Here my aim was, by using Web scraping with Python to find out the best laptops by considering key factors like the budget amount of Rs40,000–60,000 and user product rating. Above are the products which customers can consider.