

Name :- Rishabh Sharma

Registration Number :- 20MAI0082

DA-1

Importing Libraries

```
In [1]: 1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 from bs4 import BeautifulSoup as bs
5 import requests
6 from warnings import filterwarnings
7 filterwarnings("ignore")
```

Using Web Scrapping to collect the data

```
In [2]: 1 while True:
2     try:
3         link = 'https://www.flipkart.com/search?q=laptops&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off'
4         print(link)
5         page = requests.get(link)
6         print(page)
7         break
8     except:
9         pass
```

<https://www.flipkart.com/search?q=laptops&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off> (<https://www.flipkart.com/search?q=laptops&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off>)
<Response [200]>

```
In [3]: 1 soup = bs(page.content, 'html.parser')
```

```
In [4]: 1 # items = soup.find_all('div', class_="_4rR01T")
2 # items=[i.get_text() for i in items]
3 # items = [i.split(" - ")[0] for i in items]
```

```
In [5]: 1 details = soup.find_all("li", class_="rgWa7B")
2 details = [i.get_text() for i in details]
```

```
In [6]: 1 lis,newlis=[],[]
2 for i in details:
3     if ("Intel") in i or ("AMD") in i or ("M1") in i :
4         if newlis:
5             lis.append(newlis)
6             newlis=[]
7             newlis.append(i)
8         else:
9             newlis.append(i)
10    else:
11        newlis.append(i)
12 newlisnew=[]
13 for i in lis:
14     newlisnew.append(" ".join(i))
```

```
In [7]: 1 items = []
2 for i in newlisnew:
3     if 'GB DDR4' in i:
4         items.append(i.split(" GB DDR4")[0][-2])
5     elif 'GB DDR3' in i:
6         items.append(i.split(" GB DDR3")[0][-2])
7     elif 'GB LPDDR4X' in i:
8         items.append(i.split(" GB LPDDR4X")[0][-2])
9     else:
10        items.append("")
11 company = [i.split()[0] for i in items]
```

```
In [8]: 1 pages= soup.find_all('a', class_='ge-49M', href=True)
2 pages = [str(i).split(">")[0][31:] for i in pages ]
```

```
In [9]: 1 ratings = soup.find_all("div",class_='_3LWZIK')
2 ratings = [i.get_text() for i in ratings]
```

```
In [10]: 1 prices = soup.find_all("div",class_="_30jeq3 _1_WHN1")
2 prices = [i.get_text().replace("₹","").replace(",","") for i in prices]
```

```
In [11]: 1 # details = soup.find_all("i",class_="rgWa7D")
2 # details = [i.get_text() for i in details]
```

```
In [12]: 1 # indepth_link = soup.find_all("a",class_ = '_1fQZEK')
```

```
In [13]: 1 # start= str(indepth_link[0]).index("href=")
2 # end = str(indepth_link[0]).index(" rel=")
3 # links = ['https://www.flipkart.com'+str(str(i)[start+6:end-1]) for i in indepth_link]
```

```
In [14]: 1 lis,newlis=[],[]
2 for i in details:
3     if ("Intel ")in i or ("AMD ") in i or ("M1") in i :
4         if newlis:
5             lis.append(newlis)
6             newlis=[]
7             newlis.append(i)
8         else:
9             newlis.append(i)
10    else:
11        newlis.append(i)
12 newlisnew=[]
13 for i in lis:
14     newlisnew.append(" ".join(i))
```

Collecting data and filtering everything

In [15]:

```
1 processor_brand=[]
2 processor = []
3 ram = []
4 ram_type=[]
5 os =[]
6 screen_size=[]
7 ssd_present=[]
8 ssd_capacity=[]
9 hdd_capacity=[]
10 # count=0
11 for i in newlisnew:
12     if "Intel" in i:
13         # print(count)
14         processor_brand.append("Intel")
15         processor.append(i.split()[1]+" "+i.split()[2])
16     elif "AMD" in i:
17         # print(count)
18         processor_brand.append("AMD")
19         processor.append(i.split()[1]+" "+i.split()[2])
20     elif "M1" in i:
21         # print(count)
22         processor_brand.append("M1")
23         processor.append(i.split()[1]+" "+i.split()[2])
24     else:
25         processor_brand.append("")
26         processor.append("")
27     # count+=1
28     if ' GB DDR4 RAM' in i:
29         index = i.index(' GB DDR4 RAM')
30         ram.append(i[index-2:index])
31         ram_type.append("DDR4")
32     elif ' GB DDR3' in i:
33         index = i.index(' GB DDR3 RAM')
34         ram.append(i[index-2:index])
35         ram_type.append("DDR3")
36     else:
37         ram.append("")
38     if ' Operating System' in i:
39         a = i.split()
40         index = a.index("Operating")
41         if a[index-1]=="10":
42             os.append("Windows 10")
43         elif a[index-2]=='Mac':
44             os.append("Mac Os")
45         else:
46             os.append(a[index-1])
47     else:
48         os.append("")
49     if "inch" in i:
50         a = i.split()
51         if 'inches' in a:
52             index = a.index("inches")
53         else:
54             index = a.index("inch")
55         screen_size.append(a[index-1].strip("(")+" inch")
56     if (" GB SSB" in i) or (" TB SSD" in i):
57         ssd_present.append("Yes")
58         if " GB SSB" in i:
59             start = i.index(" GB SSB")-3
60             end=i.index(" GB SSB")
61             ssd_capacity.append(i[start:end])
62         elif " TB SSD" in i:
63             index = i.index(" TB SSD")
64             ssd_capacity.append(int(i[index-1])*1024)
65     else:
66         ssd_present.append("No")
67         ssd_capacity.append("")
68     if " HDD" in i:
69         index = i.index(" HDD")
70         hdd_capacity.append(i[index-4])
71     else:
72         hdd_capacity.append("")
```

Creating a dataframe in order to carry out further steps

```
In [16]: data = pd.DataFrame({
    'items':items[:len(processor)],
    'company':company[:len(processor)],
    'ratings out of 5':ratings[:len(processor)],
    'prices':prices[:len(processor)],
    'processor_brand':processor_brand,
    'processor':processor,
    'ram':ram,
    'ram_type':ram_type,
    'operating_system':os,
    'screen_size':screen_size,
    'ssd_present':ssd_present,
    'ssd_capacity':ssd_capacity,
    'hdd_capacity_in_TB':hdd_capacity,
    'Purchased': [np.random.choice(["No", "Yes"]) for i in range(0,len(processor))]) # this is generated randomly whether the customer will buy a laptop or not
    data.to_csv("Dataset.csv",index=False) # Storing data into csv file
```

```
In [17]: 1 data.shape
```

Out[17]: (23, 14)

```
In [18]: 1 data =data.replace("",np.nan)
```

```
In [19]: 1 data.head()
```

Out[19]:

	items	company	ratings out of 5	prices	processor_brand	processor	ram	ram_type	operating_system	screen_size	ssd_present	ssd_capacity	hdd_capacity_in_TB
0	AMD Ryzen 5 Quad Core Processor (3rd Gen)	AMD	4.4	48990	AMD	Ryzen 5	8	DDR4	Windows 10	14 inch	Yes	256	1
1	Intel Core i3 Processor (10th Gen)	Intel	4	35990	Intel	Core i3	8	DDR4	Windows 10	15.6 inch	Yes	256	NaN
2	Intel Core i5 Processor (9th Gen)	Intel	4.5	52990	Intel	Core i5	8	DDR4	Windows 10	15.6 inch	Yes	512	NaN
3	Intel Core i3 Processor (10th Gen)	Intel	4.2	33490	Intel	Core i3	4	DDR4	Windows 10	15.6 inch	Yes	256	NaN
4	Intel Core i3 Processor (10th Gen)	Intel	4.2	35990	Intel	Core i3	8	DDR4	Windows 10	14 inch	Yes	256	NaN

```
In [20]: 1 data.isnull().sum()
```

Out[20]: items 0
company 0
ratings out of 5 0
prices 0
processor_brand 0
processor 0
ram 0
ram_type 0
operating_system 0
screen_size 0
ssd_present 0
ssd_capacity 4
hdd_capacity_in_TB 14
Purchased 0
dtype: int64

Filling null values

```
In [21]: 1 data = data.replace(np.nan,0)
2 data.head()
```

Out[21]:

	items	company	ratings out of 5	prices	processor_brand	processor	ram	ram_type	operating_system	screen_size	ssd_present	ssd_capacity	hdd_capacity_in_TB
0	AMD Ryzen 5 Quad Core Processor (3rd Gen)	AMD	4.4	48990	AMD	Ryzen 5	8	DDR4	Windows 10	14 inch	Yes	256	1
1	Intel Core i3 Processor (10th Gen)	Intel	4	35990	Intel	Core i3	8	DDR4	Windows 10	15.6 inch	Yes	256	0
2	Intel Core i5 Processor (9th Gen)	Intel	4.5	52990	Intel	Core i5	8	DDR4	Windows 10	15.6 inch	Yes	512	0
3	Intel Core i3 Processor (10th Gen)	Intel	4.2	33490	Intel	Core i3	4	DDR4	Windows 10	15.6 inch	Yes	256	0
4	Intel Core i3 Processor (10th Gen)	Intel	4.2	35990	Intel	Core i3	8	DDR4	Windows 10	14 inch	Yes	256	0

Using train test split and selecting features

```
In [22]: 1 from sklearn.model_selection import train_test_split
2 x = data.drop(["items", 'Purchased'], axis=1)
3 y = data[["Purchased"]]
```

Transforming string to numeric

```
In [23]: 1 from sklearn.preprocessing import LabelEncoder
2 le = LabelEncoder()
3 x["company"] = le.fit_transform(x["company"])
4 x["processor_brand"] = le.fit_transform(x["processor_brand"])
5 x["processor"] = le.fit_transform(x["processor"])
6 x["ram_type"] = le.fit_transform(x["ram_type"])
7 x["operating_system"] = le.fit_transform(x["operating_system"])
8 x["screen_size"] = le.fit_transform(x["screen_size"])
9 x["ssd_present"] = le.fit_transform(x["ssd_present"])
10 y["Purchased"] = le.fit_transform(y["Purchased"])
```

```
In [24]: 1 xtrain,xtest,ytrain,ytest = train_test_split(x,y,shuffle=True,test_size=0.1)
```

Using Logistic Regression

```
In [25]: 1 from sklearn.linear_model import LogisticRegression
2 lr = LogisticRegression()
3 lr.fit(xtrain,ytrain)
4 predictions = lr.predict(xtest)
```

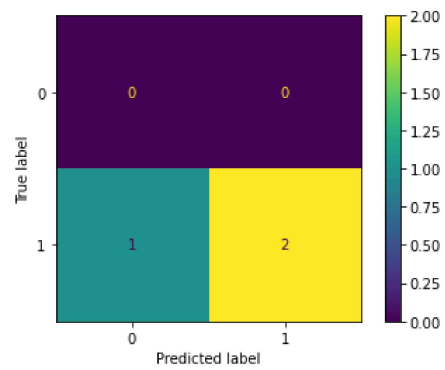
```
In [26]: 1 predictions
```

Out[26]: array([0, 1, 1])

Using Confusion Matrix and accuracy score to find how much better the algorithm performs

```
In [27]: 1 from sklearn.metrics import accuracy_score, plot_confusion_matrix
        2 plot_confusion_matrix(lr, xtest, ytest)
```

Out[27]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x2f92a0d4af0>



```
In [28]: 1 accuracy_score(predictions, ytest)
```

Out[28]: 0.6666666666666666