



RATINGS PREDICTION PROJECT

**Submitted by:
NIPAM GOGOI**

ACKNOWLEDGMENT

I would like to thank Flip Robo Technologies for providing me with the opportunity to work on this project from which I have learned a lot. I am also grateful to Miss Khushboo Garg for her constant guidance and support.

INTRODUCTION

BUSINESS PROBLEM FRAMING

We have a client who has a website where people write different reviews for technical products. Now they are adding a new feature to their website i.e. the reviewer will have to add stars (rating) as well with the review. The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars. Now they want to predict ratings for the reviews which were written in the past and they don't have rating. So we, we have to build an application which can predict the rating by seeing the review.

ANALYTICAL PROBLEM FRAMING

DATA COLLECTION - WEBSCRAPPING

```
jupyter rating_prediction_data_collection Last Checkpoint: Last Thursday at 12:11 PM (autosaved)
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3
In [1]: 1 #importing all the required libraries
2 import selenium
3 from selenium import webdriver
4 from selenium.webdriver.common.by import By
5 from selenium.webdriver.support.ui import WebDriverWait
6 from selenium.webdriver.support import expected_conditions as EC
7 import time
8 import requests
9 from bs4 import BeautifulSoup
10 import os
11 import pandas as pd
12 from selenium.common.exceptions import StaleElementReferenceException, NoSuchElementException, ElementNotInteractableException

In [2]: 1 # Opening the Webdriver
2 driver = webdriver.Chrome("C:\\chromedriver_win32\\chromedriver")

In [3]: 1 url='https://www.flipkart.com/hp-pavilion-gaming-ryzen-5-quad-core-3550h-8-gb-1-tb-hdd-windows-10-home-4-gb-graphics-nvidia-

In [4]: 1 #opening the webpage through our driver
2 driver.get(url)

In [5]: 1 #creating empty lists
2 Rating=[]
3 Full_review=[]

3 Full_review=[]

Scraping ratings and reviews of laptops

In [6]: 1 URL=[]
2 for j in range(0,147):
3     review_tags=driver.find_elements_by_xpath("//nav[@class='yFhi8N']")
4     for i in review_tags:
5         review_tags1=i.find_element_by_tag_name("a")
6         URL.append(review_tags1.get_attribute('href'))
7
8         button=driver.find_element_by_xpath("//*[contains(text(), 'Next')]")
9         driver.execute_script("arguments[0].click();", button)
10        time.sleep(3)

In [7]: 1 URL

Out[7]: ['https://www.flipkart.com/hp-pavilion-gaming-ryzen-5-quad-core-3550h-8-gb-1-tb-hdd-windows-10-home-4-gb-graphics-nvidia-gefo
rce-gtx-1650-15-ec0101ax-laptop/product-reviews/itma1af6bf593dc8?pid=COMF5FNVDXG74QXR&lid=LSTCOMF5FNVDXG74QXR8FRH2&marketpla
ce=FLIPKART&page=1',
'https://www.flipkart.com/hp-pavilion-gaming-ryzen-5-quad-core-3550h-8-gb-1-tb-hdd-windows-10-home-4-gb-graphics-nvidia-gefo
rce-gtx-1650-15-ec0101ax-laptop/product-reviews/itma1af6bf593dc8?pid=COMF5FNVDXG74QXR&lid=LSTCOMF5FNVDXG74QXR8FRH2&marketpla
ce=FLIPKART&page=1']
```

Out[16]: 1410

In [17]: 1 len(Full_review)

Out[17]: 1410

In [18]: 1 url='https://www.flipkart.com/apple-macbook-air-core-i5-5th-gen-8-gb-128-gb-ssd-mac-os-sierra-mqd32hn-a/product-reviews/itm0
2

In [19]: 1 #opening the webpage through our driver
2 driver.get(url)

In [20]: 1 for i in range(0,328):
2 for j in driver.find_elements_by_xpath("//div[@class='_3LWZlK _1BLPMq' or @class='_3LWZlK _32lA32 _1BLPMq' or @class='_3
3 Ratings.append(j.text)
4
5
6 for k in driver.find_elements_by_xpath("//div[@class='t-ZTKy']/div/div"):
7 Full_review.append(k.text.replace('\n', ' '))
8
9 time.sleep(1)
10 button=driver.find_element_by_xpath("//*[contains(text(), 'Next')]")
11 driver.execute_script("arguments[0].click();", button)
12
13 time.sleep(2)

In [21]: 1 len(Ratings)

Out[124]:



	Ratings	Full_review
0	5	Its an absolute beast if u know what are the n...
1	5	This is the best laptop in this range.I reciev...
2	5	Good product as used of now.... Everything is ...
3	5	AWESOME LAPTOP. It supports many high spec gam...
4	4	For that price... it's exceptionally good. Pla...
...
39364	5	This device completely covers my bungalow.. Goo...
39365	5	a great product from tenda,...i literally kee...
39366	3	Good router bt Flipkart fluctuates it's price....
39367	4	Too much gd...lokking nice and fit..range is a...
39368	5	Really great product, delivery was awesome. Al...

39369 rows × 2 columns


In [125]: 1 #converting into csv format
2 Reviews.to_csv("Rating_Prediction_data.csv")

In [126]: 1 #closing the driver
2 driver.close()

MODEL BUILDING AND EVALUATION

 jupyter Ratings_prediction_project_Model_Building Last Checkpoint: 7 hours ago (autosaved)  Login

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3



```
In [1]: 1 # importing all necessary modules
2 import pandas as pd
3 from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer,TfidfTransformer
4 from sklearn.ensemble import RandomForestClassifier ,AdaBoostClassifier
5 from sklearn.linear_model import LogisticRegression
6 from sklearn.metrics import precision_score, recall_score,accuracy_score, classification_report, f1_score ,confusion_matrix
7 from sklearn.model_selection import train_test_split
8 from wordcloud import WordCloud, STOPWORDS
9 import matplotlib.pyplot as plt

In [2]: 1 df = pd.read_csv("Rating_Prediction_data.csv")

In [5]: 1 df.columns
Out[5]: Index(['Unnamed: 0', 'Ratings', 'Full_review'], dtype='object')

In [6]: 1 df = df.drop("Unnamed: 0", axis=1)

2 def Word_Cloud(str_List):
3     comment_words = ''
4     stopwords = set(STOPWORDS)
5
6     # iterate through the csv file
7     for val in str_List:
8
9         # typecaste each val to string
10        val = str(val)
11
12        # split the value
13        tokens = val.split()
14
15        # Converts each token into lowercase
16        for i in range(len(tokens)):
17            tokens[i] = tokens[i].lower()
18
19        comment_words += " ".join(tokens)+" "
20
21
22        wordcloud = WordCloud(width = 800, height = 800,
23                               background_color = 'white',
24                               stopwords = stopwords,
25                               min_font_size = 10).generate(comment_words)
26
27        # plot the WordCloud image
28        plt.figure(figsize = (8, 8), facecolor = None)
29        plt.imshow(wordcloud)
30        plt.axis("off")
31        plt.tight_layout(pad = 0)
32
33        plt.show()
```

```
In [45]: 1 df.Ratings.unique()
```

```
Out[45]: array([5, 4, 3, 1, 2], dtype=int64)
```

```
In [47]: 1 for i in list(df.Ratings.unique()):  
2         print("RATINGS :", i)  
3         Word_Cloud(list(df[df.Ratings == i].sample(n=1500).Full_review))  
4         print()
```

RATINGS : 5



```
In [8]: 1 # Defining Input data and Output to be predicted  
2 X = df.Full_review  
3 y = df.Ratings
```

```
In [9]: 1 # Splitting the train_df into training and test dataset  
2 X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=88)
```

```
In [10]: 1 vectorizer = TfidfVectorizer(min_df =1,stop_words='english',use_idf=True,analyzer='word',  
2                                     ngram_range=(1,1),max_features=15000)  
3 x_train = vectorizer.fit_transform(X_train)  
4 x_test  = vectorizer.transform(X_test)
```

```
In [80]: 1 # Random Forest Model
```

```
In [11]: 1 rand = RandomForestClassifier(n_estimators=100,criterion='entropy',max_features=None,class_weight='balanced')  
2 rand.fit(x_train, y_train)
```

```
Out[11]: RandomForestClassifier(class_weight='balanced', criterion='entropy',  
                                max_features=None)
```

```
In [12]: 1 prediction = rand.predict(x_test)
```



```

In [65]: 1 df2 = df[df.Target != "Neutral"]

In [66]: 1 # Defining Input data and Output to be predicted
2 X2 = df2.Full_review
3 y2 = list(df2.Target)

In [67]: 1 # Splitting the train_df into training and test dataset
2 X_train2,X_test2,y_train2,y_test2=train_test_split(X2,y2,test_size=0.2,random_state=88)

In [68]: 1 vectorizer = TfidfVectorizer(min_df =1,stop_words='english',use_idf=True,analyzer='word',
2 ngram_range=(1,1),max_features=15000)
3 x_train2 = vectorizer.fit_transform(X_train2)
4 x_test2 = vectorizer.transform(X_test2)

In [69]: 1 rand = RandomForestClassifier(n_estimators=100,criterion='entropy',max_features=None,class_weight='balanced')
2 rand.fit(x_train2, y_train2)

Out[69]: RandomForestClassifier(class_weight='balanced', criterion='entropy',
max_features=None)

In [70]: 1 prediction = rand.predict(x_test2)

In [71]: 1 print('\n','CONFUSION MATRIX','\n',confusion_matrix(y_test2, prediction))
2 print('\n','ACCURACY','\n',accuracy_score(y_test2, prediction))
3 print('\n','REPORT','\n',classification_report(y_test2,prediction))

CONFUSION MATRIX
[[1127  324]
 [ 248 5538]]

```

```

In [70]: 1 prediction = rand.predict(x_test2)

In [71]: 1 print('\n','CONFUSION MATRIX','\n',confusion_matrix(y_test2, prediction))
2 print('\n','ACCURACY','\n',accuracy_score(y_test2, prediction))
3 print('\n','REPORT','\n',classification_report(y_test2,prediction))

CONFUSION MATRIX
[[1127  324]
 [ 248 5538]]

ACCURACY
0.9209617244714661

REPORT

```

	precision	recall	f1-score	support
0	0.82	0.78	0.80	1451
1	0.94	0.96	0.95	5786
accuracy			0.92	7237
macro avg	0.88	0.87	0.87	7237
weighted avg	0.92	0.92	0.92	7237


```

2 logisticRegr.fit(x_train2, y_train2)
Out[72]: LogisticRegression(class_weight='balanced', max_iter=1000, random_state=5,
                             solver='liblinear', tol=0.001)

In [73]: 1 prediction2 = logisticRegr.predict(x_test2)

In [74]: 1 print('\n','CONFUSION MATRIX','\n',confusion_matrix(y_test2, prediction2))
          2 print('\n','ACCURACY','\n',accuracy_score(y_test2, prediction2))
          3 print('\n','REPORT','\n',classification_report(y_test2,prediction2))

```

CONFUSION MATRIX

```
[[1263 188]
 [ 449 5337]]
```

ACCURACY

0.9119801022523145

REPORT

	precision	recall	f1-score	support
0	0.74	0.87	0.80	1451
1	0.97	0.92	0.94	5786
accuracy			0.91	7237
macro avg	0.85	0.90	0.87	7237
weighted avg	0.92	0.91	0.91	7237

In []:

1

CONCLUSION

- The Data was collected using webscrapping from Flipkart for various products, like Laptop, mobiles etc. There are 2 columns in the Dataset: Ratings and Reviews.
- EDA was done on the dataset and Word Clouds were created for all types of Reviews.
- The ML models were not performing well for predicting all the ratings. Since its hard to distinguish between neutral reviews and good or bad reviews so results in inaccurate prediction of Ratings.
- Hence transformed into binary classification problem, taking 1 or 2 ratings as Bad ratings and 4 or 5 as good ratings. The final models performed well for this set up with around 91-92 % accuracy.
- We can directly rate the cases to predicted as 5, 3 or 1 on the unseen data as ratings based on review.