RATING PREDICTION PROJECT

Submitted By:

ABHISHEK SHAHI

ACKNOWLEDGMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to Flip Robo Technologies Bangalore for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I want to thank my SME **Khushboo Garg** for providing the project and helping us to solve the problem and addressing out our Query in right time.
I would like to express my gratitude towards my parents & members of Flip Robo for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons
for giving me such attention and time .

INTRODUCTION

Rating of a Product defines the feedback or improvement of quality, comfortness, price, demand for a customer. Customers gives reviews on their purchased products on the product selling or ecommerce websites. Along with the review customers gives rating to the products out of 5 or out of 10. Normally, Out of 5 is used widely for rating of the products or even services. Companies watch these ratings and reviews and try to resolve the problem with customer, improvement of the product or services in future.

Business Problem Framing:

A particular review can tell whether a customer is satisfied with the product or not. A rating can directly indicates the likeliness of the product for the customer.

eg. A customer gives 1-2 stars out of 5 stars to product, then customer is not satisfied at all. It defines this. 3 star is moderate neither happy nor sad. 4 star defines product is good just need an improvement. 5 star defines the product is awesome and normally company ignores 5 star rating reviews. Because they know the customer is fully satisfied with the product.

Conceptual Background of the Domain Problem:

Therefore it is important a product should have average rating value which will help the product to rank with its competitors. In some cases customers don't give review but only give the rating. Here, the company's problem is a solved but if it is a low rating, then they try to find out what will upset the customer by their product. Similarly, in some company's websites the customers give the reviews without a rating. At that case company's gets the problem in their product by

Reading the reviews but they can't sort easily which is a good review or bad review without any rating points.

To overcome this problem Machine Learning along with Natural Language Processing is used to predict the rating for a particular review.

Review of Literature:

This project report will help to predict the rating of the customer's review posted for a certain product in its website

Objective:

To Build a model which can be used to predict the rating in the range of 1-5 to sort out the reviews in positive and negative portions. Also some reviews which are either 3-4 can be categorized for improvement the quality of the product.

Hardware and Software Requirements and Tools Used:

- Anaconda 2020.07
- Jupyter Notebook 6.0.3
- Python 3.8.3
- Numpy
- Pandas
- Scikit Learn
- Matplotlib
- Warnings
- Seaborn
- Machine Learning
- Natural Language Processing
- Regular Expressions
- Data Science

Analytical Problem Framing

Data Sources and their formats:

The Dataset of Rating and Review is prepared by Web Scraping of several electronics products such as laptops, Phones, Headphones, smart watches, Professional Cameras, Printers, monitors, Home theater, router from Amazon and Flipkart.

The Dataset contains Ratings and Reviews. There were also some null values. Each product data from each website is scrapped separately using four functions. Two functions were created for Amazon and other two for Flipkart. The need of functions from each website depended upon the target webpage architectures for each product on each website.

The basic Function used in Web Scraping for Amazon:

In [43]:

```
def amazon(user_ip):
    driver = webdriver.Chrome(r"F://chromedriver.exe")
    driver.get('https://www.amazon.in/')
    search_bar = driver.find_element_by_id("twotabsearchtextbox")
    search_bar.clear()
    search_bar.send_keys(user_ip)
    search_button = driver.find_element_by_xpath('//div[@class="nav-search-submit nav-sprite"]/span/input')
    search_button.click()

    HREF=[]
    n=0

    while True:

        for i in driver.find_elements_by_xpath("//h2/a"):
            HREF.append(i.get_attribute('href'))
            if len(HREF)==50:
                break

        if len(HREF)==50:
            break

        for j in driver.find_elements_by_xpath("//li[@class='a-last']/a"):
            next_page=[j.get_attribute("href")]

        for n in next_page:
            driver.get(n)

    driver.close()
    Ratings=[]
    Reviews=[]

    for i in HREF:
        driver = webdriver.Chrome(r"F://chromedriver.exe")
        driver.get(i+'#customerReviews')
        driver.maximize_window()

        time.sleep(2)

        STAR=[]

        for s in driver.find_elements_by_xpath("//td[@class='aok-nowrap']/span[@class='a-size-base']/a"):
            STAR.append(s.get_attribute('href'))

        for q in STAR:
            driver.get(q)

            try:
                for rh in driver.find_elements_by_xpath("//div[@class = 'a-row']/a[@class='a-size-base a-link-normal review-
                    Reviews.append(rh.text)

            except:
                continue
            try:
                for rt in driver.find_elements_by_xpath("//div[@class = 'a-row']/a"):
                    Ratings.append(rt.get_attribute('title'))

            except:
                Ratings.append('-')

        driver.close()


    Ratings=Ratings[0:len(Ratings):2]


    Rating_Review=pd.DataFrame({"Rating":Ratings,
                                "Review":Reviews})


    return Rating_Review
```

The Basic Function used in Flipkart:

In [37]:

```
def flipkart(user_ip):

    #Calling the driver to load the webpage
    driver=webdriver.Chrome(r"F://chromedriver.exe")
    driver.get("https://www.flipkart.com/")
    driver.maximize_window()

    #Closing the Login Pop-up
    time.sleep(1)
    Login_pop_up = driver.find_element_by_xpath("//div[@class='_2QfC02']/button[@class = '_2KpZ6l _2doB4z']")
    Login_pop_up.click()

    product_search = driver.find_element_by_xpath("//div[@class='_3OO5Xc']//input[@type = 'text']")
    product_search.send_keys(user_ip)

    search_button = driver.find_element_by_xpath("//div[@class = 'col-12-12 _2oO9oE']//button[@class = 'L0Z3Pu']")
    search_button.click()

    href=[]
    Review_details=[]
    Rating=[]
    Review=[]
    next_page=[]


    while True:

        for i in driver.find_elements_by_xpath("//a[@class='_1fQZEK']"):
            href.append(i.get_attribute('href'))
            if len(href) == 50:
                break

        if len(href) == 50:
            break

        for j in driver.find_elements_by_xpath("//a[@class = '_1LKTO3']"):
            next_page.append(j.get_attribute("href"))

        for n in next_page:
            driver.get(n)

    driver.close()
    for i in href:
        driver=webdriver.Chrome(r"F://chromedriver.exe")
        driver.get(i)

        try:
            All_Reviews=driver.find_elements_by_xpath("//div[@class='col JOpGWq']/a")
            AR=All_Reviews[0].get_attribute('href')
            driver.get(AR)
            driver.maximize_window()

        except:
            driver.close()
            continue


        try:
            for rev in driver.find_elements_by_xpath("//div[@class='col _2wzgFH K0kLPL']"):
                Review_details.append(rev.text)

        except:
            continue


        NRV=driver.find_elements_by_xpath("//select[@class='_1EDlbo tVKh2S']//option[@value='NEGATIVE_FIRST']")
        NRV[0].click()
```

```
        try:
            for nr in driver.find_elements_by_xpath("//div[@class='col _2wzgFH K0kLPL']"):
                Review_details.append(nr.text)

        except:
            continue

        driver.close()

    for i in range(len(Review_details)):
        Review_details[i]=Review_details[i].split('\n')


    for i in range(len(Review_details)):
        Rating.append(Review_details[i][0])
        Review.append(Review_details[i][1])


    Rating_Review=pd.DataFrame({"Rating":Rating,
                                "Review":Review})


    return Rating_Review
```

The Datasets from Amazon are scraped and saved as CSV files. These are named as follows with its shape excluding index column:

Dataset Name	Shape
Laptop_Rating_Review_Amazon.csv	(1315 × 2)
Phones_Rating_Review_Amazon.csv	(2205 x 2)
Headphones_Rating_Review_Amazon.csv	(2183 x 2)
Smart_Watches_Rating_Review_Amazon.csv	(1628 x 2)
Professional_Cameras_Rating_Review_Amazon.csv	(893 x2)
Printers_Rating_Review_Amazon.csv	(1768 x2)
Monitors_Cameras_Rating_Review_Amazon.csv	(1284 x 2)
Home_Theater_Rating_Review_Amazon.csv	(1844 x 2)
Router_Rating_Review_Amazon.csv	(2098 x 2)

These Datasets has 2 columns and those are Rating and Review. These are merged to make one named and saved as Rating_Review_Amazon.csv.
This Dataset has now 15218 rows and 2 columns.

Similarly from Flipkart some datas are scraped for same products as in Amazon and datas are saved as CSV files. These datasets are named as follows along with its shape excluding index column.

Dataset Name	Shape
Laptop_Rating_Review_Flipkart.csv	(778 × 2)
Phones_Rating_Review_ Flipkart.csv	(986 x 2)
Headphones_Rating_Review_ Flipkart.csv	(898 x 2)
Smart_Watches_Rating_Review_ Flipkart.csv	(576 x 2)
Professional_Cameras_Rating_Review_ Flipkart.csv	(647 x2)
Printers_Rating_Review_ Flipkart.csv	(610 x2)
Monitors_Cameras_Rating_Review_ Flipkart.csv	(546 x 2)
Home_Theater_Rating_Review_ Flipkart.csv	(532 x 2)
Router_Rating_Review_ Flipkart.csv	(968 x 2)

These Datasets have also two columns as Rating and Review. These are merged to make one Dataset named and saved as Rating_Review_Flipkart.csv.
This Dataset has now 6541 rows and 2 columns.

Data Preprocessing along with Mathematical Modeling:

Amazon and Flipkart Datasets are fed into Data Preprocessing and then merged to one after matching each other by performing following operations.

EDA Process

Both the dataframes have a extra column named as 'Unnamed:0'. So it will be dropped from both the dataframes.

```
In [5]:
1 Rating_Review_Amazon.drop(['Unnamed: 0'],axis=1,inplace=True)
2 Rating_Review_Flipkart.drop(['Unnamed: 0'],axis=1,inplace=True)
```

Checking Null Values

```
In [6]:
1 Rating_Review_Amazon.isnull().sum()

Out[6]: Rating     8
        Review    16
        dtype: int64
```

Rating_Review_Amazon has null values.

```
In [7]:
1 Rating_Review_Flipkart.isnull().sum()

Out[7]: Rating    0
        Review    0
        dtype: int64
```

Filling the Null values using mode.

Filling the Null Values

As both the columns of Rating_Review_Amazon are object type. So the null values will be filled with mode of the feature.

```
In [8]:  Rating_Review_Amazon['Rating'].mode()
```

```
Out[8]: 0    5.0 out of 5 stars
        dtype: object
```

```
In [9]:  Rating_Review_Amazon['Rating'].fillna(value='5.0 out of 5 stars',inplace=True)
```

```
In [10]: Rating_Review_Amazon['Review'].mode()
```

```
Out[10]: 0    Good
         dtype: object
```

```
In [11]: Rating_Review_Amazon['Review'].fillna(value='Good',inplace=True)
```

```
In [12]: Rating_Review_Amazon.isnull().sum()
```

```
Out[12]: Rating    0
         Review    0
         dtype: int64
```

Converting Rating Column from String to Integer.

```
In [13]: Rating_Review_Amazon['Rating'].unique()
```

```
Out[13]: array(['5.0 out of 5 stars', '4.0 out of 5 stars', '3.0 out of 5 stars',
                '2.0 out of 5 stars', '1.0 out of 5 stars'], dtype=object)
```

```
In [14]: Rating_Review_Flipkart['Rating'].unique()
```

```
Out[14]: array([4, 5, 3, 1, 2], dtype=int64)
```

Here, we have to merge both the dataframes to create a large database to fed to model for rating prediction having good results.

But in Rating_Review_Amazon, the Rating values are not in the form of a single digit as in Rating_Review_Flipkart. So we have convert and merge.

```
In [15]: for i in range(len(Rating_Review_Amazon)):
             Rating_Review_Amazon['Rating'][i]=Rating_Review_Amazon['Rating'][i][0]

         Rating_Review_Amazon['Rating'].unique()
```

```
Out[15]: array(['5', '4', '3', '2', '1'], dtype=object)
```

Now we will convert the Rating values of Rating_Review_Amazon from object to integer.

```
In [16]: Rating_Review_Amazon['Rating']=Rating_Review_Amazon['Rating'].astype(int)
```



```
In [17]: Rating_Review_Amazon['Rating']=np.int64(Rating_Review_Amazon['Rating'])
         Rating_Review_Amazon.dtypes
```

Now the Dataframes of Amazon and Flipkart are merged into one for the required dataset for model building.

In [20]:
```
Frames=[Rating_Review_Amazon,Rating_Review_Flipkart]
Rating_Review=pd.concat(Frames,ignore_index=True)
Rating_Review
```

Out[20]:

	Rating	Review
0	5	Grab before it's Gone!
1	5	Ryzen 4500u with upgradeable internals
2	5	Best Deal on Laptop I've ever got
3	5	Best value for money thinkpad e14
4	5	Wonderful Laptop from Thinkpad ,just Grab it w...
...	...	...
21754	5	Must buy!
21755	5	Classy product
21756	5	Best in the market!
21757	5	Perfect product!
21758	5	Must buy!

21759 rows × 2 columns

Rating_Review is the desired DataFrame. The describe() method shows median is higher than mean. And it is highly left skewed as 75th percentile and 100th percentile are same.

In [25]:
```
Rating_Review.describe()
```

Out[25]:

	Rating
count	21759.000000
mean	3.463257
std	1.486533
min	1.000000
25%	2.000000
50%	4.000000
75%	5.000000
max	5.000000

Model Development and Evaluation

Identification of possible problem-solving approaches (methods):

The First target to convert the reviews into numericals so that the machine can learn it.

The First thing is applied for the Review column is shortening the words using Lemmatizer, sorting, splitting and joining using regular expressions. This whole process is a part of **Natural Language Processing (NLP)**.

Now we have to start preprocessing of Reviews.

First of all the review texts will cleaned to grab the important words.

Stopwords removal is not applied here as for negative reviews stopwords like not groups are mostly used.

In [27]:
```
from nltk.stem import WordNetLemmatizer   #Lemmatizer is used to retrieve the base word
review=[]

for i in range(len(Rating_Review)):
    rev=re.sub('[^a-z A-Z]','',Rating_Review['Review'][i])  #sorting only alphabets
    rev=rev.lower()          #Converting all texts to lowercase
    rev=rev.split()          #spliting to access each word(just like tokenization)
    wl = WordNetLemmatizer()
    rev=[wl.lemmatize(word) for word in rev]
    rev=' '.join(rev)
    review.append(rev)

review
```

Out[27]:
```
['grab before it gone',
 'ryzen u with upgradeable internals',
 'best deal on laptop ive ever got',
 'best value for money thinkpad e',
 'wonderful laptop from thinkpad just grab it without a second thought',
 'great business laptop',
```

Here the review represents a list of small sentences containing important words. Here, Stopwords removal wasn't done because some stopwords like not groups are very vital for negative reviews.

This review list is then fed to a vectorizer named as **Term Frequency Inverse Document Frequency (Tf-Idf)** Vectorizer**.** This generated a matrix of same rows as in Dataframe and each word in the review is a feature. So there is 4678 words making dummy columns.

In [28]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

TV=TfidfVectorizer(smooth_idf=False)
```

In [29]:
```
def matrix(message, countvect):
    terms_doc = countvect.fit_transform(message)
    return pd.DataFrame(terms_doc.toarray(),columns=countvect.get_feature_names())
```

In [30]:
```
TV_review=matrix(review,TV)
```

In [31]:
```
TV_review
```

Out[31]:

	aa	aacha	aad	aandhon	aata	aayi	abhi	able	abnormal	about	...	zebratronic	zebronic	zebronics	zebswrucf	zebthunder	zebwarrrior	zero	zoor
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.

Now the Input Variable is ready. This is the Input variable generated from the Review column.

In [32]:
```
x=TV_review    #Input Variable
x
```

Out[32]:

	aa	aacha	aad	aandhon	aata	aayi	abhi	able	abnormal	about	...	zebratronic	zebronic	zebronics	zebswrucf	zebthunder	zebwarrrior	zero	zoor
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
21754	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
21755	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
21756	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
21757	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
21758	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.

21759 rows × 4678 columns

Target Variable is the Rating was prepared for Modeling.

```
In [35]:
1 y=pd.DataFrame(Rating_Review.iloc[:,0])   #Target Variable
2 y
```

Out[35]:

	Rating
0	5
1	5
2	5
3	5
4	5
...	...
21754	5
21755	5
21756	5
21757	5
21758	5

21759 rows × 1 columns

This Model is Classification Model as Output will be 1,2,3,4,5.

The whole data Input and Target Variable were splited into two parts Train data and Test data using **train_test_split** function.

train_x,test_x,train_y,test_y=train_test_split(x,y,test_size=0.30,random_state=42)

Testing of Identified Approaches (Algorithms):

I have trained the model with seven Classification Algorithms with ensemble technique which are as follows:

- Logistic Regression
- KNeighbors Classifier
- Decision Tree Classifier
- Support Vector Classifier(SVC)
- MultinomialNB
- Random Forest Classifier
- Ada Boost Classifier
- Gradient Boosting Classifier

Run and Evaluate selected models:

Multiple Algorithms

In [70]:

```
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier,GradientBoostingClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report

train_x,test_x,train_y,test_y=train_test_split(x,y,test_size=0.30,random_state=42)
modelclf=[LogisticRegression(),KNeighborsClassifier(),DecisionTreeClassifier(),MultinomialNB(), SVC(),
          RandomForestClassifier(),AdaBoostClassifier(),GradientBoostingClassifier()]

for mc in modelclf:
    mc.fit(train_x,train_y)
    mc_y=mc.predict(test_x)
    print("Accuracy Score of",mc,"is",accuracy_score(test_y,mc_y))
    print("Confusion Matrix\n",confusion_matrix(test_y,mc_y))
    print("Classification Report\n", classification_report(test_y,mc_y))
    print("**********************************************************************")
    print("\n\n")
```

After Splitting into train_test_split, all algorithms are applied to know the best accuracy in which model. There are three parts in measuring accuracy.

The best results are as follows:

```
Accuracy Score of SVC() is 0.6588541666666666
Confusion Matrix
 [[ 836  127   79   36   47]
 [ 233  353  127   42   51]
 [ 156  133  451  140   86]
 [  54   39  168  743  344]
 [  87   30   63  185 1918]]
Classification Report
              precision    recall  f1-score   support

           1       0.61      0.74      0.67      1125
           2       0.52      0.44      0.47       806
           3       0.51      0.47      0.49       966
           4       0.65      0.55      0.60      1348
           5       0.78      0.84      0.81      2283

    accuracy                           0.66      6528
   macro avg       0.61      0.61      0.61      6528
weighted avg       0.65      0.66      0.65      6528

**********************************************************************
```

```
Accuracy Score of RandomForestClassifier() is 0.6565563725490197
Confusion Matrix
 [[ 787  158   80   38   62]
 [ 210  379  113   42   62]
 [ 122  139  478  133   94]
 [  58   42  151  779  318]
 [  71   36   83  230 1863]]
Classification Report
              precision    recall  f1-score   support

           1       0.63      0.70      0.66      1125
           2       0.50      0.47      0.49       806
           3       0.53      0.49      0.51       966
           4       0.64      0.58      0.61      1348
           5       0.78      0.82      0.80      2283

    accuracy                           0.66      6528
   macro avg       0.62      0.61      0.61      6528
weighted avg       0.65      0.66      0.65      6528

**********************************************************************
```

Here, SVC(Support Vector Classifier) gives highest accuracy of 65.88%.

RandomForestClassifier also gives accuracy of 65.65%.

Now Saving one Model SVC() for Rating Prediction.

In [73]:
```
svc=SVC()
svc.fit(train_x,train_y)
svc_y=mc.predict(test_x)
```

Saving the Model

In [74]:
```
import pickle

filename='Rating_Review_svc.pkl'
pickle.dump(svc,open(filename,'wb'))
```

Key Metrics for success in solving problem under consideration:

Accuracy Score:

- It is used to know the accuracy of the model.

Confusion Matrix:

- It described how much True and False Predicted values occurred in the model prediction.

- The diagonal part from left top to right bottom are true in both prediction and actual.

Classification Report:

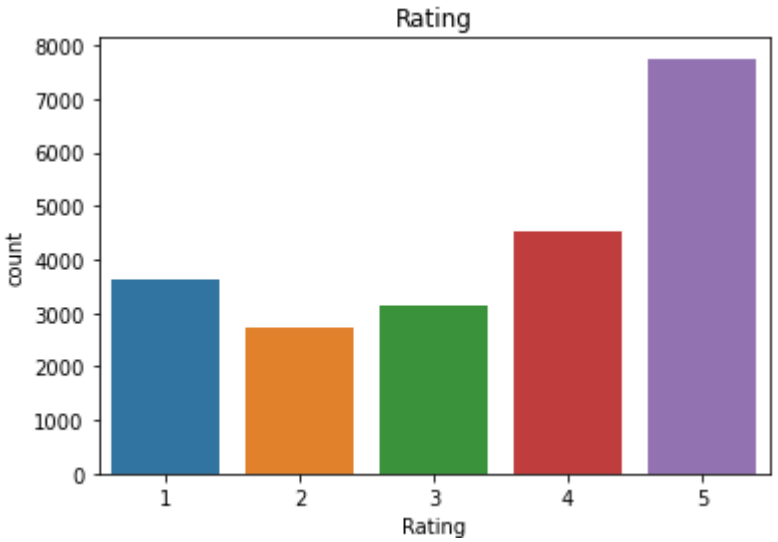
It defined all types metrics like precision, recall, f1 score and support.

Visualizations:

In Visualization, **Univariate** Analysis is done for Rating Column.

In [26]:
```
sns.countplot(x='Rating',data=Rating_Review)
plt.title('Rating')
print(Rating_Review['Rating'].value_counts())
plt.show()
```

```
5    7757
4    4524
1    3620
3    3140
2    2718
Name: Rating, dtype: int64
```



Rating 5 is much higher than other Ratings.

Rating 1-4 are balanced in this Dataset.

Interpretation of the Results:

The results shows the Rating of the reviews and it will be helpful for ranking the products and improving its quality and standards.

CONCLUSION

From this model analysis, it is understood that model accuracy is not that great. This means the review to rating prediction may be needed more learning. Some Oversampling or Undersampling may be applied for balancing the dataset. This model help to predict the reviews of any product for a company.