FLIP ROBO

Product Rating Prediction Report

Submitted by:

Jessica Ghimeliya

ACKNOWLEDGMENT

This project would not have seen the light of the day without the following people and their priceless support and cooperation. Hence I extend my gratitude to all of them.

As a student of Data Trained Education, I would first of all like to express my gratitude to FlipRobo Team and seniors for granting me permission to undertake the project report in their esteemed organization. I would also like to express my sincere thanks to Miss Sapna Mam for supporting me and being always there for me whenever I needed.

During the actual research work with FlipRobo team that set theball rolling for my project. They had been a source of inspiration through their constant guidance, personal interest, encouragement and help.

I convey my sincere thanks to them. In spite of their busy schedule they always found time to guide me throughout the project. I am also grateful to them for reposing confidence in my abilities and giving me the freedom to work on my project. Without their invaluable help I would not have been able to do justice to the project.

INTRODUCTION

Business Problem Framing:-

We have a client who has a website where people write different reviewsfor 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 a rating. So, we have to build an application which canpredict the rating by seeing the review.

Model Building Phase:-

After collecting the data, you need to build a machine learning model.Before model buildings do all data preprocessing steps involving NLP. Try different models with different hyper parameters and select the best model.
Follow the complete life cycle of data science. Include all the stepslike-

1. Data Cleaning
2. Exploratory Data Analysis
3. Data Preprocessing
4. Model Building
5. Model Evaluation
6. Selecting the best model

Conceptual Background of the Domain Problem:-

As online marketplaces have been popular during the past decades, the onlinesellers and merchants ask their purchasers to share their opinions about the products they have bought. As a result, millions of reviews are being generated daily which makes it difficult for a potential consumer to make a good decision on whether to buy the product. Analyzing this enormous amount of opinions is also hard and time consuming for product manufacturers. This thesis considers the problem of classifying reviews by their overall semantic (positive or negative).

Sentiment analysis and classification is a computational study which attempts to address this problem by extracting subjective information from the given texts in natural language, such as opinions and sentiments. Different approaches have used to tackle this problem from natural language processing, text analysis, computational linguistics, and biometrics. In recent years, Machine learning methods have got popular in the semantic and review analysis for theirsimplicity and accuracy.

Amazon is one of the e-commerce giants that people are using every day for online purchases where they can read thousands of reviews dropped by other customers about their desired products. These reviews provide valuable opinions about a product such as its property, quality and recommendations which helps the purchasers to understand almost every detail of a product. This is not only beneficial for consumers but also helps sellers who are manufacturing their ownproducts to understand the consumers and their needs better.

Review of Literature

This project is considering the sentiment classification problem for online reviews using supervised approaches to determine the overall semantic of customer reviews by classifying them into positive and negative sentiment. The data used in this studyis a set of beauty product reviews from Amazon & Flipkart that is collected from Web scrapping tool.

We analyze the information available in the literature. Aside from figuring outhow consumers are effectively using product reviews and their associated star- ratings to make purchase decisions as well as emphasizing on the imbalanced distribution of the online reviews, this chapter outlines the different approaches undertook by researchers to predict ratings from the text content of reviews.

Rating is a classification or ranking of someone or something based on a comparative assessment of their quality, standard or overall performance. This projectis more about exploration, feature engineering and classification that can be done on this data. Since we scrape huge amount of data that includes five stars rating, we can do better data exploration and derive some interesting features using the available columns.

Customer Reviews help customers to learn more about the product and decide whether it is the right product for them. Customer Reviews should give customers genuine product feedback from fellow shoppers. We have a zero tolerance policy for any review designed to mislead or manipulate customers.

The following are types of reviews that we do not allow and will remove:

- A review by someone who has a direct or indirect financial interest in the product.
- A review by someone perceived to have a close personal relationship with theproduct's owner, author or artist.
- A review by the product manufacturer, posing as an unbiased shopper.
- Multiple negative reviews for the same product from one customer.
- A review in exchange for monetary reward.
- A review of a game in exchange for bonus in-game credits.
- A negative review from a seller on a competitor's product.
- A positive review from an artist on a peer's album in exchange for receiving apositive review from them.

Motivation for the Problem Undertaken

Every day we come across various products in our lives, on the digital medium we swipe across hundreds of product choices under one category. It will be tedious for the customer to make selection. Here comes 'reviews' where customers who have already got that product leave a rating after using them and brief their experience by giving reviews. As we know ratings can be easily sorted and judged whether a product is good or bad. But when it comes to sentence reviews, we need to read through every line to make sure the review conveys a positive or negative sense.

In the era of artificial intelligence, things like that have got easy with the Natural Language Processing (NLP) technology. Therefore, it is important to minimize the number of false positives our model produces, to encourage all constructive conversation. Our model also provides beneficence for the platform hosts as it replacesthe need to manually moderate discussions, saving time and resources. Employing a machine learning model to predict ratings promotes easier way to distinguish between products qualities, costs and many other features.

There is a large number of papers that have been published in the field of machine learning. One of the most used approaches for sentiment classification is machine learning algorithms. This section attempts to cover some of them.

Analytical Problem Framing

❖ Mathematical/ Analytical Modeling of the Problem

Data contains attributes such as Ratings (which is the overall rating of the reviewer), Review Text (which contains all the thoughts of customer for that particular product), Summary (this attributes contains brief review of the customer), New review (which contains all the word count in the review text summary), Review character count (this attribute contains all the character count of review text and summary attributes).

In this project we are going to predict the ratings. We will understand the sentiments of the customers and based on that we will predict the ratings.

	Ratings	Review_Text	Summary	Full_review	new_review	Review_character_count
0	5	Amazing product in price range , good sound qu...	Fabulous!	fabulous amazing product price range good soun...	26	157
1	3	Good	Nice	nice good	2	9
2	5	Thank you flipkart for thiz amazing product..d...	Simply awesome	simply awesome thank flipkart thiz amaze produ...	11	86
3	5	Good	Terrific purchase	terrific purchase good	3	22
4	5	Awesome Product.	Simply awesome	simply awesome awesome product	4	30
...	...	...	...	...	...	...
17192	5	The camera is really budjet friendly\nThe qual...	Amazon delivered this product just in one day ...	amazon deliver product one day delivery within...	23	160
17193	5	Price, video quality is best in budget company...	Nice product	nice product price video quality best budget c...	9	60
17194	5	Value for money product. Very good stabilizati...	Smooth ... Feel standard	smooth feel standard value money product good ...	13	90
17195	5	Play Video\n Pro's:\n1. Good clarity consideri...	Osm	osm play video pro good clarity consider price...	16	96
17196	5	The product is really good at this price range...	BATTERY POWER FULL	battery power full product really good price r...	11	66

❖ Data Sources and their formats

This project was done in Three phases:-

- Data Collection Phase
  In this phase we have scrapped round 20000 reviews from Amazon and flip kart website by seleniumweb driver. For this project we have scrapped reviews for different laptops, phones, Headphones, smart watches, professional cameras, printer, and monitors.
- Data Pre-processing Phase
  In this phase we have done all the pre-processing stepslike Natural Language Process (NLP), outlier's detections and over fitting as well. Fetch an equal number of reviews for each rating, for example if you are fetching 10000 reviews then all ratings 1,2,3,4,5 should be 2000. It will balance our data set. Convert all the ratings to their round number asthere are only 5 options for rating i.e., 1, 2,3,4,5. If a rating is 4.5 convert it 5.
- Model Building Phase
  All the pre-processing done after that, we built amachine learning model.

❖ Data Pre-processing Done

Checking Missing Values:

```
1  df.isna().sum()
```

```
Ratings          0
Review_Text    274
Summary          0
dtype: int64
```

- There are some features having missing values. As we see that the numbers of missing values are too low.
- So it is good to fill them we will drop them, as it will not affect our dataset.

```
1  ## Droppping Nans
2  df.dropna(inplace=True)
3  df.shape
```

```
(16923, 3)
```

```
1  df['Ratings'].unique()
```

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

As we can see at these are the unique values in our target column

❖ Removing Unwanted characters from the review

- We can see that our reviews holds many emojis, our model will not understand to these emojies so we have to handle them.
- There are many special character used like -,:,:,...,@,(,) etc.
- Some words are in Capital letters and some are small letters.

Removing Emojies

```
Full_review=[]
for i in df.Full_Review:
    #appending text after removing the emojis from it
    Full_review.append(clean(i, no_emoji=True))
```

```
## Dropping Full review from dataframe
df.drop('Full_Review',axis=1,inplace=True)

### Adding without emojies column in our dataframe
df['Full_review']=Full_review
```

Lower Casing:-

```
## # Lowercasing the words in review
df['Full_review'] = df['Full_review'].apply(lambda x : x.lower())
```

Removing Unwanted Text and contracted words

```
def unwanted_text(string):
    string = re.sub(r"won't", "will not",string)
    string = re.sub(r"don’t", "do not",string)
    string = re.sub(r"doesn’t", "does not",string)
    string= re.sub(r"haven’t", "have not", string)
    string = re.sub(r"can't", "can not", string)
    string = re.sub(r"im ", "i am", string)
    string = re.sub(r"yo ", "you ",string)
    string = re.sub(r"n\'t", " not", string)
    string = re.sub(r"\'re", " are", string)
    string = re.sub(r"\'s", " is", string)
    string = re.sub(r"\'d", " would", string)
    string = re.sub(r"\'ll", " will", string)
    string = re.sub(r"\'t", " not", string)
    string = re.sub(r"\'ve", " have", string)
    string = re.sub(r"\'m", " am", string)
    string = re.sub(r"<br>", " ", string)
    ##removing all the urls:
    string = re.sub(r'http\S+', '', string)
    return string
```

```
## Decontracted all the reviews
df['Full_review'] = df['Full_review'].apply(lambda x : unwanted_text(x))
```

Removing Special Characters

```
### Removing Special charactors
charactors=['.',':',';','(',')','!','@','$','^','[^\w\s]','/','?','<','>']
for i in charactors:
    df['Full_review'] = df['Full_review'].str.replace(i,'')
```

Replacing Space Keys

```
space_key=['\n','_','-']
for j in space_key:
    df['Full_review'] = df['Full_review'].str.replace(j,' ')
```

Removing Stop words

```
# Removing all the stopwords
stop_charactors = stopwords.words('english')
df['Full_review'] = df['Full_review'].apply(lambda x: ' '.join([word for word in x.split() if word not in (stop_charactors)]))
```

Lemmatization:

Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which isknown as the lemma. Lemmatize minimizes text ambiguity.

Example: - Words like bicycles or bicycles are converted to base word bicycle. Basically, it will convert all words having the same meaning but different representation to their base form.

```
# Defining functiom to convert nltk tag to wordnet tags
def nltk_tag_to_wordnet_tag(nltk_tag):
    if nltk_tag.startswith('J'):
        return wordnet.ADJ
    elif nltk_tag.startswith('V'):
        return wordnet.VERB
    elif nltk_tag.startswith('N'):
        return wordnet.NOUN
    elif nltk_tag.startswith('R'):
        return wordnet.ADV
    else:
        return None

# Defining function to lemmatize our text
def lemmatize_sentence(sentence):
    # tokenize the sentence and find the pos_tag
    nltk_tagged = nltk.pos_tag(nltk.word_tokenize(sentence))
    # tuple of (token, wordnet_tag)
    wordnet_tagged = map(lambda x : (x[0], nltk_tag_to_wordnet_tag(x[1])), nltk_tagged)
    lemmatize_sentence = []
    for word, tag in wordnet_tagged:
        if tag is None:
            lemmatize_sentence.append(word)
        else:
            lemmatize_sentence.append(lemination.lemmatize(word,tag))
    return " ".join(lemmatize_sentence)

df['Full_review'] = df['Full_review'].apply(lambda x : lemmatize_sentence(x))
```

Normalization:

Normalization is **the process of converting a token into its base form**. In the normalization process, the inflectional form of a word is removed so that the base formcan be obtained.

```
## scraping noise text:
def scrap(text):
    # remove HTML markup
    text = re.sub("(<.*?>)", "", text)
    # remove non-ascii and digits
    text = re.sub("(\\W)", " ", text)
    text = re.sub("(\\d)", "", text)
    # remove white space
    text = text.strip()
    return text

df['Full_review'] = df['Full_review'].apply(lambda x : scrap(x))
```

Checking Sample Now:-

```
## printing review again:
for i in range(10):
    print(df.Full_review[i])
    print("\n**********Next Review*********\n")
```

```
fabulous amazing product price range good sound quality performance well update window display normal tn lead problem price ran
ge gb ram gb ssd upgrade gb gb

**********Next Review*********

nice good

**********Next Review*********

simply awesome thank flipkart thiz amaze productdelivery time perfectlaptop also amaze

**********Next Review*********

terrific purchase good
```

Now it looks pretty good now.

❖ Data Inputs- Logic- Output Relationships

We have use word cloud method for finding the trends betweeninput and output variable.

- Five Star Ratings

- Four Star Ratings

- Three Star Ratings

- Two Star Ratings

- One Star Ratings

Observation of Word Cloud Visualization:-

`For Rating: 1` It mostly consists of words like watch, use, bad product, waste, time, money, bad experience, issue etc

`For Rating: 2` It mostly consists of words like good, phone, use, watch, poor, issue, waste money, quality good, bad, problem etc

`For Rating: 3` It mostly consists of words like sound quality, good, use, time, camera quality, display, buy, build quality etc

`For Rating: 4` It mostly consists of words like use, buy, phone, watch, good product, good quality, good choice, nice product etc

`For Rating: 5` It mostly consists of words like price range, value money, good product, well, go, simply awesome, perfect product etc

State the set of assumptions (if any) related to the problem under consideration

We assume that exploratory data analysis shows highest word frequency of stopwords, which were removed in the subsequent pre-processing steps. No outliers werefound in the raw dataset, but some blank rows were found in row dataset which we drop as they are in less quantity.

The features of interest were found to be 'review Text', 'summary' and 'rating' asthese were most useful and relevant for model building, prediction as well as abstraction. In the 'review Text' field, mixed bag of short and long reviews were found, some of which produced no or little information about the product. In theexploratory analysis, review Text attribute was explored using the word-cloud visualization for each category.

By looking into the target variable/label, we assumed that it was a multiclass classification type of problem. Also, we observed that our dataset was imbalance sowe will have to balance the dataset for better prediction accuracy outcome.

Hardware and Software Requirements and Tools Used

The Ratings Prediction Project is about built a machine learning model that couldpredict the default case so that the company could to which products are liked by the customers and which are not. So for that we require lots of libraries and packages to work upon this project.

Hardware Required:-

- Processor:- Core i5 or above
- Ram:- 8GB or above
- SSD:- 256GB or Above
- Software Required:- Anaconda Promt/Navigator
- Libraries Required:-
  - Pandas
  - Numpy
  - Matplotlib
  - Seaborn

```
# Importing necessary libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

# Importing NLP libraries:
import re
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords', quiet=True)
nltk.download('punkt', quiet=True)
nltk.download('wordnet', quiet=True)
nltk.download('averaged_perceptron_tagger', quiet=True)

# Importing warnings:
import warnings
warnings.filterwarnings('ignore')
```

```
# Importing necessary libraries:-
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.model_selection import cross_val_score
from sklearn.metrics import f1_score, precision_score, confusion_matrix, accuracy_score, classification_report

from sklearn.svm import LinearSVC
from sklearn.naive_bayes import MultinomialNB, GaussianNB, BernoulliNB
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import SGDClassifier
from sklearn.ensemble import RandomForestClassifier, ExtraTreesClassifier,AdaBoostClassifier,GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier

```

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

Handling Outliers Using Zscore:-

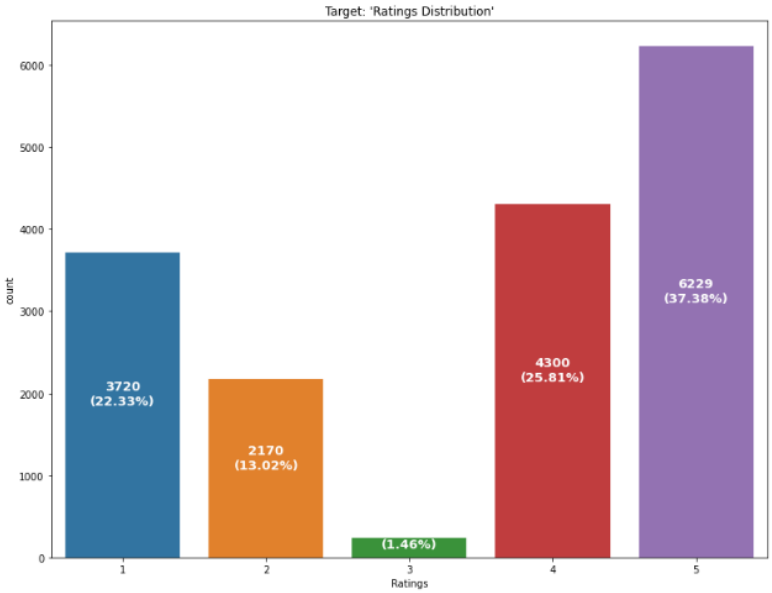
```
from scipy import stats
from scipy.stats import zscore
```

```
z_score=zscore(df[['new_review']])
abs_zscore=np.abs(z_score)
```

```
threshold=3
new_entry=(abs_zscore<threshold).all(axis=1)
df_new=df[new_entry]
print("The shape before: ", df.shape)
print("The shape after: ",df_new.shape)
```

```
The shape before:  (16923, 6)
The shape after:  (16662, 6)
```

Handling Imbalanced Dataset:



- If we failed to handle this problem then the model will become a disasterbecause modeling using class-imbalanced data is biased in favor of the majority class.
- Looking at the above count plot for our target variable (Ratings) we can say that the data set is having most number of reviews rated as 5 star and very less number of reviews rated as 2 star and 3 star.
- This will cause the Imbalance problem for our Machine Learning model and make it bias. So I am selecting equal number of reviews of each rating as an input for our model to avoid any kind of biasness.
- For that first I will shuffle the dataset so that we can select data from both web-sites (Amazon and Flipkart).Then I will select equal number of data of every category and ensure that the rating values are balanced.
- A typical example of imbalanced data is encountered in e-mail classification problem where emails are classified into ham or spam. The number of spam emails is usually lower than the number of relevant (ham) emails. So, using the original distribution of two classes leads to imbalanced dataset.

Handling Imbalanced dataset:

```
### Value Counts
df_new['Ratings'].value_counts()
```

```
5    6229
4    4300
1    3720
2    2170
3     243
Name: Ratings, dtype: int64
```

If we take 1500 records for every star then our model will not look as imbalanced dataset.

```
# Select data from every Ratings category
df1 = df[df['Ratings']==1][0:1500]
df2 = df[df['Ratings']==2][0:1500]
df3 = df[df['Ratings']==3][0:1500]
df4 = df[df['Ratings']==4][0:1500]
df5 = df[df['Ratings']==5][0:1500]
```

```
### Adding all the dataframes:
df = pd.concat([df1,df2,df3,df4,df5], ignore_index=True)
```

Testing of Identified Approaches (Algorithms)

Below listed algorithms are used for this project.

- Logistic Regression
- Decision Tree Classifier
- Random forest Classifier
- SVC
- Ada Boost Classifier
- Gradient Boosting Classifier
- Naïve Bayes Multinomial NB

Run and Evaluate selected models

The idea of distributional semantics that is implemented through 'word vectors' has been used heavily in semantic processing for a wide variety of Applications.This simple idea has probably been the most powerful and useful insight in creating semantic processing systems. Supervised techniques for word sense disambiguation require the input words to be tagged with their senses. The sense is the label assigned to the word. We defined the various classification models mentioned above and assigned them to user created variables. Then we defined the function that would train and predict the multiclass labels for us along with the evaluation metrics. I did not include the cross-validation part in the function and rather created a separate function for that metrics to evaluate only the best scored classification models amongst the original list.
We have defied few functions that will help us for finding the best random state, printing the model's score and checking the cross validation. The code is mentioned below screenshots respectively.

```
### Best Random STATE:
def Random_state(Model,Feature,Target):
    maximum_accu=0
    for i in range(11,36):
        x_train,x_test,y_train,y_test=train_test_split(Feature,Target,test_size=0.25,random_state=i)
        Model.fit(Feature,Target)
        train_pred=Model.predict(x_train)
        test_pred=Model.predict(x_test)
        accu_score=accuracy_score(y_test,test_pred)
        print("For Random State {}, the Accuracy Score is: {}".format(i,accu_score))
        if accu_score>maximum_accu:

            maximum_accu=accu_score
            j=i
    print("\n")
    print(" The Highest Accuracy SCORE is: {}".format(maximum_accu))
    print("\n The Best Random State is:")
    return j
```

Defining a Function for Priting Accuracy:

```
def print_score(clf,x,y,randomstate,train=True):
    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=randomstate)
    clf.fit(x_train,y_train)
    if train:
        y_pred= clf.predict(x_train)
        print("\n**********************************Training Score**************************************")

        print(f"Accuracy Score : {accuracy_score(y_train,y_pred)*100:.2f}%")
        print("\n **********************************Confusion Matrix**************************************")
        print(confusion_matrix(y_train,y_pred))

        print("\n \n Training Classification Report \n" ,classification_report(y_train,y_pred,digits=2))

    elif train==False:
        pred=clf.predict(x_test)
        print('\n \n')
        print("\n**********************************Test Result**************************************************")
        print(f"Accuracy_Score : {accuracy_score(y_test,pred)*100:.2f}%")
        print("\n**********************************Confusion Matrix**********************************************")
        print(confusion_matrix(y_test,pred))

        print("\n \n Test Classification Report \n", classification_report(y_test,pred,digits=2))
```

Defining A funtion for cross Validation:

```
def cross_val(Model,independent,dependent,randomstate):
    x_train,x_test,y_train,y_test=train_test_split(independent,dependent,test_size=0.2,random_state=randomstate)
    Model.fit(x_train,y_train)
    pred=Model.predict(x_test)
    for i in range(3,4):
        cv_score=cross_val_score(Model,x,y,cv=i)
        cv_mean=cv_score.mean()
        print('At cv :- ', i)
        print('Cross Validation score is :- ', cv_mean)
        print('Accuracy score is :- ',accuracy_score(y_test,pred))
        print('\n')
```

Key Metrics for success in solving problem under consideration

- Accuracy Score
- F1 Score
- Cross Validation
- Hyper parameter tuning
- Grid Search CV

❖ Logistic Regression

```
*********************************Test Result*********************************************
Accuracy_Score : 80.86%

*********************************Confusion Matrix*****************************************
[[ 668   30    0   17   12]
 [ 101  264    1   32   17]
 [   4    2   19   24    5]
 [  38   16    0  629  191]
 [  23    5    0  120 1115]]


 Test Classification Report
               precision    recall  f1-score   support

           1       0.80      0.92      0.86       727
           2       0.83      0.64      0.72       415
           3       0.95      0.35      0.51        54
           4       0.77      0.72      0.74       874
           5       0.83      0.88      0.86      1263

    accuracy                           0.81      3333
   macro avg       0.84      0.70      0.74      3333
weighted avg       0.81      0.81      0.80      3333
```

Cross Validation:

```
: 1 cross_val(LogisticRegression(),x,y,11)
```

```
At cv :-  3
Cross Validation score is :-  0.5667987036370185
Accuracy score is :-  0.8085808580858086
```

❖ Decision Tree Classifier

```
*********************************Test Result*********************************************
Accuracy_Score : 83.59%

*********************************Confusion Matrix*****************************************
[[ 652   31    4   14   16]
 [  58  310    1   35   19]
 [   0    3   43    7    3]
 [  23   19    1  679  125]
 [  15   32    4  137 1102]]


 Test Classification Report
               precision    recall  f1-score   support

           1       0.87      0.91      0.89       717
           2       0.78      0.73      0.76       423
           3       0.81      0.77      0.79        56
           4       0.78      0.80      0.79       847
           5       0.87      0.85      0.86      1290

    accuracy                           0.84      3333
   macro avg       0.82      0.81      0.82      3333
weighted avg       0.84      0.84      0.84      3333
```

Cross Validation:

```
1 cross_val(DecisionTreeClassifier(),x,y,20)
```

```
At cv :-  3
Cross Validation score is :-  0.5031808906493819
Accuracy score is :-  0.8355835583558355
```

❖ Random Forest Classifier

```
*********************************Test Result*************************************************
Accuracy_Score : 90.58%

*********************************Confusion Matrix*********************************************
[[ 692    9    0    8    8]
 [  52  326    0   22   23]
 [   1    1   43    6    5]
 [  16    2    0  723  106]
 [   7    4    1   43 1235]]


 Test Classification Report
              precision    recall  f1-score   support

           1       0.90      0.97      0.93       717
           2       0.95      0.77      0.85       423
           3       0.98      0.77      0.86        56
           4       0.90      0.85      0.88       847
           5       0.90      0.96      0.93      1290

    accuracy                           0.91      3333
   macro avg       0.93      0.86      0.89      3333
weighted avg       0.91      0.91      0.90      3333
```

Cross Validation:

```
1 cross_val(RandomForestClassifier(),x,y,20)
```

```
At cv :-  3
Cross Validation score is :-  0.5894850558156284
Accuracy score is :-  0.900090009009001
```

❖ AdaBoost Classifier

```
*********************************Test Result*************************************************
Accuracy_Score : 50.32%

*********************************Confusion Matrix*********************************************
[[619  19   0  53  66]
 [233  19   2  76  68]
 [  0   1   7  26  11]
 [180   9  10 382 298]
 [271  13  10 310 650]]


 Test Classification Report
              precision    recall  f1-score   support

           1       0.48      0.82      0.60       757
           2       0.31      0.05      0.08       398
           3       0.24      0.16      0.19        45
           4       0.45      0.43      0.44       879
           5       0.59      0.52      0.55      1254

    accuracy                           0.50      3333
   macro avg       0.41      0.39      0.37      3333
weighted avg       0.49      0.50      0.47      3333
```

Cross Validation:

```
1 cross_val(AdaBoostClassifier(),x,y,27)
```

```
At cv :-  3
Cross Validation score is :-  0.4408834473652623
Accuracy score is :-  0.5031503150315032
```

❖ Gradient Boosting Classifier

```
*********************************Test Result***********************************************
Accuracy_Score : 71.62%

*********************************Confusion Matrix*******************************************
[[663  21   0  52  16]
 [192 152   1  72  28]
 [  0   0  46   4   2]
 [ 69   9  10 543 233]
 [ 45   3   4 185 983]]


 Test Classification Report
              precision    recall  f1-score   support

           1       0.68      0.88      0.77       752
           2       0.82      0.34      0.48       445
           3       0.75      0.88      0.81        52
           4       0.63      0.63      0.63       864
           5       0.78      0.81      0.79      1220

    accuracy                           0.72      3333
   macro avg       0.73      0.71      0.70      3333
weighted avg       0.73      0.72      0.70      3333
```

Cross Validation:

```
1 cross_val(GradientBoostingClassifier(),x,y,16)
```

```
At cv :-  3
Cross Validation score is :-  0.5609770735806027
Accuracy score is :-  0.7137713771377138
```

❖ Extra Tree Classifier

```
*********************************Test Result***********************************************
Accuracy_Score : 92.05%

*********************************Confusion Matrix*******************************************
[[ 693   13    0    5    6]
 [  40  344    1   17   21]
 [   1    0   43    6    6]
 [  13    2    0  747   85]
 [   9    4    1   35 1241]]


 Test Classification Report
              precision    recall  f1-score   support

           1       0.92      0.97      0.94       717
           2       0.95      0.81      0.88       423
           3       0.96      0.77      0.85        56
           4       0.92      0.88      0.90       847
           5       0.91      0.96      0.94      1290

    accuracy                           0.92      3333
   macro avg       0.93      0.88      0.90      3333
weighted avg       0.92      0.92      0.92      3333
```

Cross Validation:

```
1 cross_val(ExtraTreesClassifier(),x,y,20)
```

```
At cv :-  3
Cross Validation score is :-  0.582223022446285
Accuracy score is :-  0.9171917191719172
```

❖ Gaussian NB Classifier

```
*********************************Test Result*************************************************
Accuracy_Score : 69.64%

*********************************Confusion Matrix*********************************************
[[ 638    5    0   24   50]
 [ 161   87    0   65  110]
 [   5    0    0   16   35]
 [  43    5    0  355  444]
 [  24    1    0   24 1241]]


 Test Classification Report
              precision    recall  f1-score   support

           1       0.73      0.89      0.80       717
           2       0.89      0.21      0.33       423
           3       0.00      0.00      0.00        56
           4       0.73      0.42      0.53       847
           5       0.66      0.96      0.78      1290

    accuracy                           0.70      3333
   macro avg       0.60      0.50      0.49      3333
weighted avg       0.71      0.70      0.65      3333
```

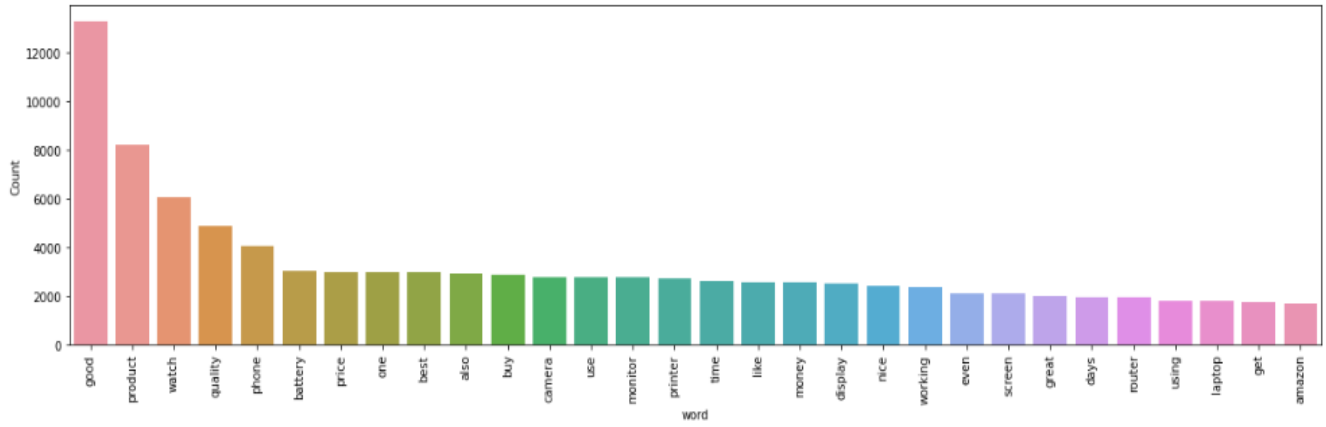
Cross Validation:

```
1 cross_val(MultinomialNB(),x,y,20)
```

```
At cv :-  3
Cross Validation score is :-  0.5377505701596447
Accuracy score is :-  0.6963696369636964
```

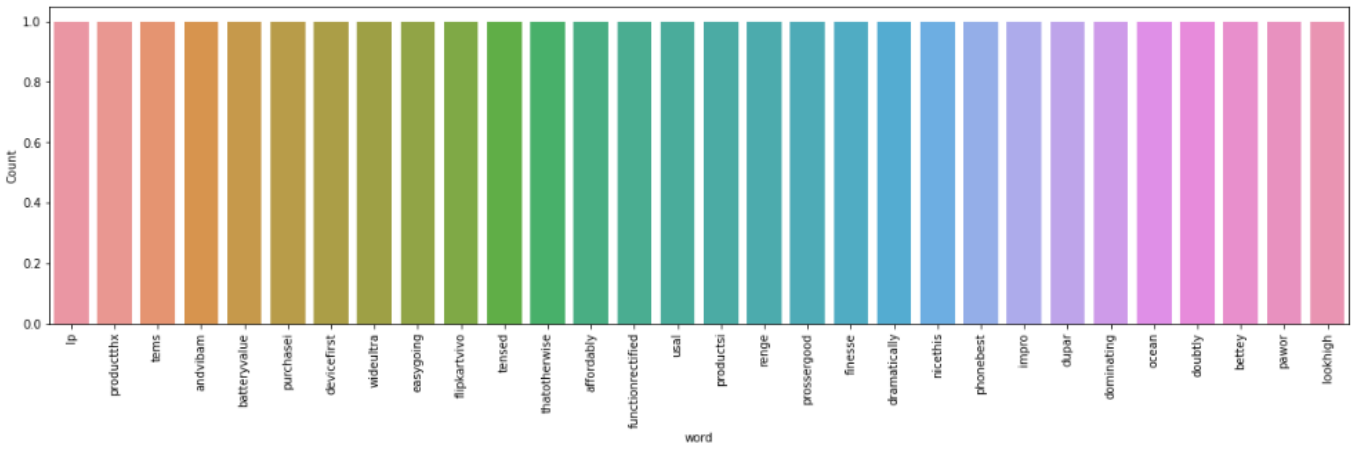
Visualizations

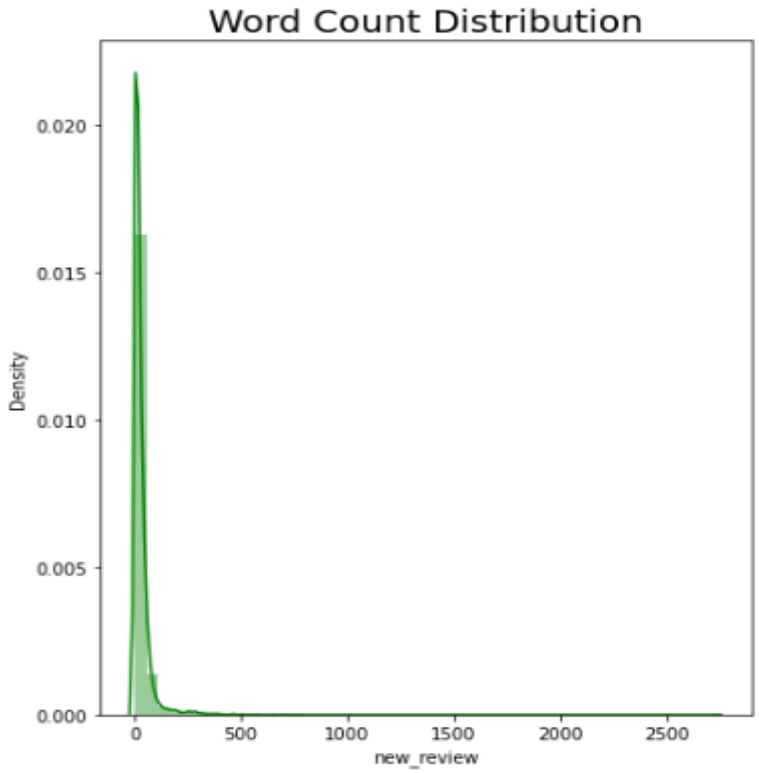
Printing Top Occuring Word in Review:



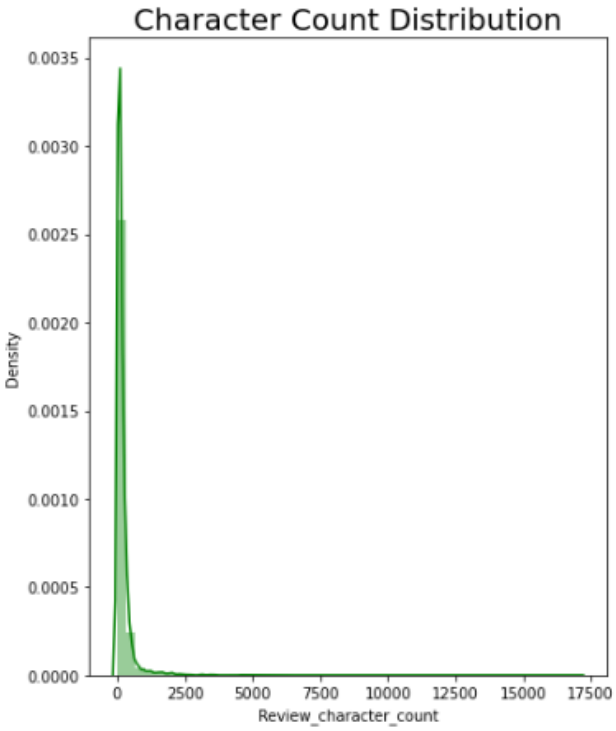
- We have printed the most accruing 20 words in our dataset.
- As we can see that the word 'good' is accruing most of the time.

Printing Rarely Occurinng Word in Review:-



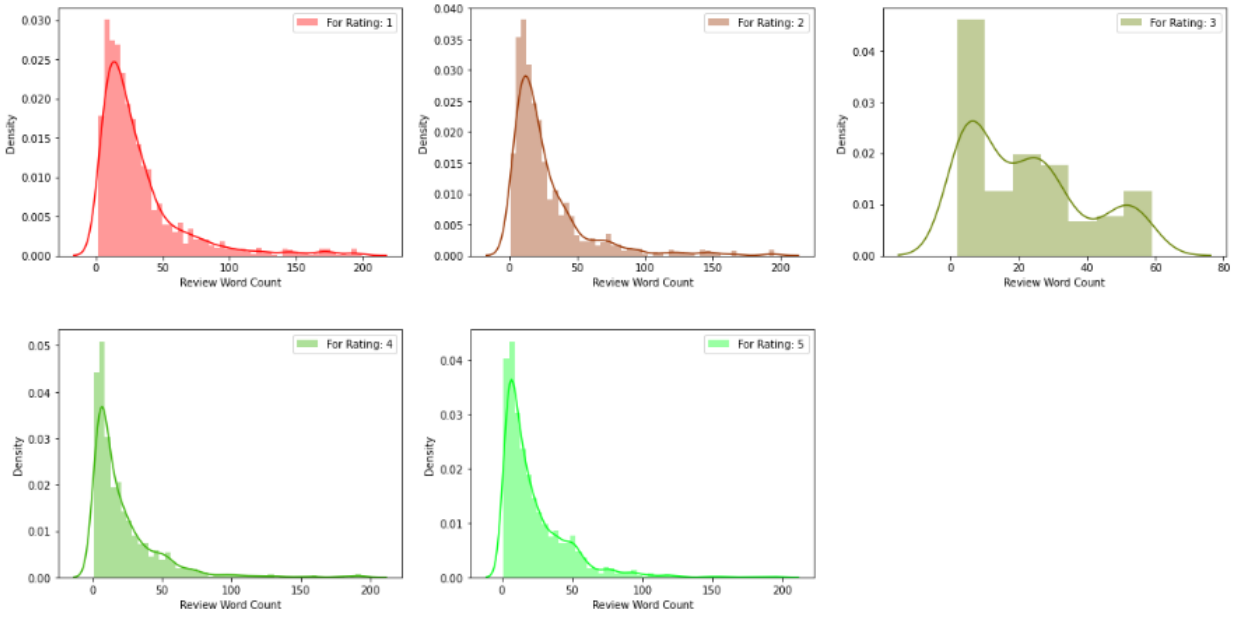


- As we can see that most of the words are accruing 0 to 10 times range.
- A Word count is highly right skewed.



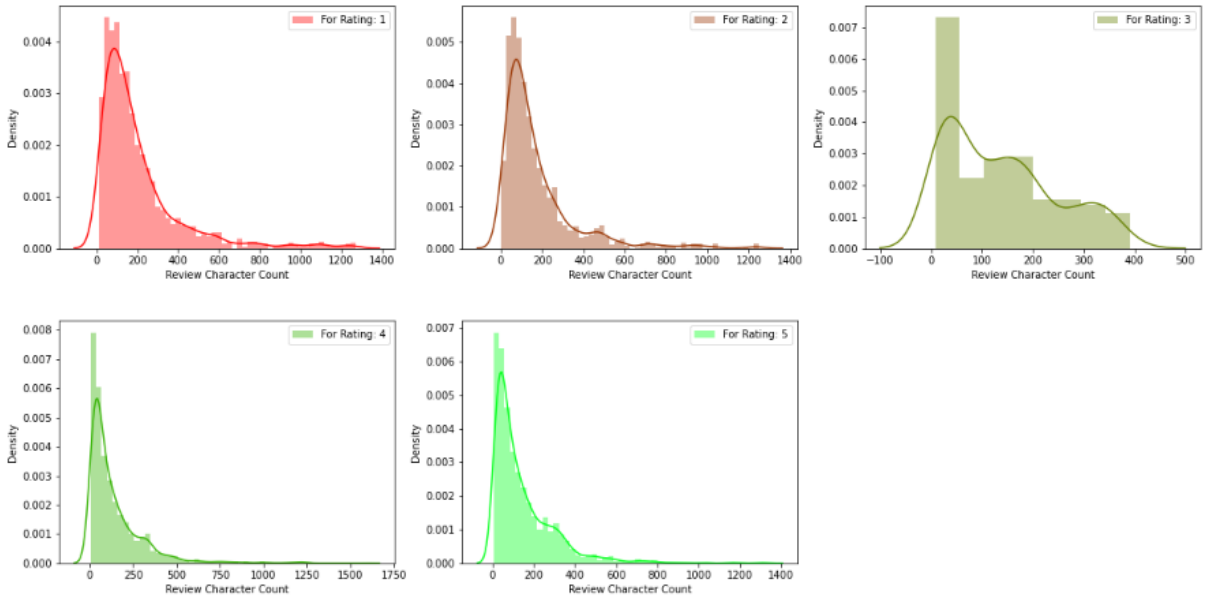
Above plot represents histogram for character count of review text, whichis quite similar to the histogram of word count.

Checking review word count distribution for each rating:-



- We noticed that every type of ratings word count distribution is highlyright skewed.
- Also we see that most of the word count falling in the range of 0 to 10.

Similarly Checking review character count distribution for each rating:-



- We noticed that for every type rating, the character count distribution ishighly right skewed.
- Similarly most of character's count is falling in the range of 0 to 10.

Interpretation of the Results

Starting with univariate analysis, with the help of count plot it was found that thedata consists of in equal amount for each rating (i.e., from 1 to 5). Moving further with the removal and replacement of certain terms (like punctuations, extra spaces, numbers, money symbols, smiley etc) as well as removal of stop words.

It was evident that the length of review text decreases by a large amount. This was also depicted by using distribution plots and histograms. With the help of word cloud, it was found that rating 1 consists of words like waste, money, slow, worst, issue, horrible etc, rating 2 consists of words like problem, issue, bad, poor, slow etc, rating 3consists of word like problem, bad, issue, slow, life, average, nice etc, rating 4 consistsof word like good, value, money, nice, performance, great, better, wonderful etc and rating 5 consists of words like excellent, must buy, great, perfect, super, awesome, mind blowing etc.

As we noticed that every model's training score showing greate as well as testing score. But after checking the Cross validation we face the reality of our models. Every model is extremely over fitted.

After Checking the Cross Validation we observe that `RandomForestClassifier` models showing highest accuracy among all of them.

Now, we will do Some tunning for this model. Let's Check if we increase their accuracy or not.

`HyperParameter Tunning:`

Random Forest Classifier:-

```
##Lets select the different parameters for tuning our best model (RandomForestClassifier)
grid_params = {'n_estimators':[100,200],
               'criterion':['gini','entropy'],
               'max_depth': [500,800],
               'bootstrap':[True,False]}

# Train the model with given parameters using GridSearchCV
GSCV =  GridSearchCV(RandomForestClassifier(), grid_params, cv=3, verbose=3,n_jobs=-1)
GSCV.fit(x_train, y_train)
```

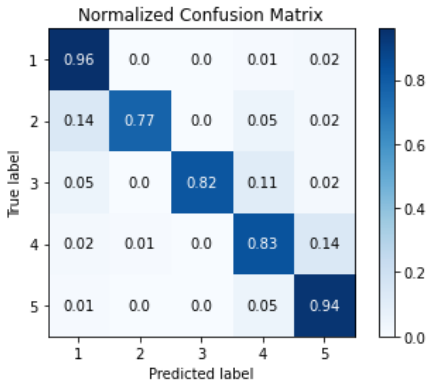
```
print(GSCV.best_params_)
```

```
### Model Evaluation

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=15)

rf=RandomForestClassifier(n_estimators=100,criterion="entropy",max_depth=800,bootstrap=False)

rf.fit(x_train,y_train)
y_pred=rf.predict(x_test)
print("\n\n")
print("**********Testing Scores**********\n")
print("Accuracy score for testing is : ", accuracy_score(y_test,y_pred))
# Printing the classification report
print(f"\nCLASSIFICATION REPORT: \n {classification_report(y_test, y_pred)}")
# Printing the Confusion matrix
print(f"\nCONFUSION MATRIX: \n {confusion_matrix(y_test, y_pred)}")
```

```
**********Testing Scores**********

Accuracy score for testing is :  0.8934229476716274

CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           1       0.89      0.96      0.92       909
           2       0.96      0.77      0.86       559
           3       0.82      0.82      0.82        55
           4       0.89      0.83      0.86      1081
           5       0.89      0.94      0.91      1562

    accuracy                           0.89      4166
   macro avg       0.89      0.87      0.87      4166
weighted avg       0.90      0.89      0.89      4166


CONFUSION MATRIX:
 [[ 876    4    0    7   22]
 [  81  433    2   30   13]
 [   3    0   45    6    1]
 [  19    6    5  899  152]
 [  10    7    3   73 1469]]
```

```
## Visualize the confusion matrix:
sktplt.metrics.plot_confusion_matrix(y_test, y_pred, normalize=True)
```

```
<AxesSubplot:title={'center':'Normalized Confusion Matrix'}, xlabel='Predicted label', ylabel='True label'>
```



Acuracy of final model Random Forest is 89%

CONCLUSION

Key Findings and Conclusions of the Study

We observe that all the models are performing quite well in training model and testing as well. But after checking the cross validation we see that all the models are getting over fitted. In summary, we have tried two types of features. For these two types of features, we tried all the algorithms we mentioned in the model part including Naïve Bayes, Support Vector Machine, Random Forest, Linear Regression, and Decision Tree. From the results, we can see that our accuracy on the test set is the best when we use Random Forest on the first type of feature.

Through the Random Forest classifier model have shown best accuracy. So that is the reason we have selected it our best fit model as well.

Learning Outcomes of the Study in respect of Data Science

In this project we were able to learn various Natural Language Processing techniques like lemmatization, stemming, removal of Stop Words, etc. This project has demonstrated the importance of sampling effectively, modeling and predicting data. Through different powerful tools of visualization, we were ableto analyses and interpret different hidden insights about the data. The few challenges while working on this project are:

1. Imbalanced Dataset.
2. Lots of Text data.

The dataset was highly imbalanced so we balanced the dataset using smote technique. We converted text data into vectors with the help of TfidfVectorizer.

Limitations of this work and Scope for Future Work

One of the main reasons our accuracy is not high enough is because of thedata imbalance. We have given equal amount of class to our model for that wehave to drop most of the data. This could be the reason of low accuracy. We will be looking into datasets obtained from E-commerce websites. This will provide better understanding of our sentiments based on demographics. And lastly we will try to work on this product to achieve a generalize form of this model.

For future we have to look into below mentioned points.

- Try to fetch data from different websites. If data is from different websites, it will help our model to remove the effect of over fitting.
- Try to fetch an equal number of reviews for each rating, for example ifyou are fetching 10000 reviews then all ratings 1,2,3,4,5 should be 2000. It will balance our data set.
- Convert all the ratings to their round number, as there are only 5 optionsfor rating i.e., 1, 2, 3, 4, 5. If a rating is 4.5 convert it 5.