

A Project Report
on
Review Prediction

*Submitted in partial fulfillment of the
requirement for the award of the degree of*

**Bachelor of Technology in Computer Science and
Engineering**



**Under The Supervision of
Dr. Priyanka
Department of Computer Science and Engineering**

Submitted By

19SCSE1010101 –Amit kumar

19SCSE1010182 – Aakarsh vats

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA, INDIA
DECEMBER - 2021**

Table of Contents

Title	Page No.
Candidates Declaration	
Acknowledgement	
Abstract	
List of Table	
List of Figures	
Acronyms	
Chapter 1 Introduction	1
1.1 REQUIREMENT OF HALL TICKET	2
1.2 DISADVANTAGE OF CURRENT SYSTEM	3
1.3 MERITS OF PROPOSED SYSTEM	
Chapter 2 Literature Survey/Project Design	5
Chapter 3 Functionality/Working of Project	9
Chapter 4 Results and Discussion	11
Chapter 5 Conclusion and Future Scope	41
5.1 Conclusion	41
5.2 Future Scope	42
Reference	43
Publication/Copyright/Product	45



**SCHOOL OF COMPUTING SCIENCE AND
ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA**

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled “ Review Prediction ” in partial fulfillment of the requirements for the award of the **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**

submitted in the **School of Computing Science and Engineering** of Galgotias University, Greater Noida, is an original work carried out during the period of **JULY-2021 to DECEMBER-2021**, under the supervision of **Dr.Priyanka, Department of Computer Science and Engineering** of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the project has not been submitted by me/us for the award of any other degree of this or any other places.

19SCSE1010101 –Amit kumar

19SCSE1010182 – Aakarsh vats

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Supervisor

(Dr. Priyanka)

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of 19SCSE1010101 –Amit
kumar 19SCSE1010182 – Aakarsh vats

has been held on _____ and his/her

work is recommended for the award of **BACHELOR OF TECHNOLOGY IN COMPUTER
SCIENCE AND ENGINEERING.**

Signature of Examiner(s)

Signature of Supervisor(s)

Signature of Project Coordinator

Signature of Dean

Date:

Place:

ABSTRACT

Consumers' reviews on ecommerce websites, online services, ratings and experience stories are useful for the user as well as the vendor. The reviewer can increase their brand's loyalty and help other customers understand their experience with the product. Similarly reviews help the vendors gain more profiles by increasing their sale of products, if consumers leave positive feedback on their product review. But unfortunately, these review mechanisms can be misused by vendors.

For example, one may create fake positive reviews to promote brand's reputation or try to demote competitor's products by leaving fake negative reviews on their product. Existing solutions with supervised include application of different machine learning algorithms and different tools like Weka.

Unlike the existing work, instead of using a constrained dataset I chose to have a wide variety of vocabulary to work on such as different subjects of datasets combined as one big data set. Sentiment analysis has been incorporated based on emojis and text content in the reviews. Review Prediction. The testing results are obtained through the application of Naïve Bayes, Linear SVC, Support Vector Machine

and Random forest algorithms.

The implemented (proposed) solution is to classify these reviews into fake or genuine. The highest accuracy is obtained by using Naïve Bayes by including sentiment classifier.

ACKNOWLEDGEMENTS

I whole heartedly show sincere gratitude to my project guide, Dr. Priyanka for guiding me with their technical expertise, providing me feedback and suggestions for improving this project and giving me an opportunity to gain and learn through my project experience. I thank Dr. Priyanka for constantly supporting and helping me with the research work.

I am also thankful my family for their love.

Chapter 1: Introduction

Everyone can freely express his/her views and opinions anonymously and without the fear of consequences. Social media and online posting have made it even easier to post confidently and openly. These opinions have both pros and cons while providing the right feedback to reach the right person which can help fix the issue and sometimes a con when these get manipulated. These opinions are regarded as valuable. This allows people with malicious intentions to easily make the system to give people the impression of genuineness and post opinions to promote their own product or to discredit the competitor products and services, without revealing identity of themselves or the organization they work for. Such people are called opinion spammers and these activities can be termed as opinion spamming.

There are few different types of opinion spamming. One type is giving positive opinions to some products with intention to promote giving untrue or negative reviews to products to damage their reputation. Second type consists of advertisements with no opinions on product. There is a lot of research work done in the field of sentiment analysis and created models while using different sentiment analysis on data from various sources, but the primary focus is on the algorithms and not on actual fake review detection. One of many other research works by E. I. Elmurngi and A. Gherbi [1] used machine learning algorithms to classify the product reviews on Amazon.com dataset [2] including customer usage of the product and buying experiences. The use of Opinion Mining, a type of language processing to track the emotion and thought process of the people or users about a product which can in turn help research work.

Opinion mining, which is also called sentiment analysis, involves building a system to collect and examine opinions about the product made in social media posts, comments, online product and service reviews or even tweets. Automated opinion mining uses machine learning, a component of artificial intelligence. An opinion mining system can be built using a software that can extract knowledge from dataset and incorporate some other data to improve its performance.

One of the biggest applications of opinion mining is in the online and e-commerce reviews of consumer products, feedback and services. As these opinions are so helpful for both the user as well as the seller the e-commerce web sites suggest their customers to leave a feedback and review about their product or service they purchased. These reviews provide valuable information that is used by potential customers to know the opinions of previous or current users before they decide to purchase that product from that seller. Similarly, the seller or service providers use this information to identify any defects or problems users face with their products and to understand the competitive information to know the difference about their similar competitors' products.

There is a lot of scope of using opinion mining and many applications for different usages:

Individual consumers: A consumer can also compare the summaries with competing products before taking a decision without missing out on any other better products available in the market.

Businesses/Sellers: Opinion mining helps the sellers to reach their audience and understand their perception about the product as well as the competitors. Such reviews

also help the sellers to understand the issues or defects so that they can improve later versions of their product. In today's generation this way of encouraging the consumers to write a review about a product has become a good strategy for marketing their product through real audience's voice. Such precious information has been spammed and manipulated. Out of many researches one fascinating research was done to identify the deceptive opinion spam.

Chapter 2: Problem Statement

People write unworthy positive reviews about products to promote them. In some cases malicious negative reviews to other (competitive) products are given in order to damage their reputation. Some of these consists of non-reviews (e.g., ads and promotions) which contain no opinions about the product.

The first challenge here is, a word can be positive in one situation while being negative in any other situation. For e.g. the word "long" in terms of a laptop's battery life being long is a positive opinion while the same word about the start time is long is a negative opinion. This shows that the opinion mining system trained about words from opinions cannot understand this nature of the word, giving a different meaning in different situations.

Another challenge is that people don't always express opinions the same way. Most of the traditional text processing techniques assume that small difference in text don't change the meaning much. However, in opinion mining, e.g. the service was great, and the service wasn't great does make a huge difference.

Finally, in some cases, people give contradictory statements which were difficult to anticipate the nature of the opinion. There could be a hidden positive sense in a negative review. And sometimes there is both positive and negative opinion about the product. A emotion factor can add a lot to what a person says or expresses. Adding a negative emoji to a positive comment or vice versa. In the millennial world of texting people have replaced long sentences with short

forms and emoticons. These emoticons when used in

text format are composed of punctuations and there is a good chance that they will be lost in data cleaning process while preprocessing the text in opinion mining.

After all these challenges, detecting the reviews that are not genuine or which are used to deviate the consumers opinion in a certain direction becomes even more difficult. Opinion spamming or fake review detection is thus significant problem for ecommerce sites and other service providers as the consumer these days rely highly on such opinions or reviews.

Chapter 3: Motivation and Related work

Lack of genuine feedback, creating fake reviews and ratings for supporting the products on their website to improve their reputation and sales is unfair and misleading. This is a common practice these days which increases the need for a fake review detector.

In a recent study a method was proposed by E.I Elmurngi and A. Gherbi [1] using an open source software tool called 'Weka tool' to implement machine learning algorithms using sentiment analysis to classify fair and unfair reviews from amazon reviews based on three different categories positive, negative and neutral words. In this research work, the spam reviews are identified by only including the helpfulness votes voted by the customers along with the rating deviation are considered which limits the overall performance of the system. Also, as per the researcher's observations and experimental results, the existing system uses Naive Bayes classifier for spam and non- spam classification where the accuracy is quite low which may not provide accurate results for the user.

Initially N. O'Brien [4] and J. C. S. Reis, A. Correia, F. Murai, A. Veloso, and F. Benevenuto [5] have proposed solutions that depends only on the features used in the dataset with the use of different machine learning algorithms in detecting fake news on social media. Though different machine learning algorithms the approach lacks in showing how accurate the results are.

B. Wagh, J.V.Shinde, P.A.Kale [6] worked on twitter to analyze the tweets posted by

users using sentiment analysis to classify twitter tweets into positive and negative. They made use of K-Nearest Neighbor as a strategy to allot them sentiment labels by

Chapter 4: Proposed solution

To solve the major problem faced by online websites due to opinion spamming, this project proposes to identify any such spammed fake reviews by classifying them into fake and genuine. The method attempts to classify the reviews obtained from freely available datasets from various sources and categories including service based, product based, customer feedback, experience based and the crawled Amazon dataset with a greater accuracy using Naïve Bayes [7], Linear SVC, SVM, Random forest, Decision Trees algorithm. In order to improve the accuracy, the additional features like comparison of the sentiment of the review, verified purchases, ratings, emoji count, product category with the overall score are used in addition to the review details.

A classifier is built based on the identified features. And those features are assigned a probability factor or a weight depending on the classified training sets. This is a supervised learning technique applying different Machine learning algorithms to detect the fake or genuine reviews,

The high-level architecture of the implementation can be seen in Figure:1 and the problem is solved in the following six steps:

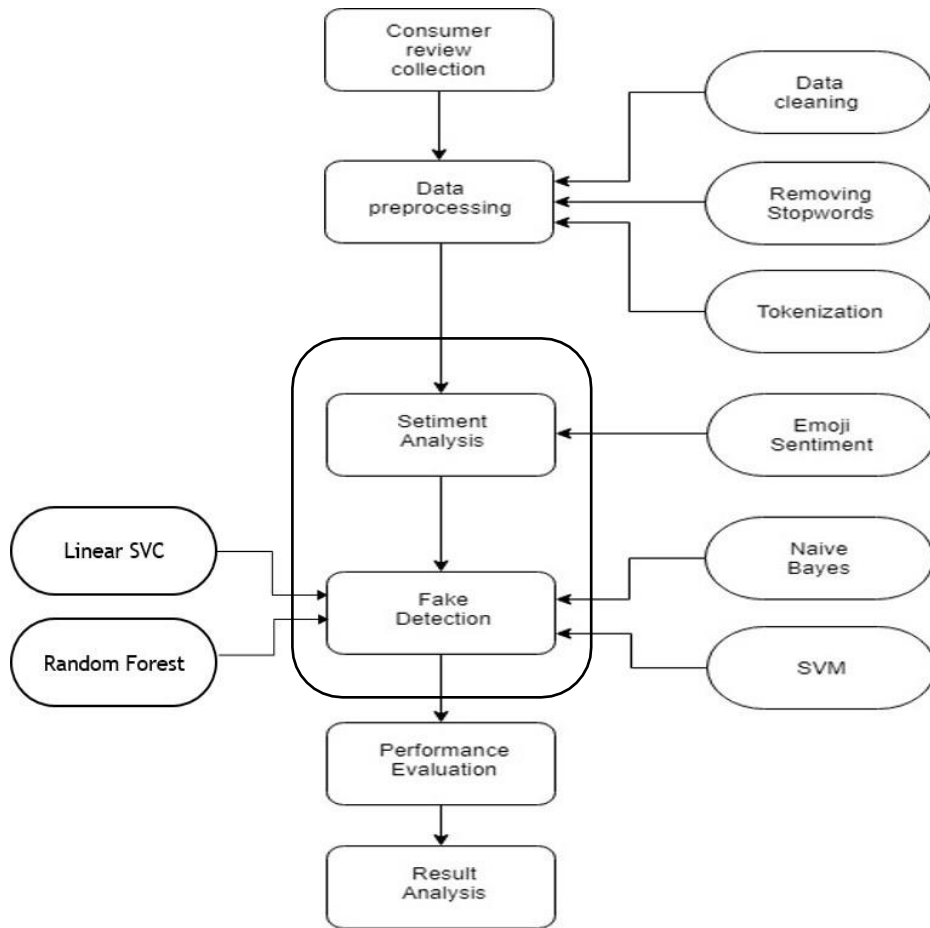


Figure 1: Implementation Architecture

4.1 Data Collection

Consumer review data collection- Raw review data was collected from different sources, such as Amazon, websites for booking Airlines, Hotel and Restaurant, CarGurus, etc. reviews. Doing so was to increase the diversity of the review data. A dataset of 21000 was created.

4.2 Data Preprocess

Processing and refining the data by removal of irrelevant and redundant information as well as noisy and unreliable data from the review dataset.

Step 1 Sentence tokenization

The entire review is given as input and it is tokenized into sentences using NLTK package.

Step 2 Removal of punctuation marks

Punctuation marks used at the starting and ending of the reviews are removed along with additional white spaces.

Step 3 Word Tokenization

Each individual review is tokenized into words and stored in a list for easier retrieval.

Step 4 Removal of stop words

Affixes are removed from the stem. For example, the stem of "cooking" is "cook", and the stemming algorithm knows that the "ing" *suffix* can be removed. A few words from the frequent word list is shown below in Figure: 2.

```
frequent_word_list = ['ourselves', 'hers', 'between', 'yourself', 'but', 'again', 'there', 'about', 'once', 'during
```

Figure 2: Frequent word list sample

4.3 Feature extraction

The preprocessed data is converted into a set of features by applying certain parameters. The following features are extracted:

Normalized length of the review-Fake reviews tend to be of smaller length. Reviewer

ID- A reviewer posting multiple reviews with the same Reviewer ID.

Rating-Fake reviews in most scenarios have 5 out of 5 stars to entice the customer or have the lowest rating for the competitive products thus it plays an important role in fake detection.

Verified Purchase-Purchase reviews that are fake have lesser chance of it being verified purchase than genuine reviews.

Thus these combination of features are selected for identifying the fake reviews.

This in turn improves the performance of the prediction models.

4.4 Sentiment Analysis

Classifying the reviews according to their emotion factor or sentiments being positive, negative or neutral. It includes predicting the reviews being positive or negative according to the words used in the text, emojis used, ratings given to the review and so on. Related research [8] shows that fake reviews has stronger positive or negative emotions than true reviews. The reasons are that, fake reviews are used to affect people opinion, and it is more significant to convey opinions than to plainly describe the facts. The Subjective vs Objective ratio matters: Advertisers post fake reviews with more objective information, giving more emotions such as how happy it made them than conveying how the product is or what it does. Positive sentiment vs negative sentiment: The sentiment of the review is analyzed which in turn help in making the decision of it being a fake or genuine review.

4.5 Fake Review Detection

Classification assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. Each data in the review file is assigned a weight and depending upon which it is classified into respective classes - Fake and Genuine.

4.6 Performance Evaluation and Results

Comparison of the accuracies of various models and classifiers with enhancements for better results, as discussed in Accuracy Enhancements.

Chapter 5: Experimental Configuration

The implementation of this project uses supervised learning technique on the datasets and the fake and genuine labels help us to cross validate the classification results of the data.

Collection of data is done by choosing appropriate dataset. Datasets for such reviews with labels is found from different sources like hotel reviews, amazon product reviews, and other free available review datasets and combined into Reviews.txt file. Firstly, the dataset is explored by loading it as csv format as shown in Figure 3.

```
1 data = pd.read_csv("D:\\Reviews.txt", delimiter = "\t")
1 data.head()
```

	DOC_ID	LABEL	RATING	VERIFIED_PURCHASE	PRODUCT_CATEGORY	PRODUCT_ID	PRODUCT_TITLE	REVIEW_TITLE	REVIEW_TEXT
0	1	__label1__	4	N	PC	B00008NG7N	Targus PAUK10U Ultra Mini USB Keypad, Black	useful	When least you think so, this product will sav...
1	2	__label1__	4	Y	Wireless	B00LH0Y3NM	Note 3 Battery : Stallion Strength Replacement ...	New era for batteries	Lithium batteries are something new introduced...
2	3	__label1__	3	N	Baby	B000I5UZ1Q	Fisher-Price Papasan Cradle Swing, Starlight	doesn't swing very well.	I purchased this swing for my baby. She is 6 m...
3	4	__label1__	4	N	Office Products	B003822IRA	Casio MS-80B Standard Function Desktop Calculator	Great computing!	I was looking for an inexpensive desk calculat...
4	5	__label1__	4	N	Beauty	B00PWSAXAM	Shine Whitening - Zero Peroxide Teeth Whitenin...	Only use twice a week	I only use it twice a week and the results are...

Figure 3: Data exploration

Then to make it readable, the labels in the dataset are clearly labelled as fake or genuine as

shown in Figure 4.

```

1 data.loc[data["LABEL"] == "__label1__", "LABEL"] = 'fake'
2 data.loc[data["LABEL"] == "__label2__", "LABEL"] = 'genuine'

```

```

1 data.head()

```

	DOC_ID	LABEL	RATING	VERIFIED_PURCHASE	PRODUCT_CATEGORY	PRODUCT_ID	PRODUCT_TITLE	REVIEW_TITLE	REVIEW_TEXT
0	1	fake	4	N	PC	B00008NG7N	Targus PAUK10U Ultra Mini USB Keypad, Black	useful	When least you think so, this product will sav...
1	2	fake	4	Y	Wireless	B00LH0Y3NM	Note 3 Battery : Station Strength Replacement ...	New era for batteries	Lithium batteries are something new introduced...
2	3	fake	3	N	Baby	B000I5UZ1Q	Fisher-Price Papasan Cradle Swing, Starlight	doesn't swing very well.	I purchased this swing for my baby. She is 6 m...
3	4	fake	4	N	Office Products	B003822IRA	Casio MS-80B Standard Function Desktop Calculator	Great computing!	I was looking for an inexpensive desk calculat...
4	5	fake	4	N	Beauty	B00PWSAXAM	Shine Whitening - Zero Peroxide Teeth Whitenin...	Only use twice a week	I only use it twice a week and the results are...

Figure 4: Data Labeled

The dataset created from multiple sources of information has many forms of redundant and unclean values. Such type of data is neither useful nor easy to model.

Preprocessing: Data has been cleaned by removing all the null values, white spaces and punctuations. This raw dataset is loaded in the form of <ID, Review text, Label> tuple using the code as shown in Figure 5 allowing to only focus on the textual review content.

```

# Loading the dataset

def dataset_load(reviewfile_path, Text=None):
    with open(reviewfile_path,encoding="utf8") as f:
        reader = csv.reader(f, delimiter='\t')
        next(reader)
        for line in reader:
            (Id, Text, Label) = parseReview(line)
            Data_raw.append((Id, Text, Label))

```

Figure 5: Dataset Load

Then the raw data is preprocessed by applying tokenization, removal of stop words and lemmatization. The code snippet used is shown in Figure 6.

```

1 # Preprocessing, removing stop words, lemmatization
2
3 from nltk.corpus import stopwords
4 from nltk.tokenize import RegexpTokenizer
5 from nltk.stem import WordNetLemmatizer
6 import string
7
8 table = str.maketrans({key: None for key in string.punctuation})
9
10 def preprocess(text):
11     lemmatizer = WordNetLemmatizer()
12     filtered_tokens=[]
13     stop_words = set(stopwords.words('english'))
14     text = text.translate(table)
15     for w in text.split(" "):
16         if w not in stop_words:
17             filtered_tokens.append(lemmatizer.lemmatize(w.lower()))
18     return filtered_tokens

```

Figure 6: Preprocessing

Feature Extraction: The text reviews have different features or peculiarities that can help to solve the classification problem. For e.g. Length of reviews (fake reviews tend to be smaller in length with less facts revealed about the product) and repetitive words (fake reviews have smaller vocabulary with words repeated). Apart from the just the review text there are other features that can contribute towards the classification of reviews as fake. Some of the significant ones that were used as additional features inclusion are Ratings, verified purchase and product category. The code snippet used to extract them is shown in Figure 7.

```

1 featureDict = {}
2
3 def toFeatureVector(Rating, verified_Purchase, product_Category, tokens):
4     localDict = {}
5
6     #Ratings
7
8     featureDict["R"] = 1
9     localDict["R"] = Rating
10
11    #Verified_Purchase
12
13    featureDict["VP"] = 1
14
15    if verified_Purchase == "N":
16        localDict["VP"] = 0
17    else:
18        localDict["VP"] = 1
19
20    #Product_Category
21
22    if product_Category not in featureDict:
23        featureDict[product_Category] = 1
24    else:
25        featureDict[product_Category] = +1
26
27    if product_Category not in localDict:
28        localDict[product_Category] = 1
29    else:
30        localDict[product_Category] = +1

```

Figure 7: Feature Extraction

Figure 8 and Figure 9 show the count of the reviews for each feature.

```

1 count_reviews = data.groupby("VERIFIED_PURCHASE").LABEL.value_counts()
2 count_reviews

```

VERIFIED_PURCHASE	LABEL	
N	fake	7623
	genuine	1679
Y	genuine	8821
	fake	2877

Figure 8: Verified Purchase Review Count

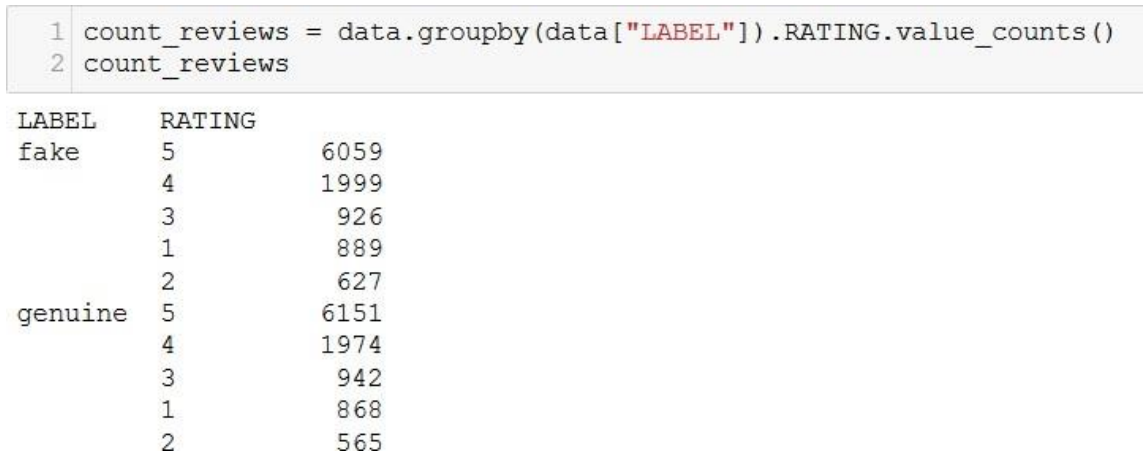


Figure 9: Rating Review count

Sentiment Analysis: This processed data is now analyzed for emotions or sentiment, if the review is positive or negative. The significant factors for doing the sentiment analysis of the reviews are use of emoticons sentiment scores and the rating of the reviews. Note that while removing the punctuation marks a list of emoticons is parsed to be exception, so we do not remove or discard them by accident, while cleaning the dataset. This is explained in more detail in chapter 6 of Accuracy Enhancements under Enhancement 4 section. Sentiment analysis is performed with use of different classification algorithms such as Naïve Bayes, Linear SVC, Non-linear SVM and Random forest to obtain better results and compare the accuracies.

Fake review Detection: This is the final goal of the project to classify these reviews into fake or genuine. The preprocessed dataset is thus classified using different classification algorithms to analyze variety of data to classify it.

5.1 NLP based Text blob Classifier:

The two classifiers used in this configuration are:

a. Naive Bayes classifier

b. Decision Tree classifier

The experimental configuration for both classifiers was kept the same, and this section consists of the configurations used to set up the models for training the Python Client. Naïve Bayes [7] and Decision Tree Classifier are used for detecting the genuine(T) and fake(F) reviews across a wide range of data set. The probability for each word is calculated is given by the ratio of (sum of frequency of each word of a class to the total words for that class). The dataset is split into 80% training 20% testing, 16800 for training and 4200 for testing. Finally, for testing the data using a test set where the probability of each review is calculated for each class. The class with the highest probability value using which the review is assigned the label i.e. true/genuine (T) or fake (F) Review. The datasets used for training are F-train.txt and T-train.txt. They include Review ID (for e.g.ID-1100) as well as the Review text (Great product) shown below in Figure 10 and Figure 11 respectively.

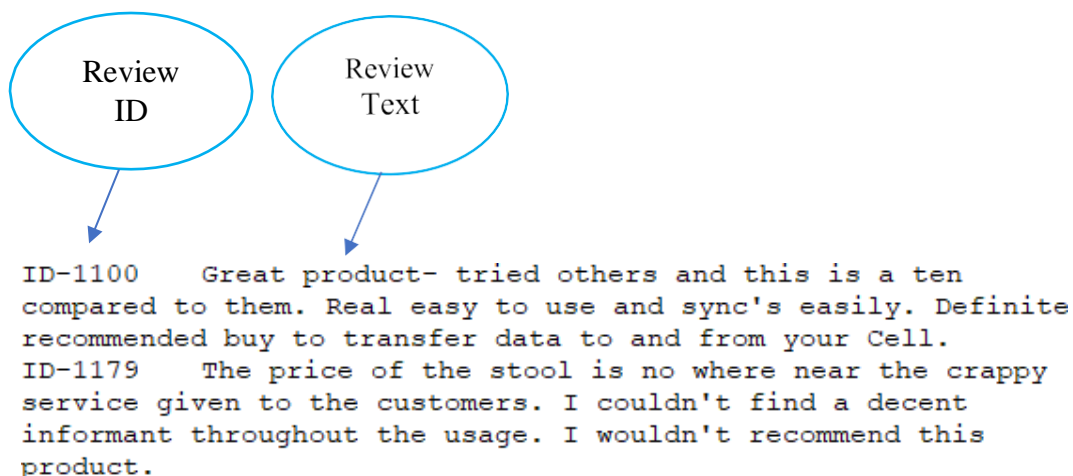


Figure 10: F-train.txt: (Fake review training dataset)

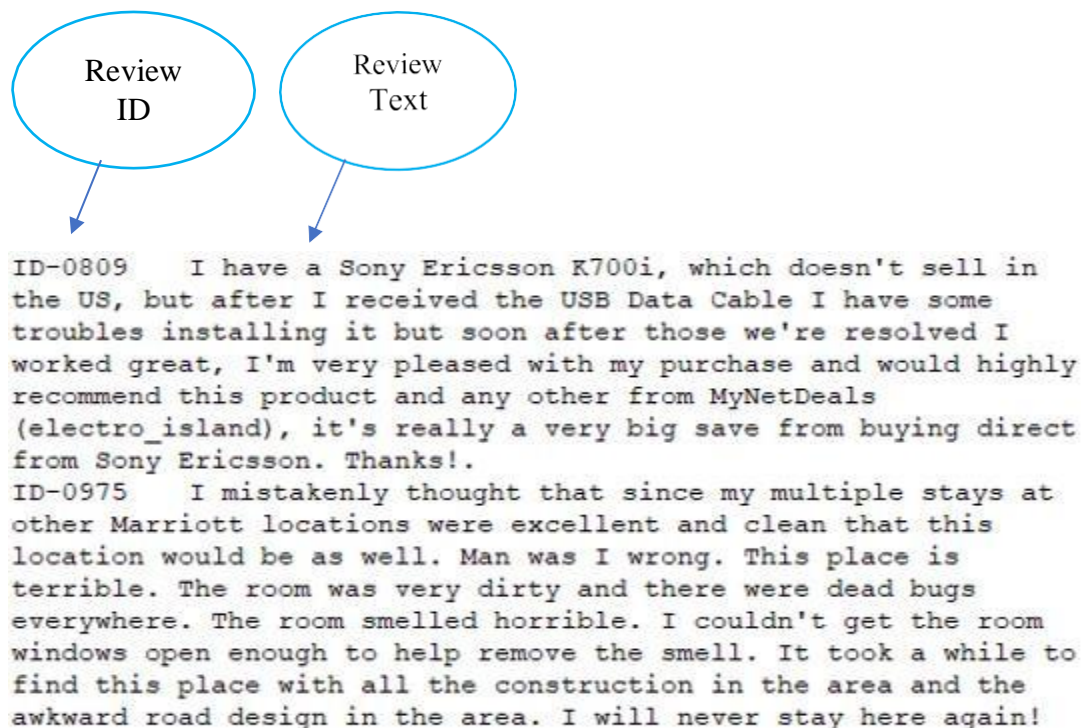


Figure 11: T-train.txt: (True review training dataset)

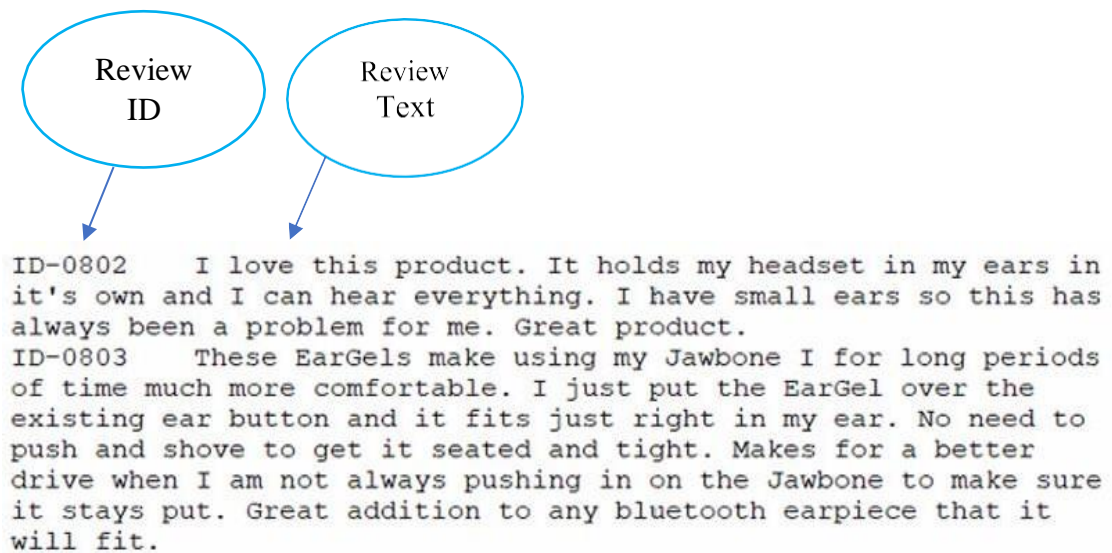


Figure 12: TestingData.txt: (Fake review testing dataset)

Figure 12 contains the testing dataset which has only the ID and text for the review and the output of this after running the model is stored in output.txt which contains the result after prediction as fake or True review alias F / T.

5.2 SKlearn Based Classifiers:

The Sklearn based classifiers were also used for classification and compared which algorithm to get better and accurate results.

a. Multinomial Naïve Bayes: Naive Bayes classifier [7] is used in natural language processing (NLP) problems by predicting the tag of text, calculate probability of each tag of a text and then output choose the highest one.

b. LinearSVC: This classifier classifies data by providing the best fit hyper plane that can be used to divide the data into categories.

c. SVC: Different studies have shown If you use the default kernel in SVC (), the Radial Basis Function (rbf) kernel, then you probably used a more nonlinear decision boundary on the case of the dataset, this will vastly outperform a linear decision boundary

d. Random Forest: This algorithm has also been used for classifying which is provided by sklearn library by creating multiple decision trees set randomly on subset of training data.

For these classifiers Reviews.txt dataset is used. Figure 13 shows the dataset.

```
DOC_ID      LABELRATING  VERIFIED_PURCHASE  PRODUCT_CATEGORY|
      PRODUCT_ID PRODUCT_TITLE  REVIEW_TITLE  REVIEW_TEXT
1  __label1__ 4      N      PC      B00008NG7N Targus PAUK10U Ultra
Mini USB Keypad, Black      useful      When least you think so,
this product will save the day. Just keep it around just in case
you need it for something.
2  __label1__ 4      Y      Wireless  B00LH0Y3NM Note 3 Battery :
Stalion Strength Replacement 3200mAh Li-Ion Battery for Samsung
Galaxy Note 3 [24-Month Warranty] with NFC Chip + Google Wallet
```

Figure 13: Reviews.txt file

After the application of all these classifiers, accuracies for each of them is compared and their performance is evaluated for classification of the fake reviews. There are some more enhancements also made to the models as discussed in the upcoming chapter 6. This provided even better accuracy results for classification of these fake reviews.

5.3 Technologies Used:

5.3.1 Hardware configuration

The machine on which this project was built, is a personal computer with the following configuration:

- Processor: Intel(R) Core i5-7200U @ 2.7GHz

- RAM: 8GB
- System: 64bit OS, x64 processor
- 512 SSD Storage

5.3.2 Software Configuration

- Windows 10
- Python 3.5.2
- Different libraries are available in Python that helps in machine learning, classification projects. Several of those libraries have improved the performance of this project. Few of them are mentioned in this section.
- First, “Numpy” that provides with high-level math function collection to support multi-dimensional matrices and arrays. This is used for faster computations over the weights (gradients) in neural networks.
- Second, “scikit-learn” is a machine learning library for Python which features different algorithms and Machine Learning function packages.
- NLTK, natural language toolkit is helpful in word processing and tokenization.

The project makes use of Anaconda Environment which is an open source distribution for Python which simplifies package management and deployment. It is best for large scale data processing.

training and testing the set using feature vectors. But the applicability of their approach to other type of data has not been validated.

Chapter 6: Accuracy Enhancements

The biggest challenge was generalizing the behavior for the datasets which it was never trained for. In a real-life situation, we can never train a model with every scenario possible. Also, it is not possible to gather the dataset for all kinds of reviews as it all depends on varied dialects. Here are a few techniques or strategies that have significantly improved the model accuracy to classify the reviews as fake or genuine. They are applied in different phases of the project, making them more efficient. These will be discussed in the following section.

6.1 Enhancement 1

Using a predefined sentiment word list to count the sentiment words in each review. This is based on the research where the results have shown that the more the number of sentiment words in a review, the more chances of it being fake. There is a list of sentiment words that the review text is compared against and ratio of words that match from the list to the total number of words. This ratio is considered as one of the factors while determining the fake reviews and is applied during the preprocessing as well as sentiment analysis phase of the experiment.

The predefined sentiment list can be glanced in following picture. B. Liu and M. Hu Sentiment Lexicon is used for sentiment words [8]. It consists of 2 sections:

- a. Positive Words
- b. Negative Words

These are included in the sentimentwordlist.txt file for further reference. A glimpse of which is shown in Figure 14 below.

#Positive Words	#Negative_Word_List
a+	2-faced
abound	2-faces
abounds	abnormal
abundance	abolish
abundant	abominable
accessible	abominably
accessible	abominate
acclaim	abomination
acclaimed	abort
acclamation	aborted
accolade	aborts
accolades	abrade
accommodative	abrasive
acomodative	abrupt
accomplish	abruptly
accomplished	abscond
accomplishment	absence
accomplishments	absent-minded
accurate	absentee
accurately	absurd
achievable	absurdity
achievement	absurdly
achievements	absurdness
achievable	abuse
acumen	abused
adaptable	abuses
adaptive	abusive
adequate	abysmal
adjustable	abysmally
	abyss
	accidental

Figure 14: Sentiment word list

The code snippet to solve make use of this sentiment list is shown in Figure 15.

```

for line in open("Sentiment_Word_List.txt",encoding="utf8").readlines():
    if line.strip():
        word_split = line.split()
        Sentiment_Word_list.append(word_split[0])

Count_senti_dict = OrderedDict()
List_Tokens_ID = []
temp = []
Ordinary_Word_Count = 0
senti_count = 0

for i in range (0,len(Sentiment_Sentence-Token_List)):
    for j in range (0,len(Sentiment_Sentence-Token_List[i])):

        word = Sentiment_Sentence-Token_List[i][j]
        word = word.lower()
        if re.match(r'id-[0-9].*',word):
            Id_Value = word
            continue
        elif word in Sentiment_Word_list:
            senti_count=senti_count+1
        else:
            Ordinary_Word_Count+=1
    Count_senti_dict.update({Id_Value:senti_count})
    senti_count = 0
Senti_Freq_Count = sum(Count_senti_dict.values())
Senti_Prob = np.log10(float(Senti_Freq_Count/float(Ordinary_Word_Count)))
return Senti_Prob

```

Figure 15: Sentiment word list code snippet

6.2 Enhancement 2

Compared the number of verbs and nouns in each review. This is based on the research where the results have shown that the more the number of verbs in a review than number of nouns, the more chances of it being fake. One of the more powerful aspects ofNLTK for Python is the part of speech tagger that is built in. This can be used in the preprocessing phase of the project. Use of NLTK part of speech tagging is done using thefollowing POS tag list:

- NN noun, singular 'desk'

- NNS noun plural 'desks'
- NNP proper noun, singular 'Harrison'
- NNPS proper noun, plural 'Americans'
- VB verb, base form takes
- VBD verb, past tense took
- VBG verb, gerund/present participle taking
- VBN verb, past participle taken
- VBP verb, sing. present, non-3d take
- VBZ verb, 3rd person sing. present takes

The review text can be tagged as verbs and nouns with use of NLTK and thus the count can be compared [9]. The code snippet to do that is shown below in Figure 16

```
def POS_Tagging(sentence):
    tagged_list = []
    tags = []
    count_verbs = 0
    count_nouns = 0
    text=nltk.word_tokenize(sentence)
    tagged_list = (nltk.pos_tag(text))

    tags = [x[1] for x in tagged_list]
    for each_item in tags:
        if each_item in ['VERB', 'VB', 'VBN', 'VBD', 'VBZ', 'VBG', 'VBP']:
            count_verbs+=1
        elif each_item in ['NOUN', 'NNP', 'NN', 'NUM', 'NNS', 'NP', 'NNPS']:
            count_nouns+=1
        else:
            continue
    if count_verbs > count_nouns:
        sentence = 'F'
    elif count_nouns > count_verbs:
        sentence = 'T'
    return sentence
```

Figure 16: POS tagging code snippet

6.3 Enhancement 3

Discount Deception Reviews:

There are reviews by some users which involves the discount prices or sale at some store to distract the buyers to buy from certain sites. These are mostly for promotional purposes, done intentionally by sellers mostly. For considering them in the fake classification, the keywords that are common in such reviews are used to identify. Some of the words on the list are:

- 1.profit
- 2.sale
- 3.percent
- 4.dollars

Use of such words are flagged as fake reviews on the testing dataset. Though it isa bit debatable to directly discard them, in this scenario it is considered as fake.

6.4 Enhancement 4

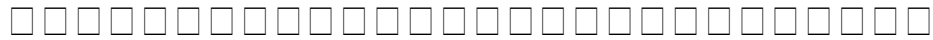
Emoticons based sentiment Classification:

Classification of reviews in datasets according to the set of emoticons used in the reviews by the reviewer, demonstrating the sentiment of the reviewer. The list of emoticons [10] that can be included in as positive negative or neutral is shown in Figure17 below

Positive emojis:



Negative emojis:



Neutral emojis:



Reference:

<https://li.st/jesseno/positive-negative-and-neutral->

Figure 17: Emojis Classification

When cleaning the dataset while preprocessing all punctuations are removed from the reviews text just like sentiment research on emojis [11]. The emojis are kept as an exception by making another list 'items_to_keep[]' in the review text and the snippet of that code is included in Figure18.

```
# Emojis are not discarded
items_to_keep = ['X-(, ':-D', 'B-)', ':(, ':-(, '=P', ':-P', '=)', ':-)', '<3', ':-|', 'X-p', ':-)']
for each_item in items_to_remove:
    if not each_item in items_to_keep:
        del Positive_Tags_Dict[each_item]
Sum_Positive_Freq = sum(Positive_Tags_Dict.values())
Length_Positive_Tags = len (Positive_Tags_Dict)
for key in Positive_Tags_Dict:
    val = np.log10 (float(Positive_Tags_Dict[key]+1)/float(Sum_Positive_Freq + Length_Positive_Tags))
    Positive_Tags_Dict[key] = val
#print.pprint (Positive_Tags_Dict)
return Positive_Tags_Dict, Negative_Tags_Dict
```

Figure 18: Including the emoticons in the data

Studies which focused their experiments on emoticons mainly distributed the intensity of an emotion as an integer polarity. Some of the most commonly used emojis are selected from a list of 751 emojis with respect to their frequency and distinction in the emoji Scores [12]. These scores are referred for finding the sentiment of the reviews that contains emojis. The scores are mentioned below in Figure 19.







Emoji Sentiment Ranking	
Emoji	Score
	0.221
	0.445
	0.746
	-0.093
	-0.173
	-0.397

Figure 19: Emoji Sentiment Ranking

The sentiment scores can be assigned according to the UTF-8 code of the emoticons to recognize the emojis in the reviews.

```

16 # Emoji Sentiment Scores
17
18 emoji_SentimentScores = {}
19
20 #happy, angry, love, sad, playful, confused
21 emoji_SentimentScores["\xF0\x9F\x98\x82"] = 0.221 #0.221*2
22 emoji_SentimentScores["\xF0\x9F\x98\xA1"] = -0.173 #-0.173
23 emoji_SentimentScores["\xe2\x9d\xa4"] = 0.746 #0.746*2
24 emoji_SentimentScores["\xF0\x9F\x98\xAD"] = -0.093 #-0.093*2
25 emoji_SentimentScores["\xF0\x9F\x98\x9C"] = 0.445 #0.445*2
26 emoji_SentimentScores["\xf0\x9f\x98\x95"] = -0.397 #0.397*2

```

Figure 20: Emoji Sentiment score code snippet

The emoticons scores recognition using UTF-8 and code snippet is included in Figure 20. This probability will be used to determine the sentiment of the review which in turn will help determining the genuineness of the reviews. Sentiment classification is done using all same classification algorithms but before actual fake review detection step.

Chapter 7:

Results

Data visualization:

The following visualizations show the kind of data that was used and each depicts how many product categories are there for each label in the Reviews.txt. Here label means fake and genuine. For e.g. for category Instruments there are 350 reviews with label fake as seen in the code snippet is in Figure 21.

```

1 count_reviews=data.groupby(data["LABEL"]).PRODUCT_CATEGORY.value_counts()
2 plt.figure(figsize=(16,8))
3 sns.barplot(count_reviews.index, count_reviews.values, alpha=0.8, color=color[1])
4 plt.ylabel('Number of Occurrences', fontsize=16)
5 plt.xlabel('(Label, Product Category)', fontsize=16)
6 plt.title('Label Vs Product Category', fontsize=18)
7 plt.xticks(rotation='vertical')
8 plt.show()

```

Figure 21: Label vs Product Category code snippet

Observing the number of occurrences of reviews with ratings vs the label they have. For eg. Number of occurrences of reviews with a fake label and rated as 5 out of 5 is more than reviews with a fake label and rated 3. The following Figure 22 shows Label vs Rating code snippet and the comparison Label vs Rating is shown in Figure 23.

```

1 count_reviews = data.groupby(data["LABEL"]).RATING.value_counts()
2 plt.figure(figsize=(16,8))
3 sns.barplot(count_reviews.index, count_reviews.values, alpha=0.8, color=color[1])
4 plt.ylabel('Number of Occurrences', fontsize=16)
5 plt.xlabel('(Label, Rating)', fontsize=16)
6 plt.title('Label Vs Rating', fontsize=18)
7 plt.xticks(rotation='horizontal')
8 plt.show()

```

Figure 22: Label vs Rating code snippet

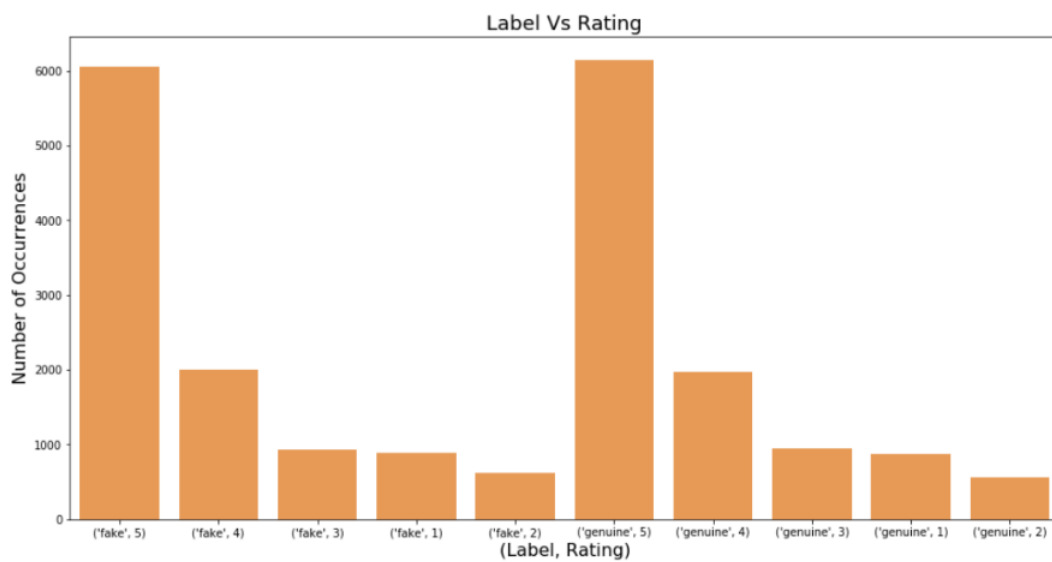


Figure 23: Label vs Rating

Observing the number of occurrences of reviews with emojis vs the label they have. For eg. Number of occurrences of reviews with a fake label and have emojis is less than reviews with a genuine label. The following Figure 24 shows the Label vs Emojis count code snippet and comparison Label vs Emojis is shown in Figure 25.

```

1 data["emojis"] = data["REVIEW_TEXT"].apply(lambda x: 1 if ";" in x.split() or ":" in x.split()
2 count_reviews = data.groupby(["LABEL"]).emojis.agg(lambda x: sum(x))
3 plt.figure(figsize=(16,8))
4 sns.barplot(count_reviews.index, count_reviews.values, alpha=0.8, color=color[3])
5 plt.ylabel('Emojis_count', fontsize=16)
6 plt.xlabel('Label', fontsize=16)
7 plt.title('Emojis_count Vs Label', fontsize=18)
8 plt.xticks(rotation='horizontal')
9 plt.show()

```

Figure 24: Label vs Emoji Count code snippet

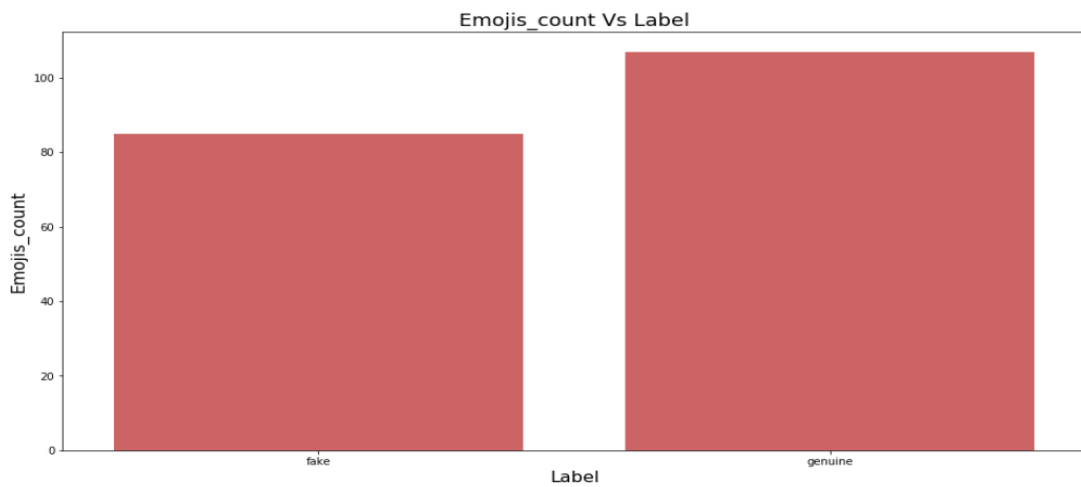


Figure 25: Label vs Emoji Count

Observing the number of occurrences of reviews with stop words counts vs the label they have. For eg. Number of occurrences of reviews with a fake label have stopwords is less than reviews with a genuine label. The following Figure 26 shows the Label vs Stopwords count code snippet and the comparison Label vs Stopwords count in Figure 27

```

1 import nltk
2 wpt = nltk.WordPunctTokenizer()
3 stop_words = nltk.corpus.stopwords.words('english')
4 def stopCount(x):
5     sum =0
6     for char in x.split():
7         sum+= char in stop_words
8     return sum
9 data['stop_count'] = data['REVIEW_TEXT'].apply(stopCount)
10
11 count_reviews = data.groupby(["LABEL"]).stop_count.agg(lambda x: sum(x)/len(x))
12 plt.figure(figsize=(16,8))
13 sns.barplot(count_reviews.index, count_reviews.values, alpha=0.8, color=color[0])
14 plt.ylabel('Stopword counts', fontsize=16)
15 plt.xlabel('Label', fontsize=16)
16 plt.title('Stopwords Counts Vs Label', fontsize=18)
17 plt.xticks(rotation='horizontal')
18 plt.show()

```

Figure 26: Label vs Stopwords Count code snippet

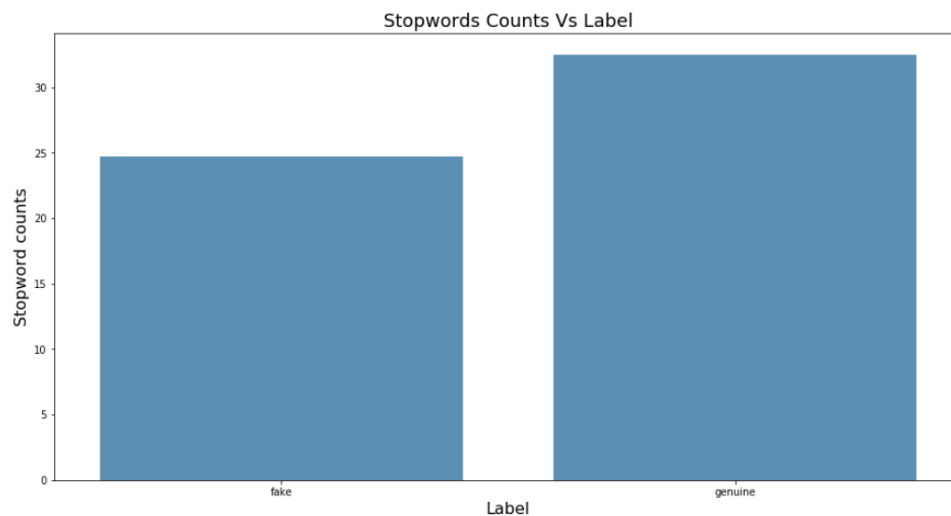


Figure 27: Label vs Stopwords Count

Observing the number of occurrences of reviews with verified purchases or not vs the label they have. For eg. Number of occurrences of reviews with a fake label have way less verified purchases than reviews with a genuine label. The following Figure 28 shows the Label vs Verified Purchases code snippet and comparison Label vs Verified Purchases in Figure 29.

```

1 count_reviews = data.groupby("VERIFIED_PURCHASE").LABEL.value_counts()
2 count_reviews
3 plt.figure(figsize=(16,8))
4 sns.barplot(count_reviews.index, count_reviews.values, alpha=0.8, color=color[1])
5 plt.ylabel('Number of Occurrences', fontsize=16)
6 plt.xlabel('(Verified Purchase (Y/N), Label)', fontsize=16)
7 plt.title('Verified_Purchase Vs Labels', fontsize=18)
8 plt.xticks(rotation='horizontal')
9 plt.show()

```

Figure 28: Label vs Verified Purchase

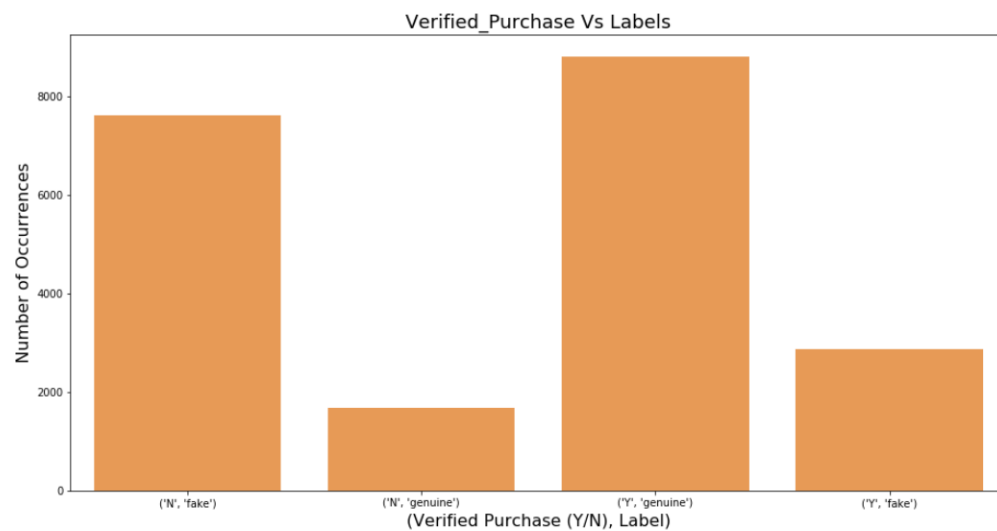


Figure 29: Label vs Verified Purchase

These snippets of code can be observed in the DataVisualization.ipynb file for further reference.

The following output.txt file is the result generated by textblob Naïve Bayes Classifier. It can be shown in Figure 30.

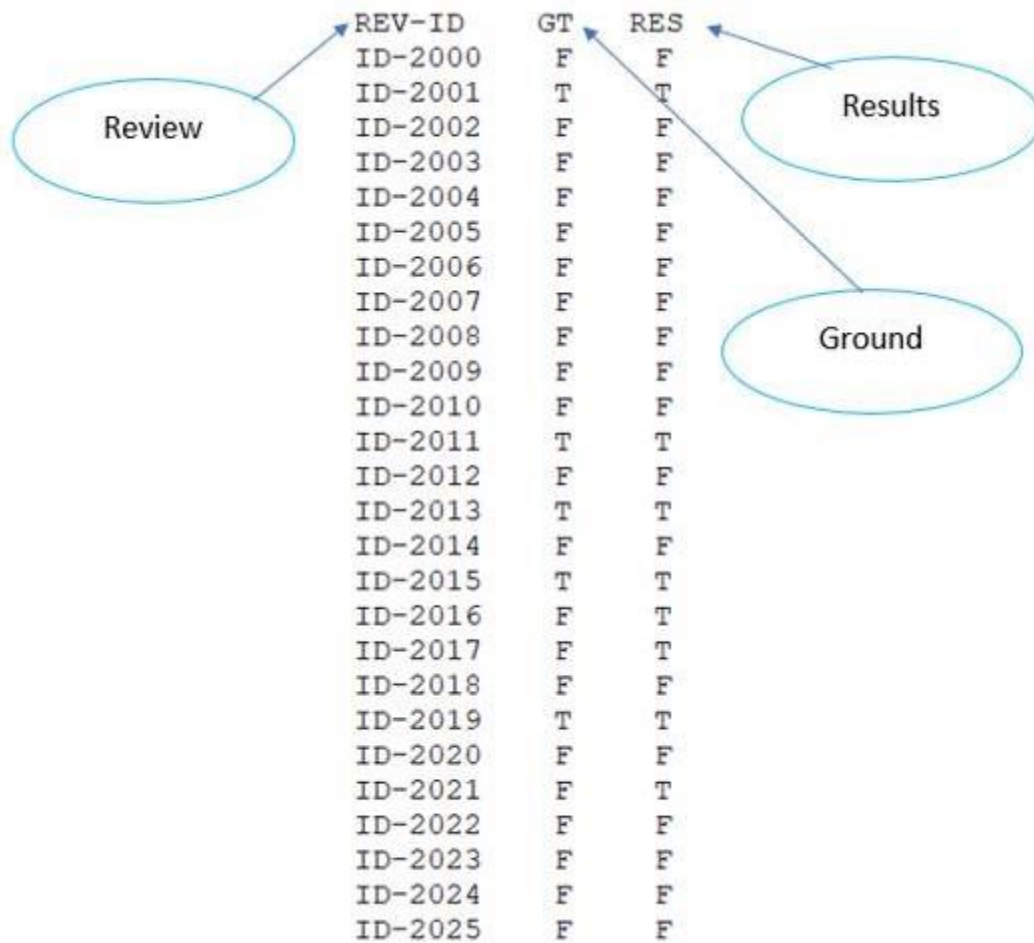


Figure 30: Output.txt: (Classified testing output dataset)

The accuracy scores obtained for this dataset are shown as follows:

Accuracy-80.542

F1 score-77.888

Precision Score-80.612

Recall-79.001

The following results were observed for each of the previously described experimental setups. The results show how the accuracy has improved after each enhancement to the model in Table 1

	Raw data w/ Tokenization	Preprocessing & Lemmatization	Feature inclusion	Testing data	Sentiment classifier
Multinomial Naïve Bayes	72%	77%	81%	80%	84%
Linear SVC	67%	70%	74%	73%	83%
SVM	69%	75%	77%	81%	81%
Random Forest	68%	70%	72%	71%	79%

Table 1: Results

Another plotting of the results is shown in Figure 31 which depicts the bar chart for each classifier with a different color for a data of 21000 in total.

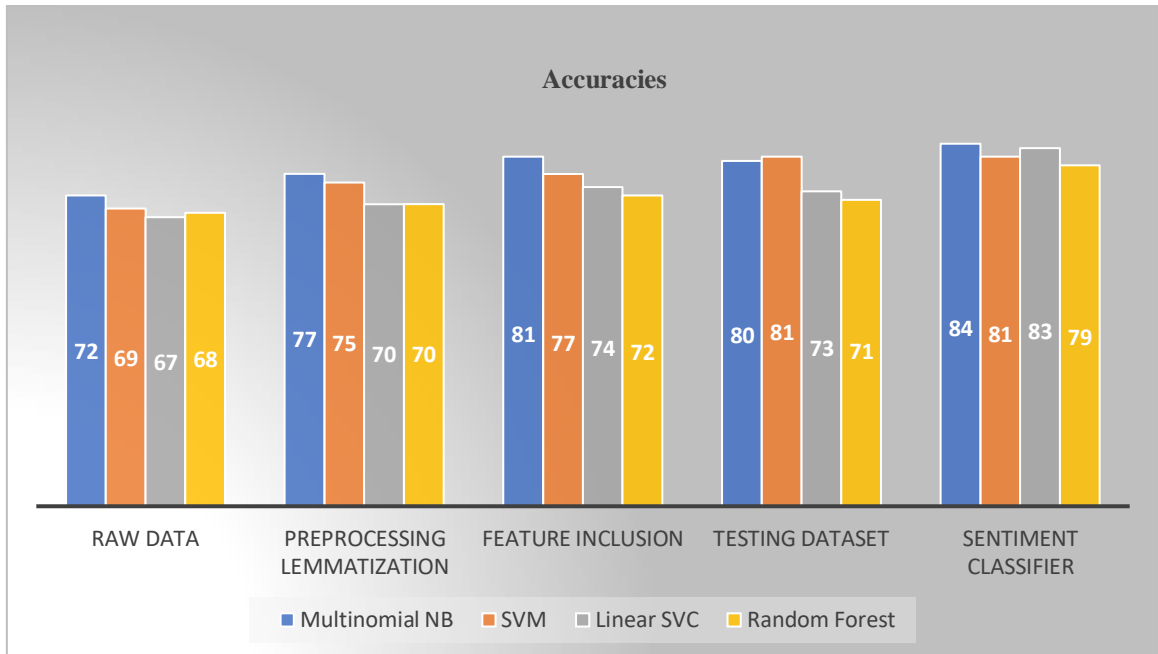


Figure 31: Results graph

Raw data is loaded from Reviews.txt file and by just parsing it and tokenizing, accuracy of each model is calculated to predict the reviews being fake or genuine. The best results were obtained using Naïve Bayes classifier as evident in the figure.

Preprocessing and lemmatization of the review text is done, accuracy of each model is calculated to predict the reviews being fake or genuine. The best results were obtained using Naïve Bayes classifier.

Additional feature inclusion works on including additional features like verified purchase, ratings, product category of the review. Previously, the data features used were only in an ID, Text, Label tuple from each review in the dataset. After utilizing these other features, the accuracy of the models increased and can see the improvements in the results for each of the classifiers.

Testing Dataset covers the classification accuracy for the reviews in the testing dataset. Here as you observed the non-linear SVM classifier performed the best and could give 81% accuracy. This shows it could generalize and predict the fake reviews more accurately compared to Naïve Bayes classification which outperformed pretty much in all the other scenarios.

Sentiment classifier includes predicting the reviews being positive or negative according to the emojis used, the count of positive or negative word ratio, ratings given to the review. This sentiment classification is in turn used in predicting the reviews being fake or genuine. The accuracy results show how each model performed on sentiment prediction of the reviews in the dataset.

Enhancement 1 is used in predicting the sentiment of the reviews using the list of positive and negative words in the review.

Enhancement 2 compares the number of verbs and nouns in each review and included in the preprocessing and lemmatization step.

Enhancement 3 is discount deceptive reviews predicted, it has increased the accuracy, but this can be regarded as infinitesimally small to be included in the results.

Enhancement 4 is using emojis has added to the overall performance of the model that helped in most accurate measure. It has improved the sentiment analysis of the reviews, and in turn helped the performance of the models to predict whether the review is fake or genuine.

Chapter 8: Conclusion

The fake review detection is designed for filtering the fake reviews. In this research work SVM classification provided a better accuracy of classifying than the Naïve Bayes classifier for testing dataset. On the other hand, the Naïve Bayes classifier has performed better than other algorithms on the training data. Revealing that it can generalize better and predict the fake reviews efficiently. This method can be applied over other sampled instances of the dataset. The data visualization helped in exploring the dataset and the features identified contributed to

the accuracy of the classification. The various algorithms used, and their accuracies show how each of them have performed based on their accuracy factors.

Also, the approach provides the user with a functionality to recommend the most truthful reviews to enable the purchaser to make decisions about the product. Various factors such as adding new vectors like ratings, emojis, verified purchase have affected the accuracy of classifying the data better.

Chapter 9: Future Work

1. To use a real time/ time based datasets which will allow us to compare the user's timestamps of the reviews to find if a certain user is posting too many reviews in a short period of time.
2. To use and compare other machine learning algorithms like logistic regression to extend the research to deep learning techniques.
3. To develop a similar process for unsupervised learning for unlabeled data to detect fake reviews.

References

- E. I. Elmurungi and A.Gherbi, "FAKE CONSUMER REVIEW DETECTION
1. " *Journal of Computer Science*, vol. 14, no. 5, pp.
 2. J. Leskovec, "WebData Amazon reviews," [Online]. Available: <http://snap.stanford.edu/data/web-Amazon-links.html> [Accessed: October 2018].
 3. J. Li, M. Ott, C. Cardie and E. Hovy, "Towards a General Rule for Identifying Deceptive Opinion Spam," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, MD, USA, vol. 1, no. 11, pp. 1566-1576, November 2014.
 4. N. O'Brien, "Machine Learning for Detection of Fake News," [Online]. Available: <https://dspace.mit.edu/bitstream/handle/1721.1/119727/1078649610-MIT.pdf> [Accessed: November 2018].
 5. J. C. S. Reis, A. Correia, F. Murai, A. Veloso, and F. Benevenuto, "Supervised Learning

for Fake News Detection,” *IEEE Intelligent Systems*, vol. 34, no. 2, pp. 76-81, May 2019.

6. B. Wagh, J. V. Shinde and P. A. Kale, “A Twitter Sentiment Analysis Using NLTK and Machine Learning Techniques,” *International Journal of Emerging Research in Management and Technology*, vol. 6, no. 12, pp. 37-44, December 2017.
7. A. McCallum and K. Nigam, “A Comparison of Event Models for Naive Bayes Text Classification,” in *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, Pittsburgh, PA, USA, vol. 752, no. 1, pp. 41-48, July 1998.
8. B. Liu and M. Hu, “Opinion Mining, Sentiment Analysis and Opinion Spam Detection,” [Online]. Available: <https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon> [Accessed: January 2019].
9. C. Hill, “10 Secrets to Uncovering which Online Reviews are Fake,” [Online]. Available: <https://www.marketwatch.com/story/10-secrets-to-uncovering-which-online-reviews-are-fake-2018-09-21> [Accessed: March 2019].
10. J. Novak, “List archive Emojis,” [Online]. Available: <https://li.st/jesseno/positive-negative-and-neutral-emojis-6EGfnd2QhBsa3t6Gp0FRP9> [Accessed: June 2019].
11. P. K. Novak, J. Smailović, B. Sluban and I. Mozeti, “Sentiment of Emojis,” *Journal of Computation and Language*, vol.10, no. 12, pp. 1-4, December 2015.

12. P. K. Novak, “Emoji Sentiment Ranking,” [Online]. Available: http://kt.ijs.si/data/Emoji_sentiment_ranking/ [Accessed: July 2019]

