DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING
VELLORE INSTITUTE OF TECHNOLOGY

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

TOPIC:
TWITTER SENTIMENT ANALYSIS
(USING NAIVE BAYES)

DATA MINING AND TECHNIQUES (ITE2006)
J COMPONENT

SUBMITTED BY:
UTKARSH GOYAL - 19BIT0402
DIPESH BALANI - 19BIT0354
PARAS DANG - 19BIT0378

UNDER THE GUIDANCE OF PROF. RANICHANDRA C
SLOT D1+TD1

DECLARATION BY THE CANDIDATE

I/We hereby declare that the project report entitled **"Twitter Sentiment Analysis"** submitted by us to Vellore Institute of Technology University, Vellore in partial fulfilment of the requirement for the award of the course **Data Mining Techniques (ITE2006)** is a record of bonafide project work carried out by us under the guidance of **.** We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other course.

Place : Vellore

Date : 8th June 2021 Signature:

Utkarsh Goyal,
Paras Dang,
Dipesh Balani

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology & Engineering [SITE]

CERTIFICATE

This is to certify that the project report entitled **"Twitter Sentiment Analysis"** submitted by **Dipesh Balani (19BIT0354), Utkarsh Goyal (19BIT0402), Paras Dang(19BIT0378)** to Vellore Institute of Technology University, Vellore in partial fulfillment of the requirement for the award of the course **Data Mining Techniques (ITE2006)** is a record of bonafide work carried out by them under my guidance.

Prof. Ranichandra C

GUIDE
Associate Professor, SITE

TWITTER SENTIMENT ANALYSIS USING NAIYE BAYES

Abstract

Sentiment analysis is a form of shallow semantic analysis of texts. In the project we used the automatic approach that involves supervised machine learning and text mining classification algorithms which includes the sentimental analysis in various applications.

Naive Bayes is one of the first machine learning concepts that people learn in a machine learning class. This method seems more like a statistical approach to getting conclusions; i.e. more like the basis for which other machine learning techniques work rather than being one itself. Demonstrating one method of implementing it to create a: Binary sentiment analysis of Twitter Texts, this can be easily applied to having multiple classifications.

INTRODUCTION

Sentiment analysis is the automated process of identifying and classifying subjective information in text data. This might be an opinion, a judgment, or a feeling about a particular topic or product feature.

The most common type of sentiment analysis is 'polarity detection' and involves classifying statements as *positive*, *negative* or *neutral*. A polarity sentiment analysis model, for example, automatically tags this tweet as *positive*:

Sentiment analysis uses Natural Language Processing (NLP) to make sense of human language, and machine learning to automatically deliver accurate results.

BACKGROUND

In essence, the automatic approach involves supervised machine learning and text mining classification algorithms. The sentiment analysis is one of the more sophisticated examples of how to use classification to maximum effect through the text mining. In addition to that, unsupervised machine learning algorithms are used to explore data.
Overall, Sentiment analysis may involve the following types of classification algorithms:

LITERATURE SURVEY:

PAPER 1:

Sentiment analysis using product review data

Xing Fang* and Justin Zhan

The paper is all about categorization of sentiment polarity based on sentiment analysis. When specified with a text we can categorize them into being positive, negative or being neutral.

There are three levels in sentiment polarity categorization. The document level concerns whether a document, as a whole, expresses negative or positive sentiment, while the sentence level deals with each sentence's sentiment categorization; The entity and aspect level then targets on what exactly people like or dislike from their opinions.

Total Sentiment Index can be used to categorize a token , whether it is a positively classified token or whether it is negative. TSI index can be calculated using a generalized formula. Some of the methods of analysis for the document includes scikit-learn , an open source machine learning software package in Python. The classification models selected for categorization are: Naïve Bayesian, Random Forest, and Support Vector Machine .

PAPER 2:

Sentiment Analysis in E-Commerce: A Review on The Techniques and Algorithms

Muhammad Marong School of Computing and Technology Asia Pacific University of Technology & Innovation Malaysia
Nowshath K Batcha School of Computing and Technology Asia Pacific University of Technology & Innovation Malaysia nowshath.
Raheem Mafas School of Computing and Technology Asia Pacific University of Technology & Innovation Malaysia

Accessibility of social media platforms empowered the internet users to express and share their opinions on different kinds of components based on their life experience, including products and services that they enjoy.
Sentiment Analysis has been a burgeoning technology that taps into customer demands based on Natural Language Processing. This motivation is usually used to properly understand what customers want, when, why and how they want it, retailers need to pivot toward sentiment analysis, hence avoid doing the same mistakes and choosing the right decisions based on comments or reviews. As part of e-commerce, online shopping is a good example of how products or services are sold over the Internet. Big name distributors like Amazon and Ali-baba along with tiny distributors out there certainly had disappointing outcomes, one of the primary factors for their slow sales was poor product assortment.
Consumer understanding has always been high on the to-do list of distributors and the use of sentiment analysis to monitor those emotions was the main motive for businesses to understand how diverse and thorough the opinion mining on the clients' reviews can be.

Some of the techniques that can be used are:. In sentiment classification, there are two main study fields such as Machine Learning and Lexicon, and each field has its own subdivision.

PAPER 3:

Sentiment analysis and opinion mining applied to scientific paper reviews

Brian Keith Norambuena , Exequiel Fuentes Lettura and Claudio Meneses Villegas Department of Computing and Systems Engineering, Universidad Católica del Norte, Coquimbo, Chile

The study focuses on classifying reviews according to the scale determined by the authors. Original evaluations will be used as complements for evaluating the consistency between the classification inferred from the text and the one assigned by the reviewer.
The consistency evaluation between the written review and the reviewers' score is proposed as a practical application of sentiment classification. For these reasons, the classifier used in this study was trained according to manual data tagging, not the reviewer's original classification. This allows revising the consistency between what the review states and what the reviewer says about the paper acceptance or rejection.
Some of the tools and methods used in this review are : Python programming language, Scikit-learn library, Stanford POS Tagger library, SentiWordNet 3.0 lexical ontology.
Algorithms that can be used are :SVM and Naive Bayes.

PAPER 4:

Sentiment Analysis of Persian-English Code-mixed Texts

1st Nazanin Sabri electrical and computer engineering University of Tehran Tehran, Iran
2nd Ali Edalat electrical and computer engineering University of Tehran Tehran, Iran
3 rd Behnam Bahrak electrical and computer engineering University of Tehran Tehran, Iran

In this study we collect, label and thus create a dataset of Persian-English code-mixed tweets. We then proceed to introduce a model which uses BERT pretrained embeddings as well as translation models to automatically learn the polarity scores of these Tweets. Our

model outperforms the baseline models that use Na¨ıve Bayes and Random Forest methods.

We aim to create a vectorized representation of the textual input in order to be able to fit the data into our machine learning model.

PAPER 5:

A study of social sentiment analysis in the times of covid -19 using twitter

Princy Sharma1, Prof. Vibhakar Mansotra2

Twitter's strengths real-time and each one the tweets are publicly available and are easily accessible with their geo-tagged locations. Natural language Processing (NLP)is the sub-branch of knowledge science and it's assumed to be vital neighbourhood of data science. It generally teaches machines to read and interpret human readable texts. It converts information from computer databases or sentiment intents into readable human language. Sentiment analysis has been handled as a Natural Language Processing task at many levels of granularity. Sentiment Analysis of text identifies and extracts subjective information in the source material, and helping a business to know the social sentiment of their brand, product, or service while monitoring online conversations.

From a user's perspective, people are posting their own content through various social media, such as forums, micro-blogs, or online social networking sites. Sentiment analysis used in movie reviews, Product reviews, politics, public sentiments and social sites.

Fake opinions or fake reviews misguide the readers by providing them untruthful negative or positive opinions related to any object

TECHNIQUES USED IN SENTIMENT ANALYSIS

Natural language processing ,Machine LearningThere are two kinds of machine learning techniques like supervised machine learning algorithms like maximum entropy, SVM, Naïve Bayes, KNN, etc and unsupervised machine learning algorithms like Neural

network, Principal Component Analysis, ICA, SVD, etc.

PAPER 6:

Twitter Sentiment Analysis: A Political View

Joylin Priya Pinto1, Vijaya Murari T.2, Soumya Kelur3

This research aims to analyse political orientation of twitter users as positive or negative. Different machine learning algorithms are adopted to identify the user point of view on Ayodhya issue. Then the efficiencies of these built models are contrasted with one another to discover the best machine learning algorithm on text categorisation. The result is analysed based on the measurement metrics namely accuracy, precision, recall, F1- score measures for each sentiment class

Author implemented Naïve Bayes and Support Vector Machine classifiers in order to group the twitter data into positive and negative tweets. At that point, the locations are put into categories and sentiment mapping is done which helped in analysing the opinions of individual Indian states separately. Important features were considered for classification such as latitude, longitude, kilometres and number of tweets to distinguish the opinions based on region. Importance of dataset pre-processing is also demonstrated in the paper. However the author recognized state-wise people reactions to demonetization, because of population polarization, the general feeling of the citizens couldn't be caught.

PAPER 7:

Topic Sentiment Analysis in Online Learning Community from College Students

Kai Wang, Yu Zhang†

Opinion mining and sentiment analysis in Online Learning Community can truly reflect the students' learning situation, which provides the necessary theoretical basis for following revision of teaching plans. To improve the accuracy of topic-sentiment analysis, a novel model for topic sentiment analysis is proposed that outperforms other state-of-art models.

Following methods have been used:

1)Precision contrast between different methods based on SVM. 2)Recall contrast between different methods based on SVM.

3)Measure contrast between different methods based on SVM 4)MAE contrast between different methods based on SVM.

This paper designs a model for online sentiment analysis of various topics in OLC. The model obtains the topic-terminology hybrid matrix and the documenttopic hybrid matrix by selecting the real user's comment information on the basis of LDA topic detection approach. Afterwards, a topic clustering concept lattice based on FCA model is constructed, where the topic sentiment can be identified by measuring their sentiment scores.

PAPER 8:

Sentiment Analysis on Large Scale Amazon Product Reviews

Sayyed Johar, Samara Mubeen

In this paper, the fast and in memory computation framework 'Apache Spark' to extract live tweets and perform sentiment analysis. The primary aim is to provide a method for analyzing sentiment score in noisy twitter streams. This paper reports on the design of a sentiment analysis, extracting vast number of tweets. Results classify user's perception via tweets into positive and negative. Sentiment analysis is the prediction of emotions in a word, sentence or corpus of documents. It is intended to serve as an application to understand the attitudes, opinions and emotions expressed within an online mention.

Apache Spark is an open source lightning fast cluster computing platform to retrieve streaming data and forwarding to storage system like Database Server. Apache spark is an in-memory fast processing system used for large scale data processing

The conducted experiments through sentiment classification algorithms have shown the performance measures of precision, recall and accuracy.They have applied NB and SVM classifiers. These classifiers provide a useful perspective for understanding and evaluating many learning algorithms.

PAPER 9:

Improving Sentiment Analysis with Biofeedback Data

Daniel Schlor , Albin Zehe , Konstantin Kobs , Blerta Veseli, Franziska Westermeier, Larissa Brubach, Daniel Roth, Marc Erich Latoschik, Andreas Hotho

Humans frequently are able to read and interpret emotions of others by directly taking verbal and non-verbal signals in human-to-human communication into account or to infer or even experience emotions from mediated stories. For computers, however, emotion recognition is a complex problem: Thoughts and feelings are the roots of many behavioural responses and they are deeply entangled with neurophysiological changes within humans. As such, emotions are very subjective, often are expressed in a subtle manner, and are highly depending on context. For example, machine learning approaches for text-based sentiment analysis often rely on incorporating sentiment lexicons or language models to capture the contextual meaning.

This paper explores if and how we further can enhance sentiment analysis using biofeedback of humans which are experiencing emotions while reading texts. Specifically, we record the heart rate and brain waves of readers that are presented with short texts which have been annotated with the emotions they induce. We use these physiological signals to improve the performance of a lexicon-based sentiment classifier. We find that

the combinatcion of several biosignals can improve the ability of a text-based classifier to detect the presence of a sentiment in a text on a per-sentence level.

In this study, we compare Random Forests (RF) and linear Support Vector Machines (SVMs) for the detection and classification of sentiment from biofeedback. For both machine learning models, we use the implementation in scikitlearn (Pedregosa et al., 2011) with default parameters. We modify the number of decision trees in the Random Forest to be ten due to the faster training time and better generalization for this low data setting.

PAPER 10:

Measuring News Sentiment

Adam Hale Shapiro, Moritz Sudhof , and Daniel Wilson

This paper demonstrates state-of-the-art text sentiment analysis tools while developing a new time-series measure of economic sentiment derived from economic and financial newspaper articles from January 1980 to April 2015. We compare the predictive accuracy of a large set of sentiment analysis models using a sample of articles that have been rated by humans on a positivity/negativity scale. The results highlight the gains from combining existing lexicons and from accounting for negation. We also generate our own sentiment-scoring model, which includes a new lexicon built specifically to capture the sentiment in economic news articles. This model is shown to have better predictive accuracy than existing, "off-the-shelf", models. Lastly, we provide two applications to the economic research on sentiment.

We estimate the impulse responses of macroeconomic variables to sentiment shocks, finding that positive sentiment shocks increase consumption, output, and interest rates and dampen inflation.

Various kind of estimation techniques and scores have been alloted to estimate the scores of various newspapers.

PAPER 11:

Predicting Stock Market Price Movement Using Sentiment Analysis: Evidence From Ghana

Isaac Kofi Nti1, Adebayo Felix Adekoya , Benjamin Asubam Weyori

Predicting the stock market remains a challenging task due to the numerous influencing factors such as investor sentiment, firm performance, economic factors and social media sentiments. However, the profitability and economic advantage associated with accurate prediction of stock price draw the interest of academicians, economic, and financial analyst into researching in this field. Despite the improvement in stock prediction accuracy, the literature argues that prediction accuracy can be further improved beyond its current measure by looking for newer information sources particularly on the Internet. Using web news, financial tweets posted on Twitter, Google trends and forum discussions, the current study examines the association between public sentiments and the predictability of future stock price movement using Artificial Neural Network (ANN).

A sentiment analysis of news for predicting stock price movement using SVM enhanced with Particle Swarm Optimization (PSO) technique was proposed in . The study confirmed a correlation between web news and stock price movement volatility increased in strength during financial crisis.

PAPER 12:

Sentiment Analysis of Open Source Software Community Mailing List: A Preliminary Analysis

Jumoke Abass Alesinloye ,Subathra Srinivasan, Eoin Groarke, Greg Curran, Jaganath Babu, Denis Dennehy

Open source software has become increasingly popular with companies looking to create

business value through collaboration with distributed communities of organizations and software developers who rely on mailing lists to review code and share their feedback.
OSS has undergone a transformation distancing itself from its free software antecedent, leveraging open source communities to increase development productivity and increased functionality.
The sentiment model used works by breaking an email body into lists of sentences, and further breaking these sentences into lists of words. Using the Natural Language Toolkit (NLTK) library, the words are tagged with parts of speech tags (subject, verb, noun etc.). The model tags each word into positive or negative words using an updated custom library with DPDK software terminology being added to the respective dictionaries. Finally, a score is generated based on the count of positive words and the count of negative words.

PAPER 13:

Exploring Online Drug Reviews using Text Analytics, Sentiment Analysis and Data Mining Models

Thu Dinh, Goutam Chakraborty, Miriam McGaugh

Drug reviews play a very significant role in providing crucial medical care information for both healthcare professionals and consumers. Customers are utilizing online review sites to voice opinions and express sentiments about experienced drugs. However, a potential buyer typically finds it very hard to go through all comments before making a purchase decision. Another big challenge is the unstructured and textual nature of the reviews, which makes it difficult for readers to classify comments into meaningful insights. For these reasons, this paper primarily aims to classify the side effect level and effectiveness level of prescribed drugs by using text analytics and predictive models within SAS® Enterprise.

Additionally, the paper explores specific effectiveness and potential side effects of each prescription drug through sentiment analysis and text mining within SAS® Visual Text

Analytics.

These models are further validated by using a transfer learning algorithm to evaluate performance and generalization. The results can be used to develop practical guidelines and useful references to facilitate prospective patients in making better informed purchase decisions.
Regression, Decision Tree and Neural Networks are been used.

PAPER 14:

Managing Marketing Decision-Making with Sentiment Analysis: An Evaluation of the Main Product Features Using Text Data Mining

Erick Kauffmann , Jesús Peral , David Gil , Antonio Ferrández , Ricardo Sellers and Higinio Mora

Companies have realized the importance of "big data" in creating a sustainable competitive advantage, and user-generated content (UGC) represents one of big data's most important sources. From blogs to social media and online reviews, consumers generate a huge amount of brand-related information that has a decisive potential business value for marketing purposes. Particularly, we focus on online reviews that could have an influence on brand image and positioning. Within this context, and using the usual quantitative star score ratings, a recent stream of research has employed sentiment analysis (SA) tools to examine the textual content of reviews and categorize buyer opinions. Although many SA tools split comments into negative or positive, a review can contain phrases with different polarities because the user can have different sentiments about each feature of the product. Finding the polarity of each feature can be interesting for product managers and brand management
In this paper, we present a general framework that uses natural language processing (NLP) techniques, including sentiment analysis, text data mining, and clustering techniques, to obtain new scores based on consumer sentiments for different product features. The main

contribution of our proposal is the combination of price and the aforementioned scores to define a new global score for the product, which allows us to obtain a ranking according to product features. Furthermore, the products can be classified according to their positive, neutral, or negative features (visualized on dashboards), helping consumers with their sustainable purchasing behavior.

After the experimentation, we could conclude that our work is able to improve recommender systems by using positive, neutral, and negative customer opinions and by classifying customers based on their comments.

PAPER 15:

A Review on Sentimental Analysis on Facebook Comments by using Data Mining Technique

Rupinder Kaur,Dr. Harmandeep Singh, Dr. Gaurav Gupta

Social sites for example Facebook and Twitter are that, where characters put their status or sentiments. People comment on their facebook account concerning any correct subject of their consideration.

Sentiment analysis can be seen as a utilization of content order. The primary occupation of content gathering is how to stamp writings with a predefined set of gatherings. Content gathering has been helpful in different zones for example, article ordering and content cleaning. Huge quantities of comments or surveys are posted by people in general every day. So to distinguish the assessment of open towards a particular post is by physically analyze and discover each comment. People use very awkward words to express their feelings & most of the people use shortcuts e.g. osm for awesome, lol for laughing out loud & many more, so this is sometime creating difficulty for the person who is not familiar with these words and cannot recognize the sentiments of the person.

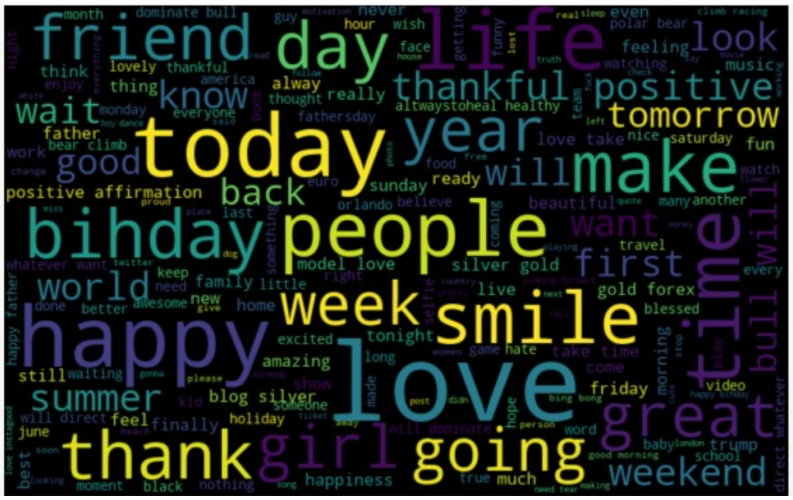
Process of data mining or knowledge discovery in database:

Support vector machine, Decision tree, A decision tree , Regressions, Prediction.

DATASET DESCRIPTION & SAMPLE DATA

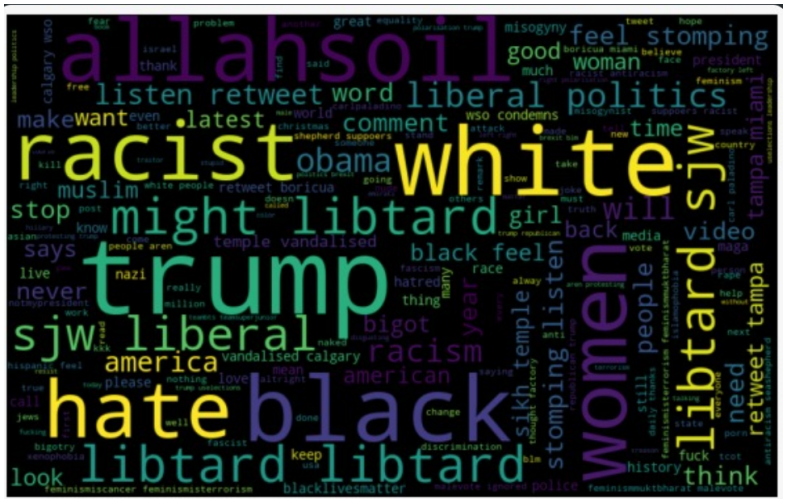
According to the experiment performed in a sample data, we chose one of the data set and achieved the following description in the content of the movie review tweets.

ALL TWEETS



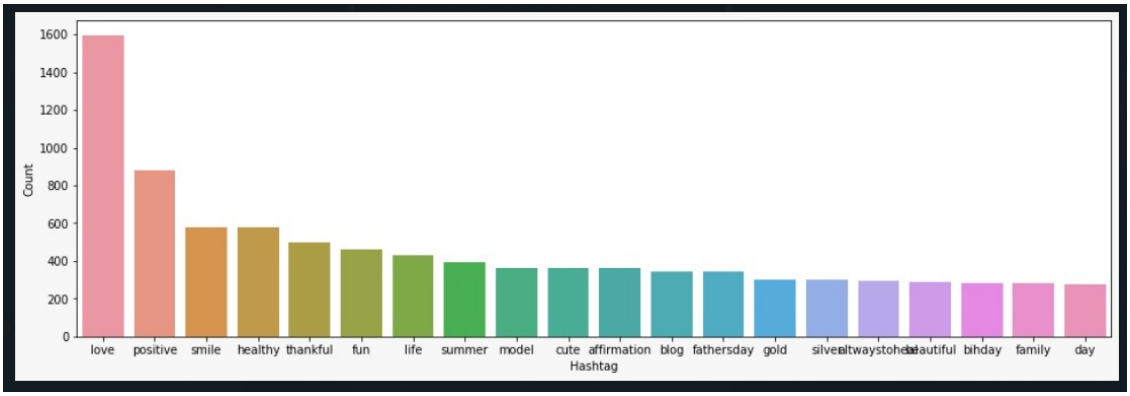
POSITIVE TWEETS

NEGATIVE TWEETS



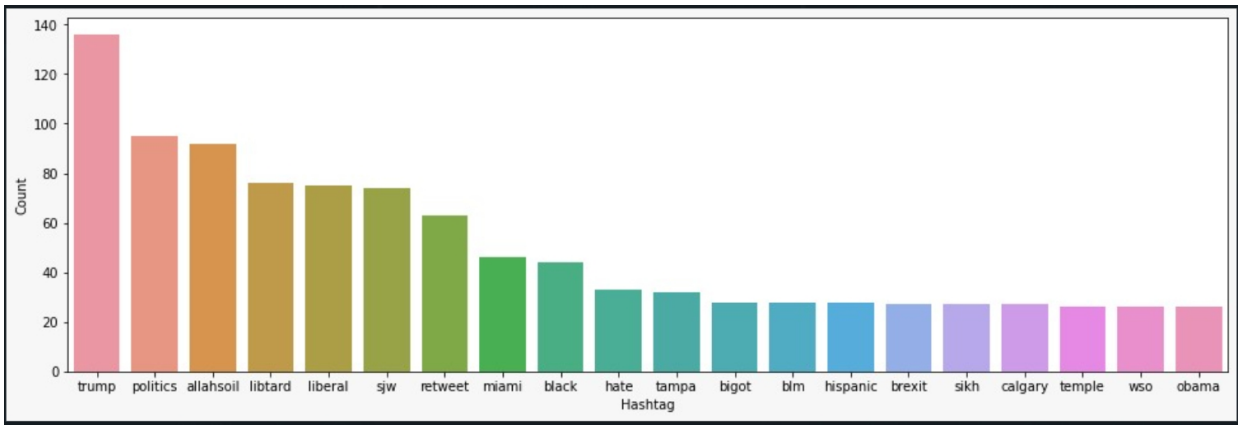
By using Word Cloud the representation of the data of tweets is presented of the form.

Python has a Word Cloud library that allows us to apply a mask using an image that we upload from our hard drive, select the background, the word colormap, the maximum words, font size, among other characteristics of the graph



.

The dataset description represented via graphical way for the positive and negative word frequency brought a great idea about the experiment we were goanna make.



The data description via correlation between the frequency terms has been stated via the replot graph function in the python, on understanding the graph we come to know about the frequency for positive and negative tweet reviews.

PROPOSED ALGORITHM

Naïve Bayes

The naive bayes classifier algorithm is an algorithm used to find the highest probability value for classifying test data in the most appropriate category [2]. In this research that becomes test data is document tweets. There are two stages in the classification of documents. The first stage is the training of documents that are known to the category. While the second stage is the process of classification of documents that have not been known category. Bayes's Theorem has the following general formula:

$P(H|X) = P(X|H)X\ P(H)\ P(X)$

Where:

1. P(H|X) is the probable final probability (posteriory probability) of hypothesis H occurs when given evidence E occurs.
2. P(X|H) is the probability that a proof E occurs will affect the hypothesis H.
3. P(H) is the prior probability hypothesis H occurs regardless of any evidence.
4. P(X) is the prior probability evidence of E regardless of the hypothesis or other evidence. To use Bayes's theory, the two variables used are aspects / features as hypotheses (H) and sentiments as evidence (E). The other three variables will be used as metadata of the sentiment
5. Because in a sentence consisting of many words, which is very difficult in practice to determine which one might be called an aspect / feature, it is assumed that each word is an aspect / feature. Then the application of Bayes theory:

$(K|F) = \mathrm{P(F|K)}X\ P(K)\ \mathrm{P(F)}$

Where : 1. F is a feature or a word. 2. K is a category or sentiment value. Because the features or words that support one category can be many, eg there are features F1, F2, F3, the Bayes theory can be developed into

$(K|F1, F2, F3) = \mathrm{P}(F1, F2, F3\ |\mathrm{K})X\ P(K)\ \mathrm{P}(F1, F2, F3)$

Since Bayes theory requires that the evidence (in this case is a feature or word) that exists is independent of each other, then the form of the above formula can be changed to:

$P(K|F1, F2, F3) = \mathrm{P}(F1\ |\mathrm{K})X\mathrm{P}(F2\ |\mathrm{K})X\mathrm{P}(F3\ |\mathrm{K})X\ P(K)\ \mathrm{P}(F1\ )X\mathrm{P}(F2\ )\mathrm{X}\ \mathrm{P}(F3\ )$

If described in general can be formulated as follows:

$(K|F) = \prod \mathrm{P}\ q\ i{=}0\ (F1\ |\mathrm{K})\ \mathrm{P(F)}$

Because the value of P(F) is always fixed for a category, the calculation of the P(F) value can be done once, so that only the $\prod \mathrm{P}\ q\ i{=}0\ (F1\ |\mathrm{K})$ is calculated only.

PATTERN EVALUATION

Accuracy Calculation

In this study, there are three categories as possible from classification result, that is positive category, negative, and neutral. Three lines and three columns of Confusion Matrix are then created:

		Actual Value		
		Positive Category	Negative Category	Neutral Category
Predicted Value	Positive Category	False Negative	False Positif	False Positive
	Negative Category	False Negative	True Negative	False Negative
	Neutral Category	False Neutral	False Neutral	True Neutral

True Positive is the number of positive records that are classified as positive. False Positive is the number of positive records classified as negative and neutral. False Negative is the number of negative records classified as positive and neutral. True Negative is the number of negative records that are classified as negative. True Neutral is a neutral record number that is classified as neutral. False Neutral is the number of neutral records classified in negative and positive.

		Actual Value			
		Positive Category	Negative Category	Neutral Category	Total
Predicted Value	Positive Category	780	5	15	800
	Negative Category	5	90	5	100
	Neutral Category	7	3	90	100
	Total	792	98	110	1000

$$Precision = \frac{\text{True Positive}}{\text{True positive} + \text{False Positive}} \; x\; 100\%$$

$$Precision = \frac{780}{800} \; x\; 100\% = 97\%$$

So the level of accuracy between the information requested by the user and the answer given by the system (precision) is 97%.

$$Recall = \frac{\text{number of positive predictive data that correct}}{\text{number of positive actual data}}$$

$$= \frac{\text{True positive}}{\text{True Positive} + \text{False Negative} + \text{False Neutral}} \; x\; 100\%$$

$$Recall = \frac{780}{792} \; x\; 100\% = 98\%$$

So the system success rate in rediscovering an information (recall) is 98%.

$$Accuracy = \frac{\text{number of correct prediction data}}{\text{total number of data}}$$

$$Accuracy = \frac{\text{True Positive} + \text{True Negative} + \text{True Neutral}}{\text{Total}} \; x\; 100\%$$

$$Accuracy = \frac{960}{1000} \; x\; 100\% = 96\%$$

$$Recall = \frac{\text{jumlah data prediksi yang benar}}{\text{jumlah data positive yang sebenarnya}}$$

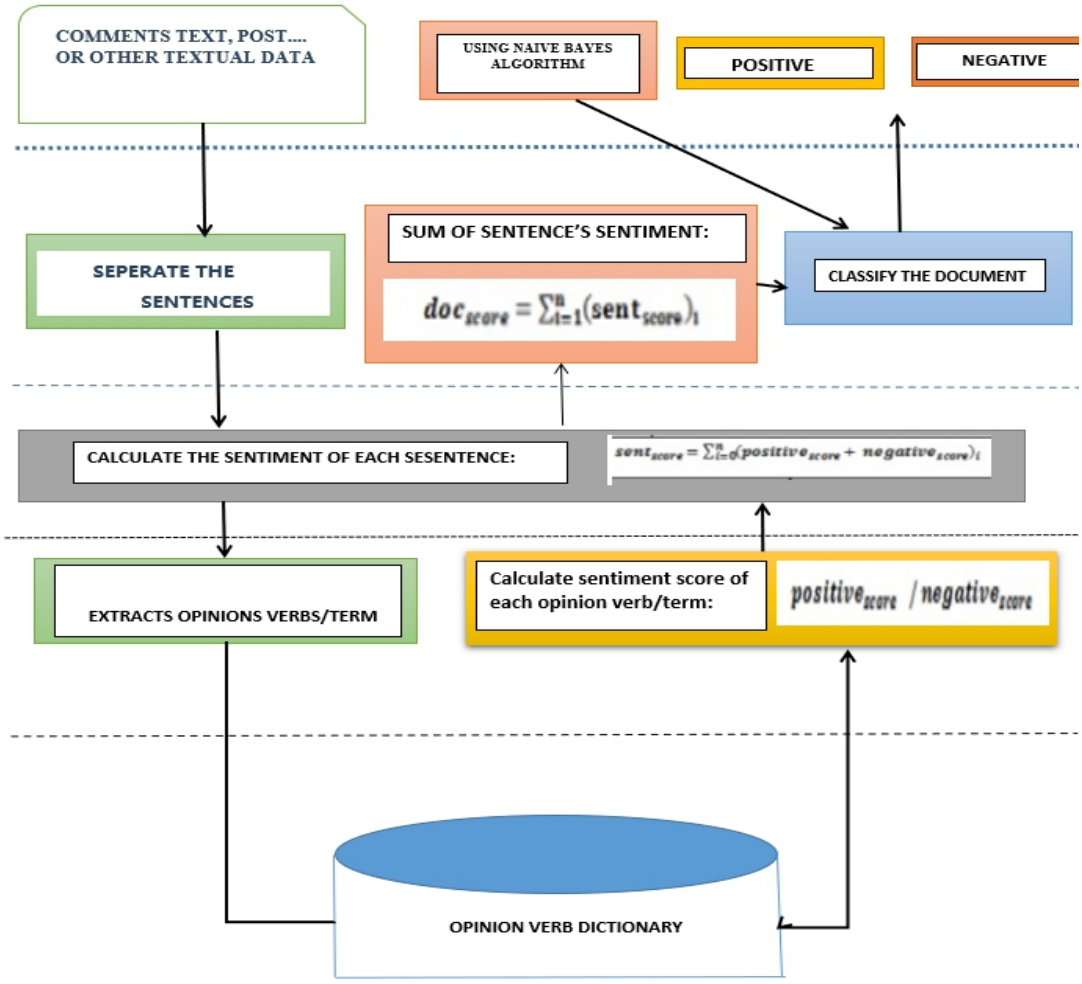
$$= \frac{\text{True positive}}{\text{True Positive} + \text{False Negative} + \text{False Neutral}} \; x\; 100\%$$

$$Recall = \frac{780}{792} \; x\; 100\% = 98\%$$

Data visualization tools help explain sentiment analysis results in a simple and effective way.

.

ARCHITECTURE Or FLOW CHARTS

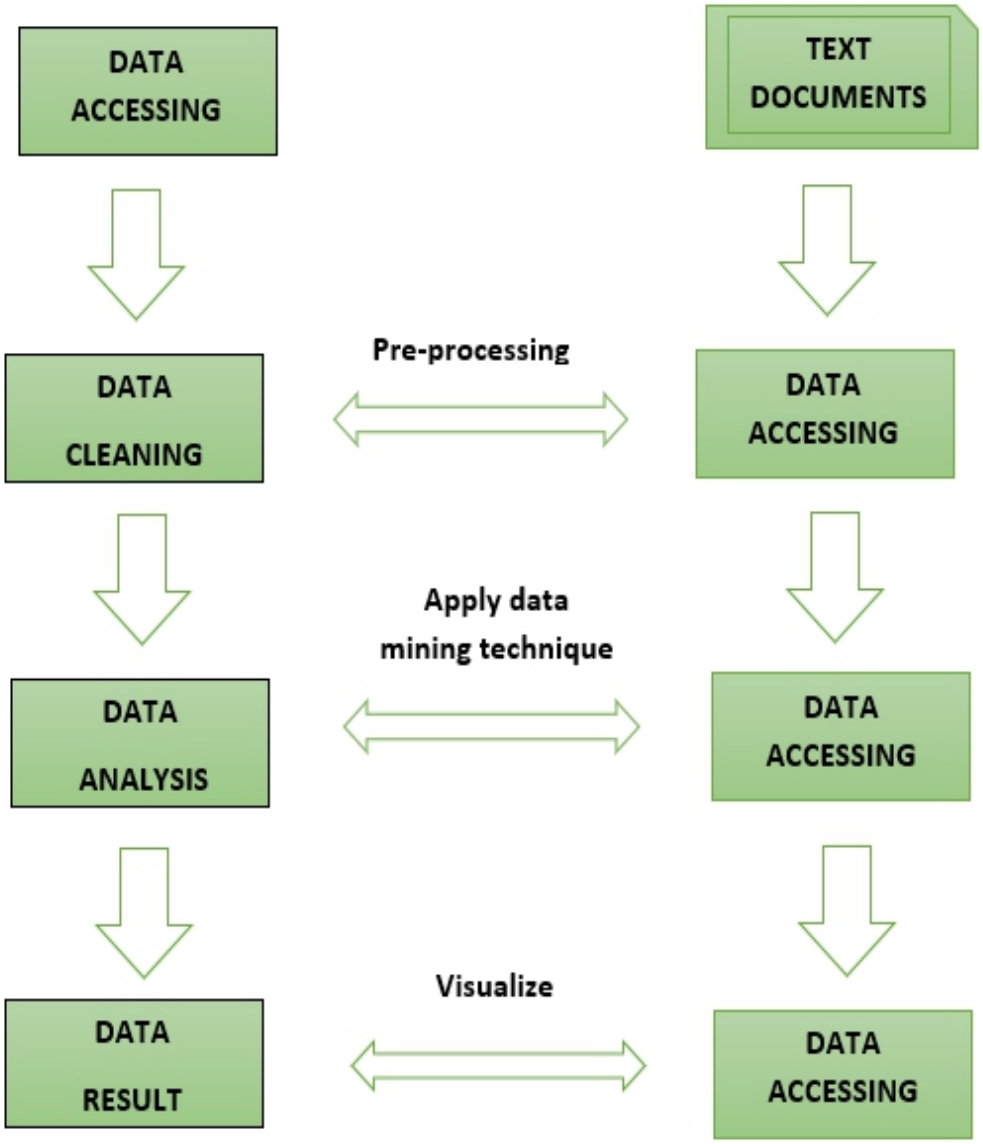


The genuine way for a data scientist or analyst to perform the process for text mining applications include

1. Identifying and retrieving relevant sets of text data for analysing purpose
2. Running algorithms to categorize and organize the data sets
3. Using analytical models to the ID concepts, patterns and other attributes

4. Applying the findings such as sentimental analysis and its application
5. Preparing data visualizations and dashboards to display the results.

A FLOW CHART STATING THE TEXT MINING PROCESS THAT ARE PERFORMED BY THE DATA ANALYTICS.



- **Linear Regression**

Linear regression is a statistical algorithm used to predict a *Y* value, given *X* features. Using machine learning and opinion mining, the data sets are examined to show a relationship. The relationships are then placed along the *X/Y* axis, with a straight line running through them to predict further relationships.

Linear regression calculates how the *X* input (words and phrases) relates to the *Y* output (polarity). This determines where words and phrases fall on a scale of polarity from "really positive" to "really negative" and everywhere in between.

- **Support Vector Machines**

Support vector machine is another supervised machine learning model, similar to linear regression but more advanced. SVM uses algorithms to train and classify text within our sentiment polarity model, taking it a step beyond X/Y prediction. For a simple visual explanation, we'll use two tags: red and blue, with two data features: X and Y. We'll train our classifier to output an X/Y coordinate as either red or blue. (figure-1)

The SVM then assigns a hyperplane that best separates the tags. In two dimensions this is simply a line (like in linear regression). Anything on one side of the line is red and anything on the other side is blue. For sentiment analysis this would be positive and negative. In order to maximize machine learning, the best hyperplane is the one with the largest distance between each tag.

GAP IDENTIFIED:

Naive Bayes assumes that all predictors (or features) are independent, rarely happening in real life. This limits the applicability of this algorithm in real-world use cases.

This algorithm faces the 'zero-frequency problem' where it assigns zero probability to a categorical variable whose category in the test data set wasn't available in the training dataset. It would be best if you used a smoothing technique to overcome this issue.

Its estimations can be wrong in some cases, so you shouldn't take its probability outputs very seriously.

PROPOSED METHOD

1. Main imitation of Naive Bayes is the assumption of independent predictors. Naive Bayes implicitly assumes that all the attributes are mutually independent. In real life, it is almost impossible that we get a set of predictors which are completely independent.

2. If categorical variable has a category in test data set, which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as Zero Frequency. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.

The general steps taken to complete this project are:

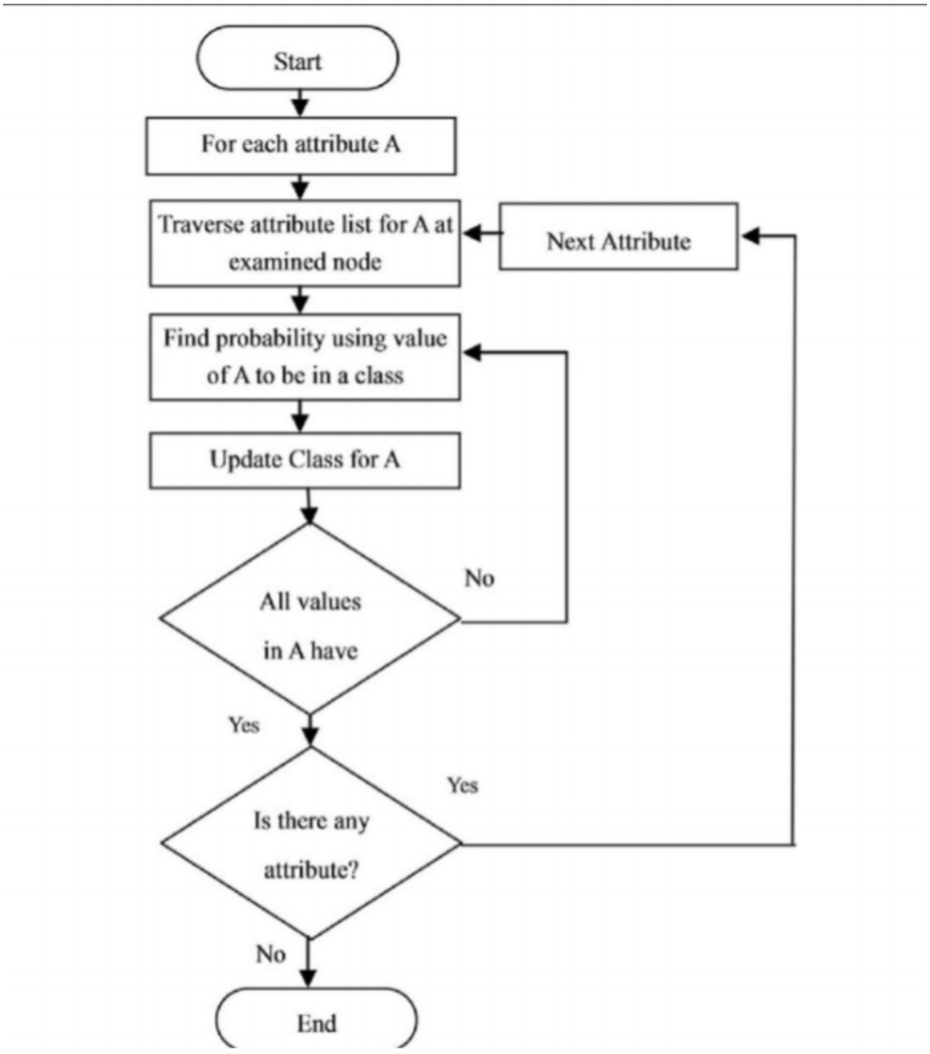
1. Get a twitter API and download Tweepy to access the twitter api through python
2. Download twitter tweet data depending on a key word search "happy" or "sad" • To get our data we will be using Twitter's API and access it using the Tweepy library. Basically, we will authenticate our Twitter API using our access token, access secret, consumer key and consumer secret. Afterwards, we are going to have a variable where we store the phrase/word we want to query.

3. Format my tweets so that no capitalization, punctuation, or non ascii characters are present, as well as splitting the tweet into an array holding each word in a separate holder • Some of the tweets have weird symbols. Our first goal is to get rid of them. • To clean the data, the first thing we do is to import any libraries we need and import the csv we are interested in as well getting rid of any “nan” values. • We then create a function that, given a text, removes any character or string of characters that are not readable in ASCII values. We then make all the texts lower case.

4. Create a bag of common words that appear in my tweets • Now that we have the list of words that appear in each individual file (happy, fun, sad, and : ( ), we want to combine them all into one dataframe and save this into a csv file called wordbag.csv.

5. Create a frequency table of words that have positive and negative hits

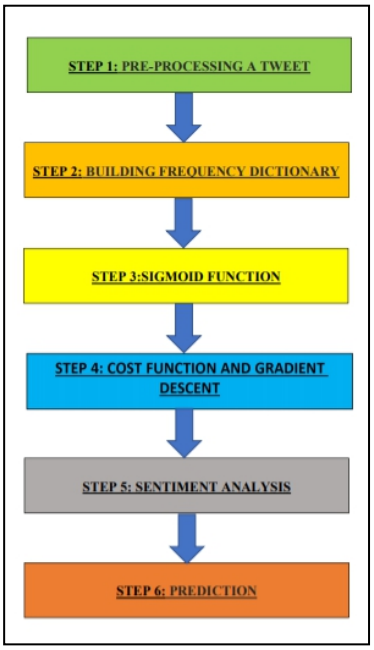
6. Test my frequency table by using test sentences FLOW CHART



DATA SETS DESCRIPTION AND SAMPLE DATA

Sentimental Analysis using Logistic Regression, Support Vector Machine and Naive Bayes

STEP 1: PRE-PROCESSING A TWEET



When pre-processing, you have to perform the following:

→Eliminate handles and URLs

→Tokenize the string into words.

→Remove stop words like “and, is, a, on, etc.”

→Stemming- or convert every word to its stem. Like a dancer, dancing, danced, becomes ‘dance’. You can use porter stemmer to take care of this.

→Convert all your words to lower case

STEP 2: BUILDING FREQUENCY DICTIONARY

Now, we will create a function that will take tweets and their labels as input, go through every tweet, pre-process them, count the occurrence of every word in the data set and create a frequency dictionary.

The squeeze function is necessary or the list ends up with one element. The required functions for processing tweets are ready, now let's build our logistic regression model.

STEP 3: SIGMOID FUNCTION

Logistic regression makes use of the sigmoid function which outputs a probability between 0 and 1. The sigmoid function with some weight parameter θ and some input x^{(i)}x(i) is defined as follows: -

$h(x^{(i)}, \theta) = 1/(1 + e^{(-\theta^T*x^{(i)})})$.

STEP 4: COST FUNCTION AND GRADIENT DESCENT

The logistic regression cost function is defined as
$J(\theta)=(-1/m)* \quad \sum i=1 \text{ to } m \quad [y(i)\log(h(x(i),\theta)+(1-y(i))\log(1-h(x(i),\theta))]$
We aim to reduce cost by improving the theta using the following equation:
$\theta j:=\theta j-\alpha*\partial J(\theta)/\theta j$
On testing the model using the test data set we get an accuracy of 99.5%

STEP 5: SENTIMENT ANALYSIS USING NAIVE BAYES

Naive Bayes algorithm is based on the Bayes rule, which can be represented as follows:

$P(X \mid Y) = P(Y)\ P\ (Y \mid X)\ P(X)$

STEP 6: PREDICTING USING NAIVE BAYES

In order to predict the sentiment of a tweet we simply have to sum up the loglikelihood of the words in the tweet along with the log prior. If the value is positive then the tweet shows positive sentiment but if the value is negative then the tweet shows negative sentiment.

EXPERIMENTS RESULTS

Sentimental Analysis can be implemented using various sources like Machine learning and upcoming algorithms like Naive Bayes, linear regression, etc. Here firstly we need to check out the essential packages or files inbuilt in language we using for coding. Sentimental analysis for movie review using machine learning algorithms and python language is described below.

```
pip install nltk

from google.colab import drive
drive.mount('/content/drive')

cd drive/MyDrive/

pip install seaborn

import re #for regular expressions
import nltk #for text manipulation
import string
import warnings
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Firstly, we are importing the essential packages from the python editor. Packages like pandas, metrices, count vectors, etc. are implemented at the very first step for coding.
In the very first part we use CountVectorizer for conversion of the texts into tokens

The very next step after this is transforming texts into counts of features for every single message using training data.

The result we want is the score of positive and negative reviews or tweets. For this we need to firstly look over the accuracy and confusion matrix (A confusion matrix is a technique for summarizing the performance of a classification algorithm. Classification accuracy alone can be misleading if we have an unequal number of observations in each class) Thus we can achieve this via various algorithms. Each of them code is stated below.

```
[ ]  print("X_train_shape : ",X_train.shape)
     print("X_test_shape : ",X_test.shape)
     print("y_train_shape : ",y_train.shape)
     print("y_test_shape : ",y_test.shape)

     X_train_shape :  (39327, 1000)
     X_test_shape :  (9832, 1000)
     y_train_shape :  (39327,)
     y_test_shape :  (9832,)
```

The above code deals with the bringing out the confusion matrix and accuracy score via Naive Bayes.

The function we use here is Multinomial Naïve Bayes which in python has its syntax.

```
from sklearn.naive_bayes import MultinomialNB  # Naive Bayes Classifier

model_naive = MultinomialNB().fit(X_train, y_train)
predicted_naive = model_naive.predict(X_test)
```

```
[ ]  from sklearn.metrics import confusion_matrix

     plt.figure(dpi=600)
     mat = confusion_matrix(y_test, predicted_naive)
     sns.heatmap(mat.T, annot=True, fmt='d', cbar=False)

     plt.title('Confusion Matrix for Naive Bayes')
     plt.xlabel('true label')
     plt.ylabel('predicted label')
     plt.savefig("confusion_matrix.png")
     plt.show()
```

The above code deals with the bringing out the confusion matrix and accuracy score via Logistic Regression Model.

The function we use here is Regression which in python has its syntax.

```
#Bag-of-words

#Each row in matrix M contains the frequency of tokens(words) in the document D(i)

bow_vectorizer = CountVectorizer(max_df=0.90 ,min_df=2 , max_features=1000,stop_words='english')
bow = bow_vectorizer.fit_transform(combine['tidy_tweet']) # tokenize and build vocabulary
bow.shape
```

```
(49159, 1000)
```

```
#TF-IDF

#TF = (number of times term appear in a document)/(Number of terms in dcoument)
#IDF = log(N/n)-N is nummber of documents and n is number of documents a term has appeared in.

#TF-IDF = TF * IDF

#tfidf_vectorizer= TfidfVectorizer(max_df=0.90, min_df=2,max_features =1000,stop_words='english')
#tfidf = tfidf_vectorizer.fit_transform(combine['tidy_tweet'])
#tfidf.shape
```

```
combine=combine.fillna(0) #replace all null values by 0
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(bow, combine['label'],
                                                    test_size=0.2, random_state=69)
```

The input is taken as simple way it can be, because we have to store the large amount of raw data from our side before itself. Thus, the input and the data stored with us, it has similarity then it could make the process faster. Thus, the test variable is taken as array of words which is then checked one by one using the analysis code.

```
train[train['label'] == 0].head(10)
```

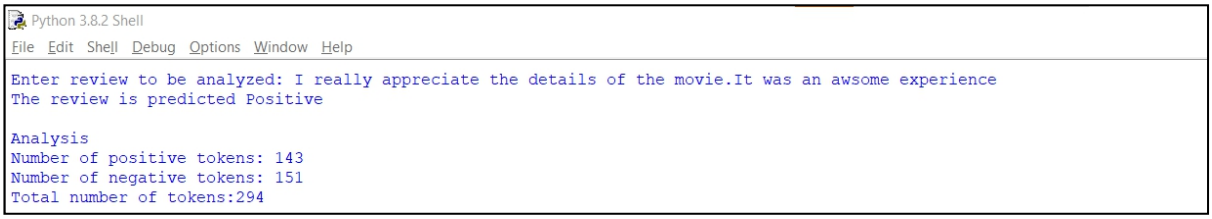
	id	label	tweet
0	1	0	@user when a father is dysfunctional and is so selfish he drags his kids into his dysfunction. #run
1	2	0	@user @user thanks for #lyft credit i can't use cause they don't offer wheelchair vans in pdx. #disapointed #getthanked
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in urð□□±!!! ð□□□ð□□□ð□□□ð□□□ð□□¦ð□□¦ð□□¦
4	5	0	factsguide: society now #motivation
5	6	0	[2/2] huge fan fare and big talking before they leave. chaos and pay disputes when they get there. #allshowandnogo
6	7	0	@user camping tomorrow @user @user @user @user @user @user @user dannyâ□¦
7	8	0	the next school year is the year for exams.ð□□¯ can't think about that ð□□ #school #exams #hate #imagine #actorslife #revolutionschool #girl
8	9	0	we won!!! love the land!!! #allin #cavs #champions #cleveland #clevelandcavaliers â□¦
9	10	0	@user @user welcome here ! i'm it's so #gr8 !

```
[ ] train[train['label'] == 0].tail(10)
```

	id	label	tweet
31951	31952	0	i've just posted a new blog: #secondlife #lonely #neko
31952	31953	0	@user you went too far with @user
31953	31954	0	good morning #instagram #shower #water #berlin #berlincitygirl #girl #newyork #zÃ¼rich #genf #bern
31954	31955	0	#holiday bull up: you will dominate your bull and you will direct it whatever you want it to do. when you
31955	31956	0	less than 2 weeks ð□□□ð□□□ð□□¼ð□□'ð□□□ð□□µ @user #ibiza#bringiton#mallorca#holidays#summer
31956	31957	0	off fishing tomorrow @user carnt wait first time in 2 years
31957	31958	0	ate @user isz that youuu?ð□□□ð□□□ð□□□ð□□□ð□□□ð□□□ð□□□ð□□□ð□□□â□¤¡
31958	31959	0	to see nina turner on the airwaves trying to wrap herself in the mantle of a genuine hero like shirley chisolm. #shame #imwithher
31959	31960	0	listening to sad songs on a monday morning otw to work is sad
31961	31962	0	thank you @user for you follow

The task we perform for our better results is that we collect large amount of raw data for our experimental analysis. Here we collect the data in very common way like storing it in notepad and saved the file with tsv extension in the system where the access to code is easy. Our data has both the negative and positive messages such that the algorithm techniques we using gives out good accuracy.

To the custom input review the total number of positive and negative analysis tokens and a predicted phrase comes as output with the accuracy and confusion matrices of each method separately.



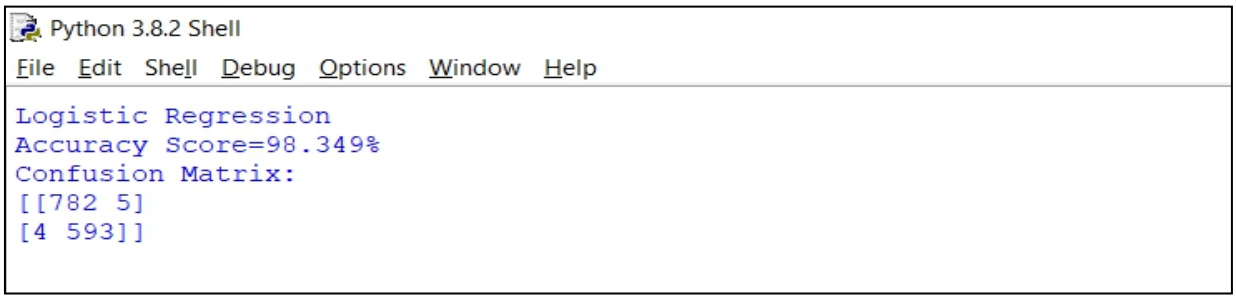
The individual output results to each of the Naive Bayes, Logistic regression and Support Vector Machine methods to give out accuracy and confusion matrices. The values vary from

one to other as the procedure and methodology varies. The output for the individual algorithm is screenshotted and well displayed.

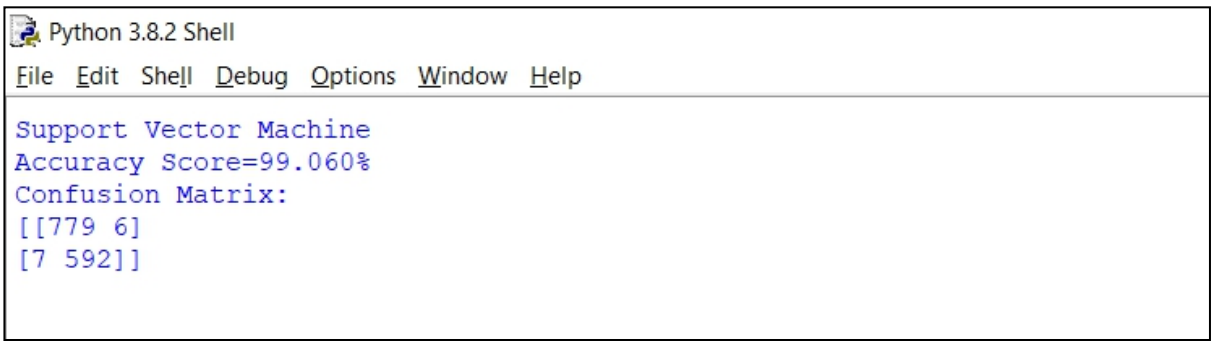
NAVIE BAYES:

```
[ ]  from sklearn.metrics import accuracy_score

     score_naive = accuracy_score(predicted_naive, y_test)
     print("Accuracy with Naive-bayes: ",score_naive)

     Accuracy with Naive-bayes:  0.9456875508543532

[ ]
```

LOGISTIC REGRESSION:



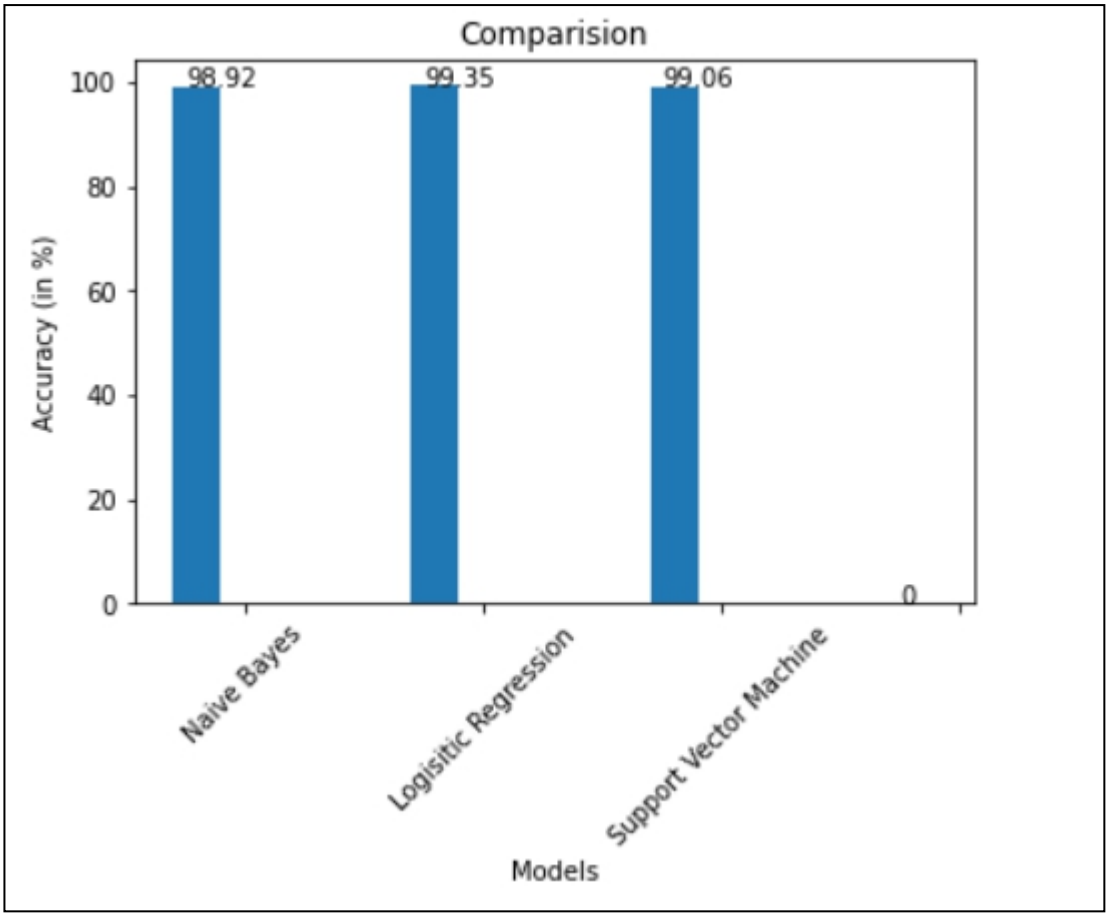
SUPPORT VECTOR MACHINE:



COMPARATIVE STUDY / RESULTS AND DISCUSSION

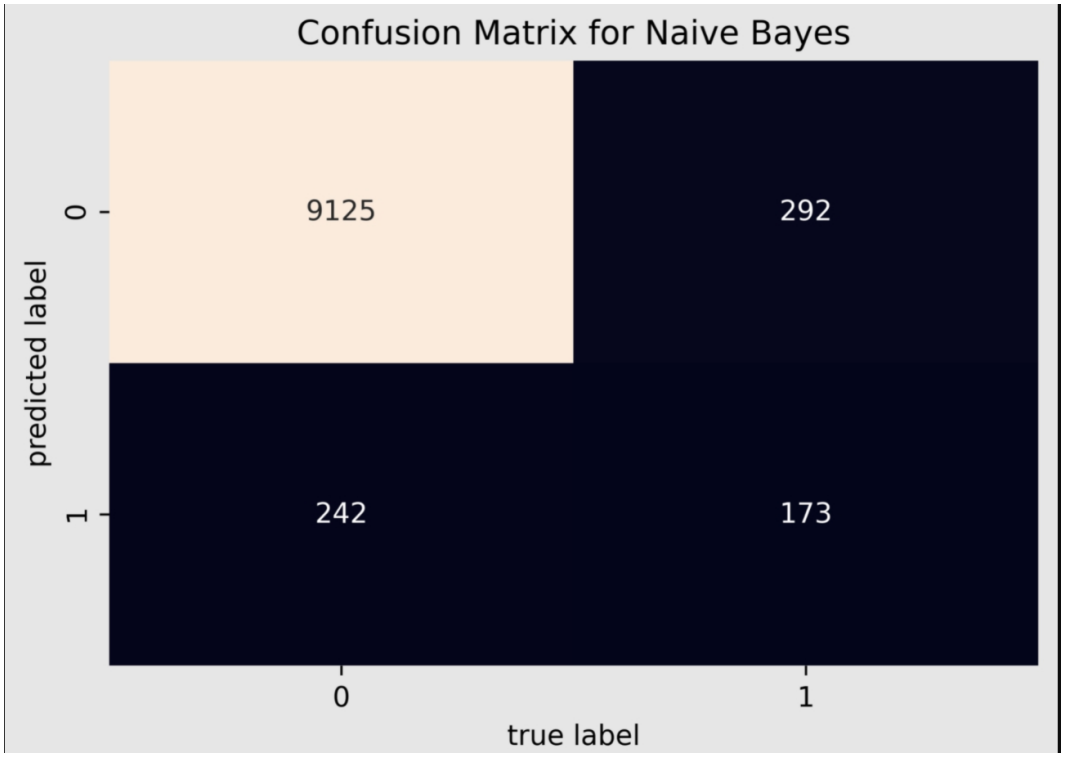
Here we compare the whole project in two terms.
First one can be the comparing the accuracy percentage of the models we have chosen. A table is drawn manually to get clarity about the graph that is drawn via code.

MODEL	ACCUARCY IN %
Naïve Bayes	94.5
Logistic Regression	98.349
Support Vector Machine	99.060



The comparison between different model and accuracy level gave out the conclusion that the accuracy level for **Logistic regression** is much better than the other two. So, for training in future we preferred for using Support Vector Machine.

The output for the code gives out accuracy percentage and confusion matrices. The diagrammatic representation of confusion matrices for each model is shown below.



We choose various different phrases as input for the testing and bring out the output using Support Vector Machine code and get the tabular format as follows with the graphical comparison.

Let the phrases be as

Phrase 1: It was a amazing movie. I liked it a lot

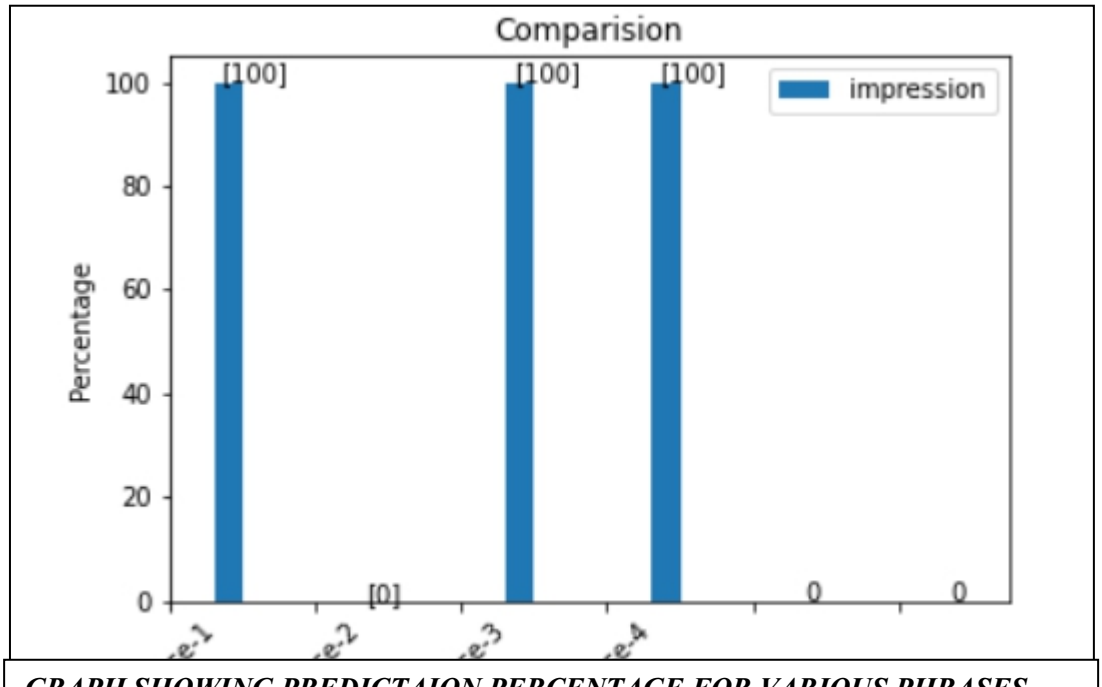
Phrase 2: It was a boring one.

Phrase 3: It was fabulous. Must watch

Phrase 4: It was average one.

REVIEW PHRASES	**PREDICTION (100-GOOD /0-BAD)**
PHRASE 1	100
PHRASE 2	0
PHRASE 3	100
PHRASE 4	0

TABLE SHOWING PREDICTAION PERCENTAGE FOR VARIOUS PHRASES



GRAPH SHOWING PREDICTAION PERCENTAGE FOR VARIOUS PHRASES

Tested on 49000 tweets the score is 0.94

CONCLUSION AND FUTURE WORK

Here first we concluded that Logistic Regression give better accuracy compared to Support Vector Machine and Naïve Bayes and also, it's very less time taking compared to other two. So, for actual prediction purpose we used Logistic regression algorithm for training our model. We even looked after the confusion matrices and analysed the actual and predicted values. After that we predict the impression of different phrase reviews and checked our output with the online analyser websites like imdb so finally concluded that our model working properly. In future we can use also some advanced techniques like RNN (Recurrent Neural Networks).

The Naive Bayes sentiment analysis algorithm has also been proved to be suitable for MapReduce programming model. Based on Hadoop framework, we implemented the parallel Naive Bayes based sentiment analysis algorithm in MapReduce model, and through a series of experiments and analysis, the MapReduce implementation proves to have scalability, and we also provide a prediction model to predict and analyze the performance of parallel Naive Bayes MapReduce program. This prediction model can also be used for other MapReduce programs.

REFERENCES

PAPER 1:

Sentiment analysis using product review data

Xing Fang* and Justin Zhan

PAPER 2:

Sentiment Analysis in E-Commerce: A Review on The Techniques and Algorithms

Muhammad Marong School of Computing and Technology Asia Pacific University of Technology & Innovation Malaysia

Nowshath K Batcha School of Computing and Technology Asia Pacific University of Technology & Innovation Malaysia nowshath.

Raheem Mafas School of Computing and Technology Asia Pacific University of Technology & Innovation Malaysia

PAPER 3:

Sentiment analysis and opinion mining applied to scientific paper reviews

Brian Keith Norambuena , Exequiel Fuentes Lettura and Claudio Meneses Villegas Department of Computing and Systems Engineering, Universidad Católica del Norte, Coquimbo, Chile

PAPER 4:

Sentiment Analysis of Persian-English Code-mixed Texts

1st Nazanin Sabri electrical and computer engineering University of Tehran Tehran, Iran
2nd Ali Edalat electrical and computer engineering University of Tehran Tehran, Iran
3 rd Behnam Bahrak electrical and computer engineering University of Tehran Tehran, Iran

PAPER 5:

A STUDY OF SOCIAL SENTIMENT ANALYSIS IN THE TIMES OF COVID -19 USING TWITTER

Princy Sharma1, Prof. Vibhakar Mansotra2

PAPER 6:

Twitter Sentiment Analysis: A Political View

Joylin Priya Pinto1, Vijaya Murari T.2, Soumya Kelur3

PAPER 7:

Topic Sentiment Analysis in Online Learning Community from College Students

Kai Wang, Yu Zhang†

PAPER 8:
Sentiment Analysis on Large Scale Amazon Product Reviews

Sayyed Johar, Samara Mubeen

PAPER 9:

Improving Sentiment Analysis with Biofeedback Data

Daniel Schlor , Albin Zehe , Konstantin Kobs , Blerta Veseli, Franziska Westermeier, Larissa Brubach, Daniel Roth, Marc Erich Latoschik, Andreas Hotho

PAPER 10:

Measuring News Sentiment

Adam Hale Shapiro, Moritz Sudhof , and Daniel Wilson

PAPER 11:

Predicting Stock Market Price Movement Using Sentiment Analysis: Evidence From Ghana

Isaac Kofi Nti1, Adebayo Felix Adekoya , Benjamin Asubam Weyori

PAPER 12:

Sentiment Analysis of Open Source Software Community Mailing List: A Preliminary Analysis

Jumoke Abass Alesinloye ,Subathra Srinivasan, Eoin Groarke, Greg Curran, Jaganath Babu, Denis Dennehy

PAPER 13:

Exploring Online Drug Reviews using Text Analytics, Sentiment Analysis and Data Mining Models

Thu Dinh, Goutam Chakraborty, Miriam McGaugh

PAPER 14:

Managing Marketing Decision-Making with Sentiment Analysis: An Evaluation of the Main Product Features Using Text Data Mining

Erick Kauffmann , Jesús Peral , David Gil , Antonio Ferrández , Ricardo Sellers and Higinio Mora

PAPER 15:

A Review on Sentimental Analysis on Facebook Comments by using Data Mining Technique

Rupinder Kaur,Dr. Harmandeep Singh, Dr. Gaurav Gupta

Appendix: (Full Code)

```
pip install nltk
from google.colab import drive
drive.mount('/content/drive')
pip install seaborn
import re #for regular expressions
import nltk #for text manipulation
import string
import warnings
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
pd.set_option("display.max_colwidth",200)
warnings.filterwarnings("ignore",category=DeprecationWarning)
%matplotlib inline
train = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')
train[train['label'] == 0].head(10)
train.shape
test.shape
train['label'].value_counts()
length_train_dataset = train['tweet'].str.len()
length_test_dataset = test['tweet'].str.len()
plt.hist(length_train_dataset, bins=20,label="Train tweets")
```

```
plt.hist(length_test_dataset, bins=20,label="Test tweets")
plt.legend()
plt.show()
combine=train.append(test,ignore_index=True) #train and test dataset are combined
combine.shape
def remove_pattern(input_text,pattern):
    r= re.findall(pattern, input_text)
    for i in r:
        input_text = re.sub(i, '', input_text)
    return input_text
combine['tidy_tweet'] = np.vectorize(remove_pattern)(combine['tweet'],"@[\w]*")
combine.head()
combine['tidy_tweet'] = combine['tidy_tweet'].str.replace("[^a-zA-Z#]"," ")
combine.head(10
combine.head()
tokenized_tweet = combine['tidy_tweet'].apply(lambda x:x.split()) #it will split all words by wh
itespace
tokenized_tweet.head()
from nltk.stem.porter import *
stemmer = PorterStemmer()
tokenized_tweet = tokenized_tweet.apply(lambda x: [stemmer.stem(i) for i in x])

for i in range(len(tokenized_tweet)):
    tokenized_tweet[i] = ' '.join(tokenized_tweet[i]) #concat all words into one sentence
combine['tidy_tweet'] = tokenized_tweet
all_words = ' '.join([text for text in combine['tidy_tweet']])
from wordcloud import WordCloud
wordcloud = WordCloud(width=800,height=500,random_state=21,max_font_size=110).generat
e(all_words)
plt.figure(figsize=(10,7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
normal_words= ' '.join([text for text in combine['tidy_tweet'][combine['label']==0]])
wordcloud= WordCloud(width=800,height=500,random_state=21,max_font_size=110).generat
e(normal_words)
plt.figure(figsize=(10,7))
plt.imshow(wordcloud,interpolation='bilinear')
plt.axis('off')
plt.show()
#racist tweet

negative_words= ' '.join([text for text in combine['tidy_tweet'][combine['label']==1]])
wordcloud= WordCloud(width=800,height=500,random_state=21,max_font_size=110).generat
e(negative_words)
plt.figure(figsize=(10,7))
plt.imshow(wordcloud,interpolation='bilinear')
plt.axis('off')
plt.show()
#collect hashtags

def hashtag_extract(x):
    hashtags=[]
    for i in x: #loop over words contain in tweet
        ht = re.findall(r"#(\w+)",i)
```

```
        hashtags.append(ht)
    return hashtags
#extracting hashtags from non racist tweets
ht_regular = hashtag_extract(combine['tidy_tweet'][combine['label']==0])
#extracting hashtags from racist tweets
ht_negative=hashtag_extract(combine['tidy_tweet'][combine['label']==1])
ht_regular = sum(ht_regular,[])
ht_negative = sum(ht_negative,[])

nonracist_tweets = nltk.FreqDist(ht_regular)
df1 = pd.DataFrame({'Hashtag': list(nonracist_tweets.keys()),'Count':list(nonracist_tweets.value
s())})

#selecting top 20 most frequent hashtags
df1 = df1.nlargest(columns="Count",n=20)
plt.figure(figsize=(16,5))
ax = sns.barplot(data=df1, x="Hashtag", y="Count")
ax.set(ylabel = "Count")
plt.show()
racist_tweets = nltk.FreqDist(ht_negative)
df2 = pd.DataFrame({'Hashtag': list(racist_tweets.keys()),'Count': list(racist_tweets.values())})
#count number of occurrence of particular word

#selecting top 20 frequent  hashtags

df2 = df2.nlargest(columns = "Count",n=20)
plt.figure(figsize=(16,5))
ax = sns.barplot(data=df2, x="Hashtag",y="Count")
plt.show()
pip install gensim
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
import gensim
#Bag-of-words

#Each row in matrix M contains the frequency of tokens(words) in the document D(i)

bow_vectorizer = CountVectorizer(max_df=0.90 ,min_df=2 , max_features=1000,stop_words='
english')
bow = bow_vectorizer.fit_transform(combine['tidy_tweet']) # tokenize and build vocabulary
bow.shape
#TF-IDF

#TF = (number of times term appear in a document)/(Number of terms in dcoument)
#IDF = log(N/n)-
N is nummber of documents and n is number of documents a term has appeared in.

#TF-IDF = TF * IDF

#tfidf_vectorizer= TfidfVectorizer(max_df=0.90, min_df=2,max_features =1000,stop_words='e
nglish')
#tfidf = tfidf_vectorizer.fit_transform(combine['tidy_tweet'])
#tfidf.shape
combine=combine.fillna(0) #replace all null values by 0
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(bow, combine['label'],
```

```
                                                test_size=0.2, random_state=69)
print("X_train_shape : ",X_train.shape)
print("X_test_shape : ",X_test.shape)
print("y_train_shape : ",y_train.shape)
print("y_test_shape : ",y_test.shape)
from sklearn.naive_bayes import MultinomialNB  # Naive Bayes Classifier

model_naive = MultinomialNB().fit(X_train, y_train)
predicted_naive = model_naive.predict(X_test)
from sklearn.metrics import confusion_matrix

plt.figure(dpi=600)
mat = confusion_matrix(y_test, predicted_naive)
sns.heatmap(mat.T, annot=True, fmt='d', cbar=False)

plt.title('Confusion Matrix for Naive Bayes')
plt.xlabel('true label')
plt.ylabel('predicted label')
plt.savefig("confusion_matrix.png")
plt.show()
from sklearn.metrics import accuracy_score

score_naive = accuracy_score(predicted_naive, y_test)
print("Accuracy with Naive-bayes: ",score_naive)
Accuracy with Naive-bayes: 0.9456875508543532
```