TWACH: THE BIAS PLATFORM

PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF TECHNOLOGY IN THE FIELD OF COMPUTER SCIENCE AND ENGINEERING

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY

SUBMITTED BY

ANJAN KUMAR DEY (123180703012)
ASHISH ABHISHEK MINZ (123180703024)
AYAN CHATTERJEE (123180703027)
DEBAJYOTI GHOSH (123180703028)
DEBRAJ KARMAKAR (123180703029)

Under the Supervision
of
Dr. Dharmpal Singh
Associate Professor and Head of the Department, Computer Science Engineering, JIS College of Engineering

Conducted
By

Department of Computer Science and Engineering
JIS College of Engineering
Kalyani-741235, Nadia, West Bengal, India

Certificate of Authenticated work

This is to certify that the project report entitled **"TWACH: THE BIAS PLATFORM"** submitted to the DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, JIS COLLEGE OF ENGINEERING in partial fulfillment of the requirement for the award of the degree of BACHELORS OF TECHNOLOGY. The matter embodied in this project is authentic and is genuine work done by the student and has not been submitted whether to this College or to any other Institute for the fulfilment of the requirement of any course of study.

…….. …….

Signature of the Student: Signature of the Professor

Date: ……... Date: ……...

DR. DHARMPAL SINGH
(HOD CSE DEPT.)

JIS College of Engineering
Block 'A', Phase-III, Kalyani, Nadia, 741235
Phone: +91 33 2582 2137, Telefax: +91 33 2582 2138
Website: www.jiscollege.ac.in, Email: info@jiscollege.ac.in

CERTIFICATE

This is to certify that **DEBRAJ KARMAKAR (123180703029).** has completed his/her project entitled **TWATCH: THE BIAS PLATFORM,** under the guidance of **DR. DHARMPAL SINGH** in partial fulfillment of the requirements for the award of the **Bachelor of Technology in Computer Science and Engineering** from JIS college of Engineering (An Autonomous Institute) is an authentic record of their own work carried out during the academic year 2021-22 and to the best of our knowledge, this work has not been submitted elsewhere as part of the process of obtaining a degree, diploma, fellowship or any other similar title.

| --- | --- | --- |
| **Signature of the Supervisor** | **Signature of the HOD** | **Signature of the Principal** |

Place:

Date:

Corporate office: 7, Sarat Bose Road, Kolkata-700 020, Phone: +91 33 2289 3944/5323, Telefax: +91 33 2289 3945

ACKNOWLEDGEMENT

The analysis of the project work wishes to express our gratitude to Dr. Dharmpal Singh for allowing the degree attitude and providing effective guidance in development of this project work. His conscription of the topic and all the helpful hints, he provided, contributed greatly to successful development of this work, without being pedagogic and overbearing influence.

We also express our sincere gratitude to Dr. Dharmpal Singh, Head of the Department of Computer Science and Engineering of JIS College of Engineering and all the respected faculty members of Department of CSE for giving the scope of successfully carrying out the project work.

Finally, we take this opportunity to thank to Prof. **(Dr.) Partha Sarkar**, Principal of JIS College of Engineering for giving us the scope of carrying out the project work.

Date:

.....

ANJAN KUMAR DEY
B. TECH in Computer Science and Engineering
4thYEAR/8th SEMESTER
Univ Roll--123180703012

.....

ASHISH ABHISHEK MINZ
B. TECH in Computer Science and Engineering
4thYEAR/8th SEMESTER
Univ Roll--123180703024

.....

AYAN CHATTERJEE
B. TECH in Computer Science and Engineering
4thYEAR/8th SEMESTER
Univ Roll—123180703027

.....

DEBAJYOTI GHOSH
B. TECH in Computer Science and Engineering
4thYEAR/8th SEMESTER
Univ Roll--123180703028

.....

DEBRAJ KARMAKAR
B. TECH in Computer Science and Engineering
4thYEAR/8th SEMESTER
Univ Roll--123180703029

TABLE OF CONTENTS

ABSTRACT

This project aims to address the problem of sentiment analysis on twitter.Classifying tweets based on the sentiment expressed in them: positive, negative or neutral. We can harvest data and analyze the tweets either by providing a username or any hashtag. The user inputs the number of tweets to be extracted and date of extraction along with the hashtag or the username. If the input is not empty then using twitter API, we scrape the tweets from twitter in real time. TheNLP technique is then used for cleaning them. The tweets are analyzed and classified according to their polarity and subjectivity scores into positive, negative or neutral lists and finally visualized. Analyzing the public sentiment or opinion on something is important for many applications such as firms trying to find out the public sentiment of their products in the market, predicting political elections and socioeconomic phenomena like stock exchange. The aim of this project is to develop a functional classifier for accurate and automatic sentiment classification of recent tweets.

Introduction

Nowadays, the age of Internet has changed the way people express their views, opinions. It is now mainly done through blog posts, online forums, product review websites, social media, etc. Nowadays, millions of people are using social network sites like Facebook, Twitter, Google Plus, etc. to express their emotions, opinion and share views about their daily lives. Through the online communities, we get an interactive media where consumers inform and influence others through forums. Social media is generating a large volume of sentiment rich data in the form of tweets, status updates, blog posts, comments, reviews, etc. Moreover, social media provides an opportunity for businesses by giving a platform to connect with their customers for advertising. People mostly depend upon user generated content over online to a great extent for decision making. For e.g., if someone wants to buy a product or wants to use any service, then they firstly look up its reviews online, discuss about it on social media before taking a decision. The amount of content generated by users is too vast for a normal user to analyze. So, there is a need to automate this, various sentiment analysis techniques are widely used.

Data analysis is the process of applying organized and systematic statistical techniques to data to describe, recap, check and condense it. It is applying these techniques to modify data to suit our requirements. Data mining is a multistep process that involves collecting, organizing, cleaning of the data. It is needed because different sources like social media, enterprise data, public data, transactions etc. generate data of increasing volume every day, and it is importantto handle, analyze and organize such big data. Social media is something that has become a game changer in the 21st century, something that we live by. It haschanged how we and enterprises look at advertising, marketing, globalization or politics. More data has been generated in the past two years than ever before inthe history of mankind. Internet users have also increased from millions to billionswith the introduction of affordable internet services.

Literature Survey

Sentiment analysis has been handled as a Natural Language Processing task at many levels of granularity. From being handled as a document level classificationtask (Turney, 2002; Pang and Lee, 2004), handled at the sentence level (Hu and Liu, 2004; Kim and Hovy, 2004) and also at the phrase level (Wilson et al., 2005;Agarwal et al., 2009). Classifying the sentiment of tweets is most similar to sentence level sentiment analysis, Given the 140-character limit on tweets. However, the informal and specialized language of using shorthand/internet slangs used in tweets, as well as the very nature of the microblogging domain make Twitter sentiment analysis a very different task. Microblog data likeTwitter, on which users post real time reactions to and opinions about "everything", poses newer and different challenges. (Barbosa and Feng 2010) exploit existing Twitter sentiment sites for collecting training data. (Davidov, Tsur, and Rappoport 2010) also use hashtags for creating training data, but they limit their experiments to sentiment/non-sentiment classification, rather than 3- way polarity classification, as we do.

Some of the early results on sentiment analysis of Twitter data are by Go et al. (2009), Pak and Paroubek (2010) and (Bermingham and Smeaton, 2010). Go etal. (2009) used distant learning to acquire sentiment data. They build modelsusing Naive Bayes, MaxEnt and Support Vector Machines (SVM), and reportedthat SVM outperforms other classifiers. In terms of feature space, they try aUnigram, Bigram model in conjunction with parts-of-speech (POS) features.Reporting that the unigram model outperforms all other models and bigrams andPOS features do not help. (Das and Chen, 2001) reported that both Naive Bayes and SVM classifiers achieve a higher accuracy without applying the negationrules. (Joachims, 1998) reported that both these two classifiers have achievedmore than 0.9 accuracy using unigram features in traditional topic-basedclassification.

SVMs were used by Yung-Ming Li and Tsung-Ying Li (2013) as a sentiment polarity classifier. Unlike the binary classification problem, they argued that opinion subjectivity and expresser credibility should also be considered for the accuracy of the model. They proposed a framework that gives a compact numericsummarization of opinions on micro-blogs platforms.

Hanhoon Kang, Seong Joon Yoo and Dongil Han (2012) proposed an improved NB classifier to correct the tendency for the positive classification accuracy to appear up to approximately 10% higher than the negative classification accuracy. This decreases the average accuracy when the accuracies of the two classes are expressed as an average value. They reported that using this algorithm with restaurant reviews narrowed the gap between the positive accuracy and the negative accuracy compared to NB and SVM. The accuracy is improved in recalland precision compared to both NB and SVM.

After analyzing these articles, it is clear that the enhancements of Sentiment Classification algorithms are still an open field for research. Naïve Bayes andSupport Vector Machines are the most used ML algorithms for solving sentimentclassification problems. They are often considered a reference model to comparemany proposed algorithms to. SVM is known to be the model giving the bestresults but, in this project, we focus only on the probabilistic model that is NaiveBayes that has been widely used in this field.

The Naive Bayes model which is well known for its simplicity and efficiency fortext classification. We report accuracy results on data that does not suffer from any known biases. We perform cross validation experiments and check for the variance in performance of the classifier across folds.

Motivation

The Database which we have chosen for the proposed study is Twitter. It is a verypopular microblogging service. Compared to conventional internet articles and web blobs we feel twitter is a better approximation of public sentiments. The reason being, the amount of relevant data is much larger for twitter and the response on twitter is more prompt and more general (as the number of people who tweet daily is substantially more than those who write web articles or blogs).Sentiment analysis of the public opinion is very important in macro-scale socioeconomic phenomena like predicting the stock market rates of a particular firm. This can be done by analyzing the overall public sentiment towards that firmwith respect to time and using various economic tools for finding the correlation.

between the firm's stock market value and public sentiment. Firms can also review how well a product is doing in the market, which areas of the market is ithaving a favorable or negative response in. Since twitter allows us to download astream of geo-tagged tweets for particular locations. If firms can get this information and analyze it, they can get the reasons behind geographically differentiated response, and can thus market their product in a more optimized manner by looking for solutions like creating suitable market segments. Predicting the results of popular political elections and polls is also an emerging application to sentiment analysis. A study was conducted by Tumasjan et al. in Germany for predicting the outcome of federal elections, which concluded that twitter is a good reflection of offline sentiment.

Scope

This project hopes to be useful to companies, political people and the common person. It will be helpful to review the public opinion on their programs. Companies can get reviews about their new product or latest hardware or software. Also, the movie makers and goers can see the reviews on the currentlyrunning movie. By analyzing the tweets, the analyzer can get results on how positive, negative or neutral people are about it.

Overview of the Project

This project is a web application which is used to analyze the tweets about a certain topic using a hashtag or a certain person using their username. We will bebuilding a classifier to perform sentiment analysis of these tweets to determine whether they are positive, negative or neutral. This application can be used by any organization office to review their works or by any other company to reviewtheir products or brands.

The main feature of our application is that it helps to determine the public opinionabout the products, government work, politics or any other by analyzing the tweets.

Domain Introduction

This project of analyzing sentiments of tweets comes under the domain of "DataMining" and "Pattern Classification". These terms are very closely related and can be formally defined as the process of discovering "useful" patterns in large sets of data. This is done either automatically (unsupervised) or semi- automatically (supervised). Techniques of Natural Language Processing (NLP) are heavily relied on in extracting significant patterns and features from the largedata set of tweets. Machine Learning techniques are also heavily relied on for accurately classifying individual unlabeled data (tweets) according to the patternmodel that best describes them.

The features used for modeling patterns and classification can be divided into two main groups: formal language based and informal blogging based. Language based features deal with formal linguistics and include prior sentiment polarity ofindividual words and phrases, and parts of speech tagging of sentences (POS). Prior sentiment polarity means that words and phrases have a natural tendency toexpress specific sentiments. For example- the word "great" has a strong positive connotation while the word "bad" has a strong negative connotation. Whenever aword with positive connotation is used in a sentence, the sentence has a higher chance to also be expressing a positive sentiment. Parts of Speech tagging is a syntactic approach, it automatically identifies which part of speech each individual word of a sentence belongs to: noun, pronoun, adjective, verb, adverb,etc. Patterns can be extracted from analyzing the frequency distribution of these parts of speech either individually or collectively with some other part of speech,in a class of labelled tweets. Twitter based features relate with how people expressthemselves online on social platforms and are more informal. They compress theirsentiments to 140 characters, which is the limited space offered by twitter for each tweet. They include twitter hashtags, word lengthening, word capitalization, presence of URL in tweets, question marks, exclamation marks, internet emoticons and slang/shorthand.

The data sets used in this field are mainly from product reviews. These reviews are important to business holders to take business decisions in accordance with the analysis of users' opinions about their products.Sentimental Analysis finds its applications in various fields such as predicting stock markets, on political debates or news articles. In political debate, we could gain insight into people's opinions on certain election candidates or political parties. The election results might also be predicted from political posts. Social networking sites or micro-blogging sites are very good sources of information aspeople freely share and discuss their opinions about various topics and can be used as data sources in the Sentiment Analysis process.

Sentiment Classification techniques can be divided into machine learning approach, lexicon-based approach and hybrid approach.

Sentiment Classification Techniques

Sentiment classification techniques can be segregated into three categories (Fig. 1.). These are machine learning, lexicon-based and hybrid approaches [22]. The first of these techniques involve popular machine learning (ML) algorithms and involves using linguistic features. The second involves analyses through a collection of sentiment terms that are precompiled into sentiment lexicon. This is further divided into dictionary- and corpus-based approaches that use semantic or statistical methods to gauge the extent of polarity of sentiment. The hybrid approach involves combining ML and lexicon-based approaches. The following illustration aims at providing an insight into more popular algorithms used in sentiment classification techniques. Machine Learning Approach In machine learning approach, machine learning (ML) algorithms are used almost exclusively and extensively to conduct SA. ML algorithms are used on conjunction with linguistic and syntactic features.

Sentiment analysis can be defined as a process that automates mining of attitudes, opinions, views and emotions from text, speech, tweets and database sources through Natural Language Processing (NLP). Sentiment analysis involves classifying opinions in text into categories like "positive" or "negative" or "neutral". It's also referred as subjectivity analysis, opinion mining, and appraisal extraction.

The words opinion, sentiment, view and belief are used interchangeably but there are differences between them.

- ***Opinion****:* A conclusion open to dispute (because different experts have different opinions).
- ***View:*** subjective opinion
- ***Belief:*** deliberate acceptance and intellectual assent
- ***Sentiment:*** opinion representing one’s feelings

An example for terminologies for Sentiment Analysis is as given below,

<SENTENCE> = The story of the movie was weak and boring

<OPINION HOLDER> =<author>

<OBJECT> = <movie>

<FEATURE> = <story>

<OPINION >= <weak><boring>

<POLARITY> = <negative>

Sentiment Analysis is a term that include many tasks such as sentiment extraction, sentiment classification, subjectivity classification, summarization of opinions or opinion spam detection, among others. It aims to analyze people's sentiments, , attitudes, opinions emotions, etc. towards elements such as, products, individuals, topics ,organizations, and services.

The **Machine Learning Approach (ML)** applies the famous ML algorithms and uses linguistic features.

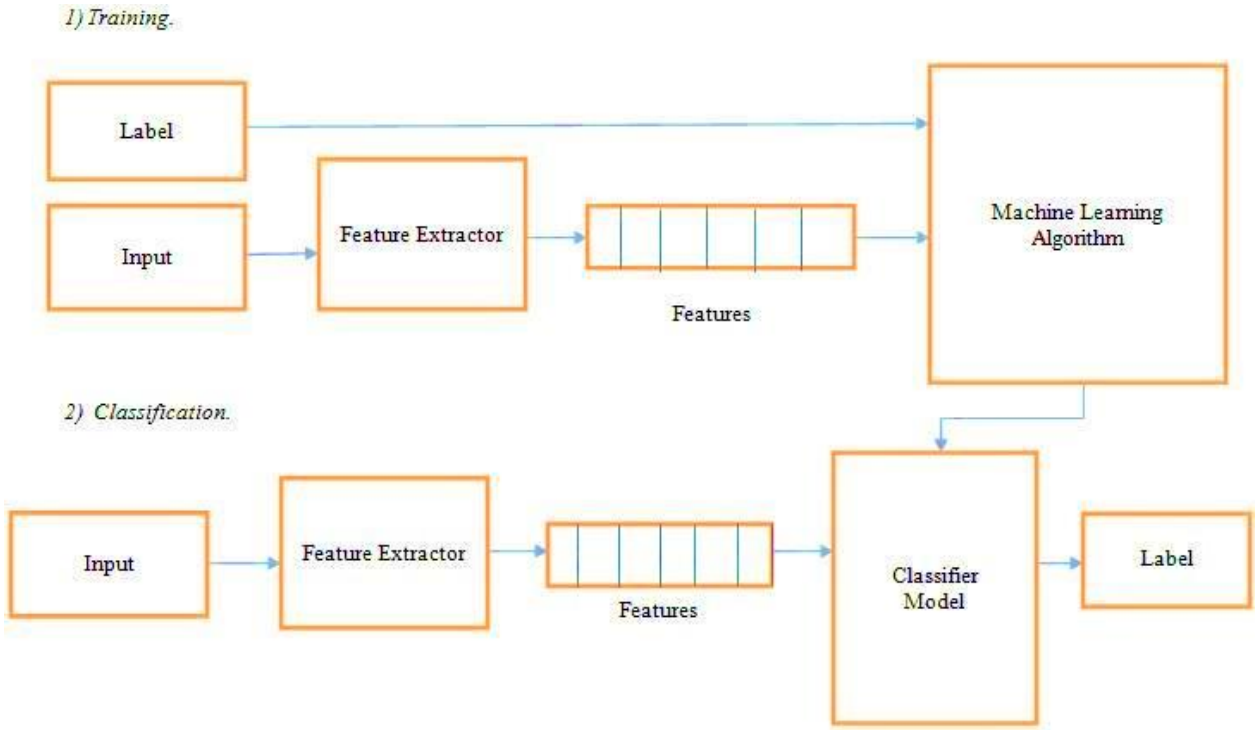


Figure1***: Machine Learning Approach***

The **Lexicon-based Approach** relies on a sentiment lexicon, a collection of known and precompiled sentiment terms. It is divided into a dictionary-based approach and corpus-based approach which use statistical or semantic methods tofind sentiment polarity.

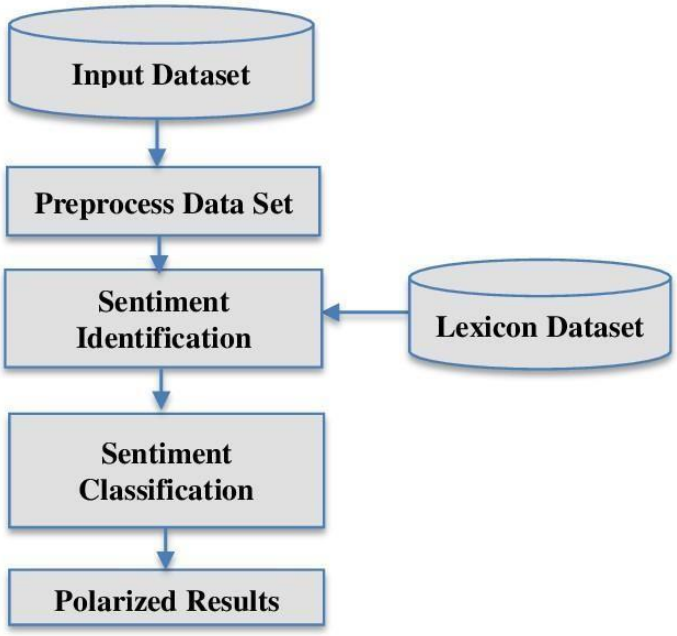


Figure 2: ***Lexicon-based Approach***

Supervised Learning: Supervised Learning involves datasets that are clearly labelled. Such datasets are share with assorted supervised learning models.

Decision Tree Classifiers: This type of classifier extends a hierarchical breakdown of training data space in which attribute values are used for data segregation [28].

This method is predicated or based upon the absence or presence of one or more words and is conducted recursively until a minimum number of records are registered with lead nodes that are used for classification.

Linear Classification: Linear classification models involve support vector machine (SVM). This is a form of classifier that is focused on segregating and isolating direct separators between different classes. It also involves neural system [29]. SVM is a form of supervised learning model and works chiefly on the principle of decision boundaries setup by decision planes. A decision plane is defined as a set of objects that are part of a range of class memberships.

Support Vector Machines Classifiers (SVM): SVMs have been designed to fundamentally identify and isolate linear separators in search space for the purpose of categorizing assorted classes. Ideally, text data is considered suitable for SVM classification in view of sparse nature accorded by text. Since, only a select number of features are irrelevant notwithstanding the tendency to be correlated and organized into linear distinguishing buckets, text data is often considered an ideal candidate for SVM [33]. With the help of SVM, a non-linear decision surface can be constructed out of original feature space. This can be achieved through mapping of data instances in a non-linear fashion to an inner product space where classes can be linearly segregated using hyperplane.

Neural Network (NN): Neural Networks are constituted of, as the term suggests, by neurons as the basic fundamental building block. Inputs to neurons are depicted using a vector and denoting word frequencies in a document across a line. A set of weights are considered for each neuron to enable computation of a function of the inputs involved. For boundaries that are nonlinear, multilayer neural networks are employed. These multiple layers are used in conjunction with multiple pieces of linear boundaries used for approximation of enclosed regions involving particular class. Neuron outputs generated in previous layers are used to feed the neurons in subsequent layers. In their research, relations were marked as being positive, neutral or unknown between two individuals. The case study involved historical information on biographies of individual of a given region and year. In terms of scoring accuracy and as a benchmark of practicability, the authors were able to demonstrate that SVM and a single layer NN (1-NN) algorithm together were effective in attaining highest scores.

Rule-based Classifiers: Data space is modelled with a set of rules in rule-based classifiers. On the left-hand side, feature set is expressed in disjunctive normal form to denote condition of the feature set itself. While on the right-hand side, class label is inserted. Conditions are usually formulated on basis of presence of term. The opposite, or term absence, is not normally invoked as it does not project informative behavior on sparse data. Several criteria are used to generate rules and training phases create rules based on criteria chosen. The two most common are being confidence and support [30]. Support criterion is considered to be the number of instances in training data set relevant for the rule. Confidence criterion involves conditional probability and denotes that the right-hand side of the rule is satisfactorily met only upon satisfying the left-hand side. The authors [31] have proposed combined rule algorithms. While decision rules and decision trees often involve encoding rules in feature space, the latter attempts to reach the goal using a hierarchical approach. In research [32], it was observed that a certain path in decision tree can be identified as being appropriate for classification rule in a text instance. The primary difference between the two is that while decision trees involve hierarchical partitioning of data space that is considered rigid, the latter is more flexible and can accommodate overlapping in decision space.

Probabilistic Classifiers: This type of classifier involves mixing of models to achieve classification. This model assumes that every class involved is a component of the mixture itself. Each component is a generative model that functions as probability sampler of any given term for that component. Alternatively, such kinds of classifiers are also known as generative classifiers. Three important probabilistic classifiers stand out. These are Bayesian Network, Naïve Byes, and Maximum Entropy.

Naïve Bayes Classifier (NB): This form of classification is used almost extensively for classifying text documents and conducting SA on such form of documents [40][41][42][43]. The technique is founded on a probabilistic approach and uses cooperative probabilities of specific terms with a text document as an input for approximation of probability of a certain group.

Bayesian Network: The Naïve Bayes classifier assumes that each and every component is complete independent of the other. This enables BN to display an acyclic graph that is not only coordinated but also relates random variables with edges representing dependencies that are in turn conditional.

BN approaches and examines factors and associations that exist between them. In such manner, an entire probability joint distributed over each element can be resolved. Considering that computational complexity of BN is expensive in terms of text mining, it is sporadically used.

Maximum Entropy Classifier (ME): A type of probabilistic classifier with a place among exponential class of models, it does not rely on the assumption that components are independent. On the other hand, ME is dependent on Principle of Maximum Entropy. It selects the model that has the largest entropy. ME classifiers find use in applications involving dialect identification, assumption investigation, point arrangement, etc.

Unsupervised Techniques: In this form of approach, classification of sentiment is achieved through comparison. Component comparisons take place involving word lexicons that are assigned sentiment values before use [3, 44]. The more popular forms of such group of techniques are hierarchical and partial clustering.

Lexicon-based Approach: This kind of approach involves determination of polarity by employing opinion words from sentiment dictionary and matches those with data. Such an approach marks sentiment scores to indicate positive, negative or objective types of words.

Lexicon-based approaches are dependent on sentiment of lexicon involving a set of precompiled and known sentiment phrases, terms and idioms. Two forms of subclassifications exist for this type of approach. These are discussed in subsequent sections.

Dictionary-based Approach: In this approach, arrangement of words is made possible through manual approach involving a group of instructions that are known beforehand. The conclusion set is generated by looking up a notable corpus – WordNet, for appropriate words and antonyms relevant for the SA [14, 15]. The process is iterative and stops only when no new words are detected. Else, subsequent iterations follow when words are progressively appended to seed list. After the process stops, a manual appraisal is conducted to evaluate and correct errors. However, this approach is not without its flaws as it is often unable to detect in certain circumstances involving specific introductions or words with spaces.

Corpus-Based: This approach involves dictionaries specific to given domain. The dictionaries are produced on the basis of seeds of opinion terms that grow out of search for related words through use of statistical or semantic procedures.

The **Hybrid Approach** combines both approaches and is very common withsentiment lexicons playing a key role in the majority of methods. The various approaches and the most popular algorithms:

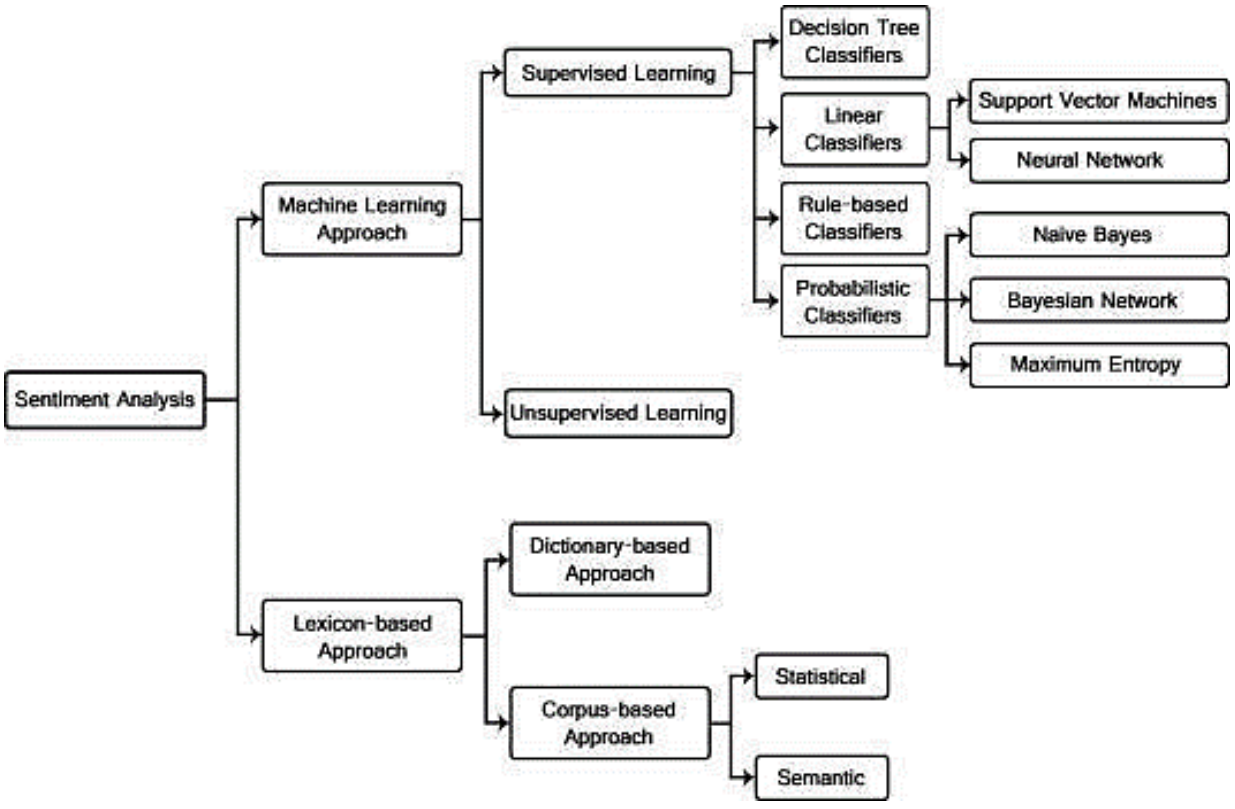


Figure 3: ***Types of Sentiment Analysis***

The text classification methods using ML approach can be roughly divided into supervised and unsupervised learning methods. The supervised methods make use of a large number of labeled training documents. The unsupervised methods are used when it is difficult to find these labeled training documents.

The lexicon-based approach depends on finding the opinion lexicon which is usedto analyze the text. There are two methods in this approach. The dictionary-based approach which depends on finding opinion seed words, and then searches the dictionary of their synonyms and antonyms. The corpus-based approach begins with a seed list of opinion words, and then finds other opinion words in a large corpus to help in finding opinion words with context specific orientations. This could be done by using statistical or semantic methods.

ML algorithms are usually used to solve the Sentiment Classification problems for its simplicity and the ability to use training data which gives it the privilege of domain adaptability. Therefore, we are going to be using a Machine Learningalgorithm for our project.

Machine Learning Approaches: Machine learning based approach uses classification technique to classify text into classes. There are mainly two types of machine learning techniques.

Unsupervised learning: It does not consist of a category and they do not provide with the correct targets at all and therefore rely on clustering.

Supervised learning: It is based on labelled dataset and thus the labels are provided to the model during the process. These labelled datasets are trained to get meaningful outputs when encountered during decision-making. The success of both this learning methods is mainly depending on the selection and extraction of the specific set of features used to detect sentiment.

The machine learning approach applicable to sentiment analysis mainly belongs to supervised classification. In a machine learning technique, two sets of data are needed:

1. Training Set
2. Test Set.

A number of machine learning techniques have been formulated to classify the tweets into classes. Machine learning techniques like Naive Bayes (NB), maximum entropy (ME), and support vector machines (SVM) have achieved great success in sentiment analysis. They can tell us how documents are represented.

The most commonly used features in sentiment classification are

- Term presence and their frequency
- Part of speech information
- Negations
- Opinion words and phrases

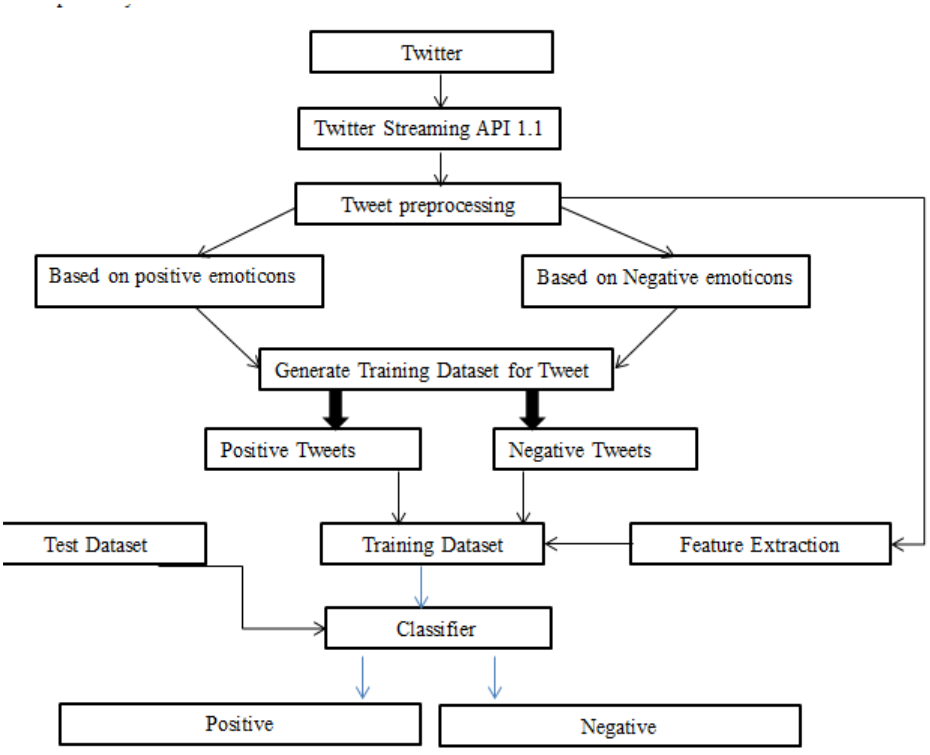


Figure 4: ***Sentiment Classification Based on Emoticons***

APPLICATIONS

The application of sentiment analysis ranges from business and marketing, health,politics to public action. Sentiment analysis provides a vast application in different areas to assist in decision making.

1. Applications that use Reviews from Websites: The Internet now has a large collection of feedback and reviews on almost everything. This includes feedback on product issues, comments about services, product reviews, etc.

A sentiment analysis system that can extract sentiments about certain services or products is therefore required which can help us automate the analysis of ratingsor feedback for a given item, program, product, etc. This would meet the needs of both the vendors and the users. Allowing us to keep track of what is being saidabout a service or product on social media and can help us detect angry and dissatisfied customers or negative mentions before they turn into a major crisis ornegatively impacts an organization's brand image.

2. Applications in Business Intelligence: People now tend to look forreviews of products online before they buy them. So, for many businesses, the online opinion becomes a deciding factor in the success or failure of their productor service.

Sentiment Analysis plays an important role in businesses in this way.It can assist businesses and organizations to take the appropriate action toimprove their product or services and business strategy. Sentiment analysis creates advantages for business owners to identify their popularity among customers and how customers think about their product or service and evaluate their business flow of stock price through social media and assessing the effectiveness and capability of business brand communication.

3. Applications as a Sub-component Technology: A sentiment predictor system can be helpful in recommender systems. These systems won't recommenditems that receive a lot of negative feedback or low ratings. Communication overthe internet can be dangerous. We can come across abusive language and other negative elements. These can be detected by identifying a highly negativesentiment and correspondingly taking action against it.

4. Applications across Domains: Recent researches in sociology and other fields like medical, sports have also been benefited by Sentiment Analysis that show trends in human emotions especially on social media.

5. Applications in Politics: Predicting the results of popular elections and polls is also an emerging application to sentiment analysis.

Data analyzed from twitter is more reliable as a platform where 94% of correlation has been found topolling data and have the potential to become a platform that is able to rival sophisticated polling techniques

6. Applications in Healthcare: We can see the application of sentiment analysis in healthcare and where the study uses Sentiment analysis as a service framework is proposed and utilizes spatio-temporal properties to identify locations of disease outbreaks.

Sentiment analysis can be used to find the level of depression of a person by observing and analyzing emotions from the text in their social media. In addition, sentiment analysis can identify sentiment needs ofpeople during a disaster and prepare an appropriate response to rescue.

Naïve Bayes Classifier (NB)

It is a Probabilistic classifier which uses mixture models for classification. The mixture model assumes that each class is a component of the mixture. Each mixture component is a generative model that provides the probability of sampling a particular term for that component. These kinds of classifiers are alsocalled *generative classifiers*.

The Naïve Bayes Classifier is the simplest and most commonly used classifier. Itcomputes the posterior probability of a class, based on the distribution of the words in the document.

The model works with the BoWs feature extraction which ignores the position ofthe word in the document. A bag-of-words model, or BoW for short, is a way of extracting features from text for use in modeling, such as with machine learning algorithms. A bag-of-words is a representation of text that describes the occurrence of words within a document. The approach is very simple and flexible,and can be used in a myriad of ways for extracting features from documents.

It uses Bayes Theorem to predict the probability that a given feature set belongs to a particular label.

$$P(label \mid features) = \frac{P(label) * P(features \mid label)}{P(features)}$$

P(*label*) is the prior probability of a label or the likelihood that a random featuresets the label. *P*(*features*|*label*) is the prior probability that a given feature set is being classified as a label. *P*(*features*) is the prior probability that a given featureset occurs.

Given the Naïve assumption which states that all features are independent, the equation could be rewritten as follows:

$$P(label \mid features) = \frac{P(label) * P(f1 \mid label) * \ldots * P(fn \mid label)}{P(features)}$$

Rather than computing P(features) explicitly, our algorithm just calculates thenumerator for each label, and normalizes them so they sum to one:

$$P(label \mid features) = \frac{P(label) * P(f1 \mid label) * \ldots * P(fn \mid label)}{SUM[l]\ (P(l) * P(f1 \mid l) * \ldots * P(fn \mid l))}$$

Despite the assumption we made above is not always held in the real word, the Naive Bayes model performs surprisingly well. (Domingos and Pazzani, 1997) show that even with some dependent features, Naive Bayes is optimal for certainproblems. Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets. It can be used for Binary as well as Multi-class Classifications.It is the most popular choice for text classification problems.

Advantages of Naïve Bayes Classifier:

- It is easy and simple to implement
- It performs well even without much training data
- It handles both continuous and discrete data
- It is highly scalable with the number of predictors and data points
- It is fast and can be used to make real-time predictions
- It is not sensitive to irrelevant feature

Text Blob

Text blob is a python library that provides text mining, text analysis and text processing modules for python developers. It reuses NLTK corpora. Text blob is a sentence level analysis tool. First, it takes a dataset as the input then it splits thereview into sentences.

A common way of determining polarity for an entire dataset is to count the number of positive and negative sentences/reviews and decide whether the response is positive and negative based on total number of positive and negative reviews. Polarity and subjectivity of a given review can beobtained using the sentiment module. It returns a named tuple with two parameters called polarity and subjectivity.

The default sentiment analyzer algorithm uses the implementation Pattern Analyzer which is based on a library called pattern. The sentiment analysis lexicon bundled in Pattern focuses on adjectives. It contains adjectives that occur frequently in customer reviews, hand-tagged with values for polarity and subjectivity.

The polarity score is ranging from -1 to 1 and subjectivity ranges are from 0 to 1 where 0 is most objective and1 is most subjective.

KNN Classifier

The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It's easy to implement and understand, but has a major drawback of becoming significantly slows as the size of that data in use grows.

The k-nearest neighbours (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It’s easy to implement and understand, but has a major drawback of becoming significantly slows as the size of that data in use grows.

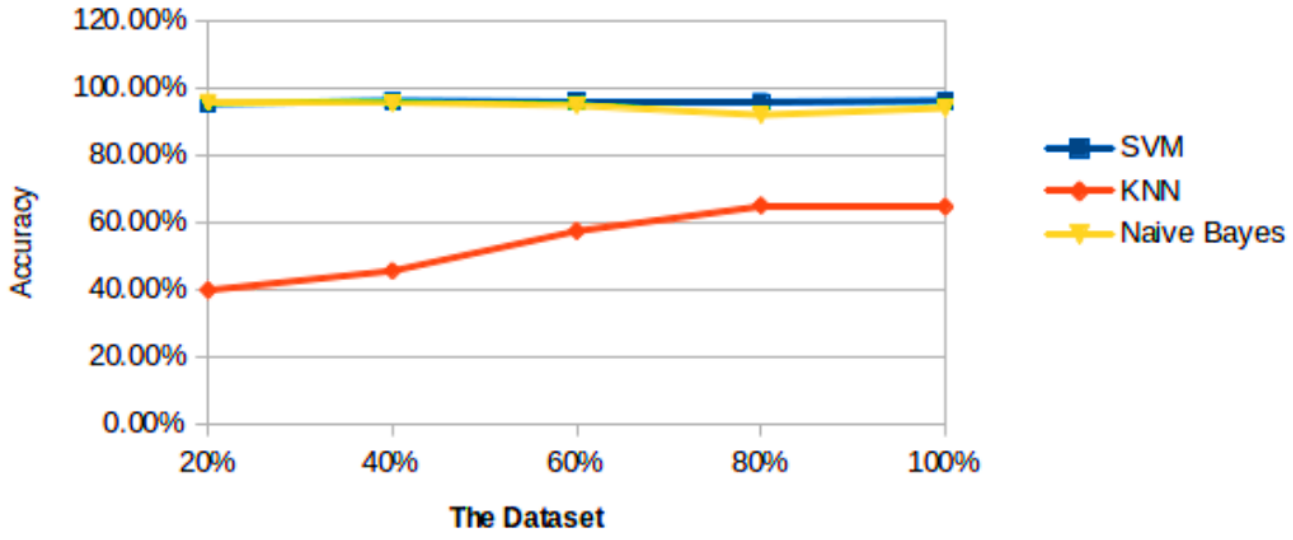
KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of classification) or averages the labels (in the case of regression).

In the case of classification and regression, we saw that choosing the right K for our data is done by trying several Ks and picking the one that works best.

Finally, we looked at an example of how the KNN algorithm could be used in recommender systems, an application of KNN-search.

COMPARISON WITH K-NN CLASSIFIER

One of the techniques used to find divorce is by doing data processing to predict the occurrence of a divorce that is by using data mining techniques such as Naive Bayes algorithm and K-Nearest Neighbor. This algorithm has a high degree of accuracy in predicting. The best level of accuracy between the two algorithms can be determined by comparison. Comparison of algorithm aims to get the algorithm that is considered the fastest and accurate to make a prediction of a problem. Result of comparison of Naive Bayes and K-Nearest Neighbor algorithm can be concluded that Naive Bayes algorithm yield 72,5% accuracy and K-Nearest Neighbor algorithm yield 57,5% accuracy.



WHY NAÏVE BAYES

Since a Naive Bayes text classifier is based on the Bayes's Theorem, which helps us compute the conditional probabilities of occurrence of two events based on the probabilities of occurrence of each individual event, encoding those probabilities is extremely useful.

Naive Bayes also assumes that the features are conditionally independent. Real data sets are never perfectly independent but they can be close. In short Naive Bayes has a higher bias but lower variance compared to logistic regression. If the data set follows the bias, then Naive Bayes will be a better classifier.

SYSTEM REQUIREMENTS STUDY

User characteristics:

Sentiment analysis is a process that automates mining of attitudes, views, opinions and emotions from text, tweets, speech and database sources through Natural Language Processing (NLP). It involves classifying opinions in text intocategories like "positive" or "negative" or "neutral". Also known as subjectivity analysis, appraisal extraction or opinion mining. The words sentiment, view, opinion and belief are used interchangeably but theyare somewhat different.

- Opinion: A conclusion open to dispute (because different experts havedifferent opinions)
- View: subjective opinion
- Belief: deliberate acceptance and intellectual assent
- Sentiment: opinion representing one's feelings

Sentiment Analysis includes many tasks such as sentiment extraction, subjectivity classification, sentiment classification, opinion spam detection, summarization ofopinions and many more. It aims to analyze people's sentiments, opinions, attitudes, emotions, etc. towards elements such as individuals, organizations, products, topics and services.
The main components of this application are:

- **UI:** The graphical interface of this web application has been developed using html, css, and streamlit which is an open-source Python library.

  It is simple and consists of a text input field for a hashtag to be searched, two sliders on the sidebar to input other fields and two buttons: one to analyses the hashtag and the other to display the raw twitter data that has been obtained from data mining through the twitter API.
- **Backend:** The backend of this web application has been developed using Python3 and its libraries.

In this project some libraries used:

- **Tweepy**, a Python library for accessing the Twitter API for scraping data or data mining.
- **Natural Language Toolkit (NLTK)**, an NLP library in Python, to clean thetweets.
- **TextBlob**, a Python (2 and 3) library for processing textual data. It providesa simple API for diving into common natural language processing (NLP) tasks. It is used for sentiment analysis.
- **Scikit-learn** (**Sklearn**) is the most useful and robust library for machine learning in Python.It provides a selection of efficient tools for machine learning and statistical modeling including various classification, regression and clustering algorithms.
- **Jupyter Notebook**, an interactive computational environment, in which you can combine code execution, rich text, mathematics, plots and rich media. It is used for faster implementation of the experiments.

The web application is hosted using **heroku**, a cloud platform.

HARDWARE REQUIREMENTS:

- x86 64-bit CPU (Intel / AMD architecture)
- 4 GB RAM
- 5 GB free disk space

SOFTWARE REQUIREMENTS:

- Windows 7 or higher
- A web browser preferably Chrome, Firefox
- Python-3
- Twitter developer account

SYSTEM ANALYSIS

Feasibility study:

Technical Feasibility: Our system is technically feasible since all the required tools are easily available. PyCharm which is a hybrid-platform IDE for python and Streamlit which is an open-source library for python makes the system more user and developer friendly. Jupyter Notebook is an easy to use, interactive data science environment.

1. **Operational Feasibility:** In this project, we have made a simplified web application for analyzing the tweets. It is simple to operate and can be used in any webpages. It is free and not costly to operate.

2. **Economic Feasibility:** This is a web-based application. Hosted on a free cloud service platform; heroku. Creation of application is not costly.

3. **Schedule Feasibility:** This application development is feasible in terms of time.

Requirement Definition

After the extensive analysis of the problems in the system, we are familiarized with the requirement that the current system needs. The requirement that the system needs is categorized into the functional and non-functional requirements.These requirements are listed below.

1. **Functional Requirements:** Functional requirements are product features or functions those developers must implement to enable users to accomplish their tasks. The functional requirements that this system must require are as follows:

    - System should be able to set twitter authentication keys.
    - System should be able to authorize tweepy and access twitter API.
    - System should be able to analyse data and classify each tweet polarity.

2. **Non-Functional Requirements:** Non-functional Requirements define system attributes such as reliability, maintainability, security, performance, scalability and usability. They serve as constraints on the design of the system across different backlogs. They ensure the usability and effectiveness of the entire system.

    Based on this the non-functional requirements are as follows:

    - User friendly
    - System should provide better accuracy
    - To perform with efficient throughput and response time
    - Easy to maintain and is reliable

SYSTEM DESIGN

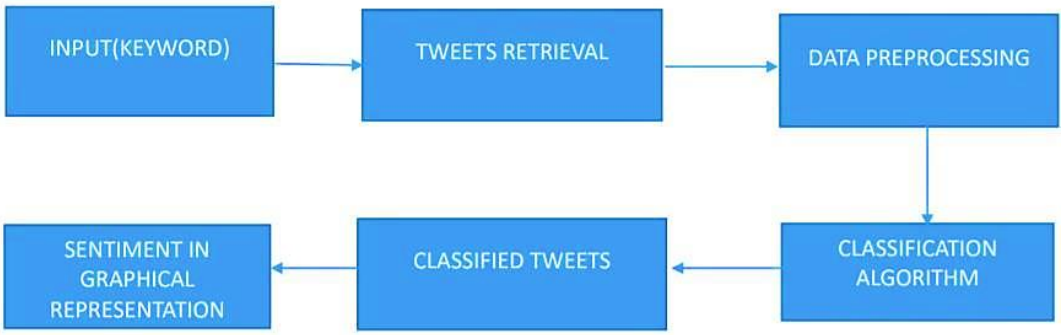
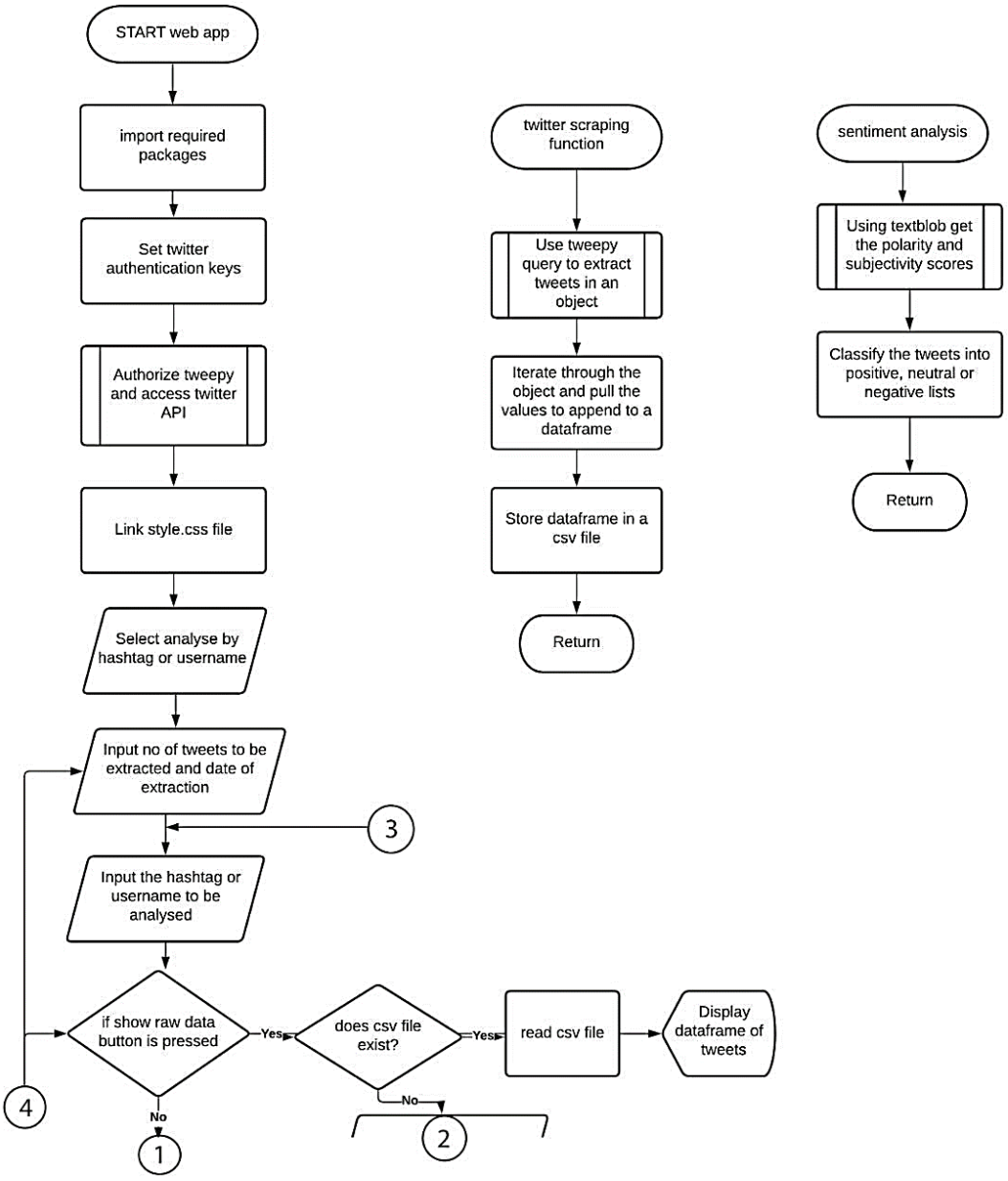


Figure 4: ***System Design***

Web Application Flowchart



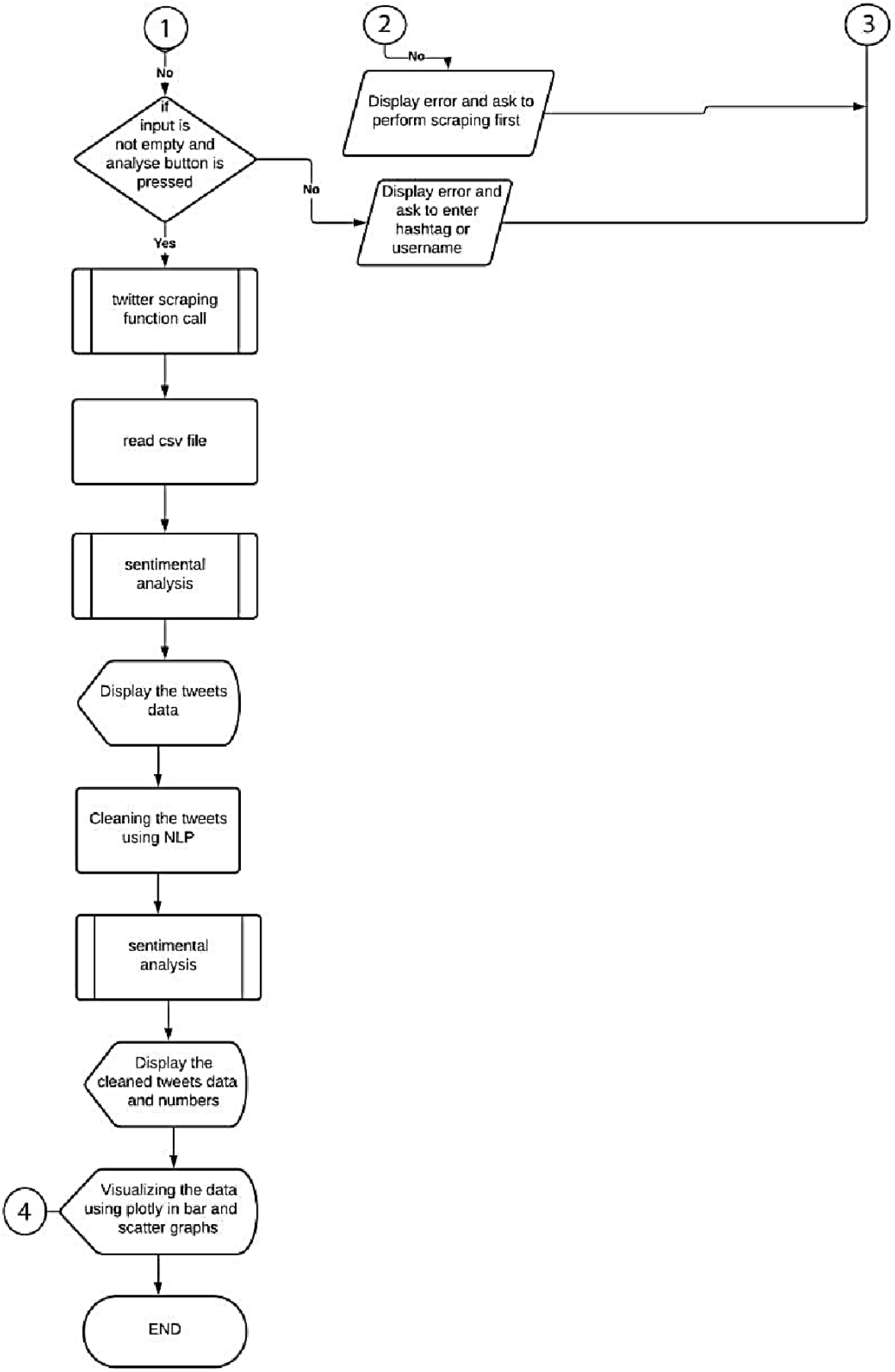


Figure 5: ***Web App Flowchart***

METHODOLOGY

For the development a dataset containing labelled tweets is extracted fromKaggle. Pre-processing takes place on the data obtained. Then by using Naive Bayes algorithm a classifier will be trained using training data. Several methodologies will be used to extract features from the source text. After featuresare added to the feature vector, each tweet in the training data is associated with the class label and passed to the classifier. Test tweets will be given to the modeland classification will be performed with the help of these trained classifiers. Weget the tweets classified into the positive and negative. We then extend the classification to include the neutral class. The model can now be saved and loadedwhen a prediction is to be made. It is ready to be implemented in our web application.

I. Loading sentiment data: The Dataset for this project is extracted from Kaggle. This data set contains more than a million tweets extracted using the twitter API. The tweets have been annotated.

(0 = negative, 4 = positive) and they can be used to detect sentiment.

It contains the following 6 fields:

1. target: the polarity of the tweet
2. ids: The id of the tweet
3. date: the date of the tweet
4. flag: The query. If there is no query, then this value is NO_QUERY.
5. user: the user that tweeted
6. text: the text of the tweet

The files contain positively labelled and negatively labelled tweets. First, the dataset is loaded. We use a fraction of this to train our model for better time efficiency. 200000 tweets were loaded. We replaced the polarity value of positive as: 4→ 1. Next, we check the number of positive and negative tagged tweets.

No. of positive tagged sentences: 99956

No. of negative tagged sentences: 100044

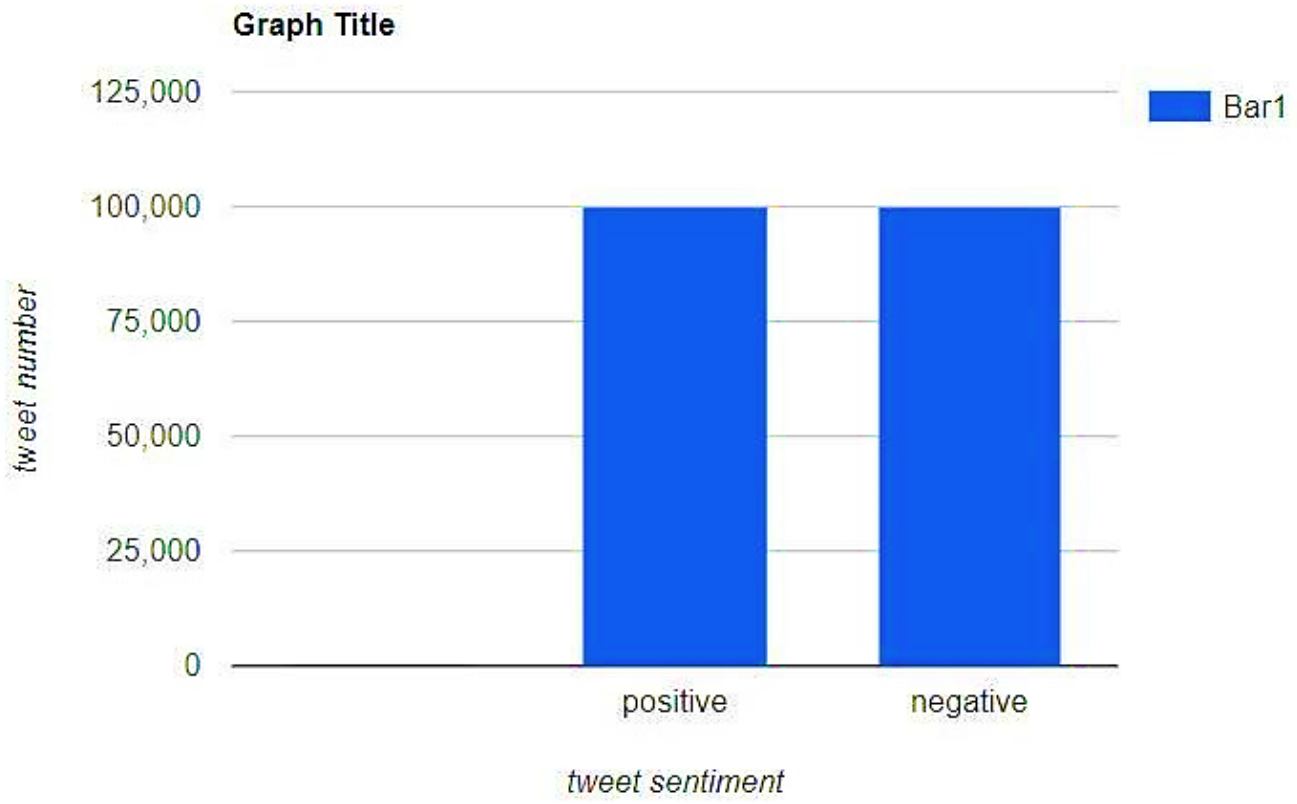


Figure 6: ***Dataset bias***

We see that the dataset is balanced. Therefore, the chance baseline is 50%. We drop the unnecessary columns leaving only polarity and text in the data.

II. Pre-processing Data: After loading data, pre-processing takes place. The Preprocessing steps taken are:

- Lower Casing: Each text is converted to lowercase.
- Removing URLs: Links starting with "http"/ "https" or "www" arereplaced by "".
- Removing Usernames: Replace @Usernames with word "".
- Removing Short Words: Words with length less than 2 are removed.
- Removing Stopwords: Stopwords are the English words which do not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence. (eg: "the", "she", "have").
- Lemmatizing: Lemmatization is the process of converting a word to its base form. (eg: runs, running, ran are all forms of the word run, thereforerun is the lemma of all these words).

III. Training Naïve Bayes Classifier:

Convert text to word frequency vectors

TF-IDF: This is an acronym that stands for Term Frequency – Inverse DocumentFrequency which are the components of the resulting scores assigned to each word.

- Term Frequency: This summarizes how often a given word appears withina document.
- Inverse Document Frequency: This downscales words that appear a lotacross documents.

Split train and test

The Preprocessed Data is divided into 2 sets of data:

- Training Data: The dataset on which the model would be trained on.Contains 80% data.
- Test Data: The dataset on which the model would be tested against.Contains 20% data.

IV. Implementation of evaluation metric: Finally, for testing the accuracy of the model, F1 score is evaluated. In statistical analysis of binary classification, the F1-score or F-measure is a measure of a test's accuracy. For the evaluation of data confusion matrix is used. A confusion matrix is a technique for summarizing the performance of a classification algorithm. Calculating a confusion matrix can give us a better idea of what our classification. model is getting right and what types of errors it is making. The number of correct and incorrect predictions are summarized with count values and broken down by each class.

There are 4 important terms in confusion metric:

1. True Positives: The cases in which we predicted YES, and actual output was also YES.
2. True Negatives: The cases in which we predicted NO, and the actual output was NO.
3. False Positives: The cases in which we predicted YES, and actual output was NO.
4. False Negatives: The cases in which we predicted NO, and actual output was YES.

On evaluating our model, we found: Accuracy of model on training data: 86.868749999 Accuracy of model on testing data: 75.697499999 The F1 score: 0.76.

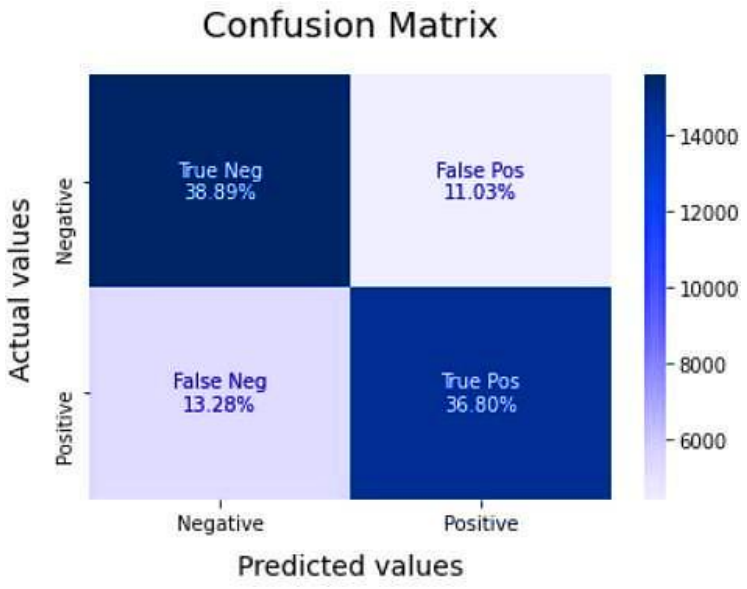


Figure 7: ***Confusion Matrix of Model***

Now we save this model and load it at the time of making a prediction.

V. Handling the Neutral Class: In the previous sections, neutral sentiment was disregarded. The training and test data only had text with positive and negative sentiments. In this section, we explore what happens when neutral sentiment is introduced.

Naive Bayes with Three Classes

Our model returns its results as a named tuple of the form:
Sentiment (classification, p_pos, p_neg)

eg: Sentiment (classification='pos', p_pos=0.5057908299783777, p_neg=0.49420917002162196)

We extended the Naive Bayes Classifier to handle 3 classes: positive, neutral and negative. By taking the difference of p_pos and p_neg as a polarity score (polarity = p_pos - p_neg).

we have classified the three classes as such:

Positive: if the polarity score is more than or equal to 0.2.
Negative: if the polarity score is less than or equal to - 0.2.
Neutral: if the polarity score is between 0.2 and -0.2.

Although rudimentary it works quite well for this type of classification.

Finally, after classifying the tweets, we can visualize their count using graphs. We also visualize the polarity and subjectivity obtained through the textblob library in a scatter plot.

Implementation

In this paper, Tweepy python library has been utilized for data extraction from Twitter API (Application programming interface). Moreover, Tweepy allows appropriate data retrieval by searching via keywords, hashtags, timelines, trends,or geo-location. In spite of having numerous restrictions from Twitter API, we have applied successive attempts to access as many posts as possible. We test outour modified Naive Bayes algorithm and compare it to the inbuilt PatternAnalyzerin TextBlob. We have used usernames to test our algorithm in contrast to hashtagsas the test data extracted varies when using hashtags in real time. In order to avoid redundancy, a specific timeframe has been considered to pull out the tweets during a week. The gathered data has been stored in CSV format, and fed to our Sentiment Analysis model and Textblob. The data gets analysed, classified and finally visualized.

Coding:

The main modules are:

Setting Tweepy authentication keys:

```
24  #authorize tweepy
25  consumer_key = '<CONSUMER_KEY>'
26  consumer_secret = '<CONSUMER_SECTRET KEY>'
27  auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
28
29  redirect_url = auth.get_authorization_url()
30  # Get access token
31  key = '<KEY>'
32  secret = '<SECRET>'
33  auth.set_access_token(key, secret)
34  # Construct the API instance
35  api = tweepy.API(auth)
36
```

The key variables CONSUMER_KEY, CONSUMER_SECRET_KEY**,** KEY,and SECRET are set through heroku for secur

Twitter Scraping function:

```
# twitter scraping function
def scraptweets(search_words, date_since, numTweets, numRuns):
    # Define a for-loop to generate tweets at regular intervals
    # We cannot make large API call in one go. Hence, let's try T times

    # Define a pandas dataframe to store the date:
    db_tweets = pd.DataFrame(columns = ['username', 'acctdesc', 'location', 'following',
                                        'followers', 'totaltweets', 'usercreatedts', 'tweetcreatedts',
                                        'retweetcount', 'text', 'hashtags']
                             )

    for i in range(0, numRuns):
        # We will time how long it takes to scrape tweets for each run:
        start_run = time.time()

        # Collect tweets using the Cursor object
        # .Cursor() returns an object that you can iterate or loop over to access the data collected.
        # Each item in the iterator has various attributes that you can access to get information about each tweet
        if option == 'Username':
            utweets = api.user_timeline(
                screen_name=search_words, count=numTweets, lang='en', tweet_mode='extended')
            db_tweets = pd.DataFrame(columns=['text'])
            for t in utweets:
                text = t.full_text
                db_tweets = db_tweets.append({'text': text}, ignore_index=True)

        else:
            tweets = tweepy.Cursor( api.search_tweets ,
                                    q=search_words, geocode=location,
                                    lang="en", until=date_since,
                                    tweet_mode='extended',
                                    count=numTweets ).items(numTweets)

            # Store these tweets into a python list
            tweet_list = [tweet for tweet in tweets]

            # Obtain the following info (methods to call them out):
            # user.screen_name - twitter handle
            # user.description - description of account
            # user.location - where is he tweeting from
            # user.friends_count - no. of other users that user is following (following)
            # user.followers_count - no. of other users who are following this user (followers)
            # user.statuses_count - total tweets by user
            # user.created_at - when the user account was created
            # created_at - when the tweet was created
            # retweet_count - no. of retweets
            # (deprecated) user.favourites_count - probably total no. of tweets that is favourited by user
            # retweeted_status.full_text - full text of the tweet
            # tweet.entities['hashtags'] - hashtags in the tweet
            # Begin scraping the tweets individually:
            noTweets = 0
```

```
        for tweet in tweet_list:
            # Pull the values
            username = tweet.user.screen_name
            acctdesc = tweet.user.description
            location = tweet.user.location
            following = tweet.user.friends_count
            followers = tweet.user.followers_count
            totaltweets = tweet.user.statuses_count
            usercreatedts = tweet.user.created_at
            tweetcreatedts = tweet.created_at
            retweetcount = tweet.retweet_count
            hashtags = tweet.entities['hashtags']
            try:
                text = tweet.retweeted_status.full_text
            except AttributeError:  # Not a Retweet
                text = tweet.full_text
                # Add the 11 variables to the empty list - ith_tweet:
            ith_tweet = [username, acctdesc, location, following, followers, totaltweets,
                         usercreatedts, tweetcreatedts, retweetcount, text, hashtags]
            # Append to dataframe - db_tweets
            db_tweets.loc[len(db_tweets)] = ith_tweet
            # increase counter - noTweets
            noTweets += 1

    # Run ended:
    end_run = time.time()
    duration_run = round((end_run - start_run) / 60, 2)

    print('no. of tweets scraped for run {} is {}'.format(i + 1, numTweets))
    print('time take for {} run to complete is {} mins'.format(
        i + 1, duration_run))

    # Once all runs have completed, save them to a single csv file:

    # Define working path and filename
    path = os.getcwd()
    filename = path + '/data/' + 'test_data_tweets.csv'
    # Store dataframe in csv with creation date timestamp

    db_tweets.to_csv(filename, index=False)
    print('Scraping has completed!')
```

Sentiment Analysis:

Sentiment Analysis of tweets before cleaning of data is done.

```
# Sentiment Analysis:
    if search_words != '#' and search_words != '' and search_words != '@':
        if pressed:
            program_start = time.time()
            scraptweets(search_words, option_location, date_since, numTweets, numRuns)
            st.success('Scraping done successfully ')
            df = pd.DataFrame()
            tweets = pd.read_csv('data/test_data_tweets.csv')
            column1, column2 = st.columns(2)
            tweetList = tweets.text

            for tweet in tweetList:
                tweet_list.append(tweet)
                bayes = tb(tweet)
                score = bayes.sentiment
                print(score)
                neg = score[2]
                pos = score[1]
                polarity = pos - neg
                print(polarity)
                if polarity < -0.2:
                    negative_list.append(tweet)
                    negative += 1
                elif polarity > 0.2:
                    positive_list.append(tweet)
                    positive += 1
                else:
                    neutral_list.append(tweet)
                    neutral += 1
```

Cleaning data:

```
# Cleaning Text (RT, Punctuation etc)
            tweet_list.drop_duplicates(inplace=True)
            # Creating new dataframe and new features
            tw_list = pd.DataFrame(tweet_list)
            tw_list["text"] = tw_list[0]

            # Removing Punctuation
            def remove_punctuation(text):
                no_punct = [
                    words for words in text if words not in string.punctuation]
                words_wo_punct = ''.join(no_punct)
                return words_wo_punct

            tw_list['text'] = tw_list['text'].apply(
                lambda x: remove_punctuation(x))

            # tokenization
            def tokenize(text):
                # Here, "\W+" splits on one or more non-word character
                split = re.split("\W+", text)
                return split

            tw_list['text'] = tw_list['text'].apply(
                lambda x: tokenize(x.lower()))

            # removing stopwords
            stopword = nltk.corpus.stopwords.words('english')

            def remove_stopwords(text):
                text = [word for word in text if word not in stopword]
                return text

            tw_list['text'] = tw_list['text'].apply(
                lambda x: remove_stopwords(x))

            # lemmetize text
            lemmatizer = WordNetLemmatizer()

            def word_lemmatizer(text):
                lem_text = [lemmatizer.lemmatize(i) for i in text]
                return lem_text

            tw_list['text'] = tw_list['text'].apply(
                lambda x: word_lemmatizer(x))

            # converting class list to string
            def list_to_string(texts):
                sentence = '-'.join(texts)
                sentence = ' '.join(texts)
                return sentence
```

Sentiment Analysis and Classification of cleaned tweets:

```
# Calculating Negative, Positive, Neutral and Compound values again
            tw_list[['polarity', 'subjectivity']] = tw_list['text'].apply(
                lambda Text: pd.Series(TextBlob(Text).sentiment))
            for index, row in tw_list['text'].iteritems():
                bayes = tb(row)
                score = bayes.sentiment
                print(score)
                neg = score[2]
                pos = score[1]
                polarity = pos - neg
                print(polarity)
                tw_list.loc[index, 'polarity'] = polarity
                if polarity < -0.2:
                    tw_list.loc[index, 'sentiment'] = "negative"
                elif polarity > 0.2:
                    tw_list.loc[index, 'sentiment'] = "positive"
                else:
                    tw_list.loc[index, 'sentiment'] = "neutral"

                tw_list.loc[index, 'neg'] = neg
                tw_list.loc[index, 'pos'] = pos


# Creating new data frames for all sentiments (positive, negative and neutral)
            tw_list_negative = tw_list[tw_list["sentiment"] == "negative"]
            tw_list_positive = tw_list[tw_list["sentiment"] == "positive"]
            tw_list_neutral = tw_list[tw_list["sentiment"] == "neutral"]

            tw_neutral_list = pd.DataFrame(tw_list_neutral)
            tw_negative_list = pd.DataFrame(tw_list_negative)
            tw_positive_list = pd.DataFrame(tw_list_positive)
            pos_num = len(tw_positive_list)
            neg_num = len(tw_negative_list)
            neu_num = len(tw_neutral_list)

            st.header('')
            st.header('After cleaning data:')
            st.write("total number of tweets: ", len(tw_list))
            st.write("number of positive number: ", pos_num)
            st.write("number of negative number: ", neg_num)
            st.write("number of neutral number: ", neu_num)

            st.write('Cleaned Tweet List:')
            tw_list.reset_index(inplace=True)
            tw_list.index += 1
            st.write(tw_list)
```

Graphical Visualization of Data:

```
# Visualizing Data
            st.header('')
            bar_chart = ['positive', 'neutral', 'negative']

            fig = go.Figure(
                [go.Bar(x=bar_chart, y=[pos_num, neu_num, neg_num])])
            fig.update_traces(marker_color='rgb(158,202,225)', marker_line_color='rgb(8,48,107)',
                              marker_line_width=1.5, opacity=0.6)
            fig.update_layout(title_text='SENTIMENTS OF TWEETS FETCHED:')
            st.write(fig)

            fig2 = go.Figure(data=go.Scatter(
                x=tw_list['polarity'],
                y=tw_list['subjectivity'],

                mode='markers',
                marker=dict(
                    size=16,
                    color=tw_list['polarity'],  # set color equal to a variable
                    colorscale='Viridis',  # one of plotly colorscales
                    showscale=True
                ),
            ))
            fig2.update_layout(title='Polarity And Subjectivity:')
            fig2.update_xaxes(title_text='Polarity')
            fig2.update_yaxes(title_text='Subjectivity')

            st.write(fig2)
```

Results And Discussion

When comparing our algorithm to the Text Blob Pattern Analyzer to classify tweets, we got the following results:

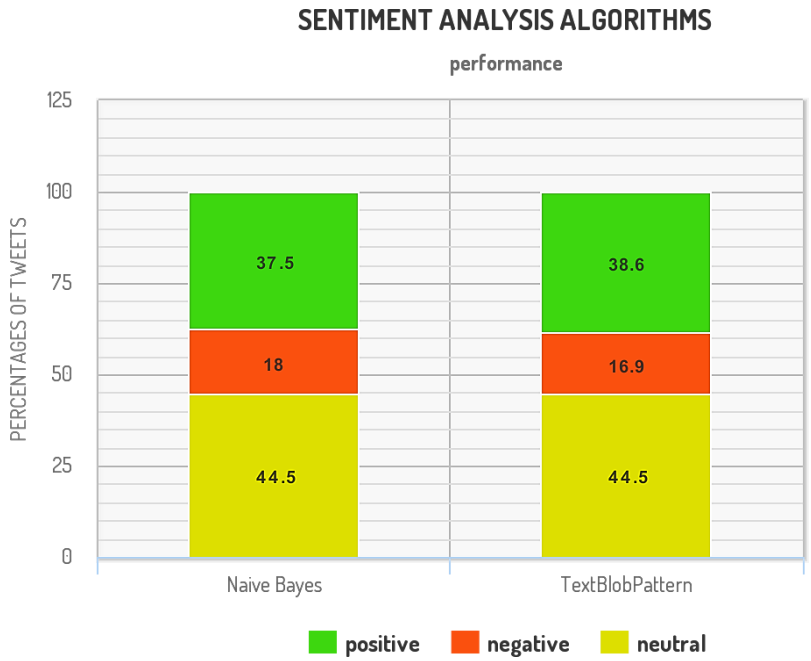


Figure 8: ***Performance Analysis of Algorithms***

Thus, we see that the tweet classification is almost the same whether using our simple modified Naive Bayes model were using machine learning, the computer automatically learned what words are associated with a positive or negative ratingor Text Blob's Pattern Analyzer which uses the pattern library which is hand- tagged with values for polarity and subjectivity. Analyses was done on this labeled dataset using various feature extraction technique.

Train data	45000
Negative	23514
Positive	21486
Test Data	44832
Negative	22606
Positive	22226

Naïve Bayes Algorithm:

- **Effect of Stop words:** When Naïve Bayes (Baseline) was run, it gave an accuracy of 73.65 percent, which is considered as the baseline result. The next thing used was removal of stop word. When stop words were removed and Naïve Bayes was run, it gave an accuracy of 74.56 percent. Following table shows the accuracy obtained at different sizes for the Naïve Bayes with stop words removed and using preprocessed data and based on unigram model.

Table 5. Accuracy of Naïve Bayes Algorithm (Stop word removal + unigram) Dataset	**Accuracy**
10	0.522305496074
50	0.583333333333
100	0.593839221984
500	0.649134546752
1000	0.673536759458
5000	0.7005710207
10000	0.717300142755
15000	0.725486259814
20000	0.731441827266
25000	0.734653818701
30000	0.738891862955
35000	0.740743219129
40000	0.742148465382
45000	0.745605817273

disappointed = True neg: pos = 28.8 : 1.0

sad = True neg: pos = 27.6 : 1.0

awful = True neg: pos = 20.3 : 1.0

ugh = True neg: pos = 19.3 : 1.0

poor = True neg: pos = 19.3 : 1.0

sucks = True neg: pos = 18.7 : 1.0

upset = True neg: pos = 18.0 : 1.0

argh = True neg: pos = 17.3 : 1.0

battery = True neg: pos = 16.6 : 1.0

- **Effect of Bigram:** Bigram uses a combination of two words as a feature. Bigram effectively captures some features in the data that unigram fails to capture. For example, words like ‟not sad‟, ‟not good‟ clearly say that the sentiment is negative. This effect can be clearly seen from the increase in accuracy from 74.56(Unigram) to 76.44 percent which is almost a 2% increase. Following table shows the accuracy obtained at different sizes for the Naïve Bayes algorithm with bigram model.

Table 6. Accuracy of Naïve Bayes Algorithm (Stopword removal+Bigram) Dataset	**Accuracy**
10	0.544990185582
50	0.593593861527
100	0.591407922912
500	0.654956281228
1000	0.67193076374
5000	0.718214668094
10000	0.730973411849
15000	0.740609386153
20000	0.746431120628
25000	0.75073608137
30000	0.755041042113
35000	0.758453783012
40000	0.762892576731
45000	0.764476266952

The most informative features for Naive Bayes with Bigrams as features.
('so', 'sad') = True neg: pos = 55.2 : 1.0

sad. = True neg: pos = 44.2 : 1.0

bummed = True neg: pos = 33.8 : 1.0

horrible = True neg: pos = 32.0 : 1.0

('USERNAME', 'welcome') = True pos: neg = 29.5 : 1.0

('welcome', 'to') = True pos: neg = 28.1 : 1.0

sad = True neg: pos = 27.5 : 1.0

('i', 'lost') = True neg: pos = 24.7 : 1.0

died = True neg: pos = 24.3 : 1.0

('miss', 'him') = True neg: pos = 24.1 : 1.0

- **Effect of using Trigram:** Running Naïve Bayes using Trigrams, bigrams and unigrams together gave an accuracy of 75.41 percent which is less than the accuracy obtained when Bigrams were used as a feature. Also, this feature combination bloats up the feature space exponentially and the execution becomes extremely slow. Hence for further analysis, the trigrams are not considered as they do not have a notice able impact on the accuracy. Following table shows the accuracy obtained at different sizes for the Naïve Bayes algorithm with Trigram model.

Table 7. Accuracy of Naïve Bayes Algorithm (Stopword removal+Trigram) Dataset	**Accuracy**
10	0.486995895789
50	0.528484118487
100	0.581571199143
500	0.634346002855
1000	0.654331727338
5000	0.703403818701
10000	0.721002855103
15000	0.731352605282
20000	0.737419700214
25000	0.742148465382
30000	0.74823786581
35000	0.748773197716
40000	0.753234296931
45000	0.754171127766

The most informative features for Naive Bayes with Trigrams as features.

('so', 'sad') = True neg :pos = 59.1 : 1.0

('lost', 'my') = True neg :pos = 38.9 : 1.0

('i', 'miss', 'my') = True neg :pos = 36.9 : 1.0

('going', 'to', 'miss') = True neg :pos = 28.5 : 1.0

('miss', 'him') = True neg :pos = 25.4 : 1.0

('happy', "mother's", 'day') = True pos :neg = 25.0 : 1.0

("can't", 'sleep') = True neg :pos = 21.5 : 1.0

('sad', 'that') = True neg :pos = 21.5 : 1.0

('miss', 'my') = True neg :pos = 21.4 : 1.0

('i', 'lost') = True neg :pos = 20.9 : 1.0

Following graph shows the summary of the results obtained by using different features and variation in the naïve bayes algorithm.

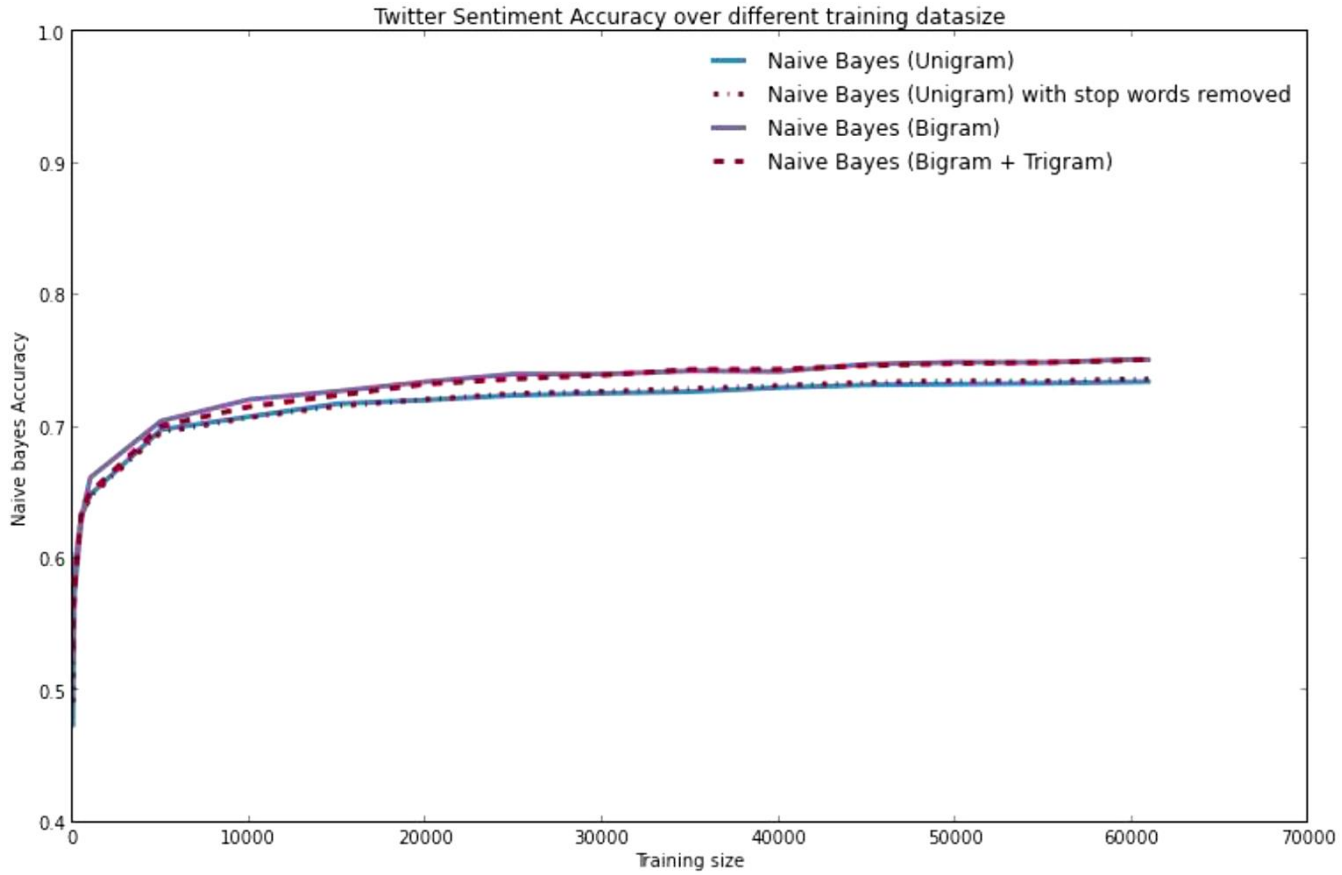


Fig.6 *Graph Representing Different results obtained for Naïve Bayes Algorithm.*

Table 8. Accuracy of Naïve Bayes Algorithm	**Accuracy**
Naïve Bayes (unigram)	74.56
Naïve Bayes (bigram)	76.44
Naïve Bayes (trigram)	75.41

Implementation In Web Application

For example: Giving the input of “uber” in the hashtag option, we can analyze thesentiments (positive, negative or neutral), polarity and subjectivity of the tweets that are tagged with #uber as shown below.

SNAPSHOTS:

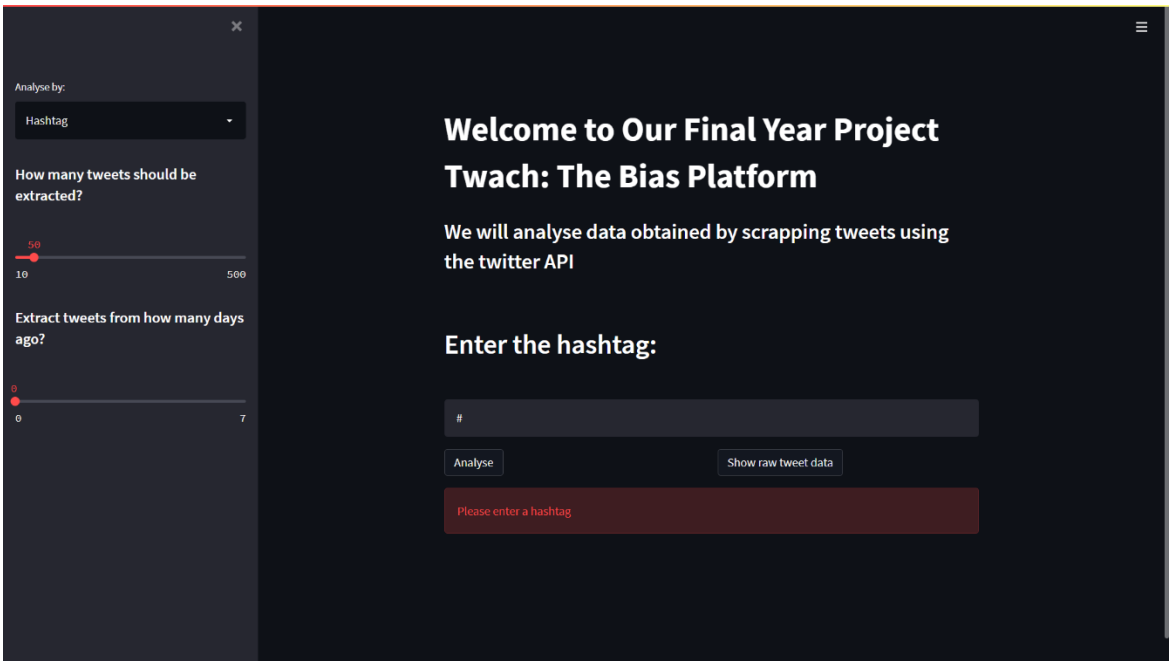


Figure 9: ***Snapshot of Web Application #1***

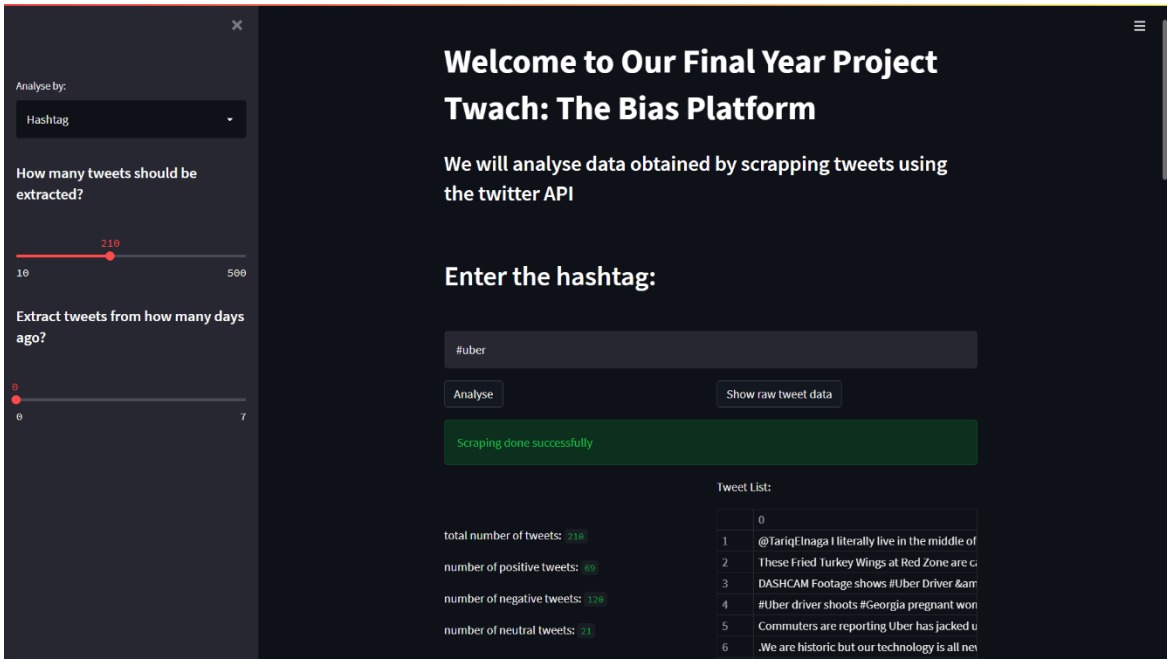


Figure 10: ***Snapshot of Web Application #2***

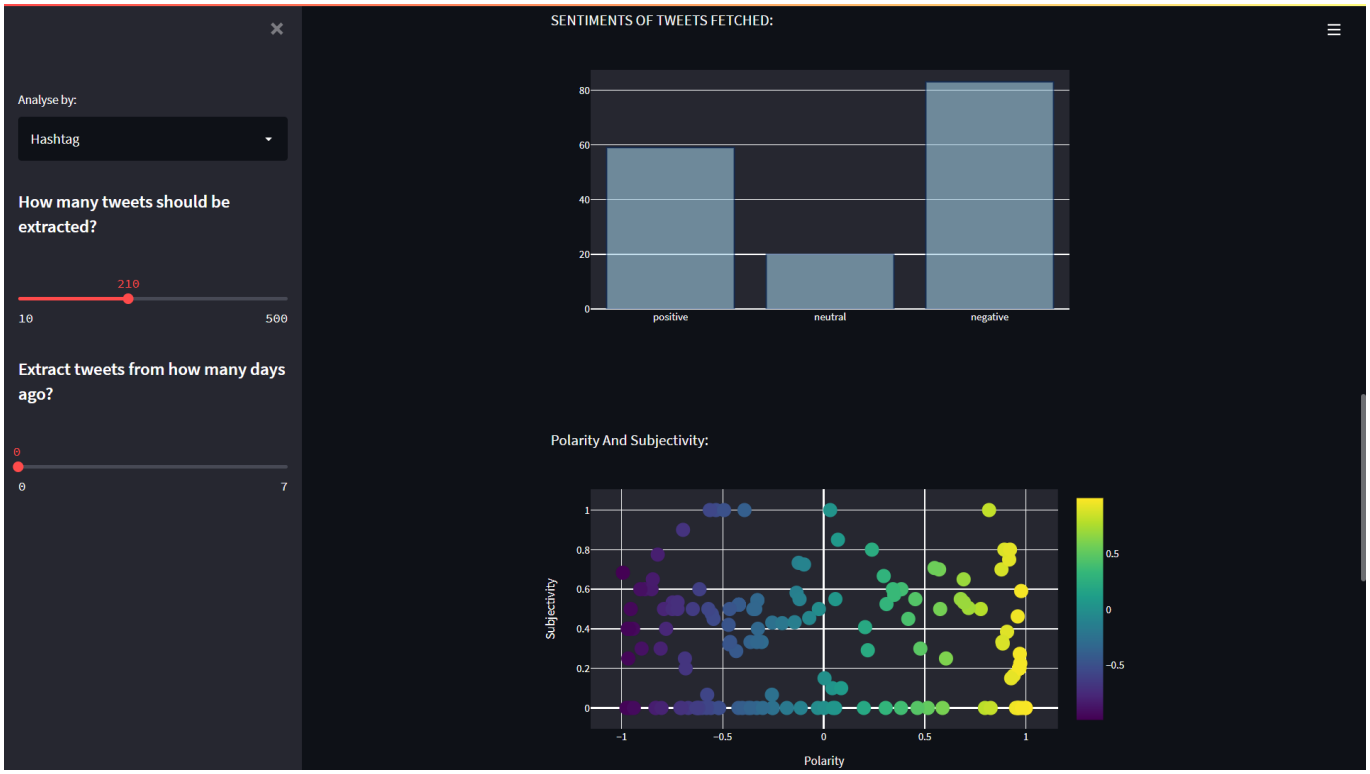


Figure 11: ***Snapshot of Web Application #3***

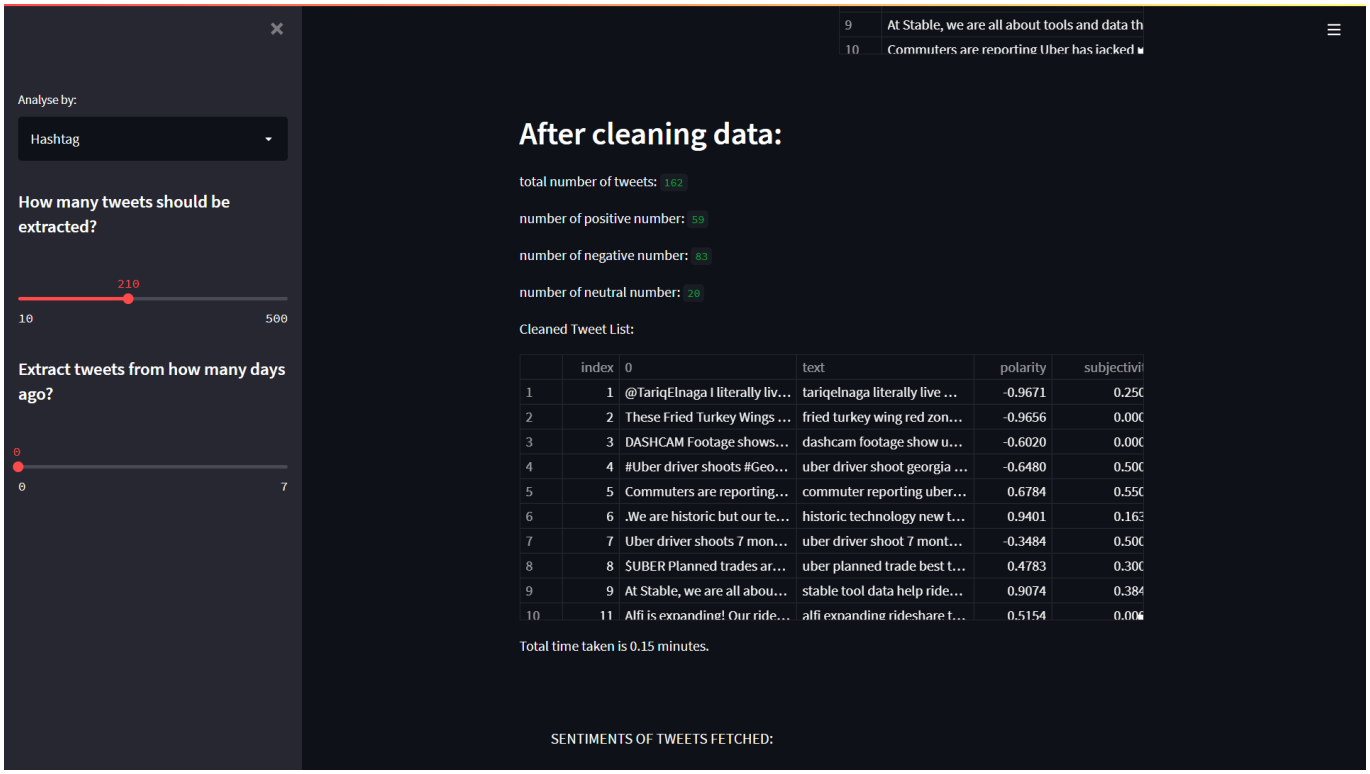


Figure 12: ***Snapshot of Web Application #4***

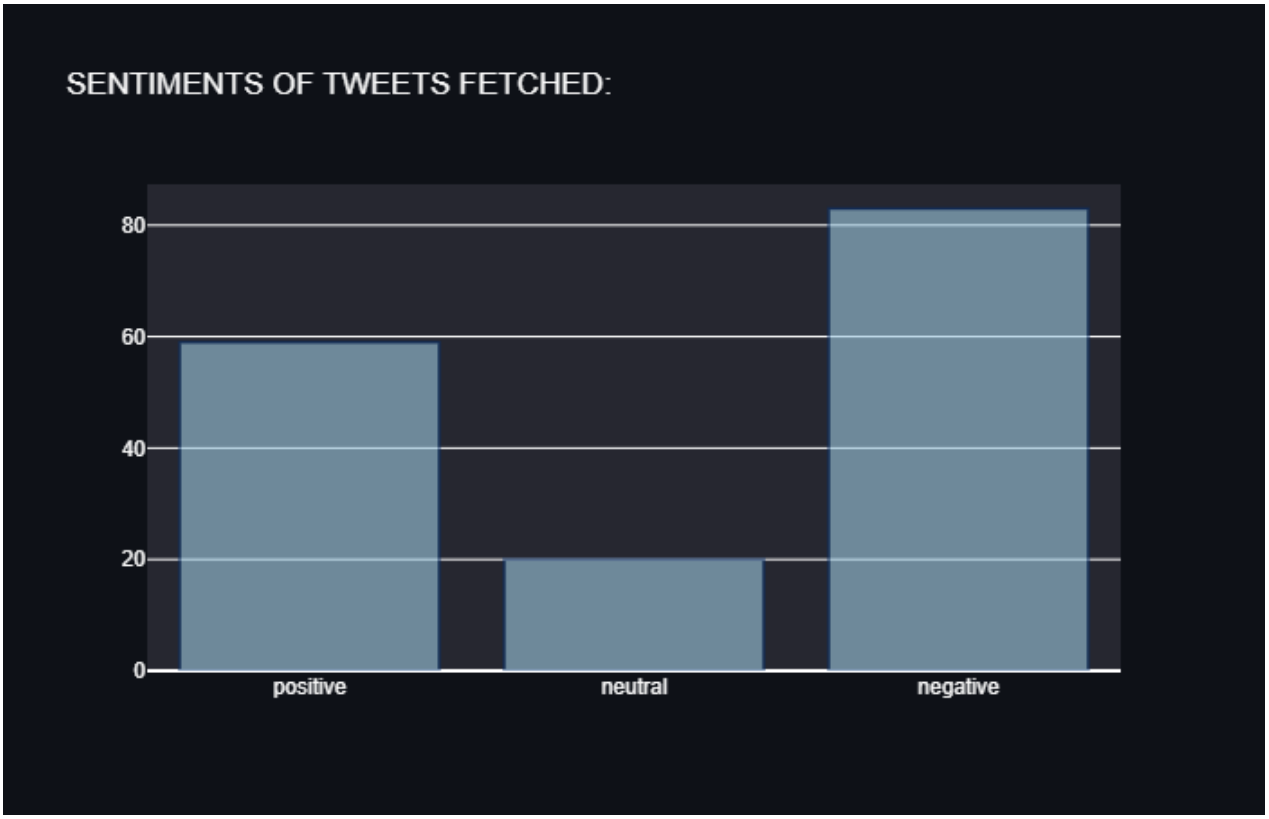


Figure 13: ***Graphical Representation of Tweet Sentiments by Bar Graph***

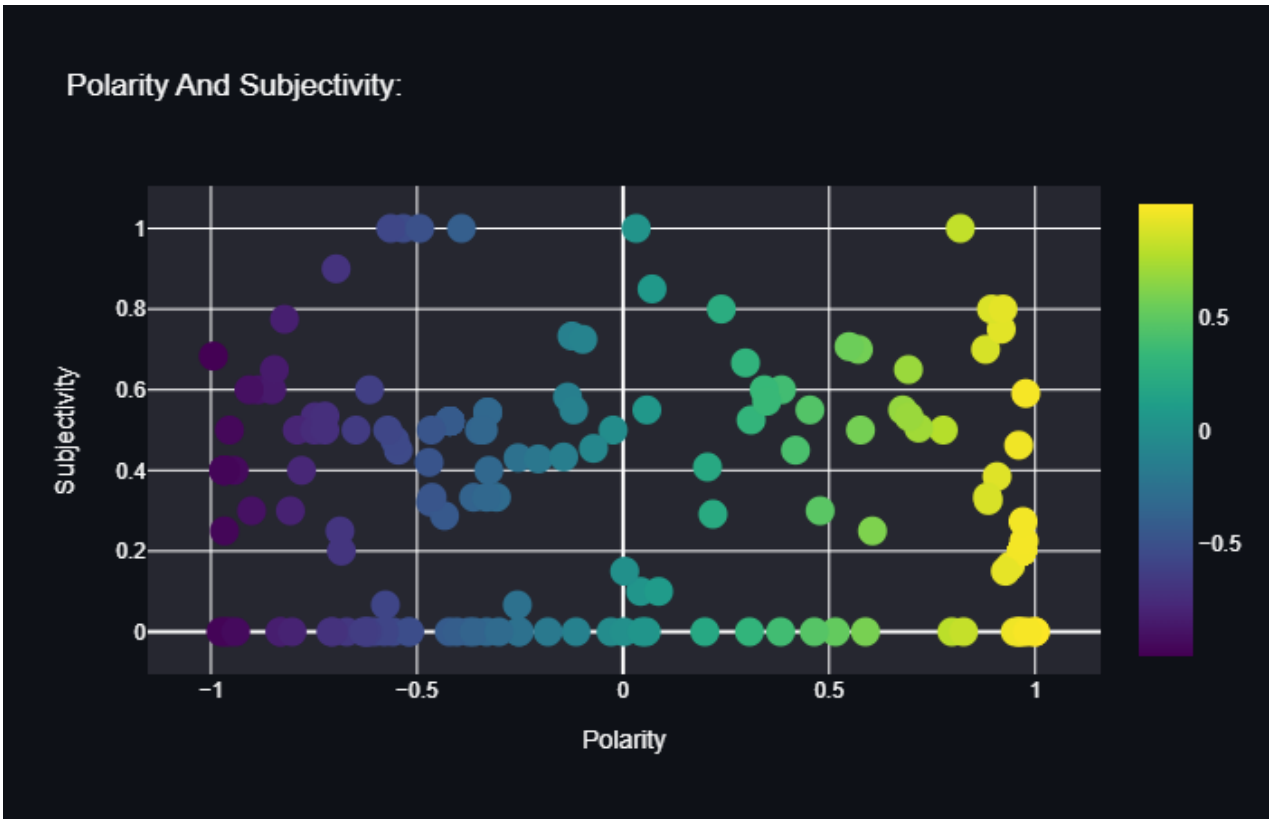


Figure 14: ***Graphical Representation of Tweet Sentiments by Scatter Plot***

Limitations

Sentiment Analysis is a very challenging task. Following are some of thelimitations of Sentiment Analysis of Twitter:

1. Identifying subjective parts of text: Subjective parts represent sentiment bearing content. The same word can be treated as subjective in one case, or an objective in some other. This makes it difficult to identify the subjective portionsof text.

Example:

1. The language of Mr Danny was very crude.
2. Crude oil is obtained by extraction from the sea beds.

The word "crude" is used as an opinion in the first sentence, while it is completely objective in the second sentence.

2. Domain dependence: The same sentence or phrase can have different meanings in different domains.

Example: The word "unpredictable" is positive in the domain of movies, dramas, etc, but if the same word is used in the context of a vehicle steering, then it has a negative opinion.

3. Sarcasm Detection: Sarcastic sentences express negative opinions abouta target using positive words in a unique way.

Example: "Wow that's amazing! Well done!"

The sentence contains only positive words but actually it expresses a negative sentiment.

4. Thwarted expressions: In some sentences only some part of the textdetermines the overall polarity of the document.

Example: "This Movie should have been amazing. It sounds like a great plot, thepopular actors, and the supporting cast is talented as well".

In this case, a simple bag-of-words approach will term it as positive sentiment,but the ultimate sentiment is negative.

5. Explicit Negation of sentiment: Sentiment can be negated in many waysas opposed to using simple no, not, never, etc. It is very challenging to identify such negations. Example: “It avoids all clichés and predictability found inHollywood movies.” Here the words clichés and predictable bear a negative sentiment, but the usage of the word “avoids” negates their respective sentiments.

6. Order dependence: Discourse Structure analysis is essential for Sentiment Analysis/Opinion Mining.
Example: Dan is older than John, conveys the exact opposite opinion from, Johnis older than Dan.

7. Building a classifier for subjective vs. objective tweets: Our current research work focuses mostly on classifying positive, negative and neutral tweetscorrectly. There is a need to look at classifying tweets with sentiment vs. no sentiment closely.

8. Handling comparisons: Bag of words model doesn't handle comparisonsvery well.
Example: "IIT’s are better than most of the private colleges", the tweet would be considered positive for both IIT’s and private colleges using the bag of words model because it doesn't take into account the relation towards "better".

9. Applying sentiment analysis to Facebook messages: There has not beenmuch work on sentiment analysis on Facebook data due to mainly the various restrictions by Facebook graph API and security policies in accessing data.

10. Internationalization: Current research work focuses on mainly English content, but Twitter has many varied users from across the globe with different languages.

Future Scope

Sentiment Analysis, especially in the domain of micro-blogging is still in the developing stage and far from being complete. We propose some ideas which wefeel are worth exploring in the future and may result in improved performance.

1. We may find more reliable sources of information on other social media platforms like Facebook, Instagram, LinkedIn, YouTube, etc. It is necessaryto explore other social media platforms to perform and analyze sentiment analysis results.

2. In our contribution, we chose to use the Naïve Bayes model. But there areother models that may provide interesting results such as lexicon-based algorithms. In this project we tried to show the basic way of classifying.

3. We are focusing on general sentiment analysis. There is potential for research in the field of sentiment analysis with partially known context. For example, users generally use our website to search with specific typesof keywords which can be divided into distinct classes: celebrities, politics/politicians, sports/sportsmen, brands/products, movies/media and music. So, we can attempt to perform separate sentiment analysis on tweetsthat only belong to one of these classes (i.e., the training data would not begeneral but specific to one of these categories) and compare the results withthose that we would get if we apply general sentiment analysis on it instead.

4. One more feature that we feel is worth exploring is whether the information about relative position of words in a tweet has any effect on the performance of the classifier. Although Pang et al. explored a similar feature and reported negative results, they worked on an extremely simplemodel and their results were based on reviews which are very different from tweets.

Conclusion

Therefore, by entering a keyword, we get the visual representation of the tweets analyzed (positive, negative, neutral) as well as the polarity and subjectivity through the web app.

The conducted systematic literature review provides information on studies on sentiment analysis in social media. There are various methods introduced by researchers. Choosing the most efficient method of sentiment analysis depends on the data itself. Both methods demonstrated a similar accuracy. The things thatwe need to consider are the structure of the text, time and amount of data. If thedata structure is messy, there is limited time available to analyze and a small amount of data, lexicon-based methods are more suitable. Whereas, machine learning based methods are more suitable for bigger data as it requires more data and time to train. In order to improve the accuracy and quality of the result, combining both lexicon and machine learning methods would be recommended.We identified the most common type of social media site to extract information from for sentiment analysis, which is Twitter. Most of the reviewed papers use twitter as their social media context. This is due to the accessibility, availability and richness of Twitter content. There are millions of tweets every day on almostany topic. This indicates that social media is becoming a precious source of information.

We demonstrate the application of sentiment analysis in social media through making a web app. Sentiment analysis has a broad application and can be utilizedin different areas such as political forecasting an election result, improving strategy and quality of products in business, monitoring disease outbreak, perception towards a particular topic or person and improving response to a disaster. This shows that sentiment analysis plays a huge role in understanding people's perception and helps in decision making.

REFERENCES

- "tweepy documentation" - https://docs.tweepy.org/en/latest/api.html
- "textblob documentation" - https://textblob.readthedocs.io/en/dev/
- "streamlit documentation"- https://docs.streamlit.io/en/stable/
- https://tealfeed.com/twitter-sentimental-analysis-using-naive-bayes-9eb73
- https://stackabuse.com/sentiment-analysis-in-python-with-textblob/
- https://investigate.ai/investigating-sentiment-analysis/comparing- sentiment analysis-tools/
- https://www.jeremyjordan.me/evaluating-a-machine-learning-model/
- https://towardsdatascience.com/nlp-in-python-data-cleaning-6313a404a470
- https://python.plainenglish.io/scraping-tweets-with-tweepy-python59413046e788
- Alexander Pak and Patrick Paroubek. Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In Proceedings of international conference onLanguage Resources and Evaluation (LREC), 2010.
- B. Pang and L. Lee. "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts" in Proceedings of ACL, 2004.
- Peter D. Turney. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL), 2002.
- Hu, M., & Liu, B. (2004)."Mining and Summarizing Customer Reviews." Proceedings of the 2004 ACM SIGKDD International Conferenceon Knowledge Discovery and Data Mining - KDD '04 (2004).
- Kang, Hanhoon, Seong Joon Yoo, and Dongil Han. "Senti-Lexicon and Improved Naïve Bayes Algorithms for Sentiment Analysis of Restaurant Reviews." Expert Systems with Applications (2012)
- Kim, Soo-Min, and Eduard Hovy. "Determining the Sentiment of Opinions."Proceedings of the 20th International Conference on Computational Linguistics - COLING '04 (2004).
- Agarwal, Apoorv, Owen Rambow, and Nandini Bhardwaj. "Predicting Interests of People on Online Social Networks." 2009 International Conference on Computational Science and Engineering (2009).
- Hassan, Anees Ul, Jamil Hussain, Musarrat Hussain, Muhammad Sadiq, andSungyoung Lee. (2017) "Sentiment Analysis of Social Networking Sites (SNS) Data Using Machine Learning Approach for the Measurement of Depression", in International Conference on Information and Communication Technology Convergence (ICTC), Jeju, South Korea: IEEE.
- Ragini, J. Rexiline, P. M. Rubesh Anand, and Vidhyacharan Bhaskar. (2018) "Big Data Analytics for Disaster Response and Recovery Through Sentiment Analysis." International Journal of Information Management 42: 13-24.
- Twitter REST API <https://dev.twitter.com/docs/api>