A project report on

PREDICTION OF ELECTION RESULTS USING TWITTER DATA

submitted in partial fulfillment of the requirement for the award of

the degree of

Bachelor of Technology in Computer Science and Engineering

By

I. Subbarao	15VV1A0517
B. Aneeth Babu	15VV1A0540
V. Tilak	15VV1A0501
J. Kumari	16VV5A0564
L. Ravin Neha	15VV1A0553

Under the esteemed guidance of

Dr. R. Rajeswara Rao

Professor

Department of Computer Science and Engineering

JNTUK-UCEV

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JNTUK - UNIVERSITY COLLEGE OF ENGINEERING VIZIANAGARAM

VZIANAGARAM – 535003, A.P., INDIA

(2015-2019)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JNTUK - UNIVERSITY COLLEGE OF ENGINEERING VIZIANAGARAM

VZIANAGARAM – 535003, A.P., INDIA

(2015-2019)

CERTIFICATE

This is to certify that the project report entitled **“PREDICTION OF ELECTION OF ELECTION USING TWETTER DATA”** that is being submitted by I. Subbarao (15VV1A0517), B. Aneeth Babu (15VV1A0540), V. Tilak (15VV1A0501), J. Kumari (16VV5A0564) and L. Ravin Neha (15VV1A0553), in the partial fulfillment for the award of Degree of Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING from Jawaharlal Nehru Technological University Kakinada- University College of Engineering Vizianagaram is a record of bonafied work carried out by them in this department.

The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma

Project Supervisor	**Head of the Department**
Dr. R.Rajeswara Rao	**Prof. A. S. N. Chakravarty**
Professor	Professor & Head
Dept. of CSE	Dept. of CSE
JNTUK-UCEV	JNTUK-UCEV

External Examiner

DECLARATION

We, *I. SUBBARAO, B. ANEETH BABU, V. TILAK, J. KUMARI, L. RAVIN NEHA* hereby declare that the project report titled *"PREDICTION OF ELECTION USING TWITTER DATA"* submitted to JNTUK University College of Engineering Vizianagaram, in partial fulfillment of the requirements for the award of the degree of B.Tech. in *COMPUTER SCIENCE AND ENGINEERING* is a record of original and independent research work done by us during the academic year 2018-2019 under the supervision of *Dr. R.RAJESWEA RAO* and it has not formed the basis for the award of any Degree or other similar title to any candidate in any university.

1. 2. 3. 4. 5.

Signature of the Candidates

1. I.V SUBBARAO	(15VV1A0517)
2. B. ANEETH BABU	(15VV1A0540)
3. V.T TILAK	(15VV1A0501)
4. J. KUMARI	(16VV5A0564)
5. L. RAVIN NEHA	(15VV1A0553)

Place:

Date:

ACKNOWLEDGEMENT

It is needed with a great sense of pleasure and immense sense of gratitude that we acknowledge the help of the individuals. We owe many thanks to many people who helped and supported us during the writing of this report.

We take the privilege to express my heartfelt gratitude to project supervisor **Dr. R. RAJESWARA RAO**, Prof., CSE, JNTUK UCEV for his valuable suggestions and constant motivation that greatly helped us in successful completion of the project. We express our sincere thanks to **Prof. A. S. N. CHAKRAVARTHY**, Head, Dept. of Computer Science and Engineering for his continuous support. We express our scincere thanks to **Prof. Ch. srinivasa Rao** Vice-Principal, JNTUK-UCEV for his continuous support. We express our sincere thanks to **Prof. G. SARASWATHI**, Principal, JNTUK UCEV for providing us with a good infrastructure and environment to carry out this project.

We are thankful to all faculty members for extending their kind cooperation and assistance. Finally, we are extremely thankful to our parents and friends for their constant help and moral support.

I. V SUBBA RAO

B.ANEETH BABU

V.T TILAK

J.KUMARI

L.RAVIN NEHA

Place:

Date:

ABSTRACT

Prediction of Indian election is quite difficult through manual survey. This project provides an automated solution for this problem by using sentiment analysis on Twitter. Sentiment analysis refers to determine whether a piece of text contains some subjective information and what it expresses. Understanding the opinions behind user-generated content automatically is of great help for commercial and political use, among others.

In this project, tweet data is collected through **Twitter** developer's account by creating twitter application. **Apache Flume** and **Tweepy** both are used for collection of data. Python used for preprocessing of data. **Naive Bayes algorithm** is used to classify the tweets in the upcoming general Indian elections.

Keywords: Twitter, Flume, Python, Sentiment analysis, Naive Bayes.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES:

1. INTRODUCTION

1.1 PROBLEM STATEMENT :

Prediction of election result is quite difficult through manual survey which requires a lot of man power, need more days to gather the each people's individual opinion and for processing those information and predicting the final result. This will be very hectic through manual survey. This project provides an automated solution for this problem by using sentiment analysis on Twitter. A huge amount of user generated data is present on the web. A lot of people are posting the several tweets regarding their interested topics using the hashtags, mentions, etc. If this content can be extracted and analyzed properly then it can act as a key factor in decision making. But manual extraction and analysis of this content is an impossible task, as the content is unstructured in nature and it is written in natural language. Sentiment Analysis is an extension of Data Mining that extracts and analyzes the unstructured data automatically. The main motive of this is to predict which party is going to win the upcoming elections using Sentiment Analysis and also presents a comparative analysis of various tools used in this area.

1.2 SENTIMENT ANALYSIS:

Sentiment analysis is the automated mining of attitudes, opinions, and emotions from text, speech, and database sources through Natural Language Processing. Sentiment analysis involves classifying opinions in text into categories like "positive" or "negative" or "neutral". It is often referred to as subjectivity analysis, opinion mining, and appraisal extraction. Main fields of research in Sentiment analysis are as follows:

1.2.1 Subjectivity Detection is a task of determining whether text is opinionated or not.

1.2.2 Sentiment Prediction is about predicting the polarity of text whether it is positive or negative.

1.2.3 Aspect Based Sentiment Summarization provides sentiment summary in the form of star ratings or scores of features

1.2.4 Text Summarization generates a few sentences that summarize the reviews of a product.

1.2.5 Contrastive Viewpoint Summarization puts an emphasis on contradicting opinions.

1.2.6 Product Feature Extraction is a task that extracts the product features from its review.

1.2.7 Detecting opinion spam is concern with identifying fake or bogus opinion from reviews.

Sentiment prediction can be done at the document level, sentence level and phrase level.

Document level [1]: analyze the overall sentiment expressed in the text and determine if the overall sentiment is positive or negative.

Sentence level [2]: examine the sentiment expressed in sentences and determines whether each sentence expressed be a positive, negative, or neutral opinion.

Aspect level [3]: aspect level performs better analysis when document level and the sentence level analysis do not find what people liked and did not like. Aspect level also called feature level.

Sentiment analysis is the task of identifying whether the opinion expressed in a text is positive or negative in general, or about a given topic. Sentiment analysis, also refers as opinion mining, is a sub machine learning task where we want to determine which is the general sentiment of a given document.

For example: **"I am so happy today, good morning to everyone**", is a general positive text, and the text**: "Baahubali is such a good movie, highly recommends 10/10"**,expresses positive sentiment towards the movie, named Baahubali, which is considered as the topic of this text.

Sometimes, the task of identifying the exact sentiment is not so clear even for humans, for example in the text: "I'm surprised so many people put Baahubali in their favorite films ever list, I felt it was a good watch but definitely not that good", the sentiment expressed by the author toward the movie is probably positive, but surely not as good as in the message that was mentioned above. In many other cases, identifying the sentiment of a given text is very difficult for an algorithm, even when it looks easy from a human perspective, for example: "if you haven't seen Baahubali, you're not worth my time. If you plan to see it, there's hope for you yet." Nowadays, when the micro-blogging platforms, such as Twitter, are commonly used, the task of sentiment analysis becomes even more interesting. Micro-blogging introduces a new way of communication, where people are forced to use short texts to deliver their messages, hence containing new acronyms, abbreviations, and grammatical mistakes that were generated intentionally. Although there are several known tasks related to Sentiment analysis. Using machine learning techniques and natural language processing we can extract the subjective information of a document and try to classify it according to its polarity such as positive, neutral or negative.

1.3 Applications of sentiment analysis:

Sentiment analysis can be used in diverse fields for various purposes. Some of them are:

1.3.1 Government:

Sentiment analysis helps government in assessing their strength and weaknesses by analyzing opinions from public. For example, “If this is the state, how do you expect truth to come out? The MP who is investigating 2g scam himself is deeply corrupt.” This example clearly shows negative sentiment about government. Whether it is tracking citizens’ opinions on a new 108 system, identifying strengths and weaknesses in a recruitment campaign in government job, assessing success of electronic submission of tax returns, or many other areas, we can see the potential for sentiment analysis.

1.3.2 Online Commerce:

The most general use of sentiment analysis is in ecommerce activities. Websites allows their users to submit their experience about shopping and product qualities. They provide summary for the product and different features of the product by assigning ratings or scores. Customers can easily view opinions and recommendation information on whole product as well as specific product features. Graphical summary of the overall product and its features is presented to users. Popular merchant websites like amazon.com provides review from editors and also from customers with rating information.

1.3.3 Voice of the Market (VOM):

Voice of the Market is about determining what customers are feeling about products or services of competitors. Accurate and timely information from the Voice of the Market helps in gaining competitive advantage and new product development. Detection of such information as early as possible helps in direct and target key marketing campaigns. Sentiment Analysis helps corporate to get customer opinion in real-time. This real-time information helps them to design new marketing strategies, improve product features and can predict chances of product failure.

1.3.4 Voice of the Customer (VOC):

Voice of the Customer is concern about what individual customer is saying about products or services. It means analyzing the reviews and feedback of the customers. VOC is a key element of Customer Experience Management. VOC helps in identifying new opportunities for product inventions. Extracting customer opinions also helps identify functional requirements of the products and some non-functional requirements like performance and cost.

1.3.5 Brand Reputation Management:

Brand Reputation Management is concern about managing your reputation in market. Customers or any other parties can damage or enhance your reputation. Brand Reputation Management (BRM) is a product and company focused rather than customer. Now, one-to-many conversations are taking place online at a high rate. That creates opportunities for organizations to manage and strengthen brand reputation. Sentiment analysis helps in determining how company's brand, product or service is being perceived by community online.

1.3.6 Social Media Monitoring:

Now social Data are increases very fast, in every area social data play an important role in every angle. Every day huge amount of data had been generated from several social media websites like Twitter, Facebook, Instagram, etc. Sentiment analysis on the social media data helps the people to know about the trends and the predictions about the events that are going to happen and how much the people showing interest towards some future events from the Twitter, Facebook. By using sentiment analysis on social media, we can get incredible insights into the quality of conversation that's happening around a brand.

1.4 Sentiment analysis advantages

- Sentiment analysis is a useful tool for any organization or group for which public sentiment or attitude towards them is important for their success - whichever way that success is defined.
- On social media, blogs, and online forums millions of people are busily discussing and reviewing businesses, companies, and organizations. And those opinions are being 'listened to' and analyzed.
- Those being discussed are making use of this enormous amount of data by using computer programs that don't just locate all mentions of their products, services, or business, but also determine the emotions and attitudes behind the words being used.
- The results from sentiment analysis help businesses understand the conversations and discussions taking place about them, and helps them react and take action accordingly.
- They can quickly identify any negative sentiments being expressed, and turn poor customer experiences into very good ones.
- They can create better products and services, and they can formulate the marketing messages they send out according to the sentiments being expressed by their target audience or customers.

- By listening to and analyzing comments on Facebook and Twitter, local government departments can gauge public sentiment towards their department and the services they provide, and use the results to improve services such as parking and leisure facilities, local policing, and the condition of roads.
- Universities can use sentiment analysis to analyze student feedback and comments garnered either from their own surveys, or from online sources such as social media. They can then use the results to identify and address any areas of student dissatisfaction, as well as identify and build on those areas where students are expressing positive sentiments.
- And by analyzing the sentiment behind customer reviews on sites like TripAdvisor and help, hotels and restaurants can not only manage their reputations by improving the services offered, but can also gauge the general customer attitude to their business or brand.

1.5 Disadvantages:

- Sentiment analysis tools can identify and analyse many pieces of text automatically and quickly.
- But computer programs have problems recognizing things like sarcasm and irony, negations, jokes, and exaggerations - the sorts of things a person would have little trouble identifying. And failing to recognize these can skew the results.
- 'Disappointed' may be classified as a negative word for the purposes of sentiment analysis, but within the phrase “I wasn't disappointed", it should be classified as positive.
- We would find it easy to recognize as sarcasm the statement "I'm really loving the enormous pool at my hotel!", if this statement is accompanied by a photo of a tiny swimming pool; whereas an automated sentiment analysis tool probably would not, and would most likely classify it as an example of positive sentiment.
- With short sentences and pieces of text, for example like those you find on Twitter especially, and sometimes on Facebook, there might not be enough context for a reliable sentiment analysis. However, in general, Twitter has a reputation for being a good source of information for sentiment analysis, and with the new increased word count for tweets it's likely it will become even more useful. So, automated sentiment analysis tools do a really great job of analyzing text for opinion and attitude, but they're not perfect.

2. LITERATURE SURVEY

2.1 A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text (2010)

Author: C.J. Hutto, Eric Gilbert

Sentiment analysis, or opinion mining, is an active area of study in the field of natural language processing that analyzes people's opinions, sentiments, evaluations, attitudes, and emotions via the computational treatment of subjectivity in text. It is not our intention to review the entire body of literature concerning sentiment analysis. Indeed, such an endeavor would not be possible within the limited space available. This paper describes the development, validation, and evaluation of VADER (for Valence Aware Dictionary for sentiment Reasoning). The inherent nature of social media content poses serious challenges to practical applications of sentiment analysis. This paper presents VADER, a simple rule-based model for general sentiment analysis, and compare its effectiveness with typical state-of-practice benchmarks including SentiWordNet, and machine learning oriented techniques relying on Naive Bayes, Maximum Entropy, and Support Vector Machine (SVM) algorithms. Using a combination of qualitative and quantitative methods, here, they first construct and empirically validate a gold standard list of lexical features (along with their associated sentiment intensity measures) which are specifically attuned to sentiment in microblog-like contexts. Then combine these lexical features with consideration for five general rules that embody grammatical and syntactical conventions for expressing and emphasizing sentiment intensity. Interestingly, using our parsimonious rule-based model to assess the sentiment of tweets, they find that VADER outperforms individual human raters (F1 Classification Accuracy = 0.96 and 0.84, respectively), and generalizes more favorably across contexts than any of their benchmarks.

2.2 Sentiment Analysis for the News Data Based on the social Media (2011)

Author: Firoj Fattulal Shahare

Now social Data are increases very fast, in every area social data play an important role in every angle, social media big data mining area welcomed by researchers from both government, academic and industry. A computing sentiment of news data is a significant component of the social media big data. The computing sentiment of news information may be a major factor of the social media massive information. In current web word range of user use social media and social network to browse and read news connected information.

Everyday range of issue area unit occurring and social media influence the news associated with this news. Existing sentiment computing ways area unit primarily supported the feeling wordbook or supervised ways, that aren't climbable to the social media massive information. As a result of once bid information suggests that information size increases this methodology result on potency. Therefore they tend to propose a replacement methodology to try and do the sentiment analysis for news data a lot of specially, supported the social media information and social news (i.e. Text and emotions text) of a happening .The word feeling computation algorithmic rule is planned to get the beginning word feeling that area unit more refined through the quality emotion wordbook. With the word emotions in hand, we are able to reason each sentence sentiments. The proposed method uses Naïve Bayes to determine the emotion into different categories from given social media news data. This method provides the excellent performance for real time news data on social media and also provides the better result in terms of accuracy.

2.3 Sentiment Analysis on Facebook Group Using Lexicon Based Approach (2013)

Author: Sanjida Akter, Muhammad Tareq Aziz

Internet is one of the primary sources of Big Data. Rise of the social networking platforms are creating enormous amount of data in every second where human emotions are constantly expressed in real-time. The sentiment behind each post, comments, likes can be found using opinion mining. It is possible to determine business values from these objects and events if sentiment analysis is done on the huge amount of data. Here, we have chosen FOODBANK which is a very popular Facebook group in Bangladesh; to analyze sentiment of the data to find out their market values. Sentiment analysis is the concept: “how a person reacts on something or some events”. It is always an important issue “what people think”. People’s thoughts are available in social Medias to be used as data source of market basket analysis. Social media plays an important role here because we can have data from people even we don’t know. Such analysis can help to determine recent market trend or market value of a particular product or event. Based on people’s interest. Here we have utilized FACEBOOK which is a very popular social media platform as our data source. For opinion mining or sentiment analysis some methods are applied like – Naive Bayes Machine Learning Classifier, SentiWordNet, Support Vector Machine. Here we applied both machine learning approach and Lexicon Based Dictionary. Most of the work on opinion mining is done with Naïve Bayes. As our data set contains Bangla language; we built a Dictionary on top of the data set. We count the occurrence of sentimental words and featured words which are tagged as the target of calculated sentiment value. This approach can find

73% cases correctly whether post is positive or negative. After comparing two results, we found that lexicon based algorithm works better here.

2.4 Statistical and Sentiment Analysis of Consumer Product Review (2014)

Author: Zeenia Singla1, Sukhchandan Randhawa2, and Sushma Jain3

Big Data commerce has given a big leap to ecommerce. It has opened up the avenues to smarter and informed decision making for large industries as well as the consumers. Online reviews on e-commerce giants like Amazon, Flipkart are one such paradigm which can be used to arrive at more profitable decisions. They are not only beneficial for the consumers but also for the product manufacturers. Online reviews have the potential to provide an insight to the buyers about the product like its quality, performance and recommendations; thereby providing a clear picture of the product to the future buyers. The usefulness of online reviews for manufacturers to realize customer requirements by analyzing helpful reviews is one such unrealized potential. Both positive and negative reviews play a big role in determining the customer requirements and extracting consumer‘s feedback about the product faster. Sentiment Analysis is a computational study to extract subjective information from the text. In this research, data analysis of a large set of online reviews for mobile phones is conducted. We have not only classified the text into positive and negative sentiment but have also included sentiments of anger, anticipation, disgust, fear, joy, sadness, surprise and trust. This delineated classification of reviews is helpful to evaluate the product holistically, enabling better-decision making for consumers.

2.5 Prediction of Election Using Twitter Sentiment Analysis (2016)

Author: Pritee Salunkhe, Avinash Surnar, Sunil Sonawane

Twitter is a micro-blogging service that provides short messages on a daily basis for events, products, entity etc. Now a day's researchers dealing with utilizing Twitter to monitor people reactions in political activities like debates and campaigns. On the basis of that prediction an election can be made. Analysis of the prediction of election results using messages of either political parties or politician. The use of tweet content considered as a valid indicator of political sentiment. Sentiment analysis is used for analysis as well as predicting the emotions from the text patterns regarding political issues, products, entity, election etc. Sentiment analysis consists an analyzing texts to extract information. Basic sentiment analysis allows to determine or measuring the polarity (negative or positive) of sentiment.

2.6 Motivation for this Project:

We surveyed all the above papers, we came to know different information about sentiment analysis using different data sources like Amazon, Facebook, Movie reviews, e.t.c. In one paper they used the sample static data from Facebook and they had done the sentiment analysis on that in order to identify the public review on an event. But this review may change based on the specific time. So, we decided to stream the real time data from the twitter.one paper, they used the Hindi twitter in order to predict the future winner in election. Then what we decided to do is sentiment analysis project on English Twitter. Another paper gave us the comparison regarding different classification algorithms for sentiment analysis. If we are using a small or medium data then one particular algorithm may work better than the other but here we are dealing with the huge amount of data. So we used Naive Bayes classification algorithm for sentiment analysis on the streamed data from Twitter. Along with the Naive Bayes algorithm, Lexicon based approach also gives the better result than the other supervised learning algorithms. So, here in this project, we extracted the real time data from the twitter and applying naive Bayes and lexicon based approaches to assign the polarity for each tweet. And also if we implemented and deploy it, this application can be used in different places like by political bodies, e.t.c.

3. SENTIMENT ANALYSIS APPROACHES

3.1 Unsupervised Learning :

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data. Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore machine is restricted to find the hidden structure in unlabeled data by our-self.

Unsupervised learning classified into two categories of algorithms:

3.1.1 Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

3.1.2Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

3.2 Supervised Learning :

Supervised learning as the name indicates a presence of supervisor as teacher. Basically supervised learning is a learning in which we teach or train the machine using data which is well labeled that means some data is already tagged with correct answer. After that, machine is provided with new set of examples (data) so that supervised learning algorithm analyses the training data (set of training examples) and produces a correct outcome from labeled data. **For instance**, suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all different fruits one by one like this:

If shape of object is rounded and depression at top having color **Red** then it will be labelled as –**Apple**.

If shape of object is long curving cylinder having color **Green-Yellow** then it will be labelled as –**Banana**.

Since machine has already learnt the things from previous data and this time have to use it wisely. It will first classify the fruit with its shape and color, and would confirm the fruit name as BANANA and put it in Banana category. Thus machine learns the things from training data (basket containing fruits) and then apply the knowledge to test data (new fruit).

Supervised learning classified into two categories of algorithms:

3.3 Regression Analysis:

Regression analysis is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a dependent_variable and one or more independent_variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed.

Regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine_learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer causal_relationships between the independent and dependent variables. However this can lead to illusions or false relationships, so caution is advisable.

3.4 Classification:

A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”. A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes. For example, when filtering emails “spam” or “not spam”, when looking at transaction data, “fraudulent”, or “authorized”. In short Classification either predicts categorical class labels or classifies data (construct a model) based on the training set and the values (class labels) in classifying attributes and uses it in classifying new data. There are a number of classification models. Classification models include logistic regression, decision tree, random forest, gradient-boosted tree, multilayer perceptron, one-vs-rest, and Naive Bayes.

3.5 Decision Trees:

A Decision tree is a classification scheme which generates a tree and a set of rules, representing the model of different classes from a dataset. Decision Tree learning uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modelling approaches used in statistics, data mining and machine learning. Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees.

3.5.1 Advantages:

- Decision Trees are able to generate understandable rules.
- They are able to handle both numerical and categories attributes.
- They provide a clear indication of which fields are most important for prediction or classification.

3.5.2 Disadvantages:

- Some Decision Trees can only deal with binary valued target classes, others are able to assign records to an arbitrary number of classes, but are error prone when the number of training examples per class gets small. This can happen rather quickly in a tree with many levels and many branches per node.
- Decision trees suffer from overfitting.

3.6 SUPPORT VECTOR MACHINE:

Support vector machine is a supervised machine learning algorithm which can be used for both classification and regression challenges. The advantage of SVM is that they can make use of certain kernels in order to transform the problem such that we can apply linear classification techniques to nonlinear data. SVM gives better results when compared to perceptron learning, if the data is not linearly separable. A main advantage of SVM classification is that SVM performs well on datasets that have many attributes, even when there are only a few cases that are available for the training process. It is the algorithm of choice for challenging high dimensional data. It is a preferred algorithm for sparse data. SVM uses maximum margin classifiers to decrease the misclassification rate. SVM can control both the model quality and performance.

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification, implicitly mapping their inputs into high-dimensional feature spaces.

3.6.1 Advantages:

- **SVMs** are effective when the number of features is quite large. It works effectively even if the number of features are greater than the number of samples. Non-Linear data can also be **classified using** customized hyperplanes built by **using** kernel trick.

- Unlike in neural networks, SVM is not solved for local optima.
- It scales relatively well to high dimensional data.
- SVM models have generalization in practice, the risk of overfitting is less in SVM.

3.6.2 Disadvantages:

- Choosing a "good" kernel function is not easy.
- Long training time for large datasets.
- Difficult to understand and interpret the final model, variable weights and individual impact.
- Since the final model is not so easy to see, we cannot do small calibrations to the model hence it's tough to incorporate our business logic.

3.7 NAIVE BAYES ALGORITHM

It is classification technique based on Bayes Theorem with an assumption of Independence among predictors. In simple terms, a Naive Bayes classifier assumes that presence of a particular features in a class is unrelated to the presence of any other features. For example, a fruit may be considered to be an apple if it is red, round, and 3 inches in diameter. Even if these features depends on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P (c|x) from P(c), P(x) and P (x|c).

$$P(c|x) = P(x|c)\, P(c)/P(x)$$

$$P(c|x) = P(x1|c) * P(x2|c) * \ldots\ldots * P(xn|c) * P(c)$$

P (c|x) is posterior probability of given predictor (x, attribute).

P(c) is the prior probability of class.

P (x|c) is the likelihood which is the probability of predictors given class.

P(x) is the prior probability of predictor.

3.7.1 Applications of Naive Bayes Algorithms:

3.7.1.1 Real time Prediction:

Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.

3.7.1.2 Multi class Prediction:

This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.

3.7.1.3 Text classification/ Spam Filtering/ Sentiment Analysis:

Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a results, it is widely used in spam filtering (Identify spam e-mail) and sentiment analysis (in social media analysis, to identify positive and negative customer sentiments).

3.7.1.4 Recommendation System:

Naive Bayes Classifier and Collaborative Filtering together build a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not.

3.7.2 Advantages of Naïve Bayes Algorithm:

- Very simple, easy to implement and fast.
- Highly scalable. It scales linearly with the number of predictors and data points.
- Can be used for both binary and multi-class classification problems.
- Can make probabilistic predictions.
- Handles continuous and discrete data.
- Not sensitive to irrelevant features.

3.7.3 Example for text classification:

The training set consist of 10 Positive Reviews and 10 Negative Reviews and considered word count are as fellow

Table 1: Examples for text classification

Positive Reviews Database	Negative Reviews Database
I=2	I=1
loved=2	hated=1
The=2	The=1
Movie=1	Movie=1
Poor=0	Poor=1
Acting=1	Acting=1

Given test as "I hated the poor acting" Find the sentiment for the given test set. It consists of the following information

Positive reviews=2

Negative reviews=2

Total no of Reviews= positive reviews +negative reviews=4

3.7.4 Prior Probability:

Table 1.1: The prior probability for reviews

The prior probability for the positive reviews	P(positive)=2/4=0.5
The prior probability for the negative reviews	P(negative)=2/4=0.5

3.7.5 Conditional Probability:

The conditional probability is the probability that a random variable will take on a particular value given that outcome for another random variable

3.7.6 Formulae:-

$$\mathrm{P}(w_k|+) = \frac{n_k + 1}{n + |vocabulary|}$$

nk =count of word wk in particular class

n=total no of words in particular class

|Vocabulary|=unique words in both the classes

The conditional probability for the word 'I' in positive review is

P (I/positive)=3/15=0.2

The conditional probability for the word 'LOVE' in positive review is

P (Loved/positive)=3/15=0.2

The conditional probability for the word 'HATED' in positive review is

P (Hated/positive)=1/15=0.066

The conditional probability for the word 'THE' in positive review is

P (The/positive)=3/15=0.2

The conditional probability for the word 'MOVIE' in positive review is

P (Movie/positive)=2/15=0.133

The conditional probability for the word 'POOR' in positive review is

P (Poor/positive)=1/15=0.066

The conditional probability for the word 'ACTING' in positive review is

P (Acting/positive)=2/15=0.133

The conditional probability for the word 'I' in negative review is

P (I/negative)=2/13=0.153

The conditional probability for the word 'LOVED' in negative review is

P (Loved/negative)=1/13=0.066

The conditional probability for the word 'HATED' in negative review is

P (Hated/negative)=2/13=0.153

The conditional probability for the word 'THE' in negative review is

P (The/negative)=2/13=0.153

The conditional probability for the word 'MOVIE' negative review is

P (Movie/negative)=2/13=0.153

The conditional probability for the word 'POOR' in negative review is

P (Poor/negative)=2/13=0.153

The conditional probability for the word 'ACTING' in negative review is

P (Acting/negative)=2/13=0.153

3.7.7 Posterior Probability:

The posterior probability is the product of prior probability and conditional probabilities

Posterior probability=prior probability*conditional probability

$$V_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_{w \in words} P(w|v_j)$$

The posterior probability for positive review and negative review is

P (positive) =0.5*0.2*0.066*0.2*0.066*0.133=0.0000115

P (negative) =0.5*0.153*0.153*0.153*0.153*0.153=0.0000419

The posterior probability for the positive reviews is less than the posterior probability of the negative review

P (positive) < P (negative)

The given test set "I HATED THE POOR ACTING" is predicted by naive Bayes as a negative sentiment.

4. SOFTWARE REQUIREMENT ANALYSIS

This chapter presents the first step taken to create the proposed system,which is focused on understanding what it is expected from the system, who are the stakeholders and how the user is meant to interact with the delivered services. To support this process, a list of requirements and a use case diagram are provided.

4.1 Requirements Elicitation

It is very important to grasp the scope of the system, what is the core functionality and what represents a ’nice to have’ feature. The elicitation step involves gathering clear and precise requirements, in order to model the system and its characteristics, a process that can be very complex in software development. Due to the fact that this project is mainly focused on research and less on providing a user oriented tool, it only uses the main techniques for analysing the system’s requirements.

4.2 Functional and non-functional requirements

When collecting and analysing the requirements of a software, there are two aspects that need to be considered. The functional one refers to the features that the system needs to deliver, while the non-functional aspect takes into account constraints and how the system should behave.

Functional Requirements are:

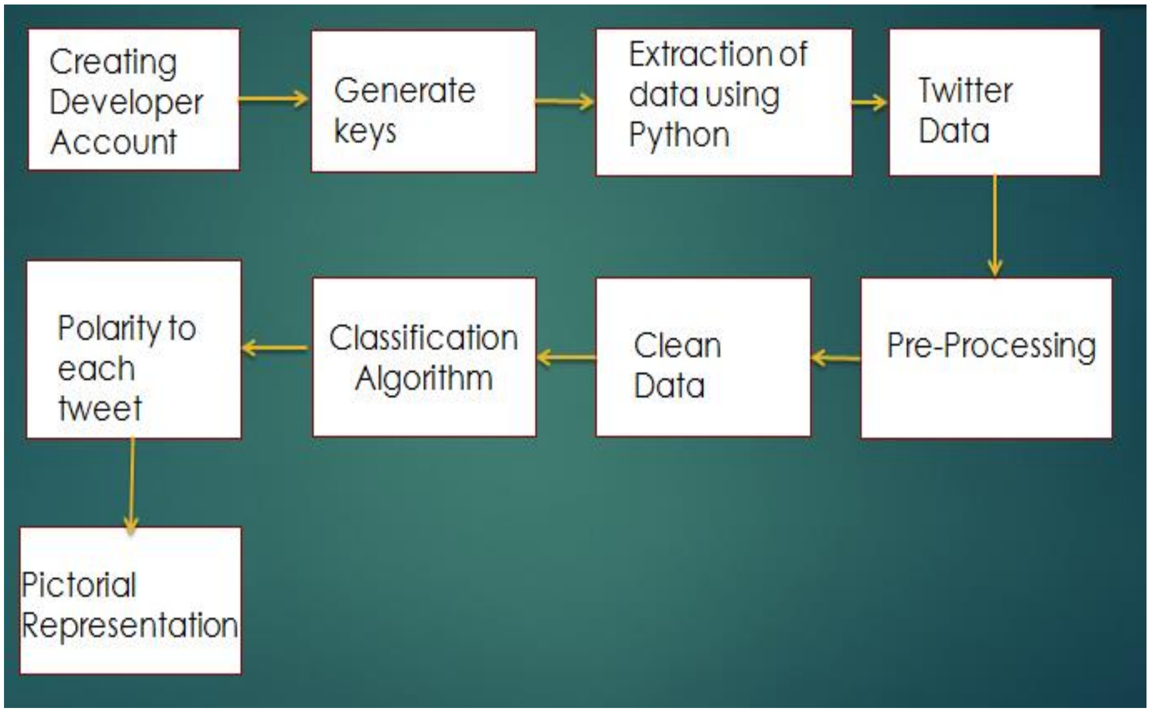
- The System should include a Naïve Bayes classification Algorithm
- The system should deliver the trained classifier
- The system should be able to handle the huge amount of data
- The system should allow user to extract and preprocess and handle the data.

Non Functional Requirements are:

- The System must be implemented in python(programming language used)
- The system should produce graphs showing performance.
- The system is compatible to handle both training and testing dataset.

5 METHODOLOGY

The methodology of the proposed work is shown in the figure:



The overall methodology of the work is divided into six phases:

Phase-1: Creating the twitter developer account for creating a twitter application.

Phase-2: After creating the twitter application, generate the authentication keys.

Phase-3: Using those keys, extract the real time twitter data.

Phase-4: Once the data had been extracted, preprocess the data.

Phase-5: Classify the tweets using classification algorithm

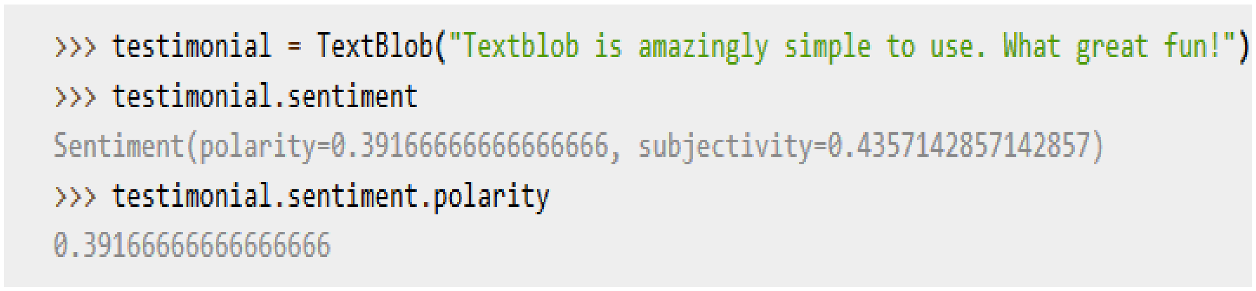
Phase-6: After calculating all the tweet polarities, plot the results in the form of graph.

6. TOOLS FOR SENTIMENT ANALYSIS

6.1 TextBlob Tool:

TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.. A good thing about TextBlob is that they are just like python strings. So, you can transform and play with it same like we did in python. TextBlob aims to provide access to common text-processing operations through a familiar interface. You can treat TextBlob objects as if they were Python strings that learned how to do Natural Language Processing.

Ex:



```
>>> testimonial = TextBlob("Textblob is amazingly simple to use. What great fun!")
>>> testimonial.sentiment
Sentiment(polarity=0.39166666666666666, subjectivity=0.4357142857142857)
>>> testimonial.sentiment.polarity
0.39166666666666666
```

6.1.1 Pros:

- Since, it is built on the shoulders of NLTK and Pattern, therefore making it simple for beginners by providing an intuitive interface to NLTK.
- It provides language translation and detection which is powered by Google Translate (not provided with Spacy).

6.1.2 Cons:

- It is little slower in the comparison to spacy but faster than NLTK. (Spacy > TextBlob > NLTK)
- It does not provide features like dependency parsing, word vectors etc. which is provided by spacy.

6.2 VADER Sentiment Analysis

VADER (Valence Aware Dictionary and Sentiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is *specifically attuned to sentiments expressed in social media*. VADER uses a combination of A sentiment lexicon is a list of lexical features (e.g., words) which are generally labelled according to their semantic orientation as either positive or negative.

VADER has been found to be quite successful when dealing with social media texts, NY Times editorials, movie reviews, and product reviews. This is because VADER not only tells about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is.

It is fully open-sourced under the MIT License. The developers of VADER have used Amazon's Mechanical Turk to get most of their ratings, You can find complete details on their Github Page.

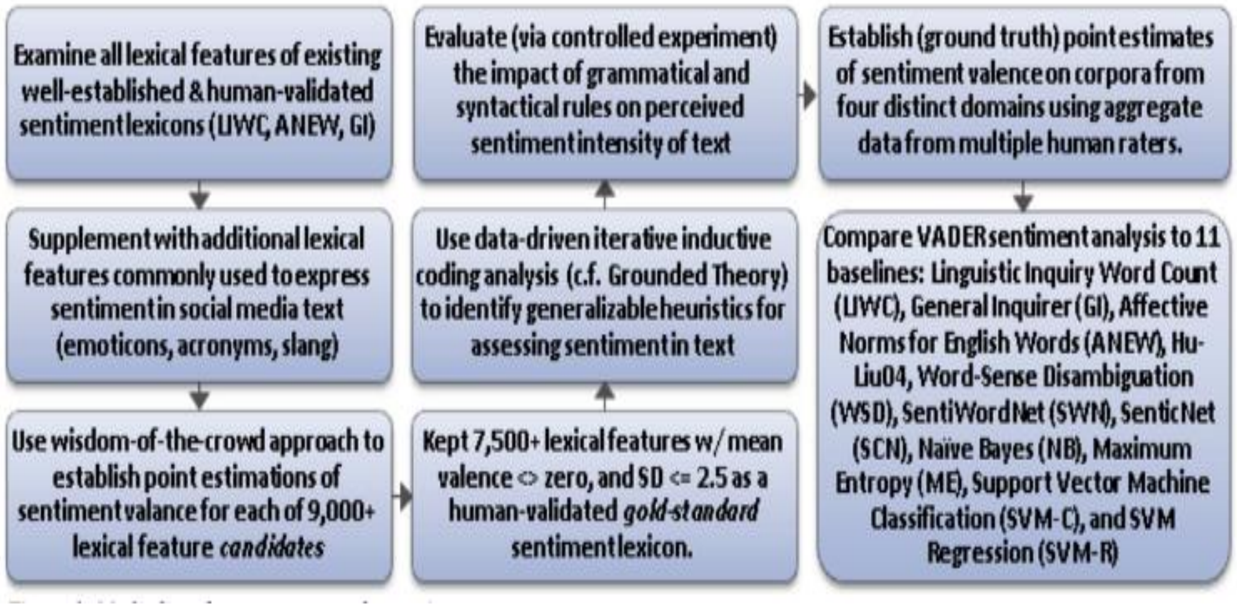

Examine all lexical features of existing well-established & human-validated sentiment lexicons (LIWC, ANEW, GI)
Supplement with additional lexical features commonly used to express sentiment in social media text (emoticons, acronyms, slang)
Use wisdom-of-the-crowd approach to establish point estimations of sentiment valance for each of 9,000+ lexical feature candidates
Kept 7,500+ lexical features w/ mean valence <> zero, and SD <= 2.5 as a human-validated gold-standard sentiment lexicon.
Use data-driven iterative inductive coding analysis (c.f. Grounded Theory) to identify generalizable heuristics for assessing sentiment in text
Evaluate (via controlled experiment) the impact of grammatical and syntactical rules on perceived sentiment intensity of text
Establish (ground truth) point estimates of sentiment valence on corpora from four distinct domains using aggregate data from multiple human raters.
Compare VADER sentiment analysis to 11 baselines: Linguistic Inquiry Word Count (LIWC), General Inquirer (GI), Affective Norms for English Words (ANEW), Hu-Liu04, Word-Sense Disambiguation (WSD), SentiWordNet (SWN), SenticNet (SCN), Naïve Bayes (NB), Maximum Entropy (ME), Support Vector Machine Classification (SVM-C), and SVM Regression (SVM-R)

Fig. 1: Methods and process approach overview

6.2.1 Advantages of using VADER:

- VADER has a lot of advantages over traditional methods of Sentiment Analysis, including:
- It works exceedingly well on social media type text, yet readily generalizes to multiple domains
- It doesn't require any training data but is constructed from a generalizable, valence-based, human-curated gold standard sentiment lexicon
- It is fast enough to be used online with streaming data, and
- It does not severely suffer from a speed-performance tradeoff.
- The source of this article is a very easy to read paper published by the creators of VADER library. You can read the paper here.
- Enough of talking. Let us now see practically how VADER analysis work for which we will have installs the library first.

6.2.2 Working & Scoring:

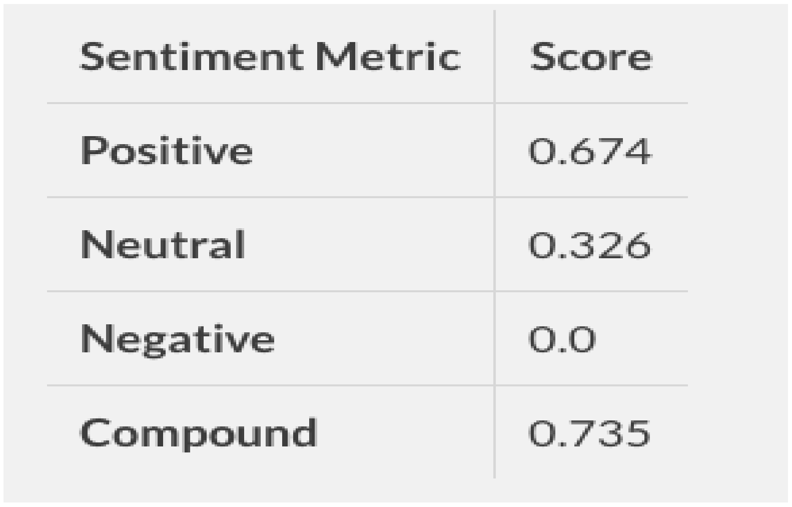
Let us test our first sentiment using VADER now. We will use the polarity scores () method to obtain the polarity indices for the given sentence.

```
def sentiment_analyzer_scores(sentence):
        score = analyser.polarity_scores(sentence)
        print ("{:-<40} {}".format(sentence, str (score)))
```

6.2.3 Let us check how VADER performs on a given review:

sentiment_analyzer_scores("The phone is super cool.")

Table 2 : Polarity valued Sentiment score

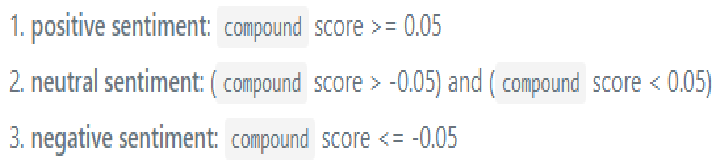


Sentiment Metric	Score
Positive	0.674
Neutral	0.326
Negative	0.0
Compound	0.735

6.3 The results obtained are:

The Positive, Negative and Neutral scores represent the proportion of text that falls in these categories. This means our sentence was rated as 67% Positive, 33% Neutral and 0% Negative. Hence all these should add up to 1.

The Compound score is a metric that calculates the sum of all the lexicon ratings which have been normalized between -1(most extreme negative) and +1 (most extreme positive). In the case above, lexicon ratings for and supercool are 2.9 and respectively 1.3. The compound score turns out to be 0.75 , denoting a very high positive sentiment.

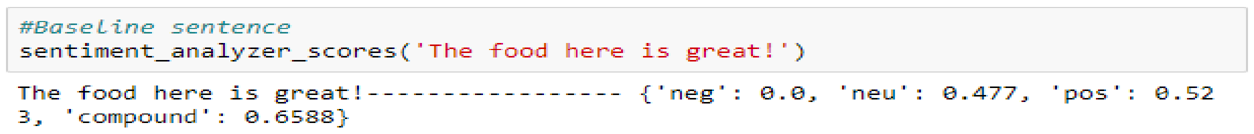


1. positive sentiment: `compound` score >= 0.05
2. neutral sentiment: (`compound` score > -0.05) and (`compound` score < 0.05)
3. negative sentiment: `compound` score <= -0.05

6.4 Advantages over other tools:

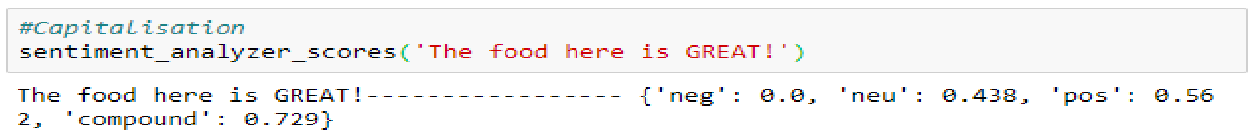
VADER analyses sentiments primarily based on certain key points:

6.4.1 Capitalization: Using upper case letters to emphasize a sentiment-relevant word in the presence of other non-capitalized words, increases the magnitude of the sentiment intensity. For example, "The food here is GREAT!" conveys more intensity than "The food here is great!"



```
#Baseline sentence
sentiment_analyzer_scores('The food here is great!')
```

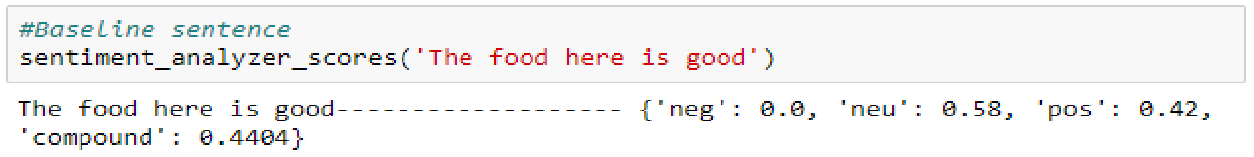
```
The food here is great!----------------- {'neg': 0.0, 'neu': 0.477, 'pos': 0.52
3, 'compound': 0.6588}
```



```
#Capitalisation
sentiment_analyzer_scores('The food here is GREAT!')
```

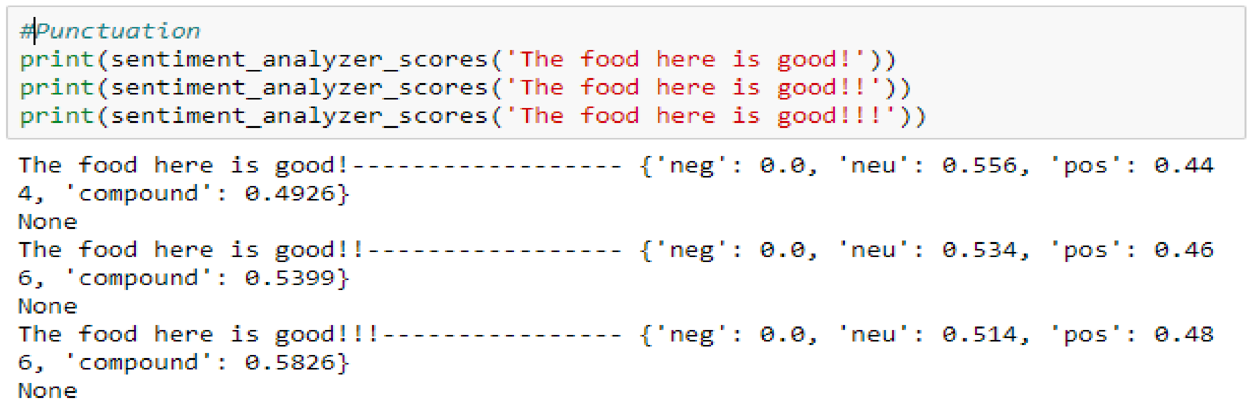
```
The food here is GREAT!----------------- {'neg': 0.0, 'neu': 0.438, 'pos': 0.56
2, 'compound': 0.729}
```

6.4.2 Punctuation: The use of an exclamation mark(!), increases the magnitude of the intensity without modifying the semantic orientation. For example, "The food here is good!" is more intense than "The food here is good." and an increase in the number of (!), increases the magnitude accordingly.



```
#Baseline sentence
sentiment_analyzer_scores('The food here is good')
```

```
The food here is good------------------- {'neg': 0.0, 'neu': 0.58, 'pos': 0.42,
'compound': 0.4404}
```



```
#Punctuation
print(sentiment_analyzer_scores('The food here is good!'))
print(sentiment_analyzer_scores('The food here is good!!'))
print(sentiment_analyzer_scores('The food here is good!!!'))
```

```
The food here is good!------------------ {'neg': 0.0, 'neu': 0.556, 'pos': 0.44
4, 'compound': 0.4926}
None
The food here is good!!----------------- {'neg': 0.0, 'neu': 0.534, 'pos': 0.46
6, 'compound': 0.5399}
None
The food here is good!!!---------------- {'neg': 0.0, 'neu': 0.514, 'pos': 0.48
6, 'compound': 0.5826}
None
```

See how the overall compound score is increasing with the increase in exclamation marks.

6.4.3 Emojis

print(sentiment_analyzer_scores ('I am 😄 today'))

print(sentiment_analyzer_scores('😊'))

print(sentiment_analyzer_scores('😥'))

print(sentiment_analyzer_scores('☹□'))

6.4.4 Output

I am 😄 today---------------------------- {'neg': 0.0, 'neu': 0.476, 'pos': 0.524, 'compound': 0.6705}

😊--- {'neg': 0.0, 'neu': 0.333, 'pos': 0.667, 'compound': 0.7184}

😥--- {'neg': 0.275, 'neu': 0.268, 'pos': 0.456, 'compound': 0.3291}

☹□-- {'neg': 0.706, 'neu': 0.294, 'pos': 0.0, 'compound': -0.34}

Emoticons

print(sentiment_analyzer_scores("Make sure you :) or :D today!"))

Make sure you :) or :D today!----------- {'neg': 0.0, 'neu': 0.294, 'pos': 0.706, 'compound': 0.8633}

We saw how VADER can easily detect sentiment from emojis and slangs which form an important component of the social media environment.

The results of VADER analysis are not only remarkable but also very encouraging. The outcomes highlight the tremendous benefits that can be attained by use of VADER in cases of micro-blogging sites wherein the text data is a complex mix of a variety of text.

6.5 DATA EXTRACTION USING APACHE FLUME:

Apache Flume is a tool/service/data ingestion mechanism for collecting aggregating and transporting large amounts of streaming data such as log files, events (etc...) from various sources to a centralized data store.

Flume is a highly reliable, distributed, and configurable tool. It is principally designed to copy streaming data (log data) from various web servers to HDFS

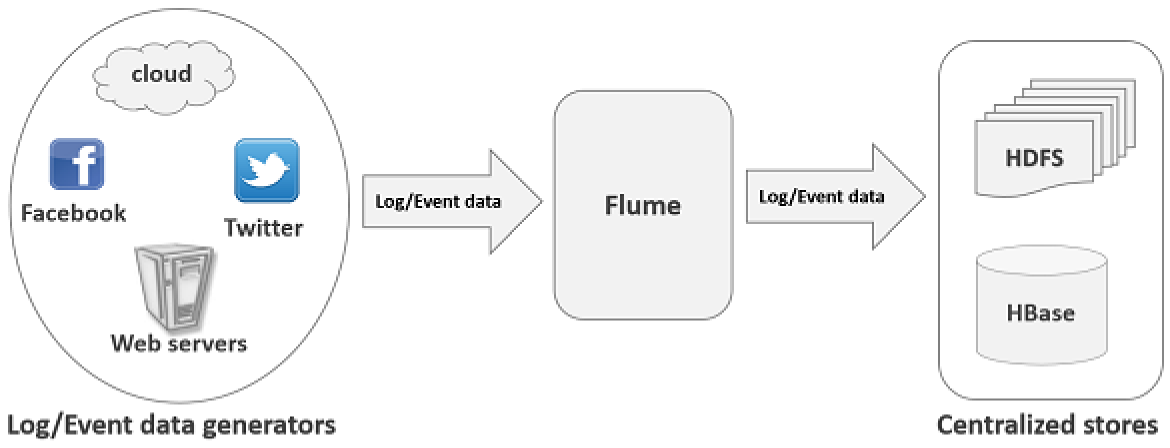


Fig. 2: Apache Flume architecture

Flume is used to move the log data generated by application servers into HDFS at a higher speed.

6.5.1 Advantages Of Using Flume

- Using Apache Flume we can store the data in to any of the centralized stores (HBase, HDFS).
- Apache Flume can handle large amount of Data.
- When the rate of incoming data exceeds the rate at which data can be written to the destination, Flume acts as a mediator between data

producers and the centralized stores and provides a steady flow of data between them.

6.5.2 Features of Flume:

- Some of the notable features of Flume are as follows −
- Flume ingests log data from multiple web servers into a centralized store (HDFS, HBase) efficiently.
- Using Flume, we can get the data from multiple servers immediately into Hadoop.
- Flume supports a large set of sources and destinations types.
- Along with the log files, Flume is also used to import huge volumes of event data produced by social networking sites like Facebook and Twitter, and e-commerce websites like Amazon and Flipkart.

6.5.2.1 Source:

A source is the component of an Agent which receives data from the data generators and transfers it to one or more channels in the form of Flume events.

Example − Avro source, Thrift source, twitter 1% source etc.

6.5.2.2 Channel:

A channel is a transient store which receives the events from the source and buffers them till they are consumed by sinks. It acts as a bridge between the sources and the sinks.

Example − JDBC channel, File system channel, Memory channel, etc.

6.5.2.3 Sink:

A sink stores the data into centralized stores like HBase and HDFS. It consumes the data (events) from the channels and delivers it to the destination. The destination of the sink might be another agent or the central stores.

Example − HDFS sink

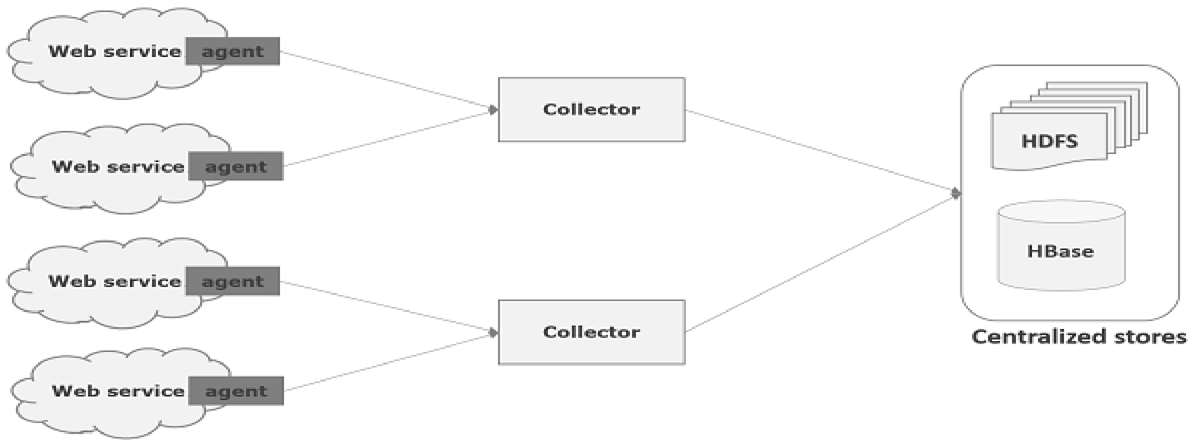


Fig. 3 : DATA FLOW IN APACHE FLUME:

These agents receive the data from the data generators. The data in these agents will be collected by an intermediate node known as Collector. Just like agents, there can be multiple collectors in Flume. Finally, the data from all these collectors will be aggregated and pushed to a centralized store such as HBase or HDFS.

6.6 Using Tweepy:

Python is great language for all sorts of things. Very active developer community creates many libraries which extend the language and make it easier to use various services. One of those libraries is tweepy. Tweepy is open-sourced library enables Python to communicate with Twitter platform and use its API.

Installing tweepy is easy,

pip install tweepy

6.6.1 Using tweepy: Tweepy supports accessing Twitter via Basic Authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication so OAuth is now the only way to use the Twitter API.

6.6.2 A sample of how to access the Twitter API using tweepy with OAuth:

//code for twitter data extraction

Import tweepy

# Consumer keys and access tokens, used for OAuth

```
Consumer key = '7EyzTcAkINVS3T2pb165'
consumer_secret = 'a44R7WvbMW7L8I656Y4l'
access_token = 'z00Xy9AkHwp8vSTJ04L0'
access_token_secret = 'A1cK98w2NXXaCWMqMW6p'
# OAuth process, using the keys and tokens
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
# Creation of the actual interface, using authentication
api = tweepy.API(auth)
# Sample method, used to update a status
api.update_status('Hello Python Central!'
```

The main difference between Basic and OAuth authentication are the consumer and access keys. With Basic Authentication, it was possible to provide a username and password and access the API, but since 2010 when the Twitter started requiring OAuth, the process is a bit more complicated.

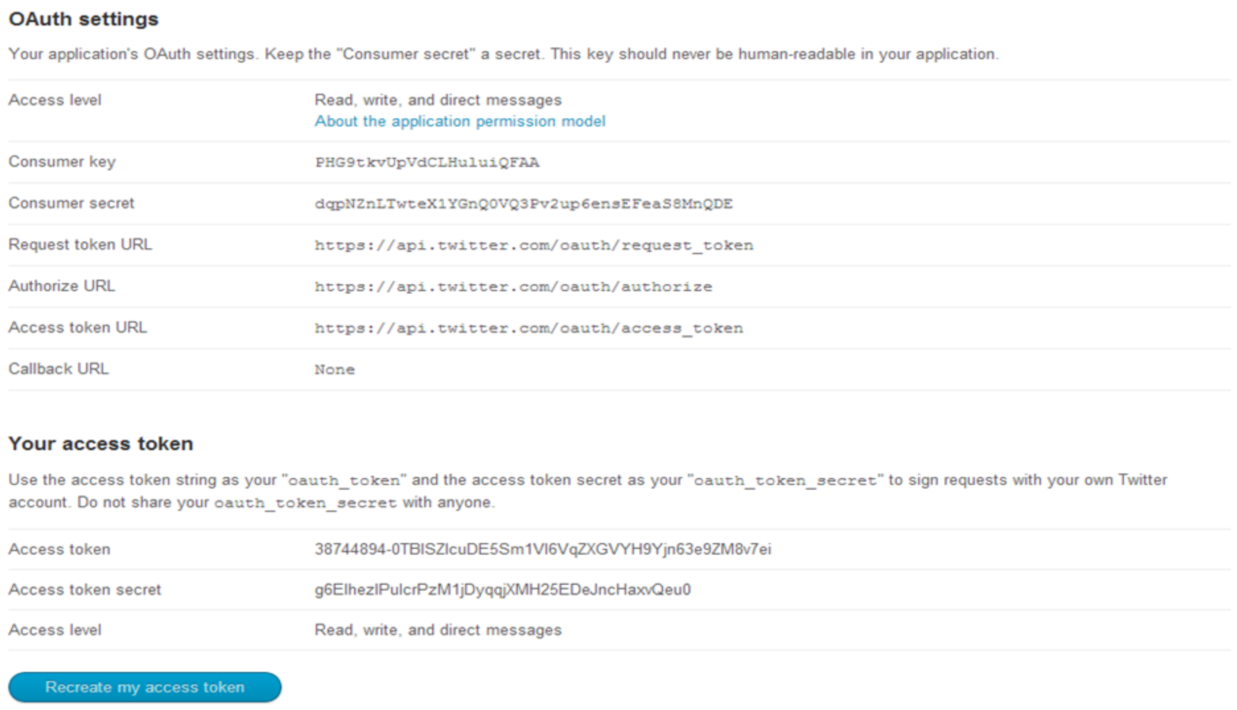
OAuth is a bit more complicated initially than Basic Auth, since it requires more effort, but the benefits it offers are very lucrative:

Tweets can be customized to have a string which identifies the app which was used. It doesn't reveal user password, making it more secure.

It's easier to manage the permissions, for example a set of tokens and keys can be generated that only allows reading from the timelines, so in case someone obtains those credentials, he/she won't be able to write or send direct messages, minimizing the risk.

The application doesn't reply on a password, so even if the user changes it, the application will still work.

After logging in to the portal, and going to "Applications", a new application can be created which will provide the needed data for communicating with Twitter API.



OAuth settings

Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

Access level	Read, write, and direct messages About the application permission model
Consumer key	PHG9tkvUpVdCLHuluiQFAA
Consumer secret	dqpNZnLTwteX1YGnQ0VQ3Pv2up6ensEFeaS8MnQDE
Request token URL	https://api.twitter.com/oauth/request_token
Authorize URL	https://api.twitter.com/oauth/authorize
Access token URL	https://api.twitter.com/oauth/access_token
Callback URL	None

Your access token

Use the access token string as your "oauth_token" and the access token secret as your "oauth_token_secret" to sign requests with your own Twitter account. Do not share your oauth_token_secret with anyone.

Access token	38744894-0TBISZIcuDE5Sm1VI6VqZXGVYH9Yjn63e9ZM8v7ei
Access token secret	g6ElhezlPulcrPzM1jDyqqjXMH25EDeJncHaxvQeu0
Access level	Read, write, and direct messages

Fig. 4: OAuth authentication

The extracted data by tweepy will be in **JSON** format :

```
{
    "text":"RT @sengineland: My Single Best... ",
    "created_at":"Fri Apr 15 23:37:26 +0000 2011",
    "retweet_count":0,
    "id_str":"59037647649259521",
    "entities":{
        "user_mentions":[{
                "screen_name":"sengineland",
                "id_str":"1059801",
                "name":"Search Engine Land",
            }],
        "hashtags":[],
        "urls":[{
                "url":"http:\/\/selnd.com\/e2QPS1",
                "expanded_url":null
            }]
    },
    "user":{
        "created_at":"Sat Jan 22 18:39:46 +0000 2011",
        "friends_count":63,
        "id_str":"241622902",
        ...
    },
    "retweeted_status":{
        "text":"My Single Best... ",
        "created_at":"Fri Apr 15 21:40:10 +0000 2011",
        "id_str":"59008136320786432",
                ...
    },
    ...
}
```

Fig. 5 : Example of tweet in JSON format

7. SYSTEM DESIGN

7.1 Use Case Diagram:

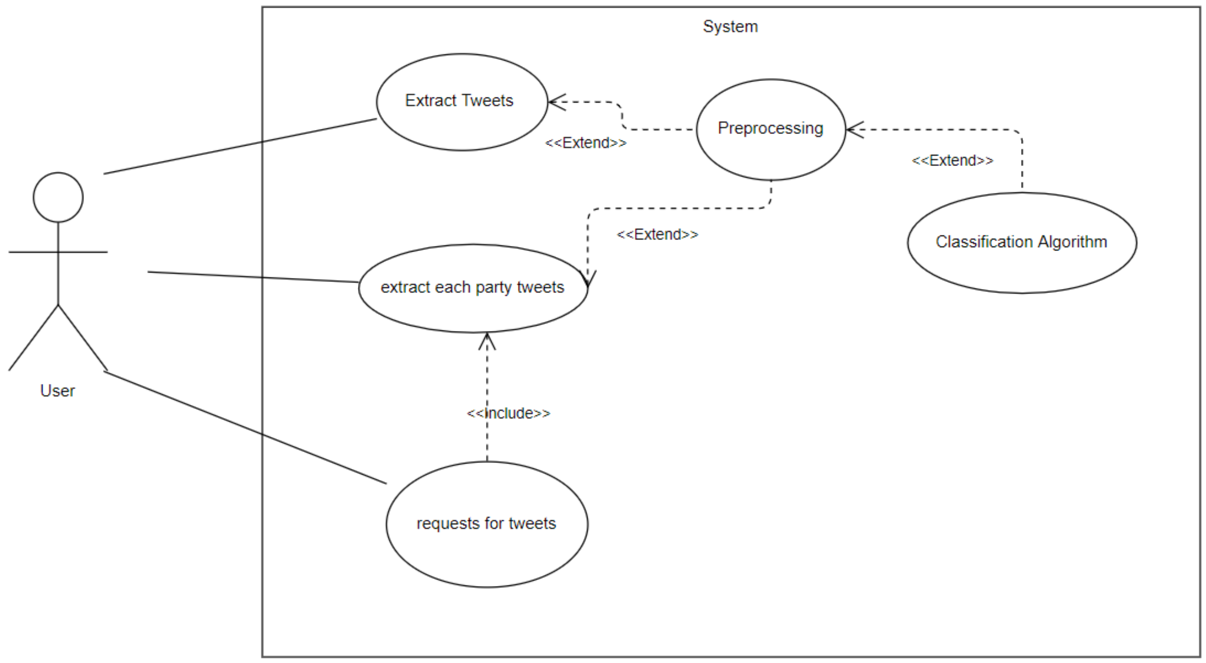


Fig. 6 : Use case diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. In the diagram above :

1. user creates a twitter application and extract the twitter data

2. preprocessing takes the twitter data and clears the noisy data

3. Then it will given to the Naïve Bayes classification algorithm to get the final polarity

4. Plot the final results for each party.

7.2 Sequence Diagram:

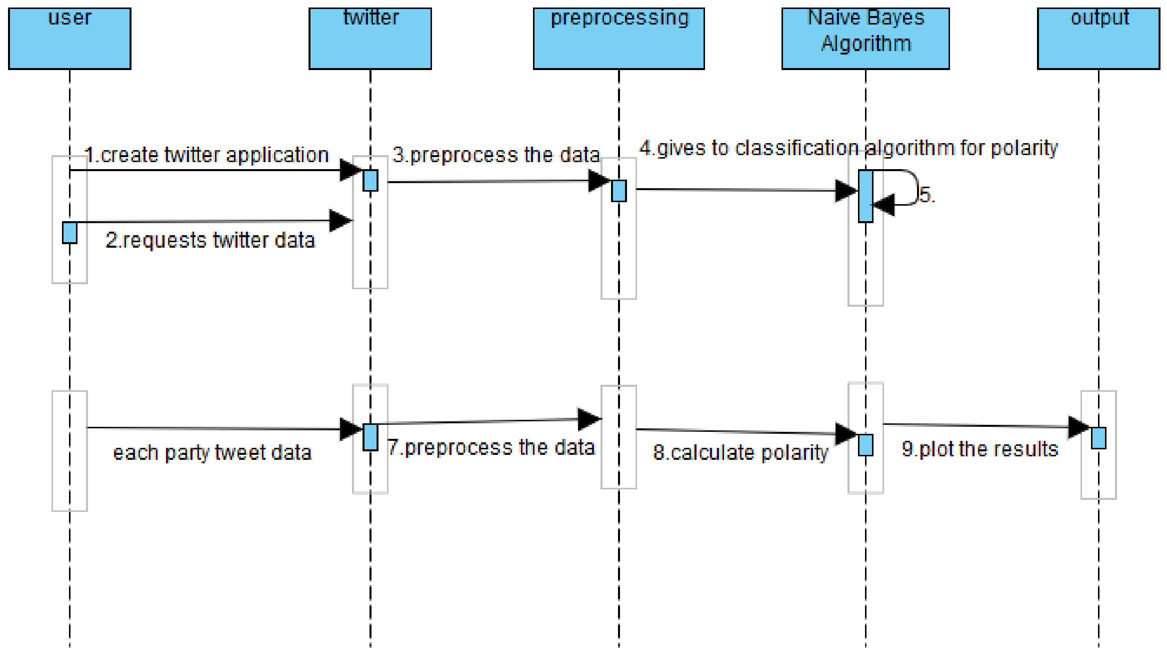


Fig. 7 : Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.In the diagram above:

1. User creates the twitter application account and generate keys
2. User requests twitter data based on the keyword
3. Preprocess the extracted trained data
4. Classify the positive and negative tweets for training
5. Extract the tweet data using keywords like janasena,e.t.c
6. Preprocess the each extracted tweet data
7. Identify the polarity for the entire party tweet data
8. Finally, plot the results using the graphs.

7.3 Activity Diagram:

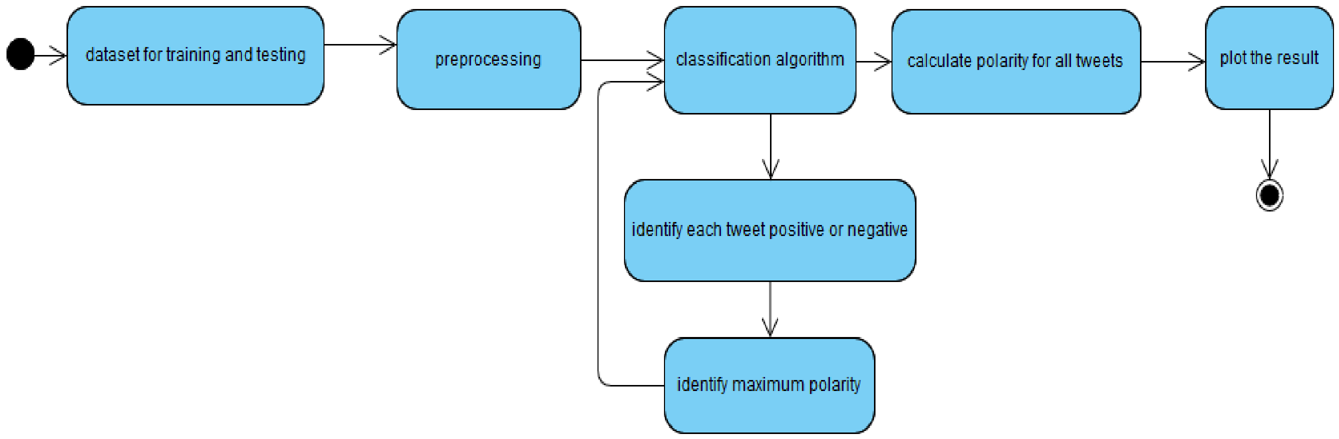

dataset for training and testing
preprocessing
classification algorithm
calculate polarity for all tweets
plot the result
identify each tweet positive or negative
identify maximum polarity

Fig. 8: Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. The above diagram clearly explains the activity done at each stage. They are:

1. Extract the twitter data using twitter application account
2. Preprocess the extracted twitter data
3. Train and Test the results using the Naïve Bayes Classification algorithm
4. Identify the polarity for all tweets
5. Finally, plot the results using graphs.

8. SOFTWARE AND HARDWARE REQUIREMENTS

This section outlines minimum software and hardware requirements for deploying sentiment analysis using classification.

8.1 Software Requirements:

Software required to run this project is:

● Windows 10 Home or Windows 10 Pro or any Linux based system

● Python version>3.5.0

● Dependencies: NLTK, Tweepy, pandas, matplotlib

8.2 Hardware Requirements:

Usage of CPU, RAM and storage space can vary significantly based on number of tweets.

Hardware required to run this project is:

● PROCESSOR: Intel i3-4210U 1.70GHz * 4 CPU or above

● RAM: 4GB or above

● Storage Space of 250MB or above

8.3 Internet Requirements:

Requires continuous Internet connection with speed above 2MBPS

9. IMPLEMENTATION AND SAMPLE CODE

9.1 Creating a Twitter Application:

In order to get the tweets from Twitter, it is needed to create a Twitter application. Follow the steps given below to create a Twitter application.

Step 1:

To create a Twitter application, click on the following link https://apps.twitter.com/. Sign in to your Twitter account. You will have a Twitter Application Management window where you can create, delete, and manage Twitter Apps.

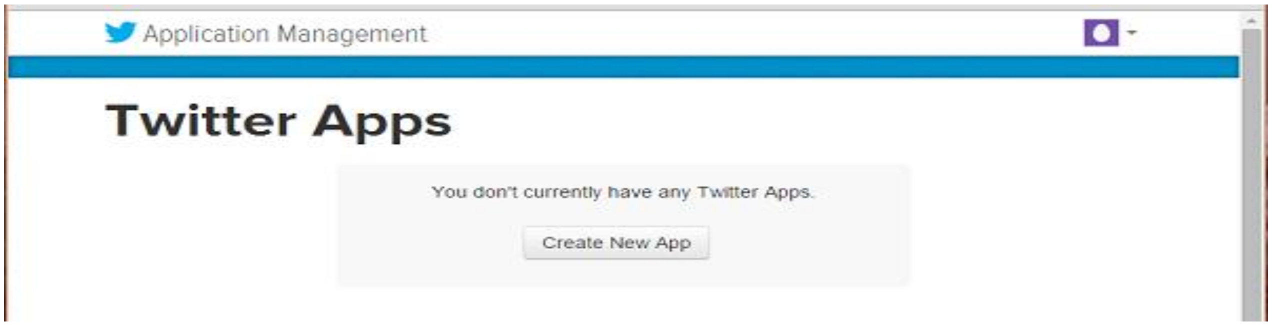
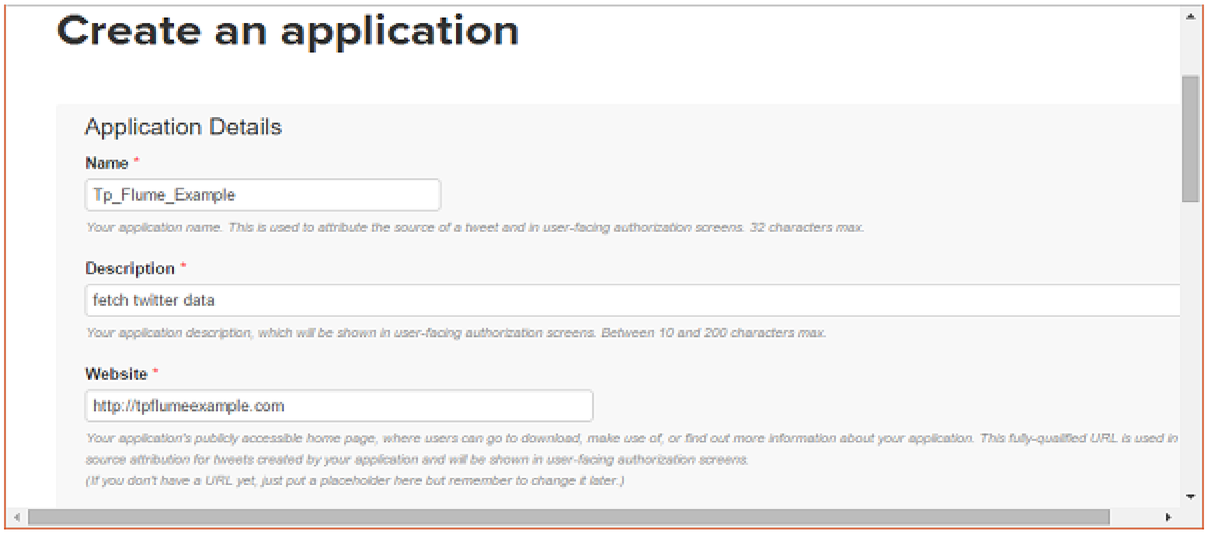


Fig. 9 : Twitter Application

Step 2:

Click on the Create New App button. You will be redirected to a window where you will get an application form in which you have to fill in your details in order to create the App. While filling the website address, give the complete URL pattern,



Create an application

Application Details

Name *

Tp_Flume_Example

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

fetch twitter data

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

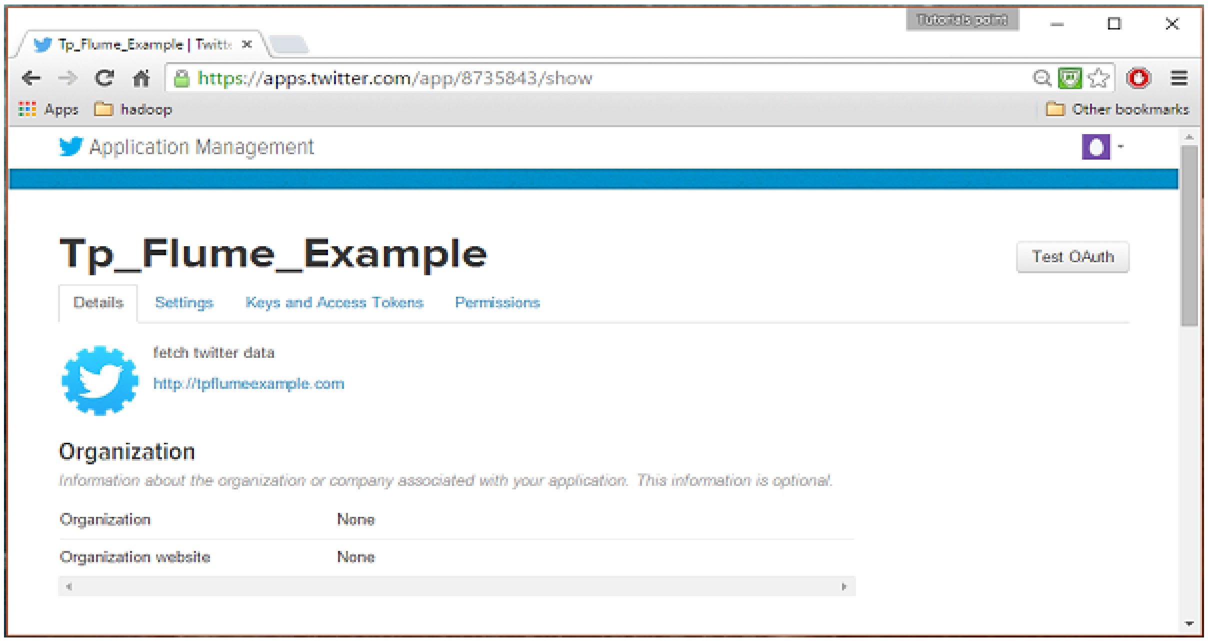
http://tpflumeexample.com

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Fig. 10 : Creating New Application

Step 3:

Fill in the details, accept the Developer Agreement when finished, click on the Create your Twitter application button which is at the bottom of the page. If everything goes fine, an App will be created with the given details as shown below.



Tp_Flume_Example

fetch twitter data

http://tpflumeexample.com

Organization

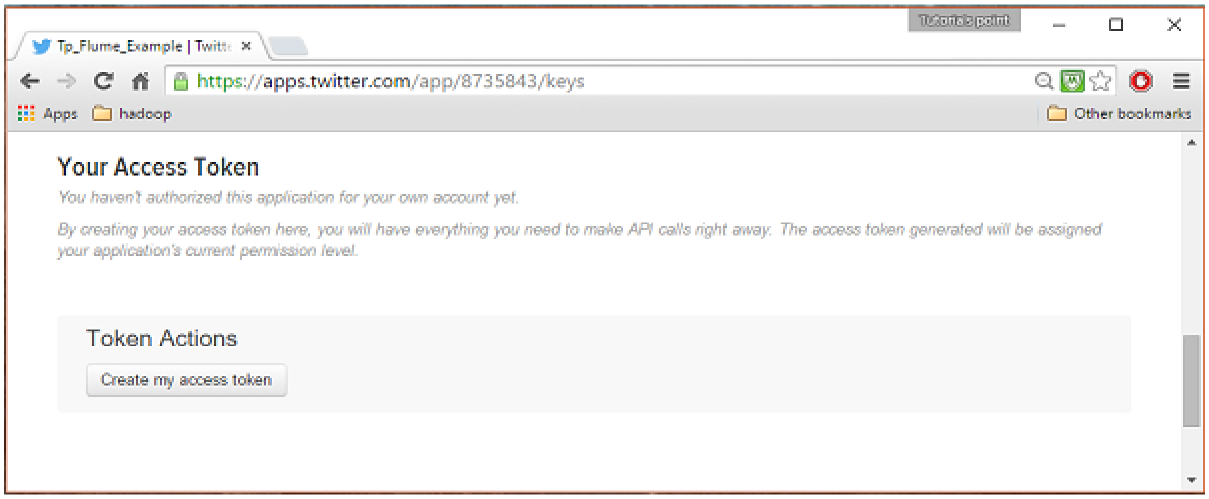
Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

Fig. 11 : TP_Flume_Example

Step 4:

Under keys and Access Tokens tab at the bottom of the page, you can observe a button named Create my access token. Click on it to generate the access token.



Your Access Token

You haven't authorized this application for your own account yet.

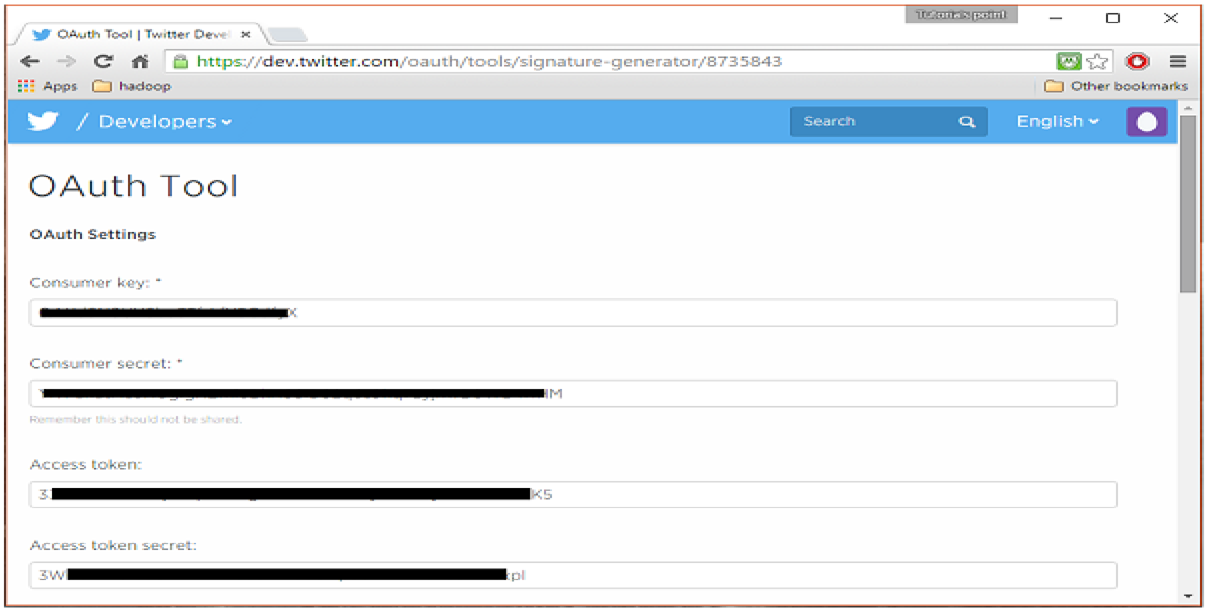
By creating your access token here, you will have everything you need to make API calls right away. The access token generated will be assigned your application's current permission level.

Token Actions

Fig. 12 : Generating access token keys

Step 5:

Finally, click on the Test OAuth button which is on the right side top of the page. This will lead to a page which displays your Consumer key, Consumer secret, Access token, and Access token secret. Copy these details. These are useful to configure the agent in Flume.


OAuth Tool | Twitter Deve
https://dev.twitter.com/oauth/tools/signature-generator/8735843
Apps
hadoop
Other bookmarks
Developers
Search
English

OAuth Tool

OAuth Settings

Consumer key: *

[redacted]x

Consumer secret: *

[redacted]IM

Remember this should not be shared.

Access token:

3[redacted]K5

Access token secret:

3W[redacted]pl

Fig. 13 : consumer, consumer secret, access token and access token secret key.

9.2 Extracting Twitter data using Tweepy:

Tweepy is one of the library that should be installed using pip. Now in order to authorize our app to access Twitter on our behalf, we need to use the OAuth Interface. Tweepy provides the convenient Cursor interface to iterate through different types of objects.

These all are the prerequisite that have to be used before getting tweets of a user.

```
# AUTHENTICATION (OAuth)
def tw_oauth(authfile):
    with open(authfile, "r") as f:
        ak = f.readlines()
    f.close()
    auth1 = tweepy.auth.OAuthHandler(ak[0].replace("\n",""), ak[1].replace("\n",""))
    auth1.set_access_token(ak[2].replace("\n",""), ak[3].replace("\n",""))
    return tweepy.API(auth1)
```

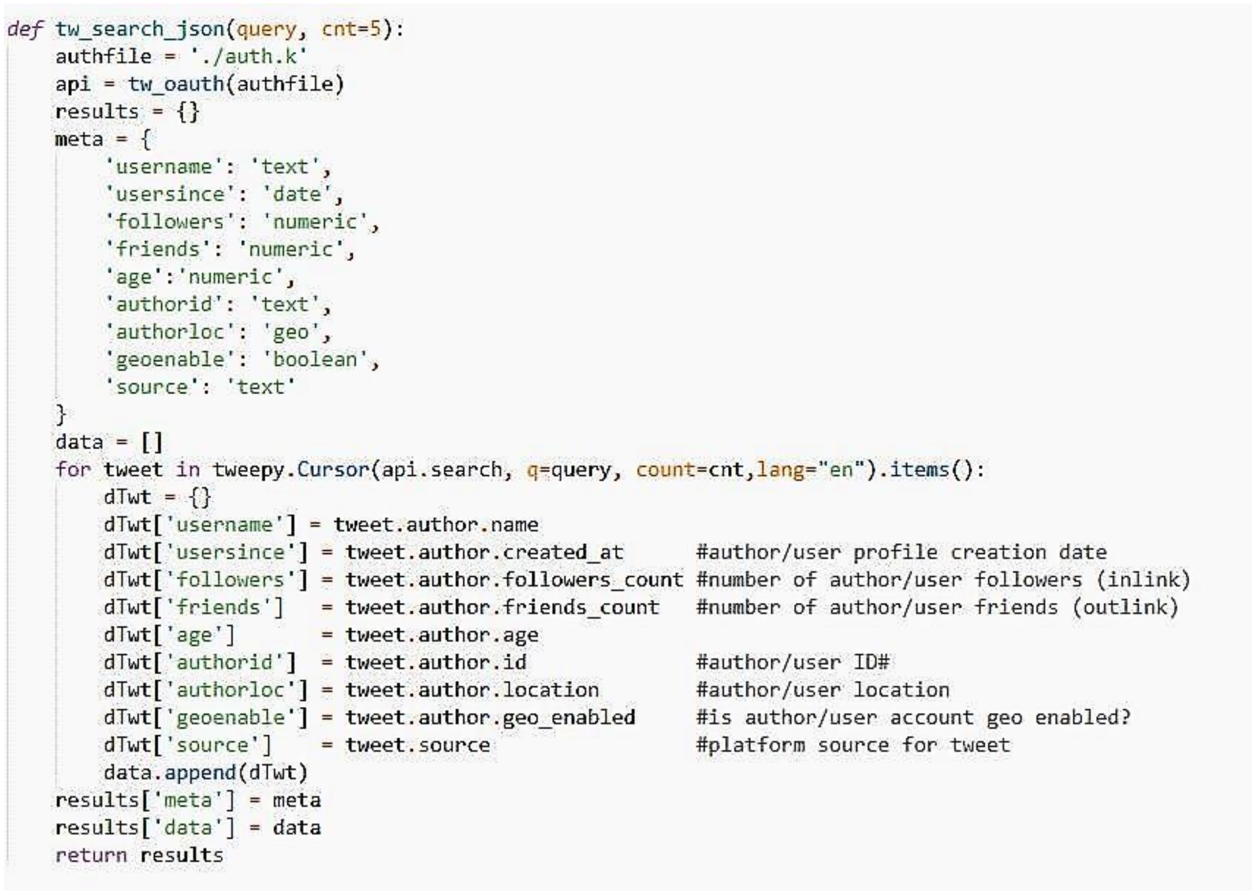
Fig. 14: Oauth for authentication with twitter

9.3 Converting the extracted data into CSV file:

JSON is an acronym standing for JavaScript Object Notation. The json library in python can parse JSON from strings or files. The library parses JSON into a Python dictionary or list. We come across various circumstances where we receive data in json format and we need to send or store it in csv format.

The python program written below will open a csv file in tmp folder and write the content of JSON file into it and close it at the end. Make sure to close the file at the end in order to save the contents.

JSON files can have much more complex structures than CSV files, so a direct conversion is not always possible. The above mentioned examples will lay the foundation for converting JSON data with high complexity to CSV.

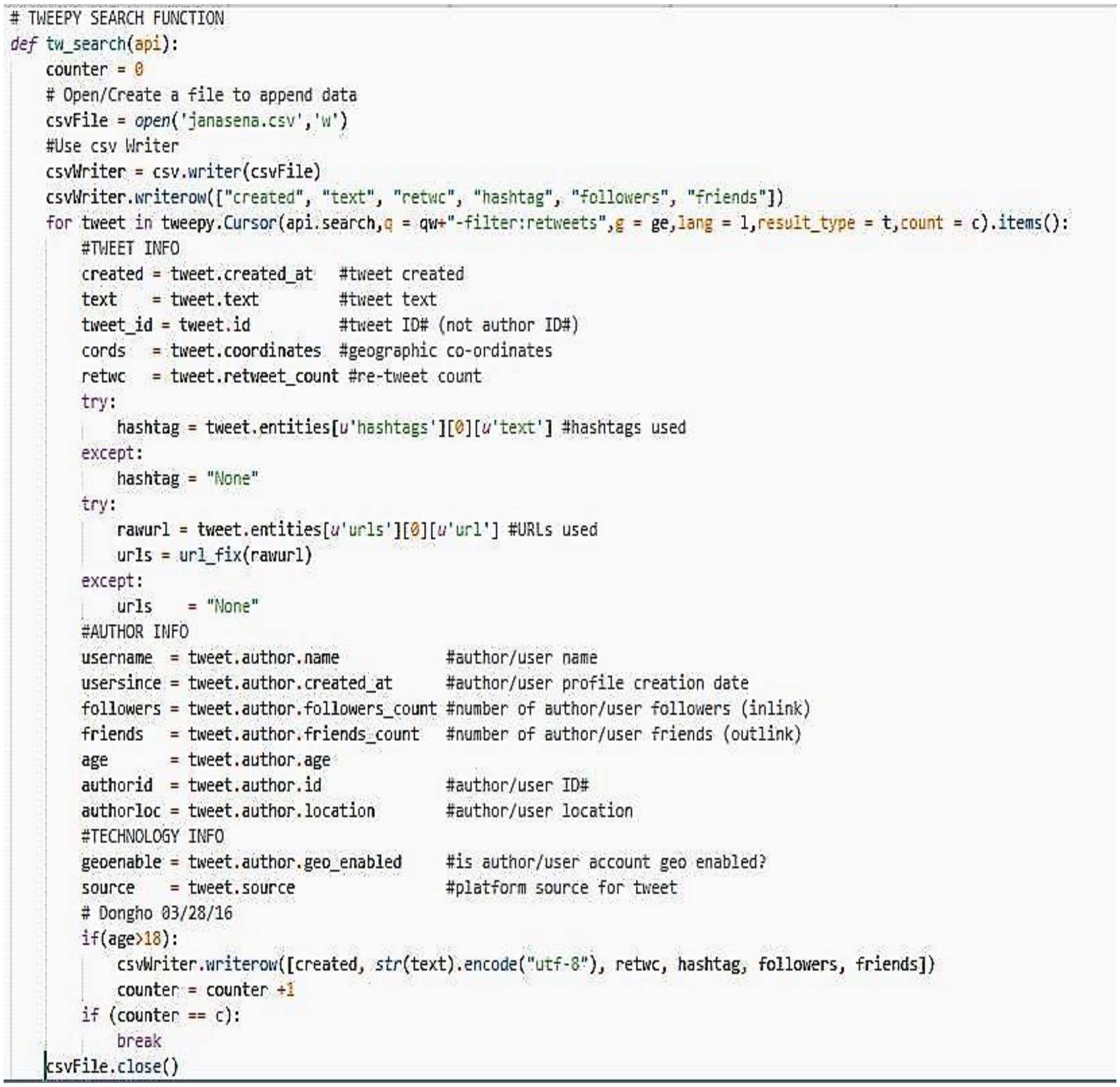


```
def tw_search_json(query, cnt=5):
    authfile = './auth.k'
    api = tw_oauth(authfile)
    results = {}
    meta = {
        'username': 'text',
        'usersince': 'date',
        'followers': 'numeric',
        'friends': 'numeric',
        'age':'numeric',
        'authorid': 'text',
        'authorloc': 'geo',
        'geoenable': 'boolean',
        'source': 'text'
    }
    data = []
    for tweet in tweepy.Cursor(api.search, q=query, count=cnt,lang="en").items():
        dTwt = {}
        dTwt['username'] = tweet.author.name
        dTwt['usersince'] = tweet.author.created_at      #author/user profile creation date
        dTwt['followers'] = tweet.author.followers_count #number of author/user followers (inlink)
        dTwt['friends']   = tweet.author.friends_count   #number of author/user friends (outlink)
        dTwt['age']       = tweet.author.age
        dTwt['authorid']  = tweet.author.id              #author/user ID#
        dTwt['authorloc'] = tweet.author.location        #author/user location
        dTwt['geoenable'] = tweet.author.geo_enabled     #is author/user account geo enabled?
        dTwt['source']    = tweet.source                 #platform source for tweet
        data.append(dTwt)
    results['meta'] = meta
    results['data'] = data
    return results
```

Fig. 15: Extracted twitter data in JSON format

Python provides an easy way to work with csv file it has csv module to read and write data in the csv file.

Also we included a main obstacle that the twitter tweets must be extracted only the age is greater than or equal to 18.



```
# TWEEPY SEARCH FUNCTION
def tw_search(api):
    counter = 0
    # Open/Create a file to append data
    csvFile = open('janasena.csv','w')
    #Use csv Writer
    csvWriter = csv.writer(csvFile)
    csvWriter.writerow(["created", "text", "retwc", "hashtag", "followers", "friends"])
    for tweet in tweepy.Cursor(api.search,q = qw+"-filter:retweets",g = ge,lang = l,result_type = t,count = c).items():
        #TWEET INFO
        created = tweet.created_at   #tweet created
        text    = tweet.text         #tweet text
        tweet_id = tweet.id          #tweet ID# (not author ID#)
        cords   = tweet.coordinates  #geographic co-ordinates
        retwc   = tweet.retweet_count #re-tweet count
        try:
            hashtag = tweet.entities[u'hashtags'][0][u'text'] #hashtags used
        except:
            hashtag = "None"
        try:
            rawurl = tweet.entities[u'urls'][0][u'url'] #URLs used
            urls = url_fix(rawurl)
        except:
            urls    = "None"
        #AUTHOR INFO
        username  = tweet.author.name            #author/user name
        usersince = tweet.author.created_at      #author/user profile creation date
        followers = tweet.author.followers_count #number of author/user followers (inlink)
        friends   = tweet.author.friends_count   #number of author/user friends (outlink)
        age       = tweet.author.age
        authorid  = tweet.author.id              #author/user ID#
        authorloc = tweet.author.location        #author/user location
        #TECHNOLOGY INFO
        geoenable = tweet.author.geo_enabled     #is author/user account geo enabled?
        source    = tweet.source                 #platform source for tweet
        # Dongho 03/28/16
        if(age>18):
            csvWriter.writerow([created, str(text).encode("utf-8"), retwc, hashtag, followers, friends])
            counter = counter +1
        if (counter == c):
            break
    csvFile.close()
```

Fig. 16 : Converting the data in JSON format into CSV file

9.4 PREPROCESSING:

In this step, we decided to process the data before we are entering into the main process of finding polarity. The various pre-processing steps we applied are,

9.4.1 Tokenization:

Here they tokenized the reviews using the tokenizer. It is an act of breaking sentence into pieces of words, keywords ,phrases, symbols and other elements called tokens…these tokens become input to the another process such as parsing or text mining.

9.4.2 Remove Punctuations:

Remove the punctuations from the reviews, this is done so that we do not have any unwanted punctuations in our resultant feature set.

9.4.3 Remove Stop-words:

This is a basic requirement, so that we can focus on words that are actually relevant to the document instead of, determiners, prepositions and coordinating conjunctions, which can appear a number of times in any given training set, if not removed. Removal of stop words is crucial to supervised learning.

9.4.4 Convert Upper case letters to lower case:

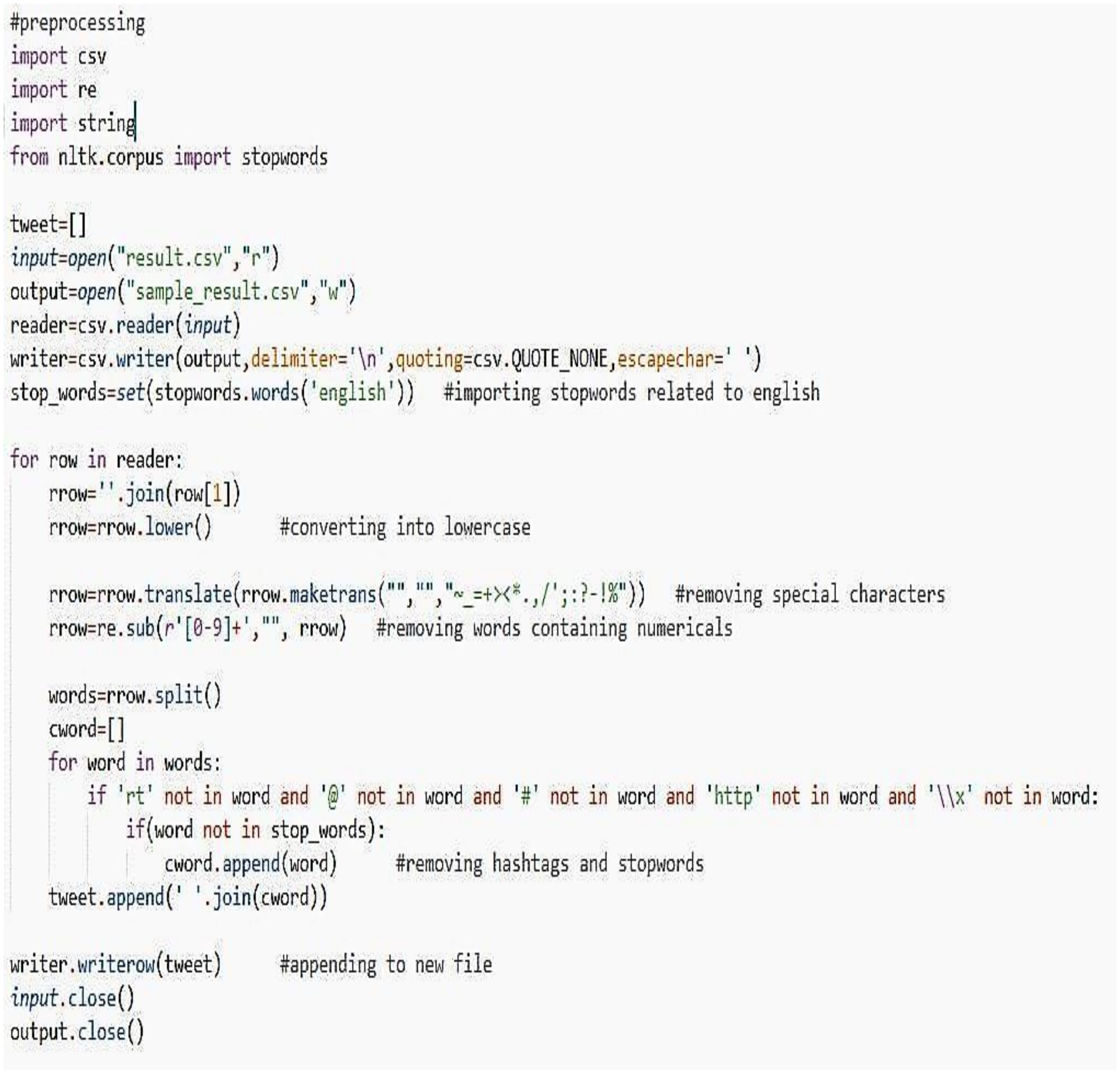
In this we convert all the upper case letters to lower case to avoid the confusion.

9.4.5 Spell-check:

Perform spell-check on the documents (reviews) to ensure that the feature-set being generated has relevant words and not commonly misspelled words, apart from this, spell-check allows for accurate frequency calculation, which is crucial when the basis of the feature set generation is frequency distribution over the set of processed documents. To accomplish this we have a big.txt file which consists of about a million words. The file is a concatenation of several public domain books from Project Gutenberg and lists of the most frequent words from Wiktionary and the British National Corpus. We then extracted the individual words from the file and trained a probability model (based on occurrence of each word). After this, we implemented Smoothing over the parts of probability distribution, that would have been zero (words that have not occurred in the big.txt file) by bumping them up to the smallest possible count. We performed this process of spell-checking two times using an edit-distance of 2, this was done after analysing that spell-checking twice gives the best result.

We are using a method called maketranslation(),**Python** String **translate**() The string **translate**() **method** returns a string where each character is mapped to its corresponding character in the translation table. The

translate() **method** takes the translation table to replace/**translate** characters in the given string as per the mapping table.



```
#preprocessing
import csv
import re
import string
from nltk.corpus import stopwords

tweet=[]
input=open("result.csv","r")
output=open("sample_result.csv","w")
reader=csv.reader(input)
writer=csv.writer(output,delimiter='\n',quoting=csv.QUOTE_NONE,escapechar=' ')
stop_words=set(stopwords.words('english'))   #importing stopwords related to english

for row in reader:
    rrow=''.join(row[1])
    rrow=rrow.lower()       #converting into lowercase

    rrow=rrow.translate(rrow.maketrans("","","~_=+><*.,/';:?-!%"))   #removing special characters
    rrow=re.sub(r'[0-9]+',"", rrow)   #removing words containing numericals

    words=rrow.split()
    cword=[]
    for word in words:
        if 'rt' not in word and '@' not in word and '#' not in word and 'http' not in word and '\\x' not in word:
            if(word not in stop_words):
                cword.append(word)      #removing hashtags and stopwords
    tweet.append(' '.join(cword))

writer.writerow(tweet)      #appending to new file
input.close()
output.close()
```

Fig. 17 : Preprocessing the data using this python code

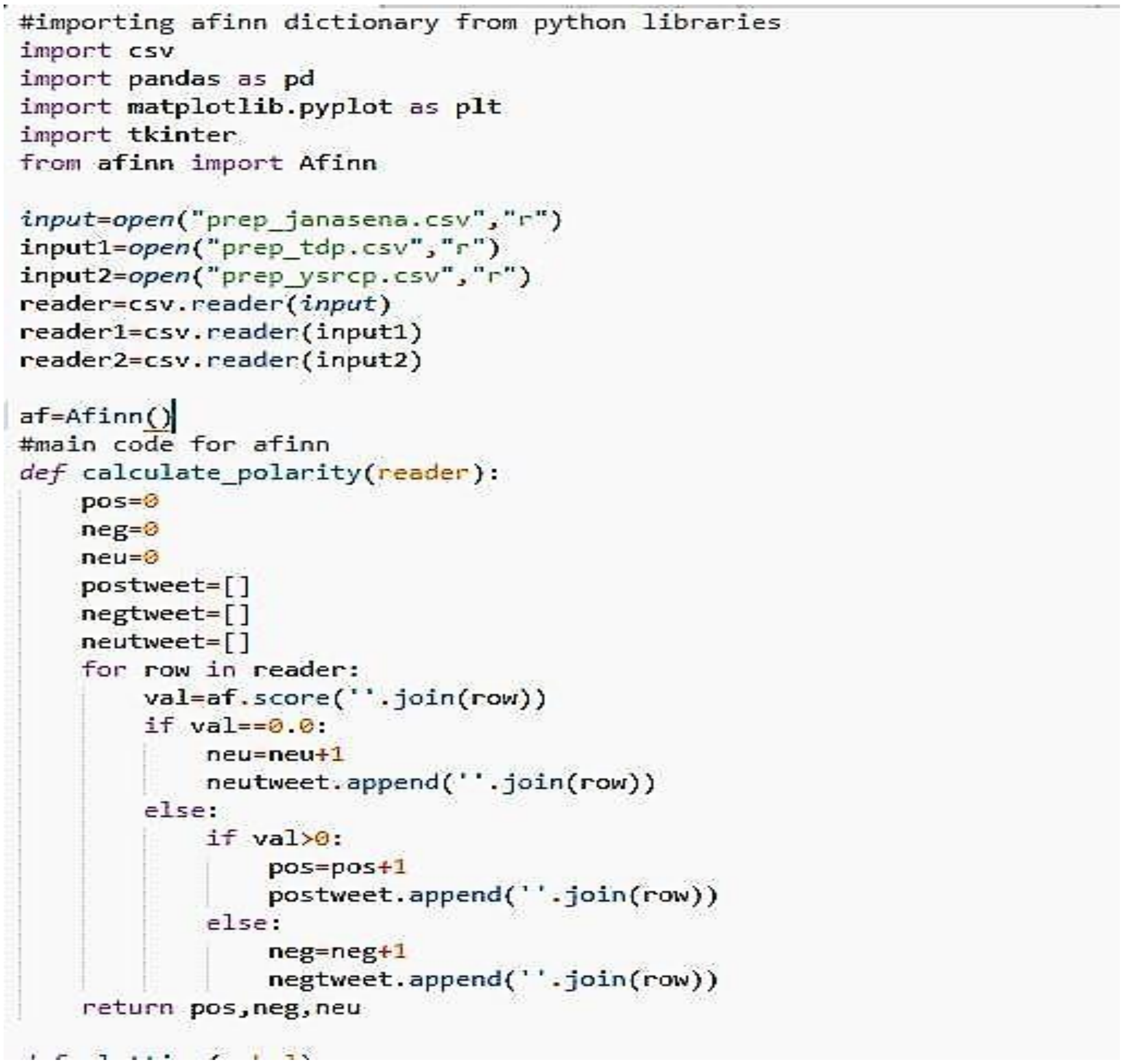
9.5 Sentiment Analysis using AFINN Dictionary:

Let’s use a dictionary called AFINN to calculate the sentiments.

AFINN is a list of English words rated for valence with an integer between -5 (negative) and +5 (positive). The words have been manually labeled by Finn Årup Nielsen in 2009-2011. The file is tab-separated.

There are two versions:

AFINN-96: 1468 unique words and phrases on 1480 lines. Note that there are 1480 lines, as some words are listed twice. The word list in notEntirely in alphabetic ordering. AFINN-111: Newest version with 2477 words and phrases.



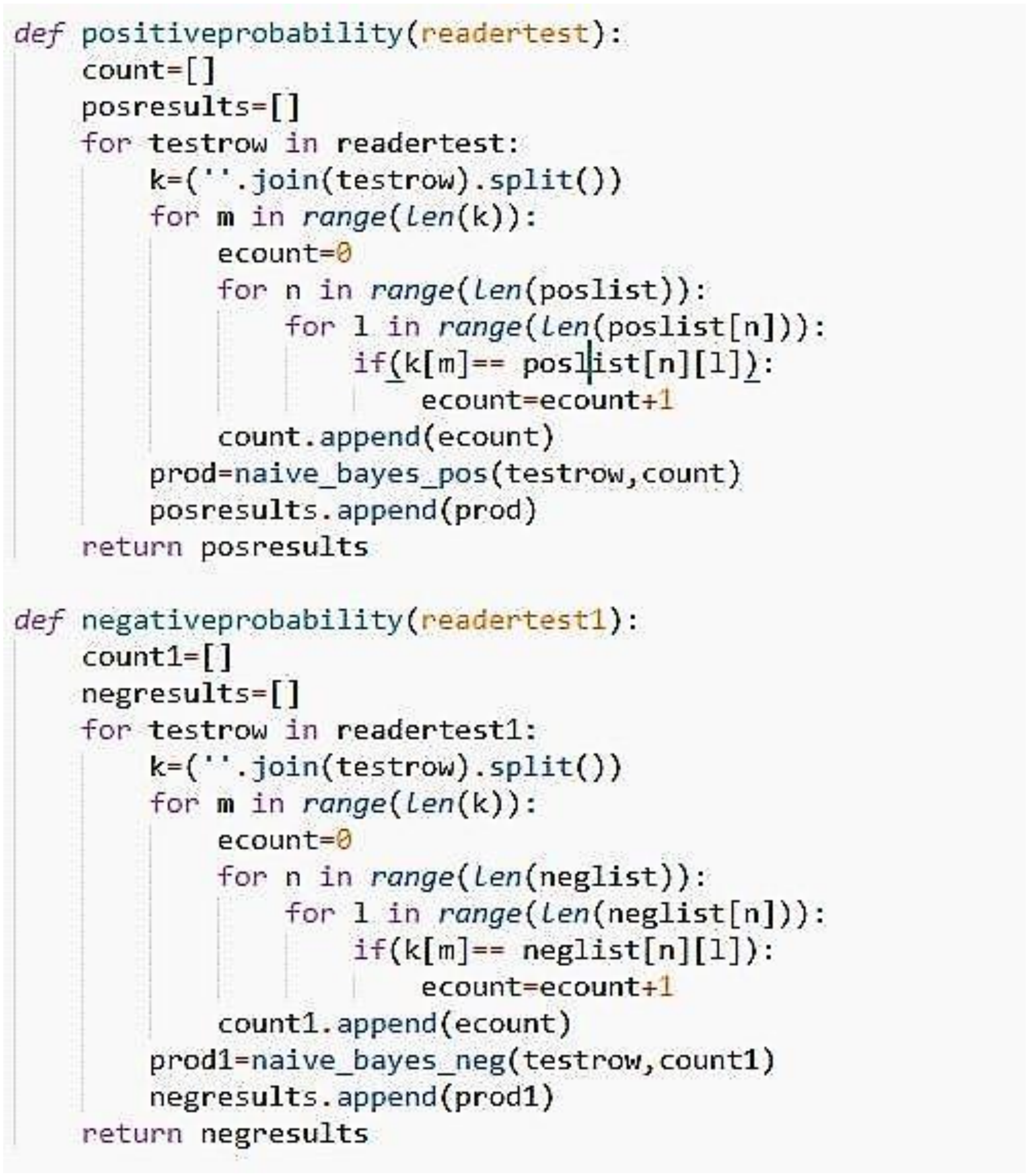
```
#importing afinn dictionary from python libraries
import csv
import pandas as pd
import matplotlib.pyplot as plt
import tkinter
from afinn import Afinn

input=open("prep_janasena.csv","r")
input1=open("prep_tdp.csv","r")
input2=open("prep_ysrcp.csv","r")
reader=csv.reader(input)
reader1=csv.reader(input1)
reader2=csv.reader(input2)

af=Afinn()
#main code for afinn
def calculate_polarity(reader):
    pos=0
    neg=0
    neu=0
    postweet=[]
    negtweet=[]
    neutweet=[]
    for row in reader:
        val=af.score(''.join(row))
        if val==0.0:
            neu=neu+1
            neutweet.append(''.join(row))
        else:
            if val>0:
                pos=pos+1
                postweet.append(''.join(row))
            else:
                neg=neg+1
                negtweet.append(''.join(row))
    return pos,neg,neu
```

Fig. 18 : python code for classification using AFINN dictionary

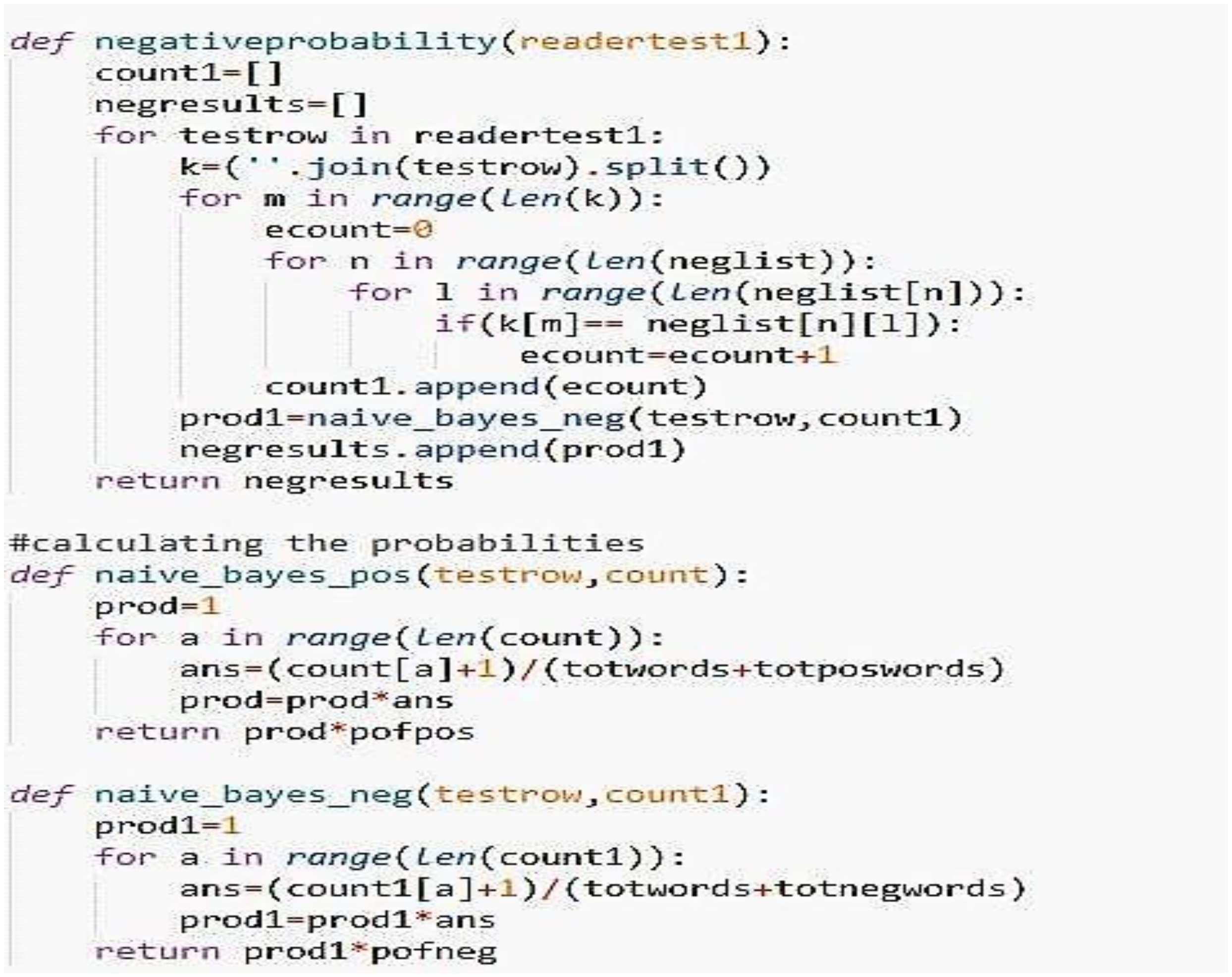
9.6 Sentiment Analysis using Naïve Bayes:



```
def positiveprobability(readertest):
    count=[]
    posresults=[]
    for testrow in readertest:
        k=(''.join(testrow).split())
        for m in range(len(k)):
            ecount=0
            for n in range(len(poslist)):
                for l in range(len(poslist[n])):
                    if(k[m]== poslist[n][l]):
                        ecount=ecount+1
            count.append(ecount)
        prod=naive_bayes_pos(testrow,count)
        posresults.append(prod)
    return posresults

def negativeprobability(readertest1):
    count1=[]
    negresults=[]
    for testrow in readertest1:
        k=(''.join(testrow).split())
        for m in range(len(k)):
            ecount=0
            for n in range(len(neglist)):
                for l in range(len(neglist[n])):
                    if(k[m]== neglist[n][l]):
                        ecount=ecount+1
            count1.append(ecount)
        prod1=naive_bayes_neg(testrow,count1)
        negresults.append(prod1)
    return negresults
```

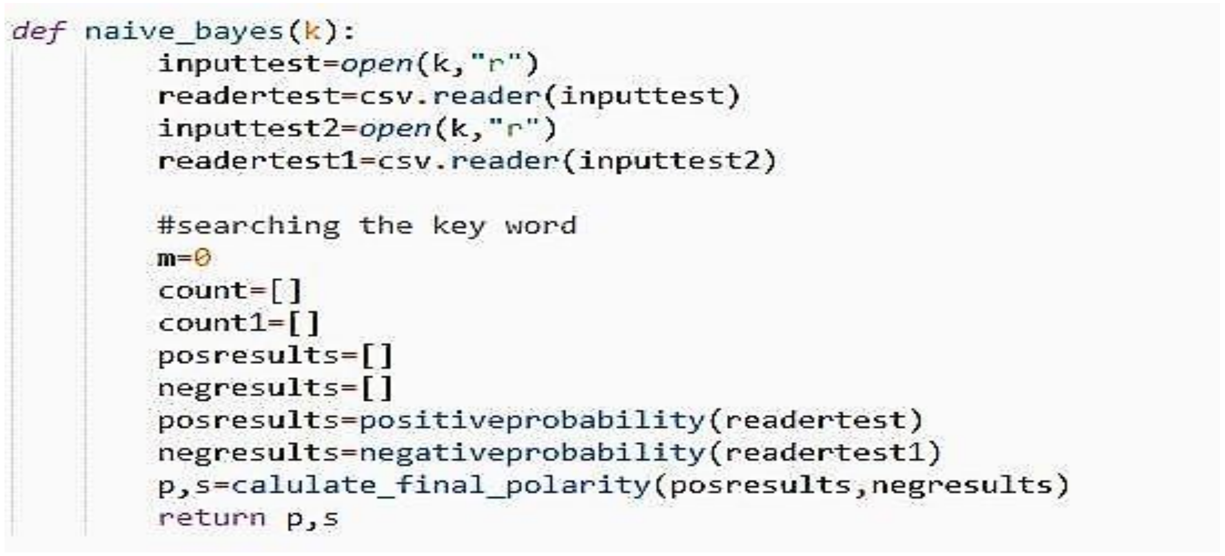
Fig. 19.1 : python code for classification using naïve bayes



```
def negativeprobability(readertest1):
    count1=[]
    negresults=[]
    for testrow in readertest1:
        k=(''.join(testrow).split())
        for m in range(len(k)):
            ecount=0
            for n in range(len(neglist)):
                for l in range(len(neglist[n])):
                    if(k[m]== neglist[n][l]):
                        ecount=ecount+1
            count1.append(ecount)
        prod1=naive_bayes_neg(testrow,count1)
        negresults.append(prod1)
    return negresults

#calculating the probabilities
def naive_bayes_pos(testrow,count):
    prod=1
    for a in range(len(count)):
        ans=(count[a]+1)/(totwords+totposwords)
        prod=prod*ans
    return prod*pofpos

def naive_bayes_neg(testrow,count1):
    prod1=1
    for a in range(len(count1)):
        ans=(count1[a]+1)/(totwords+totnegwords)
        prod1=prod1*ans
    return prod1*pofneg
```



```
def naive_bayes(k):
        inputtest=open(k,"r")
        readertest=csv.reader(inputtest)
        inputtest2=open(k,"r")
        readertest1=csv.reader(inputtest2)

        #searching the key word
        m=0
        count=[]
        count1=[]
        posresults=[]
        negresults=[]
        posresults=positiveprobability(readertest)
        negresults=negativeprobability(readertest1)
        p,s=calulate_final_polarity(posresults,negresults)
        return p,s
```

Fig. 19.2 : python code for classification using naïve bayes

9.7 Comparing the results and output:

Results Using Lexicon based Approach:

Here, we considered the 1650 tweets for each individual party and the results using lexicon approach is as follows:

We are using matplot lib in python to show output in pictorial statistical figure as shown below.

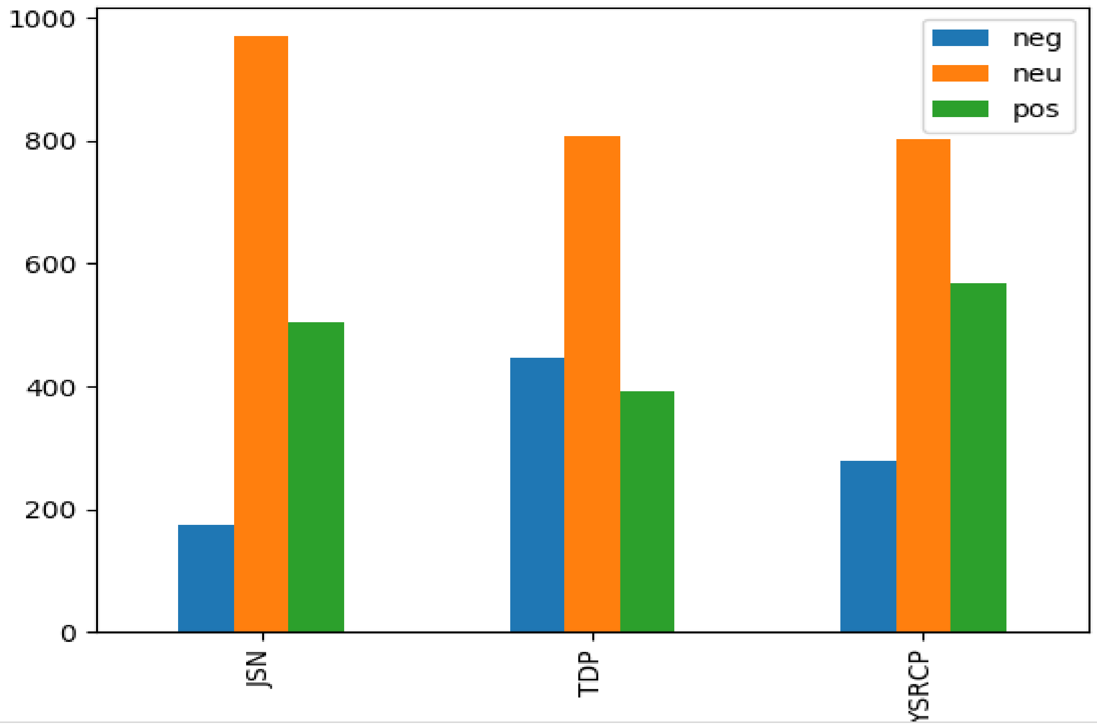


Fig. 20 : Output using Afinn Dictionary

The results obtained for each individual party is:

Table 3 : Results based on Lexicon based Approach

PARTY	POSITIVE	NEGATIVE	NEUTRAL
JSN	31%	11%	58%
TDP	24%	28%	48%
YSRCP	35%	17%	48%

Using lexicon based approach we get 31% ,11%,58% of positive,negative and neutral tweets for Janasena party similarly, we get 24%,28%,48% of positive,negative and neutral tweets for Telugu Desam Party and we get 35% ,17%,48% of positive,negative and neutral tweets for YSR congress party.

Here, we considered the 1650 tweets for each individual party and the results using Naïve Bayes classifier are as follows:

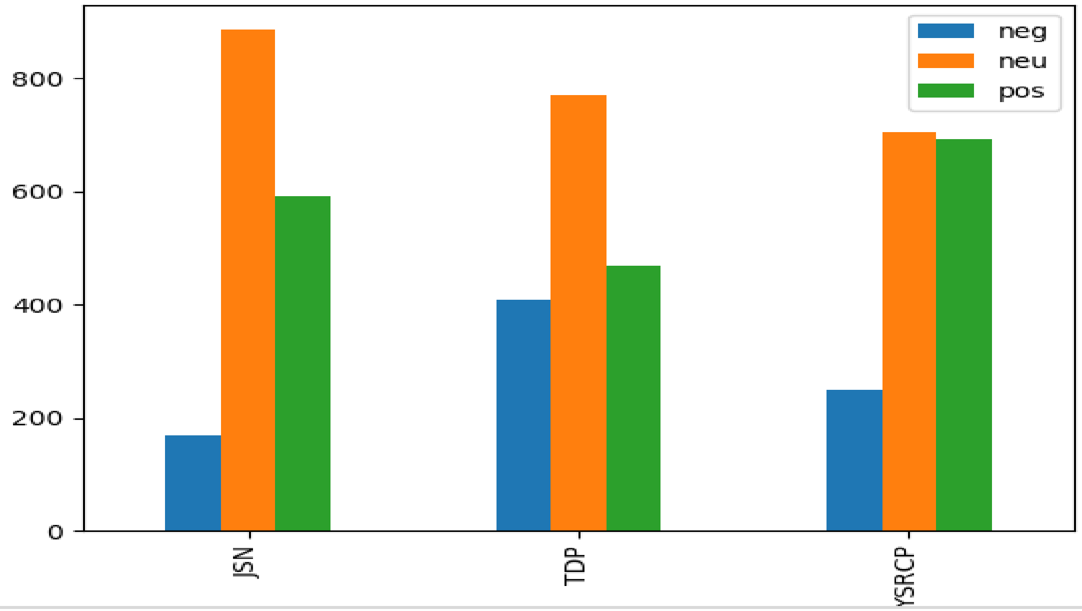


Fig. 21: Output using Naïve bayes Classifier

After Classification, the results obtained for each individual party is:

Table 4: Results obtained after classification of each individual party

PARTY	POSITIVE	NEGATIVE	NEUTRAL
JSN	40%	11%	49%
TDP	29%	25%	46%
YSRCP	42%	17%	41%

Using Naïve Bayes Classifier Approach we get 40% , 11% , 49% of positive , negative and neutral tweets for Janasena party similarly, we get 29% , 25% , 46% of positive , negative and neutral tweets for Telugu Desam Party and we get 42% , 17% , 41% of positive , negative and neutral tweets for YSR congress party.

10. FUTURE SCOPE & CONCLUSION

In this project we are used a limited amount of data and also we used a small dictionary with nearly 2500 words and their corresponding polarities to asign a polarity for each tweet. Further we may extend this to long dictionary. And we used python for the data extraction and we used statistical approach for classifying the twitter data. So in future ,we prolong this project by using APACHE FLUME for data extraction and Apache spark to process huge amount of data and also different algorithms other than the statistical approaches.

Finally, we categorised the 1650 tweets for each party into positive, negative and neutral classes. we got the results as JANASENA had got the 31% of positive tweets using Afinn dictionary and 40% of positive tweets using Naive Bayes. Similarly, tdp got the 24% of positive tweets using Afinn dictionary and 29% of positive tweets using Naïve Bayes and ysrcp got the 35% of positive tweets using Afinn dictionary and 42% of positive tweets using Naïve Bayes.

We are concluding that after analysing the above results, we are predicting that YSRCP has slightly more chances to win in the upcoming elections when compared to JANASENA and TDP.

11. REFERENCES

[1] Pak, A. &. (2010) . Twitter as a Corpus for Sentiment Analysis and Opinion Mining . LREC .

[2] Gayo-Avello , D. (2012). No, you cannot predict elections with twitter. Internet Computing , IEEE, 16(6), 91-94.

[3] El-Beltagy SR and Ali A. "Open issues in the sentiment analysis of Arabic social media: a case study." In: 9th International Conference on Innovations in Information Technology (IIT). Abu Dhabi: IEEE, 2013, pp. 215–220.

[4] L. Chih-Wei, H. Chih-Ming, C. Chih-Hung, Y. Chao-Tung, An Improvement to Data Service in Cloud Computing with Content Sensitive Transaction Analysis and Adaptation, Computer Software and Applications Conference Workshops (COMPSACW), 2013 IEEE 37th Annual, 2013, pp. 463-4681.

[5] Text Preprocessing in Python

https://medium.com/@datamonsters/text-preprocessing-in-python-steps-tools-and-examples-bf025f872908

[6] Twittwer Sentiment Analysis using Apache Flume

https://acadgild.com/blog/streaming-twitter-data-using-flume

[7] Twitter Sentiment Analysis

https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/

[8] Twitter Sentiment Analysis Using Vader Tool

https://medium.com/analytics-vidhya/simplifying-social-media-sentiment-analysis-using-vader-in-python-f9e6ec6fc52f

[9] Using AFINN Dictionary

https://acadgild.com/blog/sentiment-analysis-on-tweets-with-apache-hive-using-afinn-dictionary