TWITTER SENTIMENT ANALYSIS FOR ELECTION PREDICTION

Submitted By

SANGHAMITRA HOTA

Redg. Number: 1501106521

SHIMONA ELORA

Redg. Number: 1501106526

TULSI ACHARYA

Redg. Number: 1501106538

Under the Guidance of

Dr. Subasish Mohapatra

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING AND TECHNOLOGY, BHUBANESWAR

BIJU PATNAIK UNIVERSITY OF TECHNOLOGY, ROURKELA

2019

CERTIFICATE

This is to certify that the project entitled **Twitter Sentiment Analysis for Election Prediction** is submitted by **Tulsi Acharya** bearing registration number 1501106538, **Shimona Elora** bearing registration number 1501106526, **Sanghamitra Hota** bearing registration number 1501106521 to the Department of Computer Science and Engineering, College of Engineering and Technology, is a record bonafide research work to award partial bachelor's degree in Technology in Computer Science and Engineering under Biju Pattnaik University of Technology, Rourkela, Odisha.

Place: Bhubaneswar
Date:

Dr.Subasish Mohapatra
(HoD, Dept of CSE, CET, BBSR)

(External)

DECLARATION

We certify that the work contained in the report is original and has been done ourselves under the general supervision of my guide. The work has not been submitted to any other institute for any degree or diploma. We have followed the guidelines provided by the Institution in writing the report. Whenever I used material from other sources, we have given due credit to them by citing them in the report and giving their details. Whenever we used quotation we have used quotation marks and detailed reference.

Place: Bhubaneswar (Tulsi Acharya)

Date:

(Sanghamitra Hota)

(Shimona Elora)

ACKNOWLEDGEMENT

It is our privilege and solemn duty to express our deepest sense of gratitude to Dr. Subasish Mohapatra, Head of Department at the Department of Computer Science, under whose able guidance we carried out this work. We are indebted to him for his invaluable supervision, heart full cooperation and timely aid and advice till the completion of the report in spite of his pressing engagements. We wish to record our sincere gratitude to our respected Principal, Prof. (Dr.) P. K. Patra, for his constant support and encouragement in preparation of this report.

we take this opportunity to express our hearty thanks to all those who helped ours in the completion of our project work. We are very grateful to the author of various research papers, for helping me become aware of the research currently ongoing in the field.

We are very thankful to our parents for their constant support and love. Last, but not least, we would like to thank our classmates for their valuable comments, suggestions and unconditional support.

Place: Bhubaneswar (Tulsi Acharya)

Date:

(Sanghamitra Hota)

(Shimona Elora)

ABSTRACT

An election is conducted to view the public opinion, where a group of people choose the candidate by using votes, many methods are used to predict results. In the present day, social media platforms are playing a vital role in influencing people's sentiment in favour or against a government or an organization. We use Twitter data to analyse and predict the stand of future elections using sentiment analysis. Lexicon based approach with supervised machine learning and Naive Bayes algorithm is used to find the view point in tweets and predict sentiment score. The sentiment scores are then divided into positive, negative and neutral, to finally predict the winning party.

Keywords - Election prediction, supervised learning, sentiment analysis, Naive Bayes algorithm

LIST OF FIGURES

SERIAL NUMBER	TITLE	PAGE NUMBER
1	Histogram Example	19
2	Pie Chart Example	20
3	Web page for election analysis	24
4	Results in form of pie chart in web app	25
5	Positive and negative tweets comparison for regional parties as well as national parties	26
6	Comparison of sentiment analysis for Narendra Modi and Rahul Gandhi	31

CONTENT

SERIAL NUMBER	TITLE	PAGE NUMBER
1	INTRODUCTION	1
2	NEED OF THE PROJECT	4
3	PROCESS OF SENTIMENT ANALYSIS 3.1) DATA COLLECTION 3.2) PREPROCESSING 3.3) SENTIMENT ANALYSIS	 5 5 6
4	TOOLS USED IN THE PROJECT 4.1) PYTHON 4.2) FLASK 4.3) NLTK PACKAGE 4.4) TWEEPY 4.5) PICKLE 4.6) MATPLOTLIB	 8 9 11 16 17 18
5	STEPS INVOLVED AND CODE ANALYSIS	21
6	CONCLUSION 6.1) POPULATION 6.2) SEATS IN LOK SABHA	 26 27
7	RESULTS FOR NARENDRA MODI AND RAHUL GANDHI TWEETS	31
8	ADVANTAGES AND DISADVANTAGES OF THE PROJECT	32
9	FUTURE USE	33
10	REFERENCE	34
11	BIBLIOGRAPHY	35

INTRODUCTION

India is a federation which has a parliamentary system and is governed with the constitution of India guidelines. There is a central government which looks after the whole country and the states have their independent governments to look after the respective states.

India has the largest democracy in the world. It is a government formed by the people, of the people and for the people. One of the most efficient ways of government is a democracy if done in a free and fair way. The people elect their representatives to take decisions in the parliament and run the country. Citizens try to elect the candidate they think will bring some changes and help their community.

The general elections are held in a gap of 4 years where the citizens of India above the age of 18 are eligible to cast vote and choose the suitable candidate. The candidates are elected to be the voice of the citizens in the Lok Sabha whereas the Rajya Sabha candidates are indirectly elected. The election follows a hierarchical process where the people's representative elect the prime minister and president of India.

In less than 20 days, Indian voters from Kanyakumari to Kashmir will go to the polls to select their next parliament. The country's 2019 general election—like previous contests—will be the largest democratic exercise in world history. More than 850 million voters will be eligible to help determine which political party or alliance will form the government and, in turn, who will serve as prime minister.

With 29 states and different regional parties, it is difficult to decide which party will win the elections this time if we go by the old methods of manual queries to find out the sentiments and views of the people.

Sentiment analysis is contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product or service while monitoring online conversations. However, analysis of social media streams is usually restricted to just basic sentiment analysis and count based metrics. This is akin to just scratching the surface and missing out on those high value insights that are waiting to be discovered. So what should a brand do to capture that low hanging fruit?

With the recent advances in deep learning, the ability of algorithms to analyse text has improved considerably. Creative use of advanced artificial intelligence techniques can be an effective tool for doing in-depth research. We believe it is important to classify incoming customer conversation about a brand based on following lines:

1. Key aspects of a brand's product and service that customers care about.
2. Users' underlying intentions and reactions concerning those aspects.

These basic concepts when used in combination, become a very important tool for analyzing millions of brand conversations with human level accuracy. They are as follows:

Sentiment Analysis

Sentiment Analysis is the most common text classification tool that analyses an incoming message and tells whether the underlying sentiment is positive, negative our neutral. You can input a sentence of your choice and gauge the underlying sentiment.

Intent Analysis

Intent analysis steps up the game by analyzing the user's intention behind a message and identifying whether it relates an opinion, news, marketing, complaint, suggestion, appreciation or query.

Contextual Semantic Search (CSS)

Now this is where things get really interesting. To derive actionable insights, it is important to understand what aspect of the brand is a user discussing about. For example: Amazon would want to segregate messages that related to: late deliveries, billing issues, promotion related queries, product reviews etc. On the other hand, Starbucks would want to classify messages based on whether they relate to staff behavior, new coffee flavors, hygiene feedback, online orders, store name and location etc. But how can one do that?

We introduce an intelligent smart search algorithm called Contextual Semantic Search (a.k.a. CSS). The way CSS works is that it takes thousands of messages and a concept (like Price) as input and filters all the messages that closely match with the given concept. The graphic shown below demonstrates how CSS represents a major improvement over existing methods used by the industry.

A conventional approach for filtering all Price related messages is to do a keyword search on Price and other closely related words like (pricing, charge, $, paid). This method however is not very effective as it is almost impossible to think of all the relevant keywords and their variants that represent a particular concept. CSS on the other hand just takes the name of the concept (Price) as input and filters all the contextually similar even where the obvious variants of the concept keyword are not mentioned.

TWITTER USED FOR OPINION MINING

Microblogging today has become a very popular communication tool among Internet users. Millions of users share opinions on different aspects of life everyday. Therefore microblogging web-sites are rich sources of data for opinion mining and sentiment analysis. Because

microblogging has appeared relatively recently, there are a few research works that were devoted to this topic. In our paper, we focus on using Twitter, the most popular microblogging platform, for the task of sentiment analysis. We show how to automatically collect a corpus for sentiment analysis and opinion mining purposes. We perform linguistic analysis of the collected corpus and explain discovered phenomena. Using the corpus, we build a sentiment classifier that is able to determine positive, negative and neutral sentiments for a document. Experimental evaluations show that our proposed techniques are efficient and performs better than previously proposed methods. In our research, we worked with English, however, the proposed technique can be used with any other language.

NEED OF THE PROJECT

With so many states and over a billion people we found out the traditional way of detecting the poll is not that useful. The scientific community has turned its interest in analyzing web data, such as blog posts or social networks' users' activity as an alternative way to predict election outcomes, hopefully, more accurate. Furthermore, traditional polls are too costly, while online information is easy to obtain and freely available. This is an interesting research area that combines politics and social media which both concern today's society. It is interesting to employ technology to solve modern-day challenges.

Trying to resolve the accuracy and high-cost problem, we study the possibility of using data from social media as the data source to predict the outcome of an election. Social media has become the most popular communication tool on the internet. Hundreds of millions of messages are being posted every day in the popular social media sites such as Twitter and Facebook. Social media websites become valuable sources for opinion mining because people post everything. Users post each and every thought that they have about the current policies as well as political views. The internet is shifting from the quality and lengthy blog posts to much more numerous short posts that are posted by a lot of people. This trait is very valuable as now we can collect a different kind of people's opinions or sentiments from the social web.

One of the social media that allows researchers to use their data is Twitter. Twitter is a microblogging web service that was launched in 2006. Now, it has more than 200 million visitors on a monthly basis and 500 million messages daily. The users of Twitter can post a message (tweet) up to 140 characters. The message is then displayed at his/her personal page (timeline). Originally, tweets were intended to post status updates of the user, but these days, tweets can be about every imaginable topic.
The advantages of using tweets as a data source are as follows:
1. The number of tweets is very huge and they are available to the public.
2. Tweets contain the opinion of people including their political views.

In our project, we select data just days or weeks prior to the election, at equal intervals. The prediction could be derived by comparing the number of tweets mentioning each candidate party or by comparing the number of tweets that have positive sentiments towards each candidate party. From the sentiments, we try to find out the political sentiments of the masses and have a general idea towards whom the public views are inclined and draw a conclusion. The conclusion drawn is more accurate, time-saving and cost efficient than the previous methods.

PROCESS OF SENTIMENT ANALYSIS

This is a general description of the process that we follow through the project to obtain the tweets and analyze them.

1. DATA COLLECTION

The data collection step is the initial phase in the project, where data is collected from twitter. There are two methods on how to connect and collect tweets from Twitter. The first method is by searching tweets matching to the keywords. The second method is by collecting all the tweets provided by Twitter through streaming API, or all the tweets in a specific language, or all the tweets in a specific location, then putting all of them into our database.

Both methods have their own advantages and disadvantages. For example, the first method requires only small storage as the data is relatively small. The downside is that we cannot get data from other keywords (if we need to) from an earlier time. Twitter allows the search API only for 7 days backwards. This data collection method is suitable if the focus is on the feature extraction or the prediction method. With the second method, we can apply any set of keywords to get the best result.

As we are going for prediction and future analysis the twitter API extraction process is more helpful.

2. PREPROCESSING

Twitter analysis methods have various preprocessing steps of text. One of the most important goals of preprocessing is to enhance the quality of the data by removing noise. Another point is the reduction of the feature space size.

A. Lower Case Conversion: Because of the many ways people can write the same things down, character data can be difficult to process. String matching is another important criterion of feature selection. For accurate string matching, we are converting our complete text into lower case.

B. Removing Punctuations and Removing Numbers: All punctuations, numbers also need to be removed from reviews to make data clean and neat. Unnecessary commas, question marks, other special symbols get removed in this case. Here, we are not removing the dot (.) symbol from our reviews because we are splitting our text into sentences.

C. Stemming: Stemming is the method of conflating the variant styles of a word into a standard illustration, the stem. For example, the words: “presentation”, “presented”, “presenting” could all be reduced to a common representation “present”. This is a widely used procedure in text processing for information retrieval (IR) based on the assumption that posing a query with the term presenting implies an interest in documents containing the words presentation and

presented. Stemming in our case is helpful in correct word matching and counting case.

D. Striping White Spaces: In this preprocessing step all text data is cleaned off. All unnecessary white spaces, tabs, newline character get removed from the text.

3. SENTIMENT ANALYSIS

a) Machine Learning Approach

There are two approaches to machine learning, supervised and unsupervised. in our project we used a supervised machine learning approach. In a supervised machine learning approach, there is a finite set of classes for classification. Training dataset is also available. Most research papers do not use the neutral class, which makes the classification problem considerably easier, but it is possible to use a neutral class. Given the training data, the system classifies the document by using one of the common classification algorithms such as Support Vector Machine, Naïve Bayes etc. We used Naive Bayes algorithm for classification of tweets. We classified tweets into polarity and emotion also using Naive Bayes classifier. Naive Bayes is a machine learning algorithm for classification problems. It is based on Bayes' probability theorem. It is primarily used for text classification that involves high dimensional knowledge sets. A few examples are spam filtration, sentimental analysis, and classifying news articles. It is not only known for its simplicity, but also for its effectiveness. It is fast to build models and make predictions with the Naive Bayes algorithm.

$$P(A|B)=\frac{P(B|A)P(A)}{P(B)}$$

Where P(A|B): Probability (conditional probability) of occurrence of the event given the event B is true. P(A) and P(B): Probabilities of the occurrence of event A and B respectively. P(B|A): Probability of the occurrence of event B given the event A is true.

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

b) Lexicon Based Approach:

There are three main approaches to compiling sentiment words. They are the manual approach, dictionary-based approach, and corpus-based approach. In our research, we used a dictionary-based approach. We used eleven different variables for classification, that variables are sadness, tentativeness, anxiety, work, anger, certainty, achievement, positive words, negative words, positive hashtag and negative hashtag. We collected various word related to those eleven variables and classified them.

TOOLS USED IN THE PROJECT

To build a program to take the tweets in on a weekly basis and analyse the value according to the tweets we need various languages as well as packages. They are as follows:

1. PYTHON

Python is a general-purpose object-oriented programming language with high-level programming capabilities. It has become famous because of its apparent and easily understandable syntax, portability and easy to learn nature. Python is a programming language that includes features of C and Java. It provides the style of writing an elegant code like C, and for object-oriented programming, it offers classes and objects like Java. A Python page is a file with a .py extension.

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

The features and characteristic of python which makes it user-friendly are:

- Interpreted Language: Python is processed at runtime by Python Interpreter.
- Object-Oriented Language: It supports object-oriented features and techniques of programming.
- Interactive Programming Language: Users can interact with the python interpreter directly for writing programs.
- Easy language: Python is an easy to learn language especially for beginners.
- Straightforward Syntax: The formation of python syntax is simple and straightforward which also makes it popular.
- Easy to read: Python source-code is clearly defined and visible to the eyes.
- Portable: Python codes can be run on a wide variety of hardware platforms having the same interface.
- Extendable: Users can add low level-modules to Python interpreter.
- Scalable: Python provides an improved structure for supporting large programs then shell-scripts.

2. FLASK

Flask is a web application framework written in Python. A Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications. Werkzeug is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases. Jinga2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layers for database handling, nor does it have form validation support. Instead, Flask supports the extensions to add such functionality to the application. Importing flask module in the project is mandatory for enabling web development applications. An object of Flask class is our WSGI application.

Flask constructor takes the name of current module (__name__) as argument. The route() function of the Flask class is a decorator, which tells the application which URL should call the associated function.

- **app.route(rule, options)**

- The rule parameter represents URL binding with the function.
- The options is a list of parameters to be forwarded to the underlying Rule object.
- The route() decorator in Flask is used to bind URL to a function.
- For example:

```
@app.route('/home')
def hello_world():
        return 'Hello World'
```

Here, URL '/hello' rule is bound to the hello_world() function. As a result, if a user visits http://localhost:5000/hello URL, the output of the hello_world() function will be rendered in the browser.

3. NLTK Package

NLTK is one of the leading platforms for working with human language data and Python, the module NLTK is used for natural language processing. NLTK is literally an acronym for Natural Language Toolkit.

We can install NLTK using python 2.x and above by using the following command:

```
sudo pip install nltk
```

Then we open python and write:

```
import nltk
nltk.download()
```

A sentence or data can be split into words using the method **word_tokenize():**

```
from nltk.tokenize import sent_tokenize, word_tokenize

data = "All work and no play makes jack a dull boy, all work and no play"
print(word_tokenize(data))
```

This will output:

```
['All', 'work', 'and', 'no', 'play', 'makes', 'jack', 'dull', 'boy', ',', 'all', 'work', 'and', 'no',
'play']
```

All of them are words except the comma. Special characters are treated as separate tokens. Text may contain stop words like ‘the’, ‘is’, ‘are’. Stop words can be filtered from the text to be processed. There is no universal list of stop words in NLP research, however the NLTK module contains a list of stop words.

To find out the stop words we can do the following:

A module has to be imported:

```
from nltk.corpus import stopwords
```

We get a set of English stop words using the line:

```
stopWords = set(stopwords.words('english'))
```

The returned list stopWords contains 153 stop words on the tested computer. You can view the length or contents of this array with the lines:

```
print(len(stopWords))
print(stopWords)
```

NLTK is also used for word stemming. For it we import packages as follows:

```
from nltk.stem import PorterStemmer
from nltk.tokenize import sent_tokenize, word_tokenize
```

DESIGN OF NLTK

Several criteria were considered in the design and implementation of the toolkit. These design criteria are listed in the order of their importance. They are as follows:

1. Ease of Use - The primary purpose of the toolkit is to allow students to concentrate on building natural language processing (NLP) systems. The more time students must spend learning to use the toolkit, the less useful it is.

2. Consistency - The toolkit should use consistent data structures and interfaces.

3. Extensibility - The toolkit should easily accommodate new components, whether those components replicate or extend the toolkit's existing functionality. The toolkit should be structured in such a way that it is obvious where new extensions would fit into the toolkit's infrastructure.

4. Documentation - The toolkit, its data structures, and its implementation all need to be carefully and thoroughly documented. All nomenclature must be carefully chosen and consistently used.

5. Simplicity - The toolkit should structure the complexities of building NLP systems, not hide them. Therefore, each class defined by the toolkit should be simple enough that a student could implement it by the time they finish an introductory course in computational linguistics.

6. Modularity - The interaction between different components of the toolkit should be kept to a minimum, using simple, well-defined interfaces. In particular, it should be possible to complete individual projects using small parts of the toolkit, without worrying about how they interact with the rest of the toolkit. This allows students to learn how to use the toolkit incrementally throughout a course. Modularity also makes it easier to change and extend the toolkit.

MODULES OF NLTK

The toolkit is implemented as a collection of independent modules, each of which defines a specific data structure or task. A set of core modules defines basic data types and processing systems that are used throughout the toolkit. The token module provides basic classes for processing individual elements of text, such as words or sentences. The tree module defines data structures for representing tree structures over text, such as syntax trees and morphological trees. The probability module implements classes that encode frequency distributions and probability distributions, including a variety of statistical smoothing techniques. The remaining modules define data structures and interfaces for performing specific NLP tasks. This list of modules will grow over time, as new tasks are added and algorithms to the toolkit. The different modules are as follows:

1. Parsing Modules – The parser module defines a high-level inter-face for producing trees that represent the structures of texts. The chunk parser module defines a sub-interface for parsers that identify non-overlapping linguistic groups (such as base noun phrases) in unrestricted text.
   Four modules provide implementations for these abstract interfaces. The srparser module implements a simple shift-reduce parser. The chartparser module defines a flexible parser that uses a chart to record hypotheses about syntactic constituents. The pcfgparser module provides a variety of different parsers for probabilistic grammars. And the rechunkparser

module defines a transformational regular-expression based implementation of the chunk parser interface.

2. Tagging Modules – The tagger module defines a standard interface for augmenting each token of a text with supplementary information, such as its part of speech or its WordNet synset tag; and provides several different implementations for this interface.

3. Finite State Automata – The fsamodule defines a data type for encoding finite state automata; and an interface for creating automata from regular expressions.

4. Type Checking - Debugging time is an important factor in the toolkit's ease of use. To reduce the amount of time students must spend debugging their code, we provide a type checking module, which can be used to ensure that functions are given valid arguments. The type checking module is used by all of the basic data types and processing classes. Since type checking is done explicitly, it can slow the toolkit down. However, when efficiency is an issue, type checking can be easily turned off; and with type checking is disabled, there is no performance penalty.

5. Visualization - Visualization modules define graphical interfaces for viewing and manipulating data structures, and graphical tools for experimenting with NLP tasks. The draw.tree module provides a simple graphical interface for displaying tree structures. The draw.treeedit module provides an interface for building and modifying tree structures. The draw.plotgraph module can be used to graph mathematical functions. The draw.fsa module provides a graphical tool for displaying and simulating finite state automata. The draw.chart module provides an interactive graphical tool for experimenting with chart parsers. The visualization modules provide interfaces for interaction and experimentation; they do not directly implement NLP data structures or tasks. Simplicity of implementation is therefore less of an issue for the visualization modules than it is for the rest of the toolkit.

6. Text Classification – The classifier module defines a standard interface for classifying texts into categories. This interface is currently implemented by two modules. The classifier.naivebayes module defines a text classifier based on the Naive Bayes assumption. The classifier.maxent module defines the maximum entropy model for text classification, and implements two algorithms for training the model: Generalized Iterative Scaling and Improved Iterative Scaling. The classifier.feature module provides a standard encoding for the information that is used to make decisions for a particular classification task. This standard encoding allows students to experiment with the differences between different text classification algorithms, using identical feature sets. The classifier.feature

selection module defines a standard interface for choosing which features are relevant for a particular classification task. Good feature selection can significantly improve classification performance.

4. TWEEPY

The API class provides access to the entire twitter RESTful API methods. Each method can accept various parameters and return responses. Tweepy is open-sourced, hosted on GitHub and enables Python to communicate with Twitter platform and use its API. Installing tweepy is easy, it can be cloned from the Github repository:

```
git clone https://github.com/tweepy/tweepy.git
python setup.py install
```

Or using easy install:

```
pip install tweepy
```

Tweepy provides access to the well documented Twitter API. With tweepy, it's possible to get any object and use any method that the official Twitter API offers. For example, a User object has its documentation at https://dev.twitter.com/docs/platform-objects/users and following those guidelines, tweepy can get the appropriate information.

Main Model classes in the Twitter API are Tweets, Users, Entities and Places. Access to each returns a JSON-formatted response and traversing through information is very easy in Python.

One of the main usage cases of tweepy is monitoring for tweets and doing actions when some event happens. Key component of that is the StreamListener object, which monitors tweets in real time and catches them.

StreamListener has several methods, with on_data() and on_status() being the most useful ones.

5. PICKLE

Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it "serializes" the object first before writing it to a file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.

Advantages of using Pickle Module:

1. Recursive objects (objects containing references to themselves): Pickle keeps track of the objects it has already serialized, so later references to the same object won't be serialized again. (The marshal module breaks for this.)

2. Object sharing (references to the same object in different places): This is similar to self- referencing objects; pickle stores the object once, and ensures that all other references point to the master copy. Shared objects remain shared, which can be very important for mutable objects.

3. User-defined classes and their instances: Marshal does not support these at all, but pickle can save and restore class instances transparently. The class definition must be importable and live in the same module as when the object was stored.

6. MATPLOTLIB

Matplotlib is a 2-D plotting library that helps in visualizing figures. Matplotlib emulates Matlab like graphs and visualizations. Matlab is not free, is difficult to scale and as a programming language is tedious. So, matplotlib in Python is used as it is a robust, free and easy library for data visualization.

Matplotlib has a module called pyplot which aids in plotting figure. The Jupyter notebook is used for running the plots. We import matplotlib.pyplot as plt for making it call the package module.

- Importing required libraries and dataset to plot using Pandas *pd.read_csv()*
- Extracting important parts for plots using conditions on Pandas Dataframes.
- *plt.plot()* for plotting line chart similarly in place of plot other functions are used for plotting. All plotting functions require data and it is provided in the function through parameters.
- *plot.xlabel , plt.ylabel* for labeling x and y-axis respectively.
- *plt.xticks , plt.yticks* for labeling x and y-axis observation tick points respectively.
- *plt.legend()* for signifying the observation variables.
- *plt.title()* for setting the title of the plot.
- *plot.show()* for displaying the plot.

Histogram

A histogram takes in a series of data and divides the data into a number of bins. It then plots the frequency data points in each bin (i.e. the interval of points). It is useful in understanding the count of data ranges.

When to use: We should use histogram when we need the count of the variable in a plot.

```
import matplotlib.pyplot as plt
 from numpy.random import normal,rand
 x = normal (size=200)
 Plt.hist (x, bins=30)
 plt.show()
```

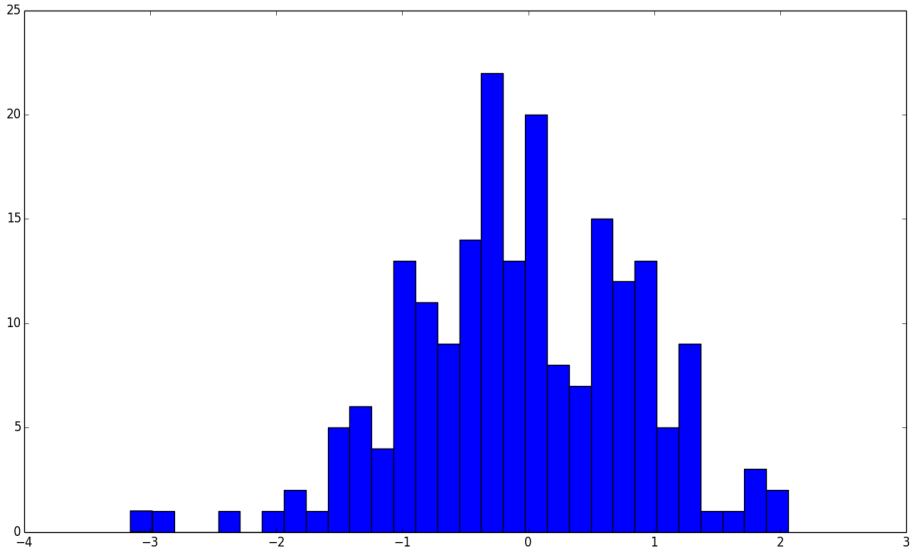


Fig 1: Histogram Example

Pie Chart

It is a circular plot which is divided into slices to illustrate numerical proportion. The slice of a pie chart is to show the proportion of parts out of a whole.

When to use: Pie chart should be used seldom used as It is difficult to compare sections of the chart. Bar plot is used instead as comparing sections is easy.

```
import matplotlib.pyplot as plt

# Data to plot
labels = 'Python', 'C++', 'Ruby', 'Java'
sizes = [215, 130, 245, 210]
colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue']
explode = (0.1, 0, 0, 0)  # explode 1st slice

# Plot
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
autopct='%1.1f%%', shadow=True, startangle=140)

plt.axis('equal')
plt.show()
```

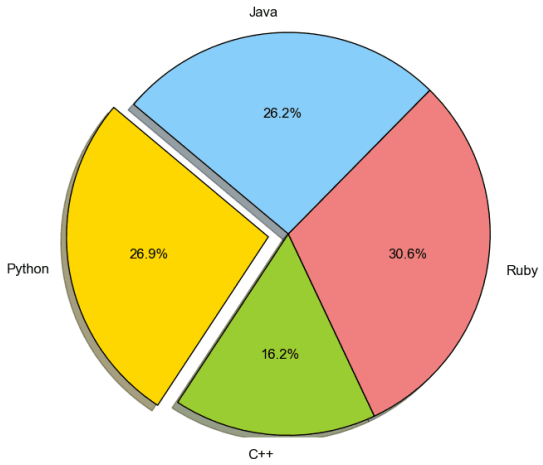


Fig 2: Pie chart Example

STEPS INVOLVED AND CODE ANALYSIS

1. Extracting tweets: The steps involved in extracting tweets is :

i) A twitter account is created. Then we login with twitter's developer account.

ii) Tweepy: provides a cursor interface to twitter web api.

iii) OAuth: to authorise our app to access twitter .

iv) Create twitter authenticated credential object: It can be done using the consumer key, access key,access secret and consumer secret.

```
import secret #files in which keys are stored
import tweepy

# authenticating with authentication variables
auth = tweepy.OAuthHandler(secret.consumer_key, secret.consumer_secret)
auth.set_access_token(secret.access_token, secret.access_token_secret)
api = tweepy.API(auth, wait_on_rate_limit=True,
wait_on_rate_limit_notify=True)
```

2. Managing the number of tweets to be searched:

i) We took input from user about what party he want to search, number of tweets and name of state or city.

ii) With the tweepy's geo-location fuction, we convert the state name or city name to a geo-location ID accepted by the search function of the API.

```
# inputs for counts taken
hash_tag = input("Which party you want to search ? \n")
number = int(input("How many Tweets do you want to analyze? \n"))
location = input("What is the location of party? \n")

# Getting Geo ID for Places
places = api.geo_search(query=location)
```

iii) As the number of tweets retrieved per page was just 100, we tried to increase the number of tweets, by using the concept of max_ids which is relevant for the tweepy API.

```
tweetsPerQry = 100  # this is the max the API permits

# If results from a specific ID onwards are reqd, set since_id to that ID.
# else default to no lower limit, go as far back as API allows
sinceId = None

# If results only below a specific ID are, set max_id to that ID.
# else default to no upper limit, start from the most recent tweet matching the search query.
max_id = -1
```

```
while tweetCount < number: #For obtaining tweets upto a the input number
  try:
        if (max_id <= 0): #incase the loop is first entered
              if (not sinceId):
                    new_tweets = api.search(q=hash_tag, count=tweetsPerQry,
                    tweet_mode='extended', lang='en', place=places[0].id)
              else:
                    new_tweets = api.search(q=hash_tag, count=tweetsPerQry,
                    since_id=sinceId, tweet_mode='extended', lang='en', place=places[0].id)
        else:
              if (not sinceId): #incase we already have a max_id after which we need the tweets
                    new_tweets = api.search(q=hash_tag, count=tweetsPerQry,
                    max_id=str(max_id - 1), tweet_mode='extended', lang='en',
                    place=places[0].id)
              else:
                    new_tweets = api.search(q=hash_tag, count=tweetsPerQry,
                    max_id=str(max_id - 1),
                    since_id=sinceId, tweet_mode='extended', lang='en', place=places[0].id)
              if not new_tweets:
                    print("No more tweets found")
                    break

        tweets = new_tweets
        for tweet in tweets: #used for storing data as a dataset
              dataset.append(tweet.full_text)
        max_id = new_tweets[-1].id #updating max_id to last tweet's ID
  except tweepy.TweepError as e:  #incase of any errors
        print(str(e))
        break
```

3. Cleaning of data:

The tweets are cleaned in python by removing:

- Extra punctuation
- Stop words (Most commonly used words in a language like the, is, at, which, and on.)
- Redundant Blank spaces
- Emoticons
- URLS

4. Sentiment Analysis:

i) Vader sentiment analysis is used to analyse if the data are positive or negative using the data semantics.

ii) Calculation of polarity score is done.

iii) Using only the polarity score of compound the positive negative tweets are found.

```
# sentiment analysis using vader sentiment anlaysis
import nltk
nltk.download('vader_lexicon')

sid = SentimentIntensityAnalyzer()

l = []
counter = Counter()

for data in dataset:
    ss = sid.polarity_scores(data) # polarity score is calculated
    l.append(ss)
    k = ss['compound'] #From the compound score, positive, negative and neutral is calculated.
    if k >= 0.05:
        counter['positive'] += 1
    elif k <= -0.05:
        counter['negative'] += 1
    else:
        counter['neutral'] += 1
```

iv) If polarity score is less than -0.05, the statement is negative, if it is greater than 0.05 the positive, if it is in between them, then neutral.

5. Visualising in a pie chart:

i) The results were visualized with the help of pie chart.

```
import matplotlib

colors = ['green', 'red', 'grey']
sizes = [positive, negative, neutral]
labels = 'Positive', 'Negative', 'Neutral'

# use matplotlib to plot the chart
plt.pie(
    x=sizes,
    shadow=True,
    colors=colors,
    labels=labels,
    startangle=90,
    autopct='%.1f%%'
)

plt.title("Sentiment of {} Tweets about {}".format(number, hash_tag))
plt.show()
```

6. Deploying the project:

i) The was deployed on a web interface using Flask and gunicorn on heroku server. (The link for the web app is : https://election-predictions.herokuapp.com/)

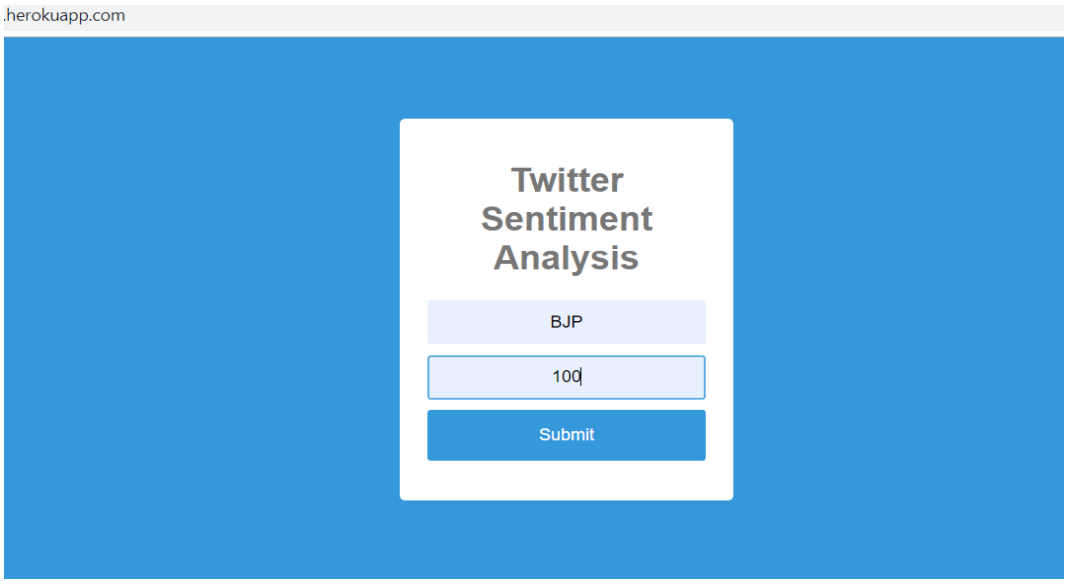


Fig 3: Web page for election analysis

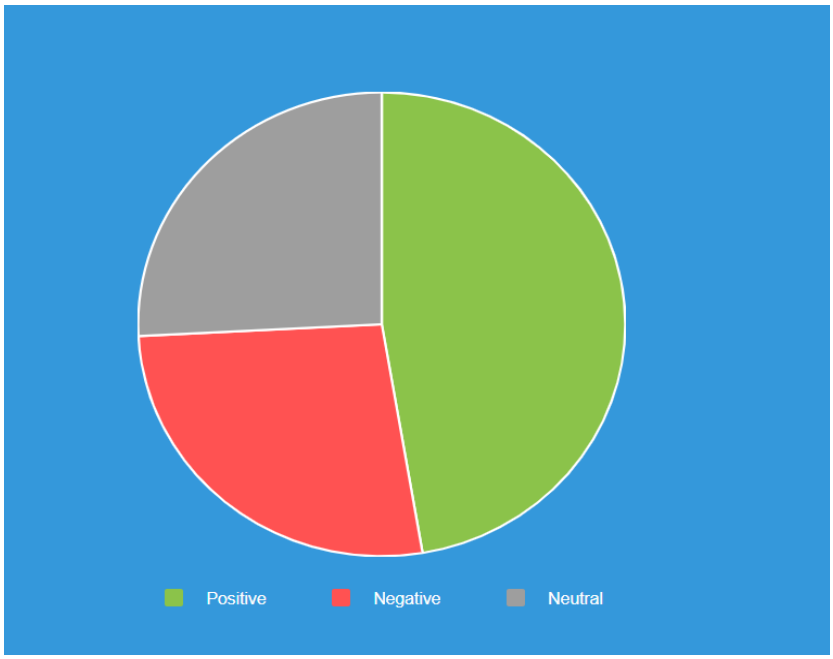


Fig 4: Results in form of pie chart in web app

CONCLUSION

We took two ways to predict the results of elections

1. **Using the population criteria**
    - As the variation by taking the ratios of positive to negative tweets was almost same. We tried considering the population factor into account for generalization purpose. We manipulated the total number of tweets as well as positive and negative tweets by dividing them with the current population percentage of the state.
    - The results derived from it showed some deviation for the national parties and the regional parties of some states.
    - The regional parties having less number of followers showed negligible tweets about them as a result the positive and negative tweets about them was unclear.

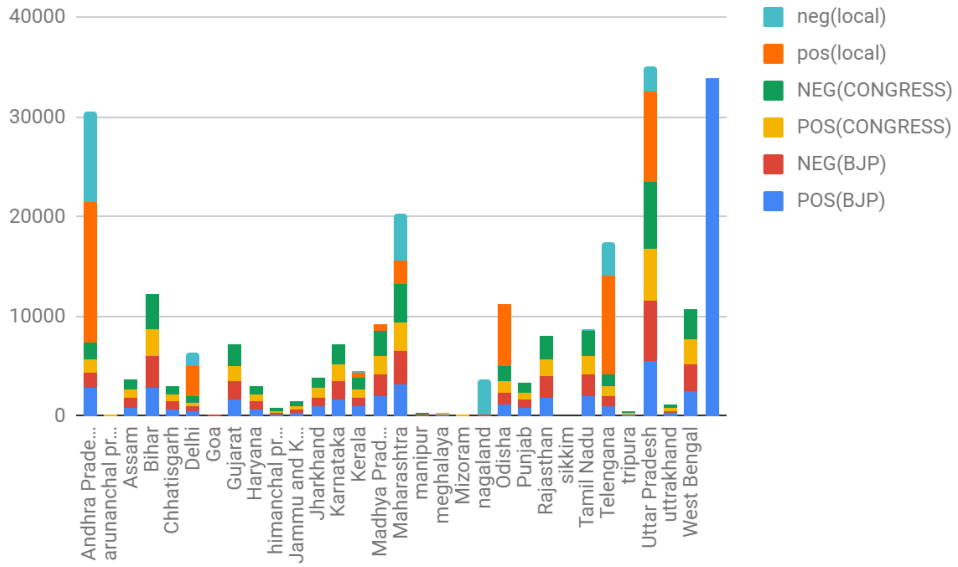


Figure 5: Positive and negative tweets comparison for regional parties as well as national parties

So we can see that,

1. States like Andhra Pradesh, West Bengal, Odisha, Tamil Nadu and Telangana have more positive sentiments as compared to negative sentiments so we can assume that winning party will be the local parties.
2. States like Gujarat, Goa, Madhya Pradesh, Uttar Pradesh and Uttarakhand have an inclination towards BJP government, giving them the winning points from those states.
3. States like Rajasthan, Haryana and Assam have a majority of congress.

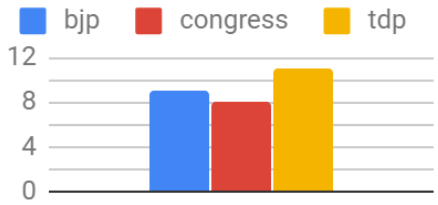
According to the graph we can take a total number of positive tweets of bjp to be 33811 and congress to be 32110.As we are calculating the winning in the lok sabha election and the major parties contesting are BJP and congress we can clearly see that BJP is winning over the people's hearts and according to that BJP wins the lok sabha election.
But there are certain drawbacks when we consider this method of calculation, which are discussed as follows:

1. When we consider the total population we are not considering the people eligible to vote.
2. There are many people whose view might change during election.
3. There are states with so less population that there vote affecting is negligible in this method.

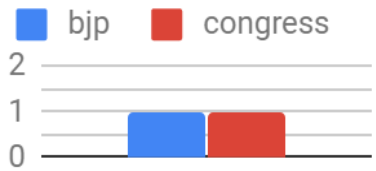
So, we chose to do the analysis in another process to get better and accurate results.

2. Using the seats in Lok Sabha

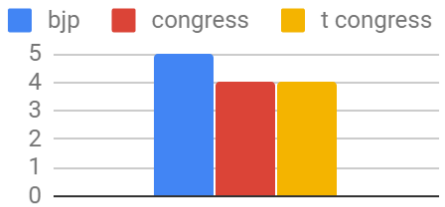
- The number of Lok Sabha seats are 543 which is contributed by each state. Representative from each state is elected for these seats. Party getting more than 271 get a majority to form the government or else parties form coalitions to get the simple majority .
- So, the total tweets was taken and the positive tweets was assumed to be as seats for the party (calculating the percentage of positive tweet and multiplying it with the seats a single states contribute).Then we calculated it for individual states and the local parties.
- Even if the local parties win in the states they affect the total seats in the Lok Sabha according to the seat percentage which gave a much clearer idea about which party would win how many seats in the election.
- The results of each state is shown below :



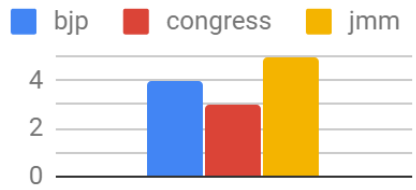
Andra Pradesh(TDP wins)



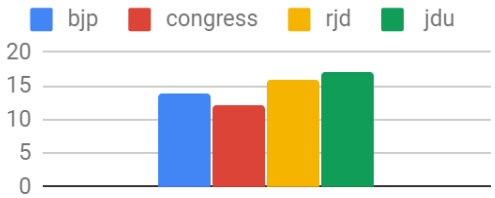
Aruachal Pradesh



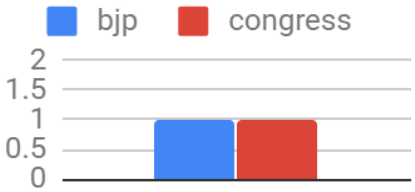
Assam(BJP wins)



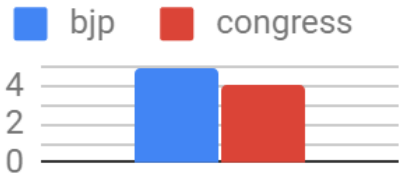
Chhattisgarh(JMM wins)



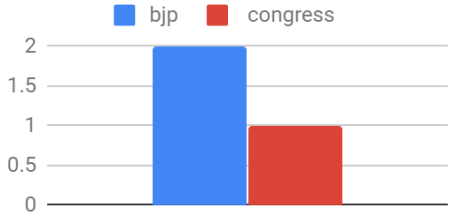
Bihar (JDU wins)



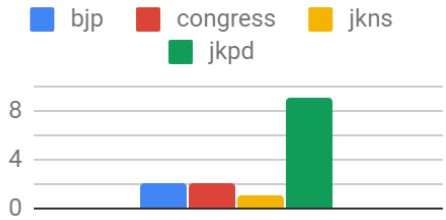
Goa (BJP=1, Congress=1)



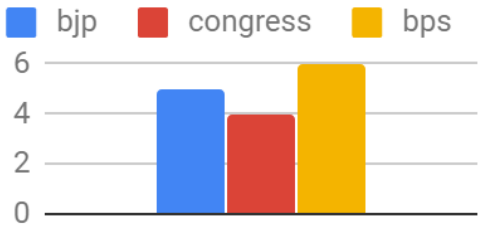
Haryana(BJP wins)



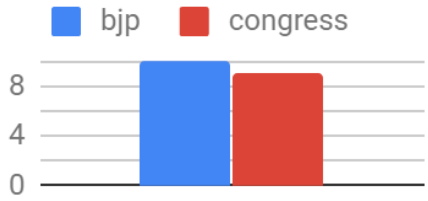
Himachal Pradesh(BJP wins)



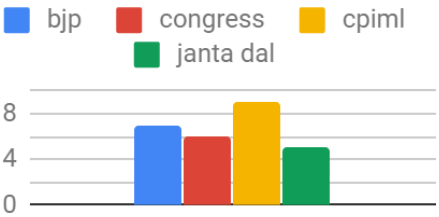
Jammu and Kashmir(JKPD wins)



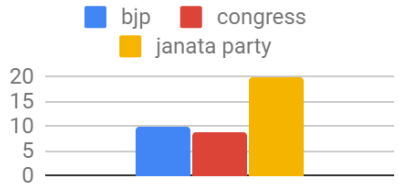
Jharkhand(BPS wins)



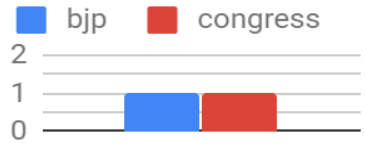
Karnataka(BJP wins)



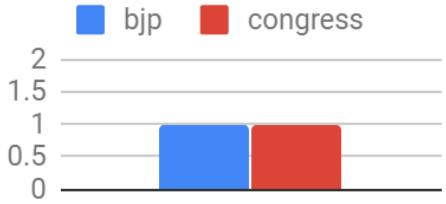
Kerela(CPIML wins)



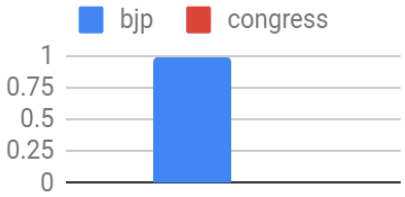
Madhya Pradesh(janta party wins



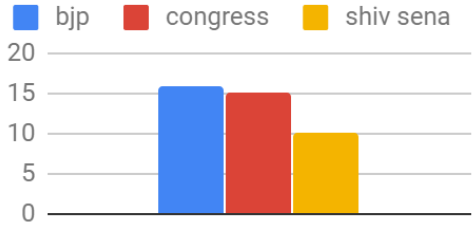
Meghalaya



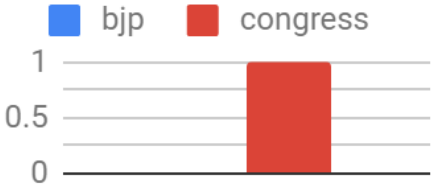
Manipur



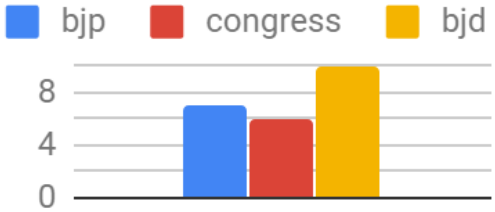
Mizoram(BJP wins)



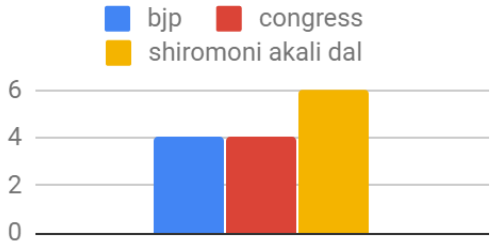
Maharashtra(BJP wins)



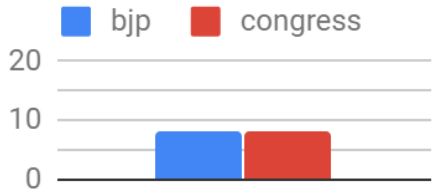
Nagaland(Congress wins)



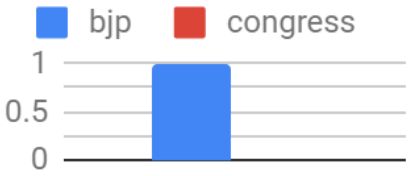
Odisha(BJD wins)



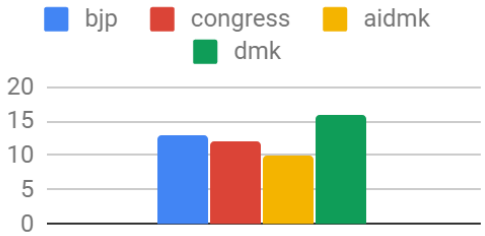
Punjab(SAD wins)



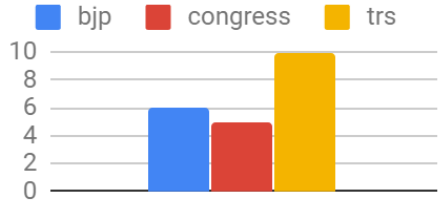
Rajasthan



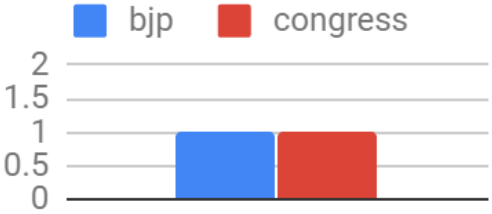
Sikkim(BJP wins)



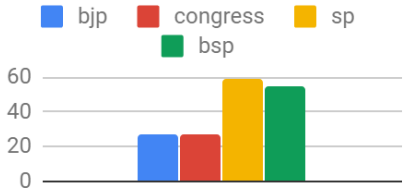
Tamil Nadu(AIDMK wins)



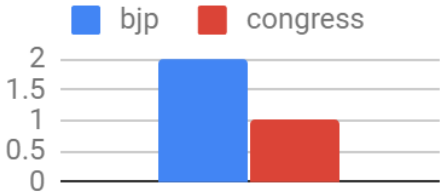
Telengana(TRS wins)



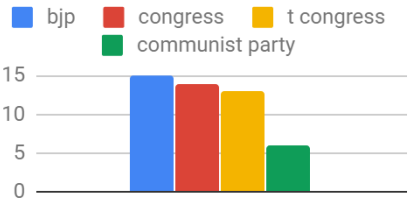
Tripura



Uttar Pradesh(SP wins)



Uttrakhand(BJP wins)



West Bengal(BJP wins

- From the value obtained by normalizing these data as can be seen from the graph we find out that *BJP wins 178 seats* in Lok Sabha and *congress wins 168 seats* in Lok Sabha. The parties have alliances with other local parties which is not taken into consideration in our project.
- If we see from the above data obtained and calculate we get that *BJP has more number of seats* so it would win the Lok Sabha elections but to have a simple majority it needs to align which other parties.

RESULTS FOR MODI AND RAHUL GANDHI TWEETS

Another way to analyze the whole prediction scenario is to compare the tweets between the leaders of the major national parties participating in the elections. This way the major sentiment of the masses towards the leaders that they are going to choose can be predicted easily.
This was performed by running the twitter sentiment analysis program with two search phrases: “Narendra Modi”, for BJP, and “Rahul Gandhi”, for Congress. The results for the two searches are as follows:

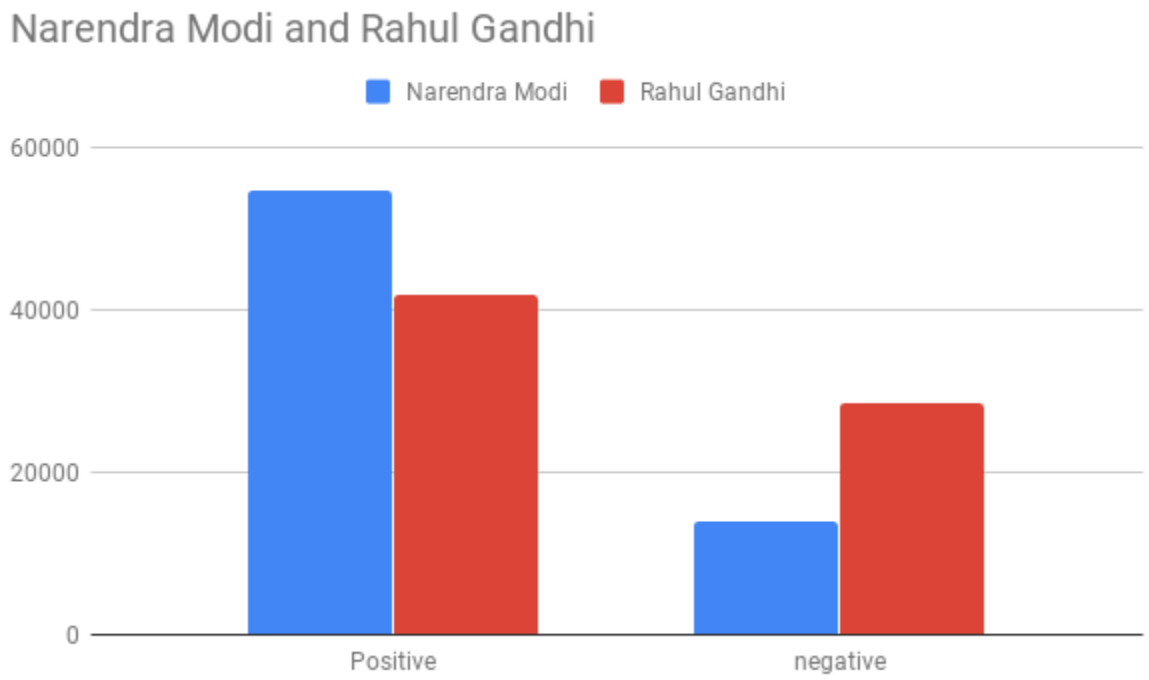


Figure 6: Comparison of sentiment analysis for Narendra Modi and Rahul Gandhi

As can be seen in the diagram, the competition is neck-to-neck between the two leaders of the major parties. However, Narendra Modi has slightly more positive and slightly less negative tweets, tipping the scale of favor towards Narendra Modi. This method however does not take into account all the different names used to reference the two.

ADVANTAGES AND DISADVANTAGES OF THE PROJECT

The advantages of the project are as follows:

- The number of tweets were increased upto 1 Lakh to get better analysis of result whereas tweepy could only retrieve 100 tweets per page.
- Analysing it using the populations and seat method gave an accuracy of nearly 90%.
- The website was deployed in heroku server for public use, which can be used for getting sentiment analysis for any phrase or word.

The disadvantages of the project are as follows:

- The long retweets couldn't be retrieved fully, as a result it was represented by "...", so the algorithm analyses it to be of neutral sentiment .
- Sarcasm was not detected in some sentences due to misuse of the semantics.
- The twitter search api could retrieve data only of past 7 days.
- Giving hashtags for shorthand representation of the party gave ambiguous results in some cases.
- Cannot get 100% accuracy in analysing the tweets.

FUTURE USE

With the help of sentiment analysis we can not only predict the outcome of the election but also use it in different fields:

- It can be used in the field of education to know what the response of the students and parents is towards the change in the curriculum and what improvements are expected by the people.
- It can be used in the field of medicine to know the use and side effects caused due to the introduction of a particular drug in the market. It also helps to know about the new diseases and the cures that can be found by the companies.
- The governments can know the response of the people about the new schemes and plans introduced by them and how it is helping the society to grow.
- It can be used by companies from IT sector to know the inclination of the present youth and develop their schemes accordingly.

REFERENCES

1. Election Vote Share Prediction using a Sentiment-based Fusion of Twitter Data with Google Trends and Online Polls, By - Parnian Kassraie1, Alireza Modirshanechi and Hamid K. Aghajan, Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran, Islamic Republic of imec, Department of Telecommunications and Information Processing, University of Gent, Gent, Belgium.
2. Twitter Based Election Prediction and Analysis, By Pritee Salunkhe, Sachin Deshmukh, Department of Computer Science and Information Technology, Dr.Babasaheb Ambedkar Marathwada University, Maharashtra, India.
3. Election result prediction using Twitter sentiment analysis(IEEE), By Jyoti Ramteke, Samarth Shah, Darshan Godhia, Aadil Shaikh
4. Natural language processing(IEEE), By A. Gelbukh
5. Loper, Edward & Bird, Steven. (2002). NLTK: the Natural Language Toolkit. CoRR. cs.CL/0205028. 10.3115/1118108.1118117.

BIBLIOGRAPHY

1. https://chatbotslife.com/twitter-data-mining-a-guide-to-big-data-analytics-using-python-4efc8ccfa219

2. https://www.digitalvidya.com/blog/twitter-sentiment-analysis-introduction-and-techniques/

3. https://towardsdatascience.com/the-real-world-as-seen-on-twitter-sentiment-analysis-part-one-5ac2d06b63fb

4. https://www.analyticsvidhya.com/blog/2018/02/the-different-methods-deal-text-data-predictive-python/