Twitter Sentiment analysis

**ABSTRACT**

In this current era, social media plays an  important role in data exchange, sharing their thoughts. Emotional Effect of a person maintains an important roleinn  their day to day life. Sentiment Analysis is a procedure of analyzing the opinions and polarity of thoughts of the person. Twitter is a main platform on sharing the thought's, opinion and sentiments on different occasions. Twitter Sentimental Analysis is method of analyzing the emotions from tweets (message posted by user in twitter).

  Tweets are helpful in extracting the Sentimental values from the user. The data provide the Polarity indication like positive, negative or unbiassed values. It is focused on the person’s tweets and the hash tags for understanding the situations in each aspect of the criteria. The paper is to analyse the famous person’s id’s  or hash tags (#IPL2018) for understanding the mindset of people in each situation when the person has tweeted or has acted upon some incidents. The proposed system is to analyze the sentiment of the people using python, twitter API, Text Blob . As the results it helps to analysis the post with a better accuracy.

**Keywords:**

*Sentiment analysis, Social Media, Machine-learning approach, Lexicon-based approach, Sentiment classification*

TABLE OF CONTENT

**CHAPTER NO. TITLE PAGE**

**NUMBER**

**ABSTRACT**

**LIST OF FIGURES**

1 **INTRODUCTION 1**

1.1 Problem definition 1

1.2 Existing applications 1

1.3 Need for the system 3

2 **LITERATURE SURVEY 4**

2.1 Lexicon-Based Approach for

real-time analysis of trending event 6

2.2 Real-time sentimental analysis on Twitter 7

2.3 Discovering public opinion 7

2.4 Production prediction 8

2.5 Handling of Voluminous Tweets 8

2.6 Sentiment Analysis on Twitter

using streaming API 8

2.7 Classification of COVID 19

Tweets 9

2.8 A Novel approach for real

time Sentiment analysis 9

# 2.9 Modelling Recommendation

# System for Real Time Analysis

# 

# of social media Dynamics 9

1. **SYSTEM ANALYSIS 11**

[System requirements 11](#_heading=h.gjdgxs)

Requirements 11

# [Packages 12](#_heading=h.30j0zll)

**4 SYSTEM DESIGN 14**

Structural diagram 14

Behavioral diagram 15

[Use case diagram](#_heading=h.3znysh7) 15

Architectural diagram 16

**5 SYSTEM IMPLEMENTATION 19**

Anaconda and Jupyter notebook 19

[Twitter API](#_heading=h.tyjcwt) 20

Data cleaning and Visualization 20

[Introduction to NLP](#_heading=h.3dy6vkm) 22

1. **PERFORMANCE METRICS 25**

**7 CONCLUSION AND FUTURE WORK 26**

* 1. [Conclusion](#_heading=h.2et92p0)

[Future work](#_heading=h.1t3h5sf)

**APPENDIX 28**

**APPENDIX A: CODE**

**APPENDIX B: SNAPSHOTS**

**REFERENCE** 37

# CHAPTER 1 INTRODUCTION

* 1. **PROBLEM DEFINITION**

# Sentiment analysis of in the domain of micro-blogging is a relatively new research topic so there is still a lot of room for further research in this area. Decent amount of related prior work has been done on sentiment analysis of user reviews, documents, web blogs/articles and general phrase level sentiment analysis. These differ from twitter mainly because of the limit of 140 characters per tweet which forces the user to express opinion compressed in very short text. The best results reached in sentiment classification use supervised learning techniques such as Naive Bayes and Support Vector Machines, but the manual labelling required for the supervised approach is very

# expensive. Some work has been done on unsupervised and semi-supervised approaches, and there is a lot of room of improvement. Various researchers testing new features and classification techniques often just compare their results to base-line performance. There is a need of proper and formal comparisons between these results arrived through different features and classification techniques in order to select the best features and most efficient classification techniques for particular applications.

# 

# 1.2 EXISTING APPLICATIONS

1. **Brand Monitoring and Reputation Management using Machine Learning**

Brand monitoring is one of the most popular applications of sentiment analysis in business. Bad reviews can snowball online, and the longer you leave them the worse the situation will be. With [sentiment analysis tools](https://monkeylearn.com/blog/sentiment-analysis-with-an-online-tool/), you will be notified about negative brand mentions immediately.

Not only that, you can keep track of your brand’s image and reputation over time or at any given moment, so you can monitor your progress. Whether monitoring news stories, blogs, forums, and social media for information about your brand, you can transform this data into usable information and statistics.

# Customer Support using Natural Language Understanding (NLU)

[Customer support management](https://monkeylearn.com/blog/customer-experience-management/) presents many challenges due to the sheer number of requests, varied topics, and diverse branches within a company – not to mention the urgency of any given request.

Sentiment analysis with [natural language understanding (NLU)](https://monkeylearn.com/blog/natural-language-understanding/) reads regular human language for meaning, emotion, tone, and more, to understand customer requests, just as a person would. You can automatically process customer support tickets, online chats, phone calls, and [emails by sentiment](https://monkeylearn.com/blog/email-sentiment-analysis/) to prioritize any urgent issues.

### **c)**  **Business Intelligence Buildup using Machine Learning (ML)**

Digital marketing plays a prominent role in business. Social media often displays the reactions and reviews of the product. When you are available with the sentiment data of your company and new products, it is a lot easier to estimate your customer retention rate.

Sentiment analysis enables you to determine how your product performs in the market and what else is needed to improve your sales. You can also analyze the responses received from your competitors. Based on the survey generated, you can satisfy your customers needs in a better way. You can make immediate decisions that will help you to adjust to the present market situation.

# 1.3 NEED FOR THE SYSTEM

Sentiment analysis can improve customer loyalty and retention through better service outcomes and customer experience.

A customer makes a support request through email or chat. The NLP machine learning model generates an algorithm that performs sentiment analysis of the text from the customer's email or chat session. The algorithm detects the customer's emotional state. In this scenario, the customer feels agitation. Business rules related to this emotional state set the customer service agent up for the appropriate response. In this case, immediate upgrade of the support request to highest priority and prompts for a customer service representative to make immediate direct contact. Finally, the service representative's awareness of the customer's emotional state results in a more empathetic response than a standard one, leading to a satisfying resolution of the issue and improvement in the customer relationship.

The [importance of customer sentiment](https://www.techtarget.com/searchcustomerexperience/tip/How-to-gather-and-evaluate-customer-sentiment) extends to what positive or negative sentiment the customer expresses, not just directly to the organization, but to other customers as well. People commonly share their feelings about a brand's products or services, whether they are positive or negative, on social media. If a customer likes or dislikes a product or service that a brand offers, they may post a comment about it -- and those comments can add up. Such posts amount to a snapshot of customer experience that is, in many ways, more accurate than what a customer survey can obtain.

**CHAPTER 2**

**LITERATURE SURVEY**

Sentiment analysis is the process of analysis of the text from many levels. First level is document level [3],the classification task determine the class of an object based on its attributes (Turney, 2002; Pang and Lee, 2004), and after that it can analysed at the sentence level[5] for classifying the sentence based on the negative, positive and neutral sentiments (Hu and Liu, 2004; Kim and Hovy, 2004) and next level is the phrase level[4] for defining if an expression is unbiassed or polar and then remove uncertainty of meaning from the polarity of the polar expressions (Wilson et al., 2005; Agarwal et al., 2009. Birmingham and Smeaton(2010) and Pak and Paroubek (2010). Go et al. (2009) they used distant learning algorithm to obtain the sentiment data [8]. In this techniques, positive emoticons symbols in tweets such as “:)” “:-)”and negative emoticons symbols in tweets such as like “:(” “:-(”. They proposed the models using Naive Bayes algorithm for analysis the text and the report are generated and visualized. They used two methods such as unigrams for identifying single word repeating over the context and bigrams for identifying double word repeating over the context along with Parts-of-Speech (POS) for analyzing the tweets. But the unigram method had reached a better way of analysis but the bigrams and POS had failed to attempt his purpose. Pak and Paroubek (2010) [2] collect the following tweets considered as data which really helped them in similar distant learning paradigm for setting a model for analysis. They perform classification of task such as subjective, objective. For subjective the information are get from the user tweets by means of text or image or symbols as Go et al. (2009) [8]. For objective information thee information are obtained from verification of the data such as famous newspapers like “Times of India”, “Washington Posts” etc. Information which is taken for analysis is casual sample of flowing tweets collected by using queries. In the past year there have been numerous documents observing the Twitter sentiment and buzz [1], [2], [4] (Jansen et al. 2009; Pak and Paroubek 2010; O’Connor et al. 2010; Tumasjan et al. 2010; Bifet and Frank 2010; Barbosa and Feng 2010; Davidov, Tsur, and Rappoport 2010). Further scientists have started to discover the usage of part-of-speech structures but results remain mixed. It has enormous interesting chances to develop the innovative applications, because success of many business depends on accessible information on online sources such as blogs, twitter and other social networks. Barbosa and Feng (2010)[4] has analyzed the sentimental classification on Twitter data. The test data of tweets are collected, they have taken some of the syntax features for analysis of tweets which contains symbols, retweet, emoticons, tags, link, punctuation and exclamation marks, semicolon are in the combination with structures for identifying the polarity of words. Kamp’s et al. (2002) [12] has analyzed the data by using the lexical database. Lexical database is description of lexemes. Lexical database such as WordNet are used. This contains the emotional content of a word. The distance metric of words are used to determined semantic polarity of adjectives. Researchers are also trying to find different ways of analyzing tweets based on the ideas they had while understanding the concept. Researchers tried this analysis using some of the specified fields such as Machine learning which uses Naive Bayes, Maximum entropy and SVM alongside the Semantic Orientation based Word Net which extracts equivalent words and similitude for the content feature, then Lexicon based analysis based on the created dataset which consists of pre-processed tweets and lastly, Hybrid approach where some researchers combined the supervised machine learning and lexicon based approaches together to improve sentiment classification performance. Gamon (2004) [9] has done sentiment analysis on feedback data from the Global Support Services survey. They are used query to identify the role of features like Part of Speech tags. The accuracy of classifier can be obtained by some of factors such as feature selection, from the testing data and demonstrate the abstract linguistic analysis feature for accuracy of data. Devaki P, et al (2017)[15] has done analysis on twitter data for election. It indicates the popularity of parties in the election based on positive tweets. This system uses Naïve Bayes classifier algorithm are used to classify the positive and negative tweets. A comparative study of existing techniques for mining the data which includes machine learning, Interdependent Latent Dirichlet Allocation, lexicon-based approaches, together with cross domain , cross-lingual methods and some evaluation metrics. The concept level sentences analysis uses the Combining Lexicon and Learning based Approach. As the result of study, machine learning methods such as Support Vector Machine and Naive Bayes have the highest accuracy and can be regarded as the baseline learning methods, while lexicon-based methods are very effective in some cases. More research is needed to determine whether the POS features are just of poor quality due to the results of the tagger or whether POS features are just less useful for sentiment analysis in this domain. Features from an existing sentiment lexicon were somewhat useful in conjunction with microblogging features, but the microblogging features (i.e., the presence of intensifiers and positive/negative/neutral emoticons and abbreviations) were clearly the most useful. In this paper, we perform extensive feature analysis of tweets using hashtags, ID’s and building model classifications.

1. **Real time analysis of top trending event on Twitter - Lexicon based approach**

Social media has become an integral part of everyone's daily routine in today's digital era. It has evolved in an immense way with its growing number of channels, never imagined of a generation ago. Nowadays, it has become rare to see anyone without their mobiles in their hand, browsing and logging in through their social media network such as Facebook, Twitter, WhatsApp and so on. Enormous amounts of real-time data being generated every second across the world mostly in unstructured text messages. It brings in a huge challenge and opportunity to analyze in order to discover answers to the various questions and solve many real time problems.

# Real Time Sentimental Analysis on Twitter

# 

Real-time analysis is one of the vital things where it is used widely in Big Data Analytics (BDA). Here a cluster or group which is a set of tweets that are created by different users on the Twitter website were developed and intimated to the user about the behavior which is mainly focusing on sentimental analysis. Here, Flume is used to collect the real-time data which actually integrate with the Twitter (website) developer account. To ensure the data security, real-time and fast information processing, the most famous or popular tools that are being used like Flume, Hive and ML algorithms in python to obtain the results more accurately.

**3.** **Discovering Public opinions by Performing Sentimental Analysis on Real Time Twitter Data**

# Data streaming is an evolutionary concept in big data where the size of data increases a lot from social media, trending websites, and mobile applications. Nowadays, streaming data tends to collect data from live streaming to run analysis and generate reports for data prediction. This process requires skilled professionals for acquiring data from live stream using complex coding and queries. The above drawback is overcome in this research work by implementing a streaming algorithm to fetch data FRO Twitterer using a keyword search. The Twitter data visualization application is designed for data visualization, report generation and analysis. The live twit twitter is fetched by configuring the system with Hadoop, Hive warehouse and, Apache Flume. By using flume agent, the keyword file is placed on Had the oop cluster to acquire relevant data via the flume channel and then sink the collected data in Hadoop Distributed File System. Twitteritter data application creates a database in Hive and imports the collected data to the Hive table for visualization.

# Production Prediction based on News using Sentimental Analysis

# In the current scenario, market is changing with a drastic rate. In such a phase taking right decision at right time with respect to organization benefits is a challenging task. Industries are indented to fulfill customers need and the news plays a vital role to change the sentiments of the targeted customers. News from the online sources is mined and combined with sentiment analysis to produce the best suitable suggestions.

1. **Handling of Voluminous Tweets and Analyzing the Sentiment of Tweets**

Nowadays, Social Media is used as a medium to express the views of the people about the recent trends, or issues. This results in generating large amount of days everyday. These data's need to be handled in more efficient manner to analyse the ambiguous. These real time streaming of data are to be stored and processed efficiently for further analysis and decision by using big data tools. Twitter is used as data source and sentimental analysis is performed over the tweets. In addition, big data tools such as Flume, Kafka, Hdfs, Hbase and Solr were used for effective handling of tweets.

# Sentiment Analysis on Twitter Using Streaming API

# 

# In general, opinion mining has been used to know about what people think and feel about their products and services in social media platforms. Millions of users share opinions on different aspects of life every day. Spurred by that growth, companies and media organizations are increasingly seeking way to mine information. It requires efficient techniques to collect a large amount of social media data and extract meaningful information from them. This paper aims to provide an interactive automatic system which predicts the sentiment of the review/tweets of the people posted in social media using hadoop, which can process the huge amount of data.

# Classification of COVID19 Tweets based on Sentimental Analysis

# The year 2020, has seen the advent of a pandemic that has affected the world as we know it globally. The origin reportedly from Wuhan, China, this pandemic is caused by COVID-19 which belongs to the family of Coronavirus. The increase of infection and mortality has shot up exponentially and has left mankind bewildered amongst the remains of the unseen disaster. During these times of hardship mankind has to face with a series of emotions. Analysis of all these emotions becomes a primary target for the well-being of an individual and mankind as a whole.

# A Novel Approach to Predict the Real Time Sentimental Analysis by Naive Bayes & RNN Algorithm during the COVID Pandemic in UAE

At present, each and every part of the globe are facing a COVID crisis, which is affecting an individual physically, mentally and on the other hand, it is affecting the nation economically. Also, the unemployment scenario will be at its peak in the upcoming years as reported by UNGA. To combat this scenario, all the country are working on fostering their fiber network and so the sectors apart from the manufacturing, will tend to work from their home and contribute to the economy. But there are many problems arising to implement this culture practically, since it affects the mindsets of the people who have to endure this transformation within a very short span. Hence, in this research work, it has been decided to focus on this current issue for which the usage of certain apps in UAE such as zoom, totok, botim for internet calling have been identified since this is the only way of connectivity with the outside world.

# Modeling Recommendation System for Real Time Analysis of Social Media Dynamics

With the increasing popularity of twitter, Sentiment analysis of data from twitter has become a research trend. With the help of Twitter API, large number of tweets can be retrieved in real time related to our interest for the analysis. Millions of tweets are posted daily which contain opinions of users around the world. The aim of this project is to develop a desktop application which present users with tweets they may have an interest in. This model analyzes the most used keyword and most mentioned username from the user timeline and henceforth recommend recent tweets with the same keyword from that user.

# Performance Prediction of Product/Person UsingReal-Timee Twitter Tweets

# Over a previous decade people have experienced an exponential boom in the usage of online resources in specific social media and microblogging internet site such as Twitter, Facebook, Instagram and YouTube. Many businesses and agencies has identified these sources as a wealthy mine of marketing information. On such platforms, massive quantities of records are produced (e.g.: 5000 tweets per 2d on twitter), this representing an chance for companies to check their social impact and people opinions towards their products, and even frequent people can additionally discover out what is a performance of a certain product or the overall performance of a particular political personality.

**CHAPTER 3 SYSTEM ANALYSIS**

# SYSTEM REQUIREMENTS

The following specifications were those required by the system for the software’s successful implementation and functioning

# REQUIREMENTS SOFTWARE REQUIREMENTS

# Anaconda

# Python3

# Jupyter notebook

# HARDWARE REQUIREMENTS

# Processor :Intel(R) Core(TM) i3-6006U CPU @2.00GHz 2.00 GHz

# Installed RAM : 4.00 GB

# System type : 64-bit operating system, x64-based processor

# PYTHON

The python programming language is a high-level, object-oriented, and interpreted programming language. Python is of best use when it comes to Rapid Application Development and this due to the high-level built-in data structures which are combined with dynamic typing and dynamic binding. This programming language has a simple and easy-to-learn syntax which emphasizes readability and reduces the overall cost of program maintenance. This high-level programming language can also be used as a scripting or glue language to connect the existing components. It supports a lot of modules and packages and encourages program modularity and code reusability. Python has an interpreter and an extensive standard library that are made available in source or binary form without charge for all major platforms, and it is an open-source language.

# PACKAGES

These are the packages that were downloaded and utilized in our project

* Panda == 1.4.2
* tweepy== 4.10.0
* RE == 3.10.5
* numpy == 1.20.1
* nltk == 3.5

**Panda**

pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way towards this goal.

**Tweepy**

[Tweepy](https://github.com/tweepy/tweepy) is a popular package in Python used by students, researchers and developers for interacting with the Twitter API. Recently, the [version 4.0 of this package was released](https://twittercommunity.com/t/tweepy-v4-0-0-has-been-released/159845) that supports the Twitter API v2 and the academic research product track. In this guide, we will learn how to use the various functionality available in the Twitter API v2, using Tweepy.

**Regular Expression**

Regular expressions use the backslash character ('\') to indicate special forms or to allow special characters to be used without invoking their special meaning. This collides with Python’s usage of the same character for the same purpose in string literals.

**NumPy**

NumPy is a Python library that enables users to interact with arrays. It also provides numerous tools in linear algebra, the Fourier transform, and matrices, including a huge handful of international mathematical functions to handle these arrays.

**NLTK**

[Natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing) (NLP) is a field that focuses on making natural human language usable by computer programs. **NLTK**, or [Natural Language Toolkit](https://www.nltk.org/), is a Python package that you can use for NLP.

A lot of the data that you could be analyzing is [unstructured data](https://en.wikipedia.org/wiki/Unstructured_data) and contains human-readable text. Before you can analyze that data programmatically, you first need to preprocess it.

# CHAPTER 4 SYSTEM DESIGN

# OBJECT ORIENTED DESIGN

Identifying the objects in a system would be what OO (Object Oriented) analysis and design have always been about identifying their relationships. Generating a design that could be transformed into executables utilising object- oriented programming languages.

UML (Unified Modelling Language) is a standard language that uses, designs, constructing, and documenting software components. The Unified Modelling Language (UML) is an effective instrument enabling Object- Oriented Analysis and Design. The Object Management Group (OMG) created it, and in January 1997, the OMG proposed UML 1.0 as a specification draught. It started as a way to capture the behaviour of vast software and non-software systems and has since grown into such an international standard. UML is created to be process generic, indicating it could be used in a variety of circumstances. It can be used for a spectrum of uses. Business analysts, software architects, and developers are all using UML as a common language. It can be used to describe, specify, build, and document the system's business processes, including its structural and behavioural artefacts.

UML is a modelling language used to create software blueprints. Diagrams are categorized into two divisions, which are further divided into

* **Structural Diagrams**
* **Behavioral Diagrams**

# STRUCTURAL DIAGRAMS

The static aspect of the system is portrayed by the structural diagrams. These static aspects are the components of a diagram that describes the main structure and are thus stable. Classes, interfaces, objects, components, and nodes are often used to represent them.

Some of the structural diagrams are:

* + - Class diagram
    - Object diagram
    - Component diagram
    - Deployment diagram

# BEHAVIORAL DIAGRAMS

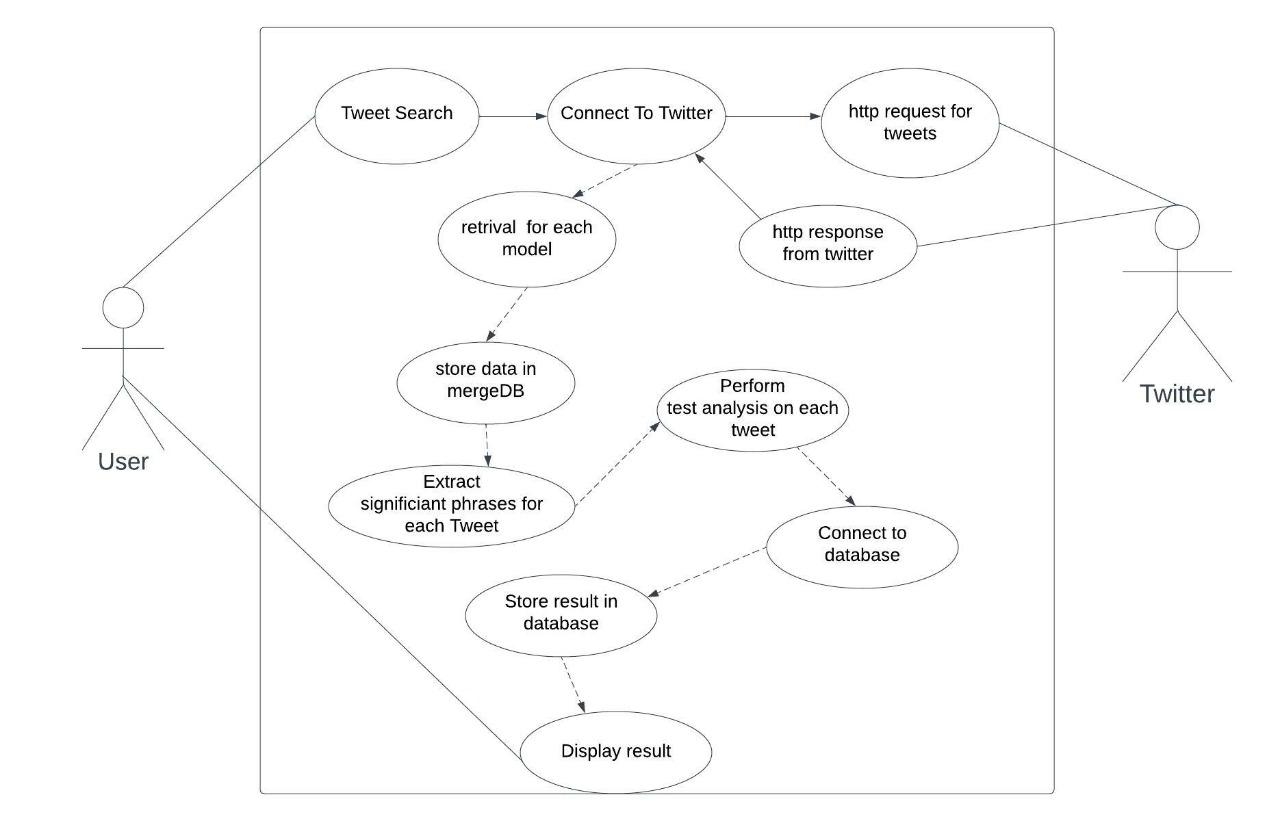
Behavioral diagrams illustrate a system's complex nature. The changing/moving parts of a system are usually known for the dynamic aspect. The following five types of behavioural diagrams are supported in UML:

* + - Use case diagram
    - Sequence diagram
    - Collaboration diagram
    - Statechart diagram
    - Activity diagram

# USE CASE DIAGRAM

The use case diagrams show the system's behavior concerning to the deployment environment. It illustrates the proposed system's users. A use case is a system analysis methodology for finding, explaining, and monitoring programs needs.

A use case is a collection of conceivable sequences of interactions between systems and users in a specific environment, all of which are tied to a specific purpose. A use case is represented by an ellipse with the name of the use case. A stick figure with a name is used to represent an actor.

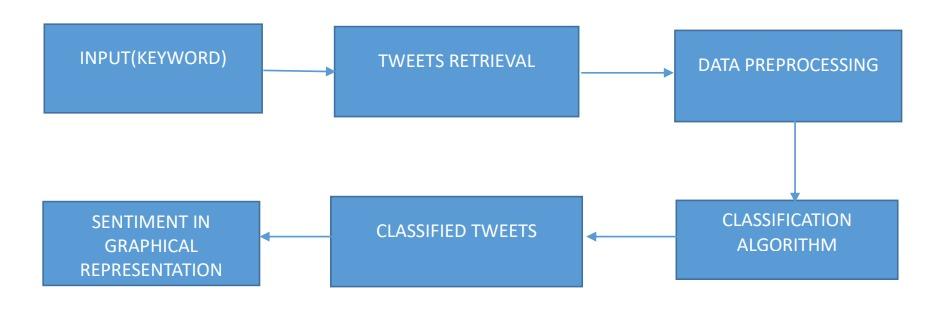
/

*Fig 1.1 Use Case diagram*

# ARCHITECTURE DIAGRAM

Two webcams are used in our system. One is placed on the dashboard which is behind the steering wheel and the other one is placed above the number plate. The webcam placed on the dashboard monitors the driver’s eyes. From the extracted frames the landmarks are found and the eye region is extracted. The Eye Aspect Ratio is calculated and the threshold is compared. Simultaneously the lane deviation is detected by the other webcam. The decision is taken and when drowsiness is detected and the car deviates from the lane, an alarm is sent to the

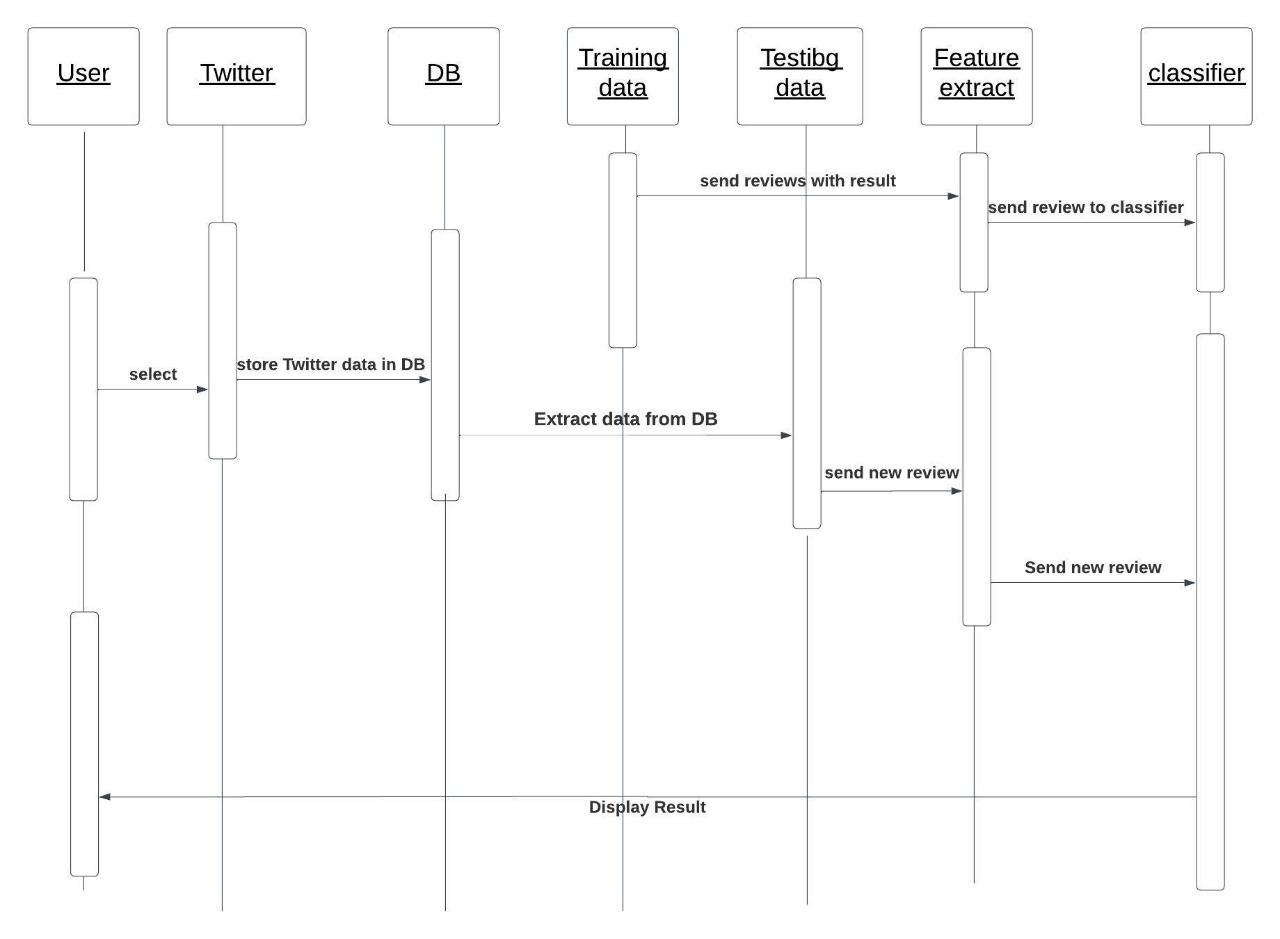
driver.



*Fig 1.2 Architecture diagram*

**SEQUENCE DIAGRAM**

The sequence diagram is used primarily to show the interactions between objects in the sequential order that those interactions occur. Much like the class diagram, developers typically think sequence diagrams were meant exclusively for them. However, an organization's business staff can find sequence diagrams useful to communicate how the business currently works by showing how various business objects interact. Besides documenting an organization's current affairs, a business-level sequence diagram can be used as a requirements document to communicate requirements for a future system implementation.

****

# CHAPTER 5 SYSTEM IMPLEMENTATION

* 1. **ALGORITHMS AND PROPOSED TECHNIQUES**

**ANACONDA AND JUPYTER NOTEBOOK**

Anaconda is an open-source software that contains Jupyter, spyder, etc that are used for large data processing, data analytics, heavy scientific computing. Anaconda works for R and python programming language. Spyder(sub-application of Anaconda) is used for python. Opencv for python will work in spyder. Package versions are managed by the package management system called conda.

To install Jupyter using Anaconda, **Launch Anaconda Navigator**  and

**Click on the Install Jupyter Notebook Button** then **Beginning the Installation** next **Loading Packages** thenFinished Installation finally **Launching Jupyter.**

Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Jupyter has support for over 40 different programming languages and [Python](https://www.geeksforgeeks.org/python-language-introduction/) is one of them. Python is a requirement (Python 3.3 or greater, or Python 2.7) for installing the Jupyter Notebook itself.

# TWITTER API

# Python access to API

Step 1: Create a [twitter account](https://twitter.com/) if you do not have one.

Step 2: On the [twitter developer account page](https://developer.twitter.com/en/apply/user), you will be asked to answer a few questions. For example, I was asked for a phone number, country, and use case. The next step is to read and agree to the developer agreement.

Step 3: Verify your email.

Step 4: After verifying your email, you will be sent to a welcome screen. Name your app and click on Get keys.

Step 5: You now have access to your keys. Make sure to save your information to a secure location. You will need them to access data using the twitter API.

# DATA CLEANING AND VISUALIZATION

**Clean text** is human language rearranged into a format that machine models can understand. Text cleaning can be performed using simple Python code that eliminates stop words, removes Unicode words, and simplifies complex words to their root form.

While text cleaning, like [data pre-processing](https://monkeylearn.com/blog/data-preprocessing/) as a whole, has greatly benefited from a number of new [self-service tools](https://monkeylearn.com/blog/data-cleaning-tools) that can standardize and clean your data for you, it is still important to understand the underlying code.

Enter the [Natural Language Toolkit (NLTK)](https://www.nltk.org/), a python toolkit [specifically designed for raw text to NLP transformation.](https://towardsdatascience.com/text-cleaning-methods-for-natural-language-processing-f2fc1796e8c7)

With an understanding of a few basic NLTK processes you can easily grasp the foundation of most text cleaning programs, and from there modify and customize them to best serve your purposes.

1. Normalize Text
2. Remove Unicode Characters
3. Remove Stop words
4. Perform Stemming and Lemmatization

**NORMALIZING TEXT**

**Normalizing text** is the process of standardizing text so that, through NLP, computer models can better understand human input, with the end goal being to more effectively perform [sentiment analysis](https://monkeylearn.com/sentiment-analysis/) and other types of analysis on your [customer feedback](https://monkeylearn.com/customer-feedback/).

Specifically, normalizing text with Python and the NLTK library means standardizing capitalization so that machine models don’t group capitalized words (Hey) as different from their lowercase counterparts (hey).

This is called **case normalization.**

**REMOVING UNICODE CHARACTERS**

Punctuation, Emoji’s, URL’s and @’s confuse AI models because they are unique signatures that either end up being translated unhelpfully into Unicode (Smiley face becomes \u200c or similar), or are unique (in the case of @’s and hyperlinks).

Punctuation also creates noise and impedes NLP understanding because it relates to the tone of the specific sentence, not necessarily the word it is attached to.

**INTRODUCTION TO NLP**

The essence of Natural Language Processing lies in making computers understand the natural language. That’s not an easy task though. Computers can understand the structured form of data like spreadsheets and the tables in the database, but human languages, texts, and voices form an unstructured category of data, and it gets difficult for the computer to understand it, and there arises the need for Natural Language Processing.

**Step 1:** Sentence Segmentation

Breaking the piece of text in various sentences.

**Step 2:** Word Tokenization

Breaking the sentence into individual words called as tokens. We can tokenize them whenever we encounter a space, we can train a model in that way. Even punctuations are considered as individual tokens as they have some meaning.

**Step 3:** Predicting Parts of Speech for each token

Predicting whether the word is a noun, verb, adjective, adverb, pronoun, etc. This will help to understand what the sentence is talking about. This can be achieved by feeding the tokens( and the words around it) to a pre-trained part-of-speech classification model. This model was fed a lot of English words with various parts of speech tagged to them so that it classifies the similar words it encounters in future in various parts of speech. Again, the models don’t really understand the ‘sense’ of the words, it just classifies them on the basis of its previous experience. It’s pure statistics.

**Step 4:** Lemmatization

Feeding the model with the root word.

**Step 5:**  Identifying stop words

There are various words in the English language that are used very frequently like ‘a’, ‘and’, ‘the’ etc. These words make a lot of noise while doing statistical analysis. We can take these words out. Some NLP pipelines will categorize these words as stop words, they will be filtered out while doing some statistical analysis. Definitely, they are needed to understand the dependency between various tokens to get the exact sense of the sentence. The list of stop words varies and depends on what kind of output are you expecting.

**Step 6:**  Dependency Parsing

This means finding out the relationship between the words in the sentence and how they are related to each other. We create a parse tree in dependency parsing, with root as the main verb in the sentence. If we talk about the first sentence in our example, then ‘is’ is the main verb and it will be the root of the parse tree. We can construct a parse tree of every sentence with one root word(main verb) associated with it. We can also identify the kind of relationship that exists between the two words.

**Step 7:** Named Entity Recognition(NER)

San Pedro is a town on the southern part of the island of Ambergris Caye in the 2. Belize District of the nation of Belize, in Central America.  
Here, the NER maps the words with the real world places. The places that actually exist in the physical world. We can automatically extract the real world places present in the document using NLP.

**Step 8: :** Coreference Resolution:

San Pedro is a town on the southern part of the island of Ambergris Caye in the Belize District of the nation of Belize, in Central America. According to 2015 mid-year estimates, the town has a population of about 16, 444. It is the second-largest town in the Belize District and largest in the Belize Rural South constituency.

**LEXICAL BASED APPROACH**

Social media has become an integral part of everyone's daily routine in today's digital era. It has evolved in an immense way with its growing number of channels, never imagined of a generation ago. Nowadays, it has become rare to see anyone without their mobiles in their hand, browsing and logging in through their social media network such as Facebook, Twitter, WhatsApp and so on. Enormous amounts of real-time data being generated every second across the world mostly in unstructured text messages. It brings in a huge challenge and opportunity to analyze in order to discover answers to the various questions and solve many real time problems. This paper mainly focusses on two aspects. One is to identify the real-time top trending event on twitter for a particular location and next is to consider the first trending event for doing sentiment analysis. Sentiment analysis is performed using lexicon based approach classified and visualized in ten different emotion parameters.

**CHAPTER 6**

**PERFORMANCE METRICS**

This system presents performance criteria evaluation based on a comparison between sentiment techniques. The target is measuring the sentiments performance through several significant perspectives in sentiment analysis. This measurement is very tight of accuracy evaluating for sentiments. However, evaluating sentiments is a hard challenge for language technologies, and achieving good results is much more difficult than some human think. Also, we introduce a comprehensive study for different sentiment techniques based on proposed performance criteria. The performance evaluation plays a vital role in accuracy measurement through a sentiment analysis word level. The performance criteria include two types of performance measurement namely F-measure and Runtime. These criteria include the balance of performance perspectives priorities. These types include a relationship between perspectives of performance to improve it. There are different performance perspectives: F-measure and speed of run time, memorability, and sentiment analysis challenges. It helps in understanding the contextual meaning and getting a score in less time and higher accuracy. The comparisons are based on the sentiment analysis word-level. They can understand some phrases as do not directly through caring with the classification of reviews. Finally, we show the efficiency and effectiveness of the proposed criteria.

# CHAPTER 7 CONCLUSION AND FUTURE WORKS

The task of sentiment analysis, especially in the domain of micro-bloging, is still in the developing stage and far from complete. So we propose a couple of ideas which we feel are worth exploring in the future and may result in further improved performance. Right now we have worked with only the very simplest unigram models; we can improve those models by adding extra information like closeness of the word with a negation word. We could specify a window prior to the word under

consideration and the effect of negation may be incorporated into the model if it lies within that window. The closer the negation word is to the unigram word whose prior polarity is to be calculated, the more it should affect the polarity. For example if the negation is right next to the word, it may simply reverse the polarity of that word and farther the negation is from the word the more minimized ifs effect should be.Apart from this, we are currently only focusing on unigrams and the effect of bigrams and trigrams may be explored. As reported in the literature review section when bigrams are used along with unigrams this usually enhances performance. However for bigrams and trigrams to be an effective feature we need a much more labeled data set than our meager 9,000 tweets.Right now we are exploring Parts of Speech separate from the unigram models, we can try to incorporate POS information within our unigram models in future. So say instead of calculating a single probability for each word like P(word | obj) we could instead have multiple probabilities for each according to the Part of Speech the word belongs to. For example we may have P(word | obj, verb), P(word | obj, noun) and P(word | obj, adjective). Pang et al. used a somewhat similar approach and claims that appending POS information for every unigram results in no significant change in performance (with Naive Bayes performing slightly better and SVM having a slight decrease in performance), while there is a significant decrease in accuracy if only adjective unigrams are used as features. However these results are for classification of reviews and may be verified for sentiment analysis on micro blogging websites like Twitter. One more feature we that is worth exploring is whether the information about relative position of word in a tweet has any effect on the performance of the classifier. Although Pang et al. explored a similar feature and reported negative results, their results were based on reviews which are very different from tweets and they worked on an extremely simple model. In this research we are focussing on general sentiment analysis. There is potential of work in the field of sentiment analysis with partially known context. For example we noticed that users generally use our website for specific types of keywords which can divided into a couple of distinct classes, namely: politics/politicians, celebrities, products/brands, sports/sportsmen, media/movies/music. So we can attempt to perform separate sentiment analysis on tweets that only belong to one of these classes (i.e. the training data would not be general but specific to one of these categories) and compare the results we get if we apply general sentiment analysis on it instead.

# APPENDIX APPENDIX A - SOURCE CODE

**VISUALISATION.PY**

# import tweepy

# import pandas as pd

# import re

# import matplotlib.pyplot as plt

# from wordcloud import WordCloud

# consumer\_key = '78F6GWmlPoJX4CtW8E5A4dQYf'

# consumer\_secret = 'Uj3ZCkYe47HOLG0OOyXUly3wyvwhFAG8GLuQqZHVqse6VipfwJ'

# access\_token = '1310463220281495552-D8AOegt4AcMXgiC738DgD6STjQbaVi'

# access\_token\_secret = 'PrcCzW7N4LmEcLIKnxv7VcX8ytwwCsAIdal2EWXROhKAh'

# #Authenticate with credentials

# auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

# auth.set\_access\_token(access\_token, access\_token\_secret)

# api = tweepy.API(auth)

# key\_word = 'bitcoin'

# limit = 500

# def TextClean(tweet):

# tweet = tweet.lower()

# tweet = re.sub(r'@[a-z0-9\_]\S+', '', tweet)

# tweet = re.sub(r'#[a-z0-9\_]\S+', '', tweet)

# tweet = re.sub(r'&[a-z0-9\_]\S+', '', tweet)

# tweet = re.sub(r'[?!.+,;$%&"]+', '', tweet)

# tweet = re.sub(r'rt[\s]+', '', tweet)

# tweet = re.sub(r'\d+', '', tweet)

# tweet = re.sub(r'\$', '', tweet)

# tweet = re.sub(r'rt+', '', tweet)

# tweet = re.sub(r'https?:?\/\/\S+', '', tweet)

# return tweet

# def tweet\_search(key\_word):

# i = 0

# tweets\_df = pd.DataFrame(columns = ['Datetime', 'Tweet', 'Username', 'Retweets', 'Followers'])

# for tweet in tweepy.Cursor(api.search, q = key\_word, count = 100, lang = 'en', tweet\_mode = 'extended').items():

# print('Tweets downloaded:', i, '/', limit, end = '\r')

# if tweet.user.followers\_count > 500:

# tweets\_df = tweets\_df.append({'Datetime': tweet.created\_at,

# 'Tweet': tweet.full\_text,

# 'Username': tweet.user.screen\_name,

# 'Retweets': tweet.retweet\_count,

# 'Followers': tweet.user.followers\_count,}, ignore\_index = True)

# i += 1

# if i >= limit:

# break

# else:

# pass

# tweets\_df['Datetime'] = pd.to\_datetime(tweets\_df['Datetime'], format = '%Y.%m.%d %H:%M:%S')

# tweets\_df.set\_index('Datetime', inplace = True)

# tweets\_df['CleanTweet'] = tweets\_df['Tweet'].apply(TextClean)

# #tweets\_df.to\_csv(key\_word + '.csv', encoding = 'utf-8')

# return tweets\_df

# tweets\_df = tweet\_search(key\_word)

# tweets\_df

# #Store all tweets as one big string

# all\_tweets = ' '.join(tweet for tweet in tweets\_df['CleanTweet'])

# all\_tweets

# WordCloud = WordCloud(width = 800, height = 400, random\_state = 21, max\_font\_size = 100, collocations = False).generate(all\_tweets)

# plt.figure(figsize = (20, 10))

# plt.imshow(WordCloud, interpolation = 'bilinear')

# plt.axis('off')

# plt.style.use('ggplot')

word\_frequency = pd.DataFrame.from\_dict(data = WordCloud.words\_, orient = 'index')

# word\_frequency = word\_frequency.head(20)

# plt.figure(figsize = (20, 10))

# plt.bar(word\_frequency.index, word\_frequency[0])

# plt.xticks(rotation = 90)

# VECTORIZATION.PY

tokens = word\_tokenize(all\_tweets)

tokens

#Lemmatization

lemmatizer = WordNetLemmatizer()

lemma = [lemmatizer.lemmatize(tweet, pos = 'v') for tweet in tokens]

print(tokens[:20])

print(lemma[:20])

#Stemming

porter\_stemmer = PorterStemmer()

stemm = [porter\_stemmer.stem(tweet) for tweet in tokens]

print(tokens[:20])

print(stemm[:20])

#Print lenghts of lists to check if we didn't lose any tokens in the process

print('Number of word in tweets:', len(all\_tweets))

print('Number of tokens:', len(tokens))

print('Number of lemmas:', len(lemma))

print('Number of stemms:', len(stemm))

#Store results in dataframe for easier comparison

df = pd.DataFrame(columns = ['Tokens', 'Stemm', 'Lemma'])

df['Tokens'] = tokens[:50]

df['Stemm'] = stemm[:50]

df['Lemma'] = lemma[:50]

df

**VADER AND TEXTBLOB.PY**

#----------------------------VADER----------------------------

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

vader = SentimentIntensityAnalyzer()

print(vader.polarity\_scores('This is good'))

print(vader.polarity\_scores('This is really good'))

print(vader.polarity\_scores('This is great'))

print(vader.polarity\_scores('This is terrible'))

print(vader.polarity\_scores('How are you? :)'))

print(vader.polarity\_scores('How are you? :('))

print(vader.polarity\_scores('This is fine'))

print(vader.polarity\_scores('This is fine!'))

print(vader.polarity\_scores('This is FINE'))

print(vader.polarity\_scores('I used to like him, but now I see he is jerk'))

#----------------------------TextBlob----------------------------

from textblob import TextBlob

print(TextBlob('This is good').sentiment)

print(TextBlob('This is really good').sentiment)

print(TextBlob('This is great').sentiment)

print(TextBlob('This is terrible').sentiment)

print(TextBlob('How are you?').sentiment)

print(TextBlob('How are you? :)').sentiment)

print(TextBlob('How are you? :(').sentiment)

print(TextBlob('This is fine').sentiment)

print(TextBlob('This is fine!').sentiment)

print(TextBlob('This is FINE').sentiment)

print(TextBlob('I used to love him, but now I see he is jerk').sentiment)

**LEXCION BASED SENTIMENT**

import tweepy

import pandas as pd

import numpy as np

import re

import matplotlib.pyplot as plt

from nltk.tokenize import TweetTokenizer

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

from textblob import TextBlob

consumer\_key = '78F6GWmlPoJX4CtW8E5A4dQYf'

consumer\_secret = 'Uj3ZCkYe47HOLG0OOyXUly3wyvwhFAG8GLuQqZHVqse6VipfwJ'

access\_token = '1310463220281495552-D8AOegt4AcMXgiC738DgD6STjQbaVi'

access\_token\_secret = 'PrcCzW7N4LmEcLIKnxv7VcX8ytwwCsAIdal2EWXROhKAh'

#Authenticate with credentials

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

auth.set\_access\_token(access\_token, access\_token\_secret)

api = tweepy.API(auth)

key\_word = 'tesla'

limit = 500

def TextClean(tweet):

tweet = tweet.lower()

tweet = re.sub(r'@[a-z0-9\_]\S+', '', tweet)

tweet = re.sub(r'#[a-z0-9\_]\S+', '', tweet)

tweet = re.sub(r'&[a-z0-9\_]\S+', '', tweet)

tweet = re.sub(r'[?!.+,;$%&"]+', '', tweet)

tweet = re.sub(r'rt[\s]+', '', tweet)

tweet = re.sub(r'\d+', '', tweet)

tweet = re.sub(r'\$', '', tweet)

tweet = re.sub(r'rt+', '', tweet)

tweet = re.sub(r'https?:?\/\/\S+', '', tweet)

return tweet

def tweet\_search(key\_word):

i = 0

tweets\_df = pd.DataFrame(columns=['Datetime', 'Tweet', 'Username', 'Retweets', 'Followers'])

for tweet in tweepy.Cursor(api.search, q = key\_word, count = 100, lang = 'en', tweet\_mode = 'extended').items():

print('Tweets downloaded:', i, '/', limit, end = '\r')

if tweet.user.followers\_count > 500:

tweets\_df = tweets\_df.append({'Datetime': tweet.created\_at,

'Tweet': tweet.full\_text,

'Username': tweet.user.screen\_name,

'Retweets': tweet.retweet\_count,

'Followers': tweet.user.followers\_count}, ignore\_index = True)

i += 1

if i >= limit:

break

else:

pass

tweets\_df['Datetime'] = pd.to\_datetime(tweets\_df['Datetime'], format = '%Y.%m.%d %H:%M:%S')

tweets\_df.set\_index('Datetime', inplace = True)

tweets\_df.drop\_duplicates(subset = ['Tweet'], inplace = True)

tweets\_df['CleanTweet'] = tweets\_df['Tweet'].apply(TextClean)

tweet\_tokenizer = TweetTokenizer()

tweets\_df['CleanTweet'] = tweets\_df['CleanTweet'].apply(tweet\_tokenizer.tokenize)

tweets\_df['CleanTweet'] = [', '.join(map(str, token)) for token in tweets\_df['CleanTweet']]

#tweets\_df.to\_csv(key\_word + '.csv', encoding='utf-8')

return tweets\_df

tweets\_df = tweet\_search(key\_word)

tweets\_df

def vader\_compound\_score(tweet):

vader = SentimentIntensityAnalyzer()

if vader.polarity\_scores(tweet)['compound'] >= 0.05:

return 'Positive'

elif vader.polarity\_scores(tweet)['compound'] <= -0.05:

return 'Negative'

else:

return 'Neutral'

def textblob\_sentiment(tweet):

analysis = TextBlob(tweet)

if analysis.sentiment.polarity > 0:

return 'Positive'

elif analysis.sentiment.polarity == 0:

return 'Neutral'

else:

return 'Negative'

tweets\_df['Vader\_sent'] = tweets\_df['CleanTweet'].apply(vader\_compound\_score)

tweets\_df['TextBlob\_sent'] = tweets\_df['CleanTweet'].apply(textblob\_sentiment)

tweets\_df['Different\_sent'] = np.where(tweets\_df['Vader\_sent'] != tweets\_df['TextBlob\_sent'], 1, 0)

tweets\_df

#Visualisation

vader\_pie = [len(tweets\_df[tweets\_df['Vader\_sent'] == 'Positive']),

len(tweets\_df[tweets\_df['Vader\_sent'] == 'Negative']),

len(tweets\_df[tweets\_df['Vader\_sent'] == 'Neutral'])]

blob\_pie = [len(tweets\_df[tweets\_df['TextBlob\_sent'] == 'Positive']),

len(tweets\_df[tweets\_df['TextBlob\_sent'] == 'Negative']),

len(tweets\_df[tweets\_df['TextBlob\_sent'] == 'Neutral'])]

labels = ['Positive', 'Negative', 'Neutral']

colors = ['aquamarine', 'tomato', 'skyblue']

print(len(tweets\_df[tweets\_df['Different\_sent'] == 1]), 'times two approaches show different results')

plt.style.use('ggplot')

plt.figure(figsize = (20, 10))

plt.subplot(1, 2, 1)

plt.pie(vader\_pie, labels = labels, colors = colors, autopct = '%1.1f%%')

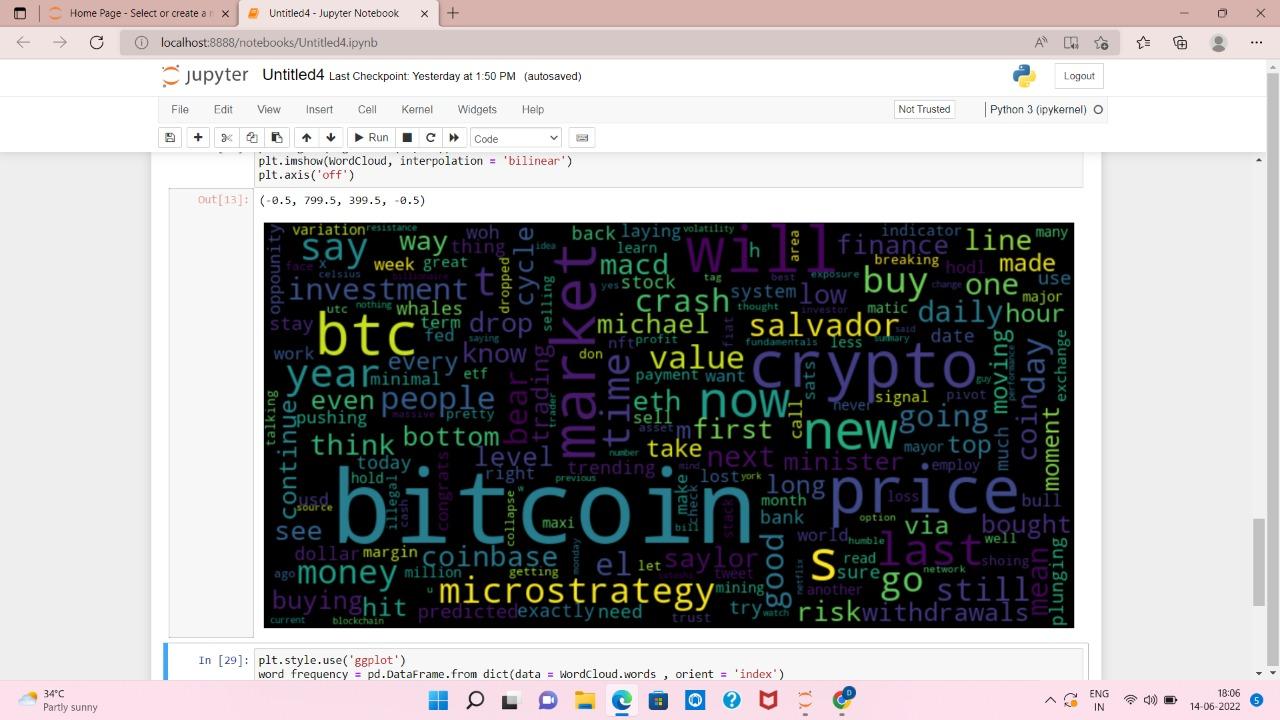
plt.title('Vader')

plt.subplot(1, 2, 2)

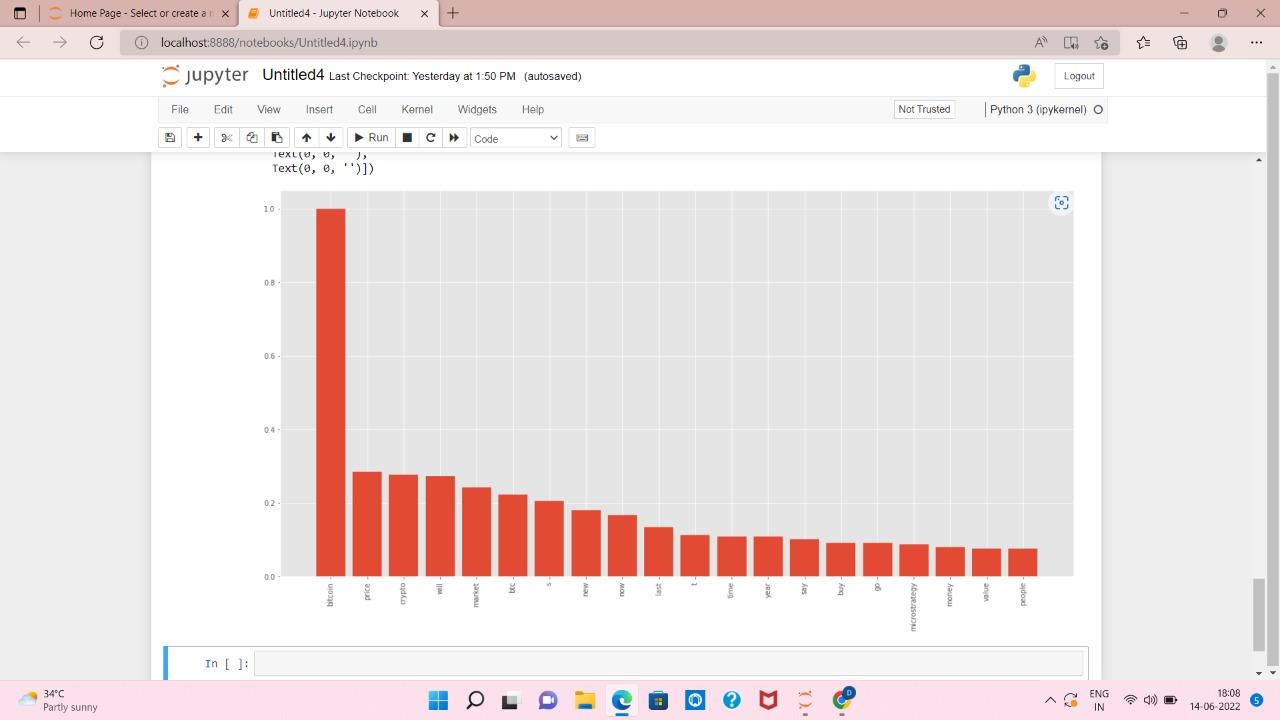
plt.pie(blob\_pie, labels = labels, colors = colors, autopct = '%1.1f%%')

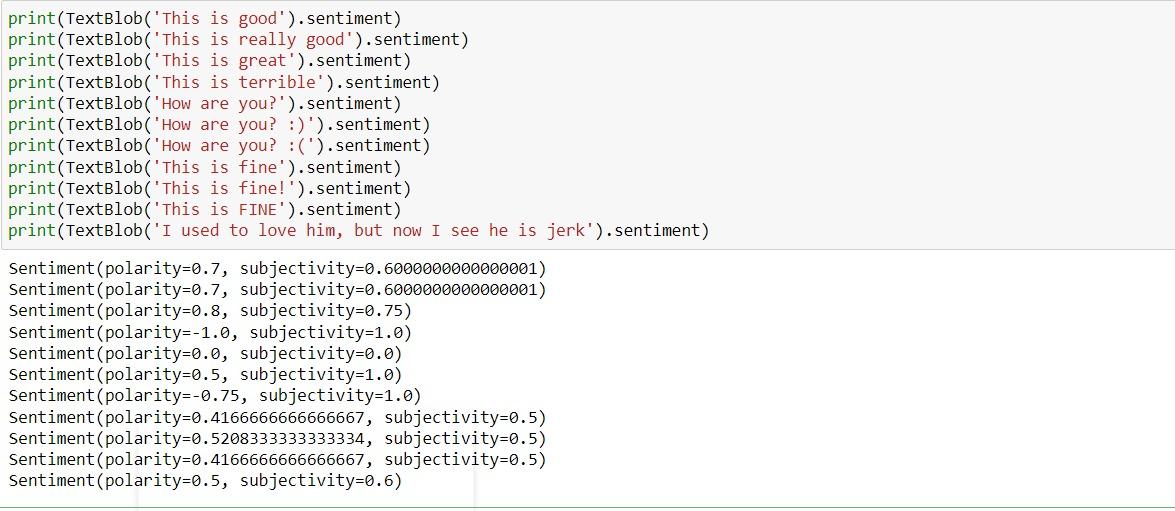
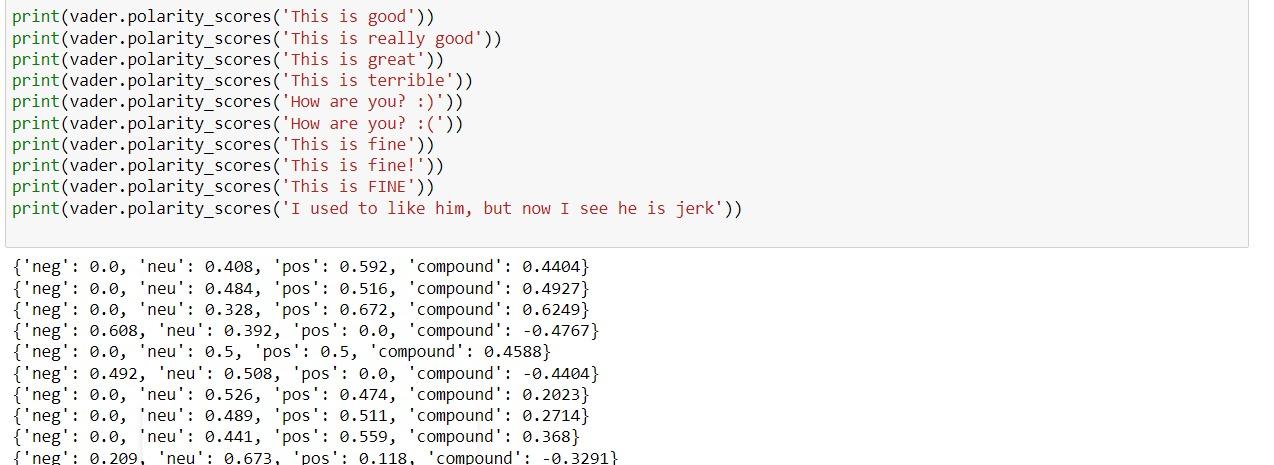
plt.title('TextBlob')

**APPENDIX B**



*Related keywords about the product*







*Tweet analyzed in pie chart*

# REFERENCES

[1].[A. Hari Priya](https://ieeexplore.ieee.org/author/37086253157) and [Santoshi Kumari](https://ieeexplore.ieee.org/author/37086258036) “Real time analysis of top trending event on twitter: Lexicon based approach” [2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)](https://ieeexplore.ieee.org/xpl/conhome/8123903/proceeding).

[2]. K Sai Madhu ,B Chakradhar Reddy ,CH Damarukanadhan ,M Polireddy and

N Ravindher “Real Time Sentimental  Analysis on Twitter” [2021 6th International Conference on Inventive Computation Technologies (ICICT)](https://ieeexplore.ieee.org/xpl/conhome/9358464/proceeding).

[3]. G.Kavitha ,B.Saveen and Nomaan Imtiaz “Discovering Public opinions by Performing  Sentimental Analysis on Real Time Twitter Data” [2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)](https://ieeexplore.ieee.org/xpl/conhome/8811519/proceeding).

[4]. Nishant Singh , Aprna Tripathi and Vivek Kumar “Production Prediction

based on News using Sentimental Analysis” [2019 4th International Conference on Information Systems and Computer Networks (ISCON)](https://ieeexplore.ieee.org/xpl/conhome/9018167/proceeding).

[5]. S.Siva Parvathy and M.Bhuvaneswri” Handling of Voluminous Tweets and Analyzing the Sentiment of Tweets” [2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)](https://ieeexplore.ieee.org/xpl/conhome/8722826/proceeding)

[6]. M.Trupthi ,Suresh Pabboju and G.Narasimha” Sentiment Analysis on Twitter Using Streaming API” [2017 IEEE 7th International Advance Computing Conference (IACC)](https://ieeexplore.ieee.org/xpl/conhome/7974917/proceeding)

[7]. Sagar Banik , Aniket Ghosh , Sumit Banik , Anumpam and Mukherjee “Classification of COVID19 Tweets based on Sentimental Analysis” [2021 International Conference on Computer Communication and Informatics (ICCCI)](https://ieeexplore.ieee.org/xpl/conhome/9402219/proceeding)

[8]. Abdulrahman Radaideh , Fikri Dweiri and Mohammad Obaidat” A Novel Approach to  Predict the Real Time Sentimental Analysis by Naive Bayes  and RNN Algorithm during the COVID Pandemic in UAE “[2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)](https://ieeexplore.ieee.org/xpl/conhome/9256400/proceeding).

[9]. Shipra Goel ,Muskan Banthia and Adwitiya Sinha “Modeling Recommendation System for Real Time Analysis of Social  Media Dynamics” [2018 Eleventh International Conference on Contemporary Computing (IC3)](https://ieeexplore.ieee.org/xpl/conhome/8509799/proceeding).

[10]. Devesh Bhangale , Sneha Poojary,Sameer Ahire and Priyanka Shingane”Performance Prediction of Product/Person Using Real Time Twitter Tweets” [2021 International Conference on Computational Intelligence and Computing Applications (ICCICA)](https://ieeexplore.ieee.org/xpl/conhome/9697106/proceeding)