



# **OPINION MINING ON TWITTER DATA USING MACHINE LEARNING**

*Major project submitted in partial fulfilment of the requirements for  
the award of the degree*

## **BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY BY**

<b>J. Subash</b>	<b>(14241A1225)</b>
<b>K. Venkata Raju</b>	<b>(14241A1228)</b>
<b>K. Sri Vardhan</b>	<b>(14241A1231)</b>
<b>N. Venkat Ramana Reddy</b>	<b>(14241A1243)</b>

*Under the esteemed guidance of*

**Mrs. L. Sukanya**  
**Assistant Professor**



**Department of Information Technology**  
**Gokaraju Rangaraju Institute of Engineering and Technology**  
**(AUTONOMOUS)**  
**HYDERABAD 500090**  
**2018**



**Department of Information Technology**  
**Gokaraju Rangaraju Institute of Engineering and Technology**  
**(AUTONOMOUS)**

**CERTIFICATE**

This is to certify that the Major Project titled **“OPINION MINING ON TWITTER DATA USING MACHINE LEARNING”** is a bona fide work done by **J. Subash (14241A1225), K. Venkata Raju (14241A1228), K. Sri Vardhan (14241A1231)** and **N. Venkat Ramana Reddy (14241A1243)**, in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Information Technology** and submitted to the Department of Information Technology, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad. The work embodied in this report has not been submitted earlier at any other University or Institute for award of any degree.

**L. Sukanya**  
**(Internal Guide)**

**Dr. Y. Vijayalata**  
**(Head of Department)**

**(External Examiner)**

## ACKNOWLEDGEMENT





We take the immense pleasure in expressing gratitude to our Internal Guide **L. Sukanya**, Assistant Professor, Department of Information Technology. We express our sincere thanks for her encouragement, suggestions and support which provided the impetus and paved the way for the successful completion of the project work.

We are grateful to **Dr. Y. Vijayalata**, Head of Department of Information Technology and Members of Project Coordinators **G. Vijender Reddy**, Associate Professor and **Dr. S.V. Appaji**, Associate Professor for their valuable suggestions.

We express our sincere thanks to our Director, **Dr. Jandhyala N. Murthy** and Principal **Dr. J. Praveen**, of Gokaraju Rangaraju Institute of Engineering and Technology for providing us the conducive environment for carrying through our academic schedules and project with ease.

We also take this opportunity to convey our sincere thanks to the teaching and non- teaching staff of Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad for their kind cooperation throughout.

Thanking You.

		<b>Address:</b> House no. 2-65, Cherlapally, Jadcherla Mandal, Mahabub Nagar - 509301
		<b>E-mail:</b> subashjogu9999@gmail.com
		<b>Phone Number:</b> +91 9542402075
	<b>J. Subash</b> (14241A1225)	
		<b>Address:</b> Flat no. 306 Rambadri Apartment, Satyam Heights, Rajiv Gandhi Nagar, Bachupally, Hyderabad - 500090
		<b>Email-id:</b> kanumurivenkata@gmail.com
		<b>Phone Number:</b> +91 9581846677
	<b>K. Venkata Raju</b> (14241A1228)	
		<b>Address:</b> House no. 5-85, Pardi (b), Kubeer, Mandal, Nirmal - 504103
		<b>Email:</b> srivardhankandike@gmail.com
		<b>Phone Number:</b> +91 739647946
	<b>K. Sri Vardhan</b> (14241A1231)	
		<b>Address:</b> Plot no. 126/3, Saraswathi Nagar Colony, Lothukunta, Hyderabad - 500015
		<b>Email-id:</b> venkat.narahaari96@gmail.com
		<b>Phone Number:</b> +91 9985535643
	<b>N. Venkat Ramana Reddy</b> (14241A1243)	

## DECLARATION

This is to certify that the Mini Project titled **“OPINION MINING ON TWITTER DATA USING MACHINE LEARNING”** is a bona fide work done by us in partial fulfilment of the requirements for the award of the degree Bachelor’s in Technology in Information Technology and submitted to the Department of Information Technology, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad.

We also declare that this project is a result of our effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references.

The work embodied in this report has not been submitted earlier at any other university or institute for the award of any degree.

J. Subash	(14241A1225)
K. Venkata Raju	(14241A1228)
K. Sri Vardhan	(14241A1231)
N. Venkat Ramana Reddy	(14241A1243)

## **ABSTRACT**

Social Network Services is a domain of its own that has created a digital platform where people can post their diverse opinions on people, products, politics, services, enterprises, entertainment and others, thus influencing the trends in these wide fields.

The large amount of data available on these social services can give deep insights which can help device various strategies for organizations, politics and government. Opinion Mining is one such way where text can be analysed to get knowledge on polarity or sentiment of a product or a service on scale by linguistic computations with natural language processing and text mining.

This is further automated by the use of machine learning which uses pre-processing for maintaining consistency and supervised learning algorithm which uses support vector machines. As the machine gets exercised with more data the results get less imperfect and more accurate.

# **INDEX**

## **CONTENT**

<b>Topic</b>	<b>Page Number</b>
<b>CERTIFICATE</b>	<b>I</b>
<b>ACKNOWLEDGEMENT</b>	<b>II-III</b>
<b>DECLARATION</b>	<b>IV</b>
<b>ABSTRACTS</b>	<b>V</b>
<b>INDEX</b>	<b>VI-XI</b>
<b>1. Introduction</b>	<b>1-2</b>
1.2 Motivation	2
1.3 Objective	2
<b>2. Literature Survey</b>	<b>3-4</b>
2.1 Feasibility Study	4
2.1.1 Operational Feasibility	4
2.1.2 Technical Feasibility	4
2.1.3 Economic Feasibility	4
<b>3. Analysis</b>	<b>5-10</b>
3.1 Existing System	6
3.2 Proposed System	6
3.3 Requirement Specification	6-10
3.3.1 Software Requirements	6-9
3.3.1.1 Operating System	6
3.3.1.2 Libraries/Toolkits	7
3.3.1.3 Languages	8



3.3.1.4 Integrated Development Environment	8
3.3.1.5 Application Programming Interface	8
3.3.1.6 Package Manager	9
3.3.2 Hardware Requirements	9
3.3.2.1 Processor	9
3.3.2.2 Main Memory	9
3.3.2.3 Hard Disk	9
3.3.3 Pre-Opinionated Data Sources	9-10
3.3.3.1 Stanford Sentiment 140	9
3.3.3.2 Cornell Polarity Dataset	10
3.3.3.3 Michigan Kaggle Social Media Dataset	10
<b>4. Design</b>	<b>11-19</b>
4.1 Architecture	11
4.2 UML Diagrams	12-19
4.2.1 Use Case Diagram	13-14
4.2.2 Class Diagram	14-15
4.2.3 Sequence Diagram	15-16
4.2.4 Collaboration Diagram	16
4.2.5 Activity Diagram	17
4.2.6 Component Diagram	18
4.2.7 Deployment Diagram	19
<b>5. Technology</b>	<b>20-41</b>
5.1 Opinion Mining	20-26
5.1.1 Types	20-22
5.1.1.1 Subjectivity/Objectivity Identification	22
5.1.1.2 Feature /Aspect based	22
5.1.2 Methods and Features	22-24
5.1.3 Evaluation	24
5.1.4 Web 2.0	25
5.1.5 Application in Recommender System	25-27
5.2 Machine Learning	27-41
5.2.1 Machine Learning Application	29-30
5.2.2 Relation to Statistics	30-31

5.2.3 Approaches	31-35
5.2.4 Model Assessments	35
5.2.5 Ethics	36
5.2.6 Software Suite	36
5.2.7 Support Vector Machines	37-41
5.2.7.1 Linear Support Vector Machines	40
5.2.7.2 Implementation of SVM	41
<b>6. Implementation</b>	<b>42-49</b>
6.1 Modules	42-43
6.1.1 Module Description	42-43
6.1.1.1 Fetch Tweets Module	42
6.1.1.2 Pre-process Module	42
6.1.1.3 Stemming Module	42
6.1.1.4 Classifier Module	42
6.1.1.5 Polarity Module	43
6.1.1.1 Visualizing Analytics Module	43
6.2 Sample Codes	43-49
6.2.1 Fetch Tweets Module	44-45
6.2.2 Pre-process Module	45-46
6.2.3 Stemming Module	46
6.2.4 Classifier Module	47
6.2.5 Polarity Module	48
6.2.6 Visualizing Analytics Module	48-49
<b>7. Testing</b>	<b>50-54</b>
7.1 Software Testing	50
7.2 Testing Methodologies	50-51
7.2.1 Verification	50
7.2.2 Validation	50
7.2.3 White Box Testing	50
7.2.4 Black Box Testing	51
7.3 Testing Levels	51
7.3.1 Unit Testing	51
7.3.2 Integration Testing	51

7.4 Test Cases	51-52
7.5 Result and Analytics	52-54
<b>8. Advantages and Applications</b>	<b>55-58</b>
8.1 Advantages and Applications of SVM	55-56
8.2 Disadvantages of SVM	56
8.3 Advantages and Applications of Opinion Mining	56
8.4 Disadvantages of Opinion Mining	56-57
8.5 Advantages and Applications of Machine Learning	57-58
<b>9. Conclusion</b>	<b>59</b>
<b>10. Future Enhancements</b>	<b>60</b>
<b>11. References</b>	<b>61</b>

## FIGURES

4.1 System Architecture	11
4.2.1.1 Use case Diagram 1	13
4.2.1.2 Use case Diagram 2	13
4.2.2 Class Diagram	14
4.2.3 Sequence Diagram	15
4.2.4 Collaboration Diagram	16
4.2.5 Activity Diagram	17
4.2.6 Component Diagram	18
4.2.7 Deployment Diagram	19
5.2.7.1 Hyperplanes	38
5.2.7.2 Kernel Machine	39
5.2.7.3 Linear SVM	40

## **SCREENSHOTS**

6.2 Creating Twitter Developer Application	43
7.5.1 Sentiment Analytics on Jeep Car	53
7.5.2 Sentiment Analytics on Nifty Stock Exchange	53
7.5.3 Precision Support	54

## **TABLES**

7.4 Test Cases	52
----------------	----

# 1. Introduction

With the dawn of the internet and digital age, websites have become a regular way of exchange of information. Today over a billion websites are influencing our lives in ways unnoticed. Websites have been growing rapidly in the digital age. In the past decade, new forms of communication, such as micro-blogging and text messaging have emerged and become ubiquitous. While there is no limit to the range of information conveyed by tweets and texts, often these short messages are used to share opinions and sentiments that people have about what is going on in the world around them. Tweets and texts are short: a sentence or a headline rather than a document. The language used is very informal, with creative spelling and punctuation, misspellings, slang, new words, URLs, and genre-specific terminology and abbreviations, such as, RT for “re-tweet” and #hash tags, which are a type of tagging for Twitter messages. Another aspect of social media data such as Twitter messages is that it includes rich structured information about the individuals involved in the communication. For example, Twitter maintains information of who follows whom and re-tweets and tags inside of tweets provide discourse information.

Opinion Mining (also known as Sentiment Analysis) refers to the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials. Sentiment Analysis (SA) is one of the most widely studied applications of Natural Language Processing (NLP) and Machine Learning (ML). This field has grown tremendously with the advent of the Web. The Internet has provided a platform for people to express their views, emotions and opinions towards products, people and life in general. Thus, the Internet is now a vast resource of opinion rich textual data. The goal of opinion mining is to harness this data in order to obtain important information regarding public opinion, that would help make smarter business decisions, political campaigns and better product consumption. The recent trends in opinion mining techniques have moved towards building generative models that can capture complex contextual phenomena. Conversely, due to the unavailability of annotated data, the focus is moving towards unsupervised approaches that use the power of co-occurrence to solve the problem. Since, the web has a huge amount of opinionated data, in the form of blogs, reviews, etc., the unsupervised approaches flourish.

## **1.1 Motivation**

Many consumers take the suggestions and opinions from others before they buy goods and services. The impact of digital social media has become one of the medium of sharing and viewing their perspective. The public use this as a source of reviews on goods and services which can impact the future of it. Thus, there is a plethora of opinions available on the internet, from a consumers' point of view extracting opinions about a particular entity is very important.

While the consumers opinions change the producer uses this opinions as a source of improvement of product or change in the market in which it is consumed. It also provides a way of knowing the loyal customers and brand ambassadors of the product. These opinions thus shape the future of the product or the service. The vendors need a system that can identify trends in customer reviews and use them to improve their product or service and also identify the requirements

Apart from the economic value of the opinions, these sentiments can also have an impact on how governments are selected, shaped and administered. Thus, the value for these sentiments is enormous and can have potential value of how these can be used for a better purpose.

Hence this prospective data can be implemented for data analysis and can be used for various commercial and administrative strategies.

## **1.2 Objective**

Twitter is one of the micro blogging social networking website that has a large user base which generates opinions in the form of tweets that range upto billions. This can be one of the potential way of determining the sentiments of public over various topics.

This project is designed and implemented to do as the above statements refer by using machine learning methodologies such as support vector machines and existing libraries for natural language processing of tweets about opinions and determine the polarity of the tweets.

The classification model helps us to train the data over different pre-evaluated and opinionated texts to make the model more accurate and less imperfect.

## 2. Literature Survey

The importance of the data generated by the Web 2.0 phenomena is apparent. The widespread availability of consumer generated media (CMS) such as blogs, message boards, and news articles post great opportunities as well as risks to today's enterprises. As of 2009 companies have already been applying this realization. The complexity issue is still relevant even when narrowing the search space to a single source of information. Facebook is a good example of an extraordinarily popular social media platform that generates a large amount of text that could be analysed through its API.

The useful data is just as plentiful as the irrelevant. There are endless amounts of both being produced in outlets across the internet. It is the relevant subjective human opinion that is a rich and useful source for marketing intelligence, social psychologists, and others interested in extracting and mining opinions, views, moods, and attitudes. With this information sentiment analysis can begin. The challenge that exists after the search space is established is to locate the relevant data. After the relevant data is established it can then be assessed for sentiment. These two stages are commonly referred to as subjectivity classification and polarity classification.

Subjectivity classification is a task to investigate whether a paragraph presents the opinion of its author or reports facts. Subjectivity classification can prevent the polarity (i.e: sentiment) classifier from considering irrelevant or even potentially misleading text. Depending on the application, contextual matching or similar may be applied to the resulting data that is already deemed subjective.

This multi-class sentiment approach will likely be the standard of the future. It will only evolve to learn how to meet our needs more effectively. The Sentiment Analysis provides the following uses to the user, it provides users an easy and effective way of knowing about a person or product etc by simply looking at the graph without wasting their valuable time in reading either the whole tweets or wiki data and analysing it. The opinion mining using machine learning concepts provides a efficient way to understand and produce the sentiments on various topics through processing wide array of data. This document furnishes the Use Case diagrams, Class diagrams, Sequence diagrams, Collaboration diagram, activity diagram, component diagram and deployment diagram helpful to understand the project on opinion mining implemented using support vector machine classifier.

## **2.1 Feasibility Study**

Feasibility study is an important phase in the software development process. It enables the developer to have an assessment of the product being developed. It refers to the feasibility study of the product in terms of outcome, operational use and technical support required for implementing it. Feasibility study should be performed on the basis of various parameters by various feasibility studies which are: 1) Operational feasibility 2) Technical feasibility 3) Economic feasibility.

### **2.1.1 Operational Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. Users level of confidence must be raised so that user is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

### **2.1.2 Technical Feasibility**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### **2.1.3 Economic Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.



### **3. Analysis**

The Analysis Phase is where the project lifecycle begins. The Analysis Phase is where you break down the deliverables in the high-level Project Charter into the more detailed business requirements. The Analysis Phase is also the part of the project where you identify the overall direction that the project will take through the creation of the project strategy documents. Gathering requirements is the main attraction of the Analysis Phase. The process of gathering requirements is usually more than simply asking the users what they need and writing their answers down. Depending on the complexity of the application, the process for gathering requirements has a clearly defined process of its own. This process consists of a group of repeatable processes that utilize certain techniques to capture, document, communicate, and manage requirements. This formal process, which will be developed in more detail, consists of four basic steps;

- 1) Elicitation – I ask questions, you talk, I listen
- 2) Validation – I analyse, I ask follow-up questions
- 3) Specification – I document, I ask follow-up question
- 4) Verification – We all agree

Although gathering requirements is the main focus during the Analysis Phase, there are other important activities during this phase as well. One is to create a Requirement Management Plan to define how the requirements will be documented, communicated, tracked and changed throughout the rest of the project life-cycle. This plan will specifically address establishing a baseline, a change control process, and a way to track the requirements through the rest of the lifecycle. Another important activity is to set the overall direction for work that does not take place until later. This is accomplished through a series of direction-setting strategy documents. For instance, once you have your requirements, you can start to set the overall direction for training in a Training Strategy document. The strategies are at a high-level and are later defined at a lower level before they are finally implemented toward the end of the project. The analysis gives an overview about the

needs for creating a system to implement opinion mining and classification model of SVM.

### **3.1 Existing System**

Existing system is defined to be the computational system that already exists to do a similar task currently accepted and feasible. In the current project, the existing system is the use of simple raw data for classification from the like of techniques such as Naive Bayes and Decision trees. Also using static data derived at one point of time it can have higher time complexity.

### **3.2 Proposed System**

Proposed system is a new and innovative change or feature that is implemented on existing system or is built from ground up. The system is an opinion miner that generates sentiment using support vector machine(SVM) classification algorithm which falls under supervised machine learning. Selection of SVM is due to its better use case for text classification.

### **3.3 Requirement Specification**

The system analysis is done keeping in mind the following key requirements:

1. Provide a simple user-friendly interface for client for effective use.
2. Provide precise opinion mining insights.

#### **3.3.1 Software Requirements**

- Operating System: Windows 8 and above
- Programming Language: Python
- IDE: Spyder
- API: Twitter Search API
- Package Manager: Anaconda
- Libraries/Toolkits: Tweepy, Sci-kit learn, Matplotlib and NLTK

##### **3.3.1.1 Operating System:**

###### **Windows 10:**

An operating system (OS) is system software that manages computer hardware and software resources and provides common services for computer programs. Windows being the most widely used operating system around the world

is very resourceful with the current project.

### **3.3.1.2 Libraries/ Toolkits**

#### **Sci-kit Learn 0.19.1:**

Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

#### **NLTK 3.2.5:**

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language. NLTK includes graphical demonstrations and sample data. It is accompanied by a book that explains the underlying concepts behind the language processing tasks supported by the toolkit, plus a cookbook. NLTK is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, cognitive science, artificial intelligence, information retrieval, and machine learning. NLTK supports classification, tokenization, stemming, tagging, parsing, and semantic reasoning functionalities.

#### **Tweepy 3.5.0:**

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. One of the main usage cases of tweepy is monitoring for tweets and doing actions when some event happens. Although the documentation for tweepy is a bit scarce, the fact that it heavily relies on the Twitter API, which has excellent documentation, makes it probably the best Twitter library for Python.

#### **Matplotlib 2.2.2:**

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural pylab interface based on a state machine (like OpenGL).

### **3.3.1.3 Languages**

#### **Python 3.6.4:**

Python is an interpreted high-level programming language for general-purpose programming. Python has a design philosophy that emphasizes code readability. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

### **3.3.1.4 Integrated Development Environment**

#### **Spyder:**

Spyder (formerly Pydee) is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates NumPy, SciPy, Matplotlib and IPython, as well as other open source software. Spyder is extensible with plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments.

### **3.3.1.5 Application Programming Interfaces**

#### **Twitter Search API:**

This search API searches against a sampling of recent Tweets published in the past 7 days. Part of the ‘public’ set of APIs. Returns a collection of relevant Tweets matching a specified query. Please note that Twitter’s search service and, by extension, the Search API is not meant to be an exhaustive source of Tweets.

### **3.3.1.6 Package Manager**

#### **Anaconda:**

Anaconda is a free and open source distribution of the Python and R programming languages for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment. Package versions are managed by the package management system conda. It has a powerful collaboration and package management for open source and private projects. It is one of the fastest growing open source community.

### **3.3.2 Hardware Requirements**

- Processor: 2.5GHz and above.
- Main Memory: 4GB and above.
- Hard Disk: 256GB and above.

#### **3.3.2.1 Processor**

The processor should be a 64 bit with a minimum speed of 2.5GHz. Preferably Intel Core i3 5<sup>th</sup> Gen.

#### **3.3.2.2 Main Memory**

The system should have a minimum RAM of 4 GB for faster computation since large data sets are analysed.

#### **3.3.2.3 Hard Disk**

Hard disk size can start from 256GB and above depending on the size of training data that should be available for classification.

### **3.3.3 Pre-Opinionated Data Sources**

- Stanford sentiment 140
- Cornell polarity dataset
- Michigan Kaggle Social media dataset

#### **3.3.3.1 Stanford Sentiment 140**

Pre-created csv file which contain 16,00,000 tweets and classified and subjected made by Stanford university.

### **3.3.3.2 Cornell Polarity Dataset**

Includes 1000 positive and 1000 negative pre-processed sentiment-based movie reviews created by Cornell University library.

### **3.3.3.3 Michigan Kaggle Social Media Dataset**

Contains 7086 trained data with positive tweets represented with 1 and negative with 0 integer made by Michigan University for Kaggle contest.

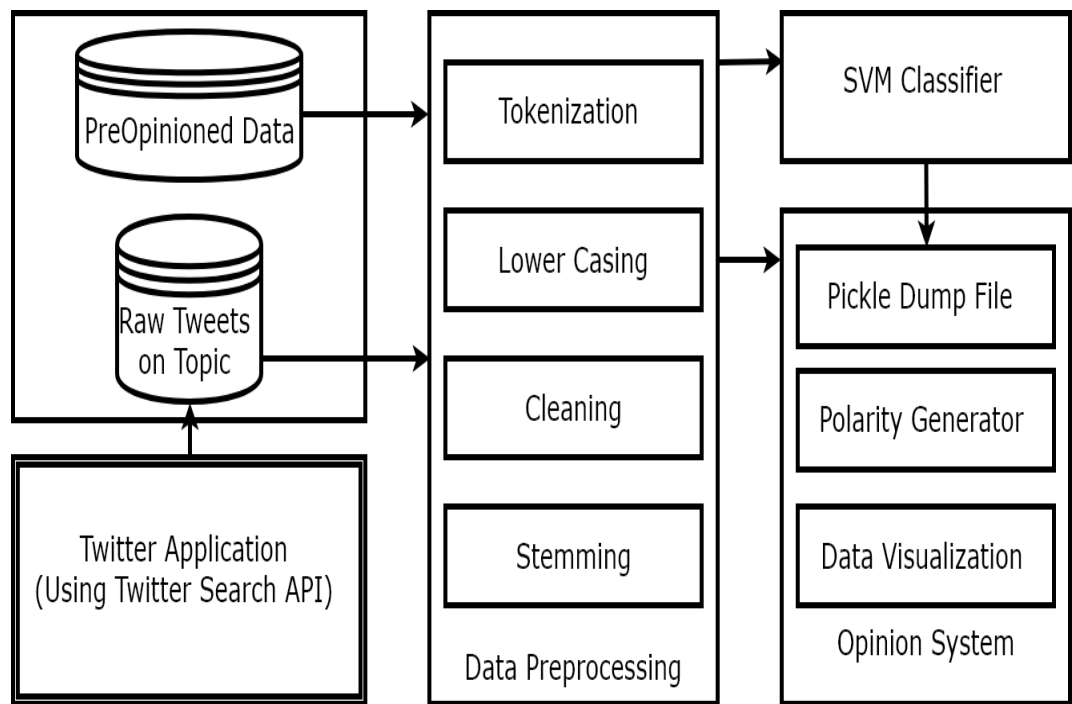
## 4. Design

A model is a simplification of reality. A model provides the blueprints of a system. A model may be structural, emphasizing the organization of the system, or it may be behaviour, emphasizing the dynamics of the system.

The objective of this phase is to transform business requirements identified during previous phases, into a detailed system architecture which is feasible, robust and brings value to the organization.

### a. Architecture

The whole system is designed in a way that the communication between various units of processing whether it is cleaning data or classifying occur in a stable way.



**Fig. 4.1 System Architecture**

The way it functions can be seen in the above Fig. 4.1 where first the pre-opinionated data is sent through data pre-processing and later classified into clusters of data and dumped into pickle file since the pre-processing over training data takes a lot of time. After classification of training data, we use live twitter tweets over various topics to be sent, which uses the clustering of training data and fits the current tweets and generates polarity over each sentence. The mean of each polarity is analysed and visualized in the form of graph and pie chart.

## 4.2 UML Diagrams

UML diagrams is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct and document the artifacts of a software intensive system. UML (Unified Modelling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

UML offers a way to visualize a system's architectural blueprints in a diagram, including elements such as: any activities (jobs), individual components of the system, and how they can interact with other software components, how the system will run, how entities interact with others (components and interfaces), external user interface.

Although originally intended for object-oriented design documentation, UML has been extended to a larger set of design documentation (as listed above) and been found useful in many contexts. It is important to distinguish between the UML model and the set of diagrams of a system. A diagram is a partial graphic representation of a system's model. The set of diagrams need not completely cover the model and deleting a diagram does not change the model. UML diagrams represent two different views of a system model:

- Static (or structural) view: emphasizes the static structure of the system using objects, attributes, operations and relationships. It includes class diagrams and composite structure diagrams.
- Dynamic (or behavioural) view: emphasizes the dynamic behaviour of the system by showing collaborations among objects and changes to the internal states of objects. This view includes sequence diagrams, activity diagrams and state machine diagrams.

Here we define 7 major UML diagrams that elaborates the structure and behaviour of the system using visual and text data. The structural diagrams consist of Class, Component and Deployment diagrams. The behavioural diagrams consist of Use case, Sequence, Collaboration and Activity diagrams. Together these 7 visualizations help to understand the design of the system.



### 4.2.1 Use case Diagram

A use case specifies the behaviour of a system or a part of a system and is a description of set of sequences of actions, including variants, that a system performs to yield an observable result of value to an actor.

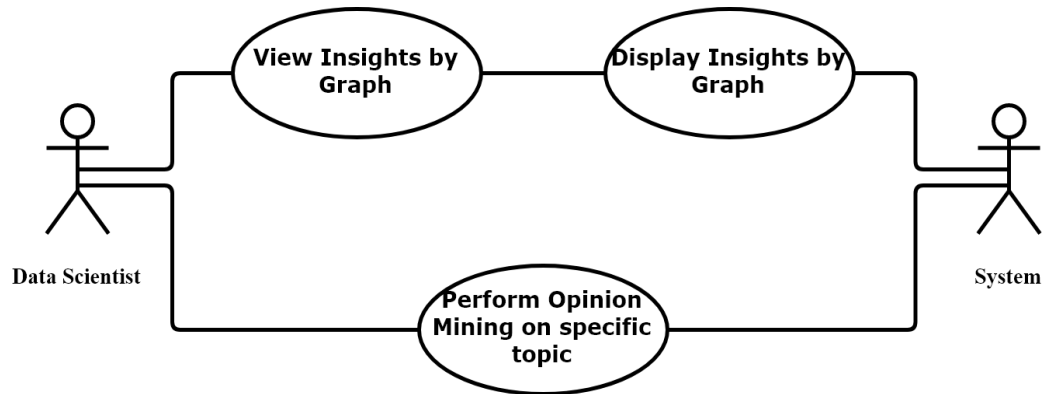


Fig. 4.2.1.1 Use case Diagram 1

From the Fig. 4.2.1.1 we can interpret how the data scientist or analyst wants to perform opinion mining on a topic through the system and view the graphs that the system plots.

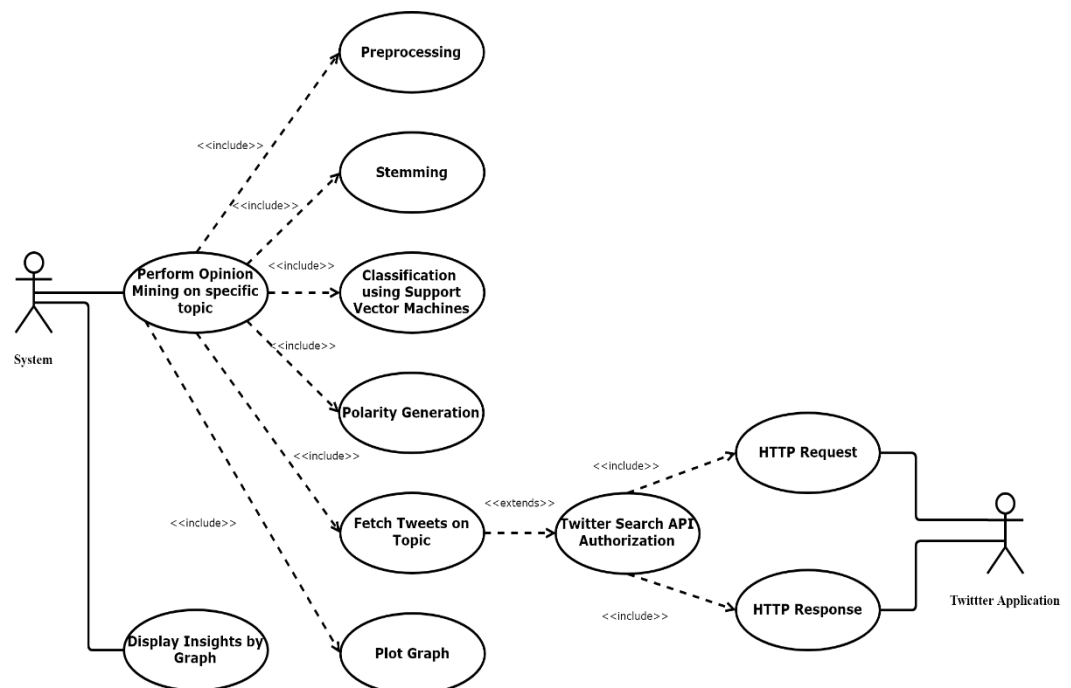
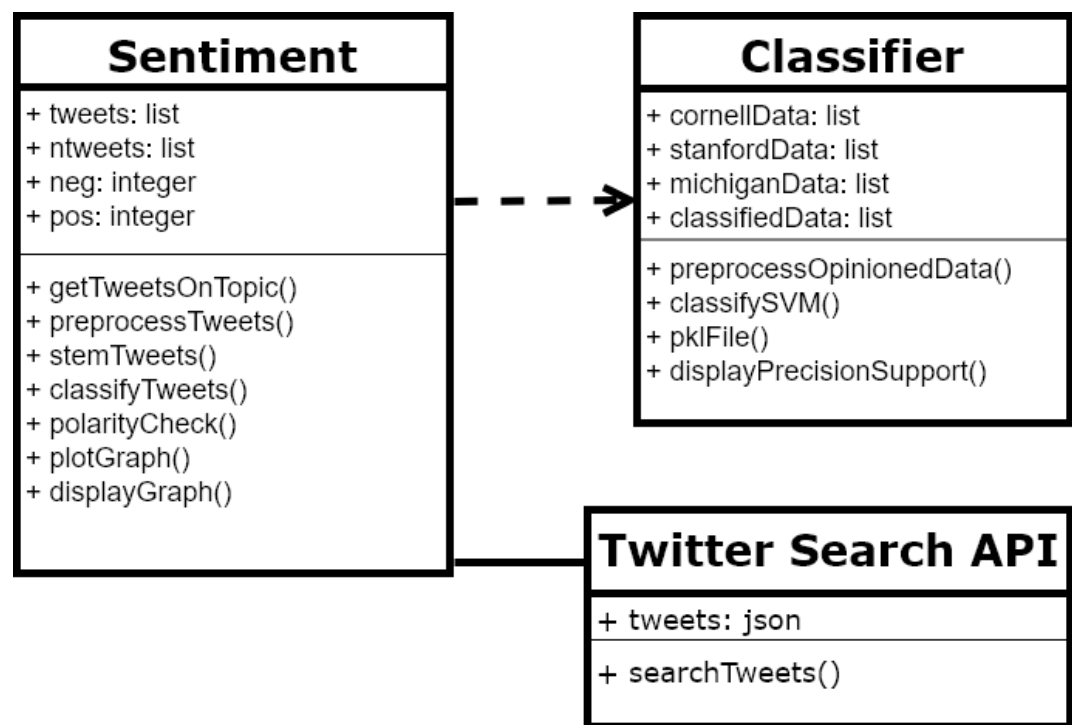


Fig. 4.2.1.2 Use Case Diagram 2

The Fig. 4.2.1.2 describes about how the opinion mining is carried out which has many decomposed use cases as pre-processing, stemming, classification, polarity generation, plot graph and fetching tweets. The actor twitter application represents the request-response cycle for twitter data transfer with the system through twitter search API.

#### 4.2.2 Class Diagrams

A class diagram shows a set of classes, interfaces, and collaborations and their relationships. It is used to model the static design view of the system. Class diagrams are useful for constructing executable systems through forward and reverse engineering.



**Fig. 4.2.2 Class Diagram**

The class diagram in the Fig. 4.2.2 explains the classes in the application responsible for sentiment analysis and other for classification training. The main class contains attributes such as tweets, ntweets, neg and pos and functions responsible for fetching tweets, pre-processing, stemming, polarity generation and plotting the graph.

The main class is dependent on the classifier based on Support vector machine which is trained with already opinionated data to classify the current data accurately. The classifier contains methods for pre-processing, classifying

functions, displaying precision support and dumping the data into a pickle file.

The class diagram helps to properly understand how each of the methods are dependent on the classes and its attributes which will help in proper sentiment processing over time. Note that the class is dependent on several libraries which is not depicted in the class diagram.

#### 4.2.3 Sequence Diagram

A Sequence diagram is an interaction diagram that emphasizes the time ordering of messages. Graphically a sequence diagram is a table that shows objects arranged in x-axis and messages ordered in increasing time along y-axis. They are called event diagrams or event scenarios. A sequence diagram shows as parallel vertical lines, different processes or objects that live simultaneously and horizontal arrows the messages exchanged between them in order which they occur.

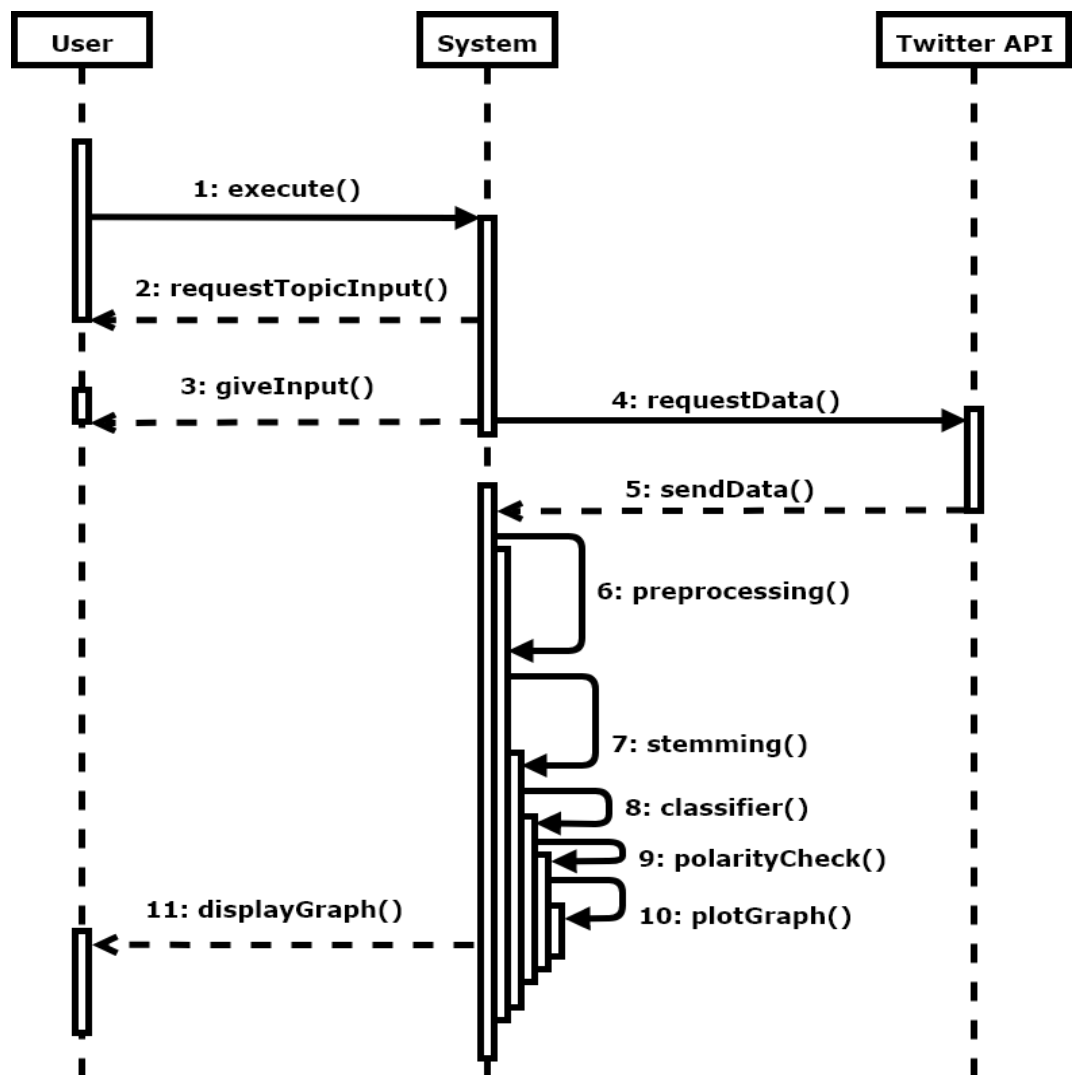
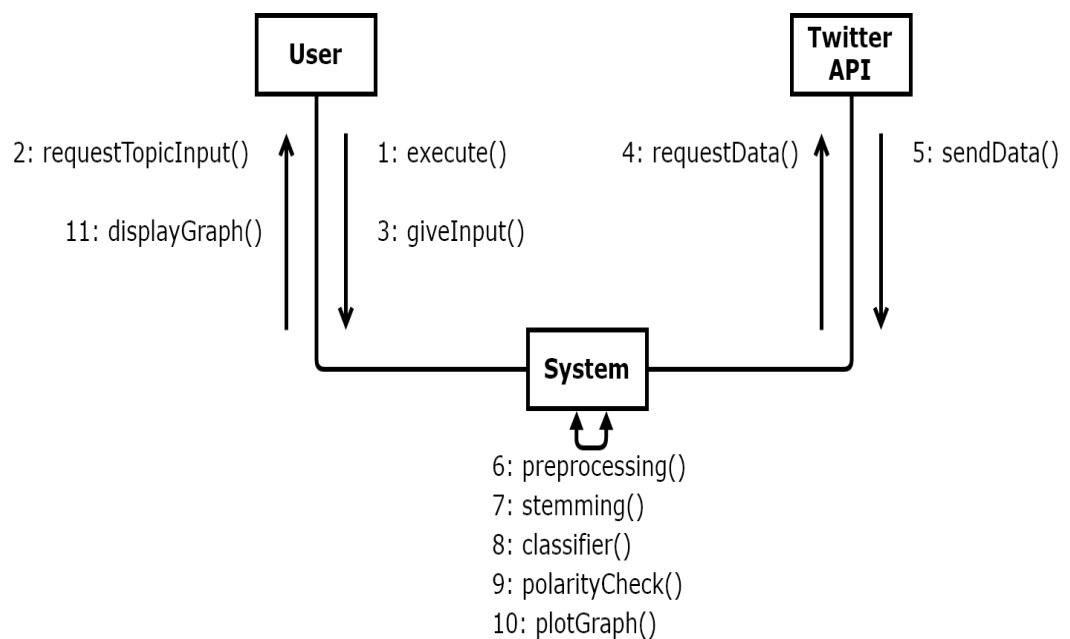


Fig. 4.2.3 Sequence Diagram

The sequence diagram tells us the timing of operations that take place between objects user, system and twitter application as depicted in the Fig. 4.2.3. The sequence starts with the execution from the user with system giving a prompt to user to enter the topic for analysis and later the tweets are fetched from the twitter API to perform opinion mining and plot the graph.

#### 4.2.4 Collaboration Diagram

A collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Graphically the collaboration diagram is a collection of vertices and arcs. It portrays the roles, functionality and behaviour of individual objects as well as the overall operation of the system in real time. They are best suited for showing the simple interactions among objects.



**Fig. 4.2.4 Collaboration Diagram**

The collaboration diagram is like sequence diagram but only shows the exchange of messages between the different objects i.e.: user, system and twitter application. The collaboration diagram is a communication diagram and is better for visualizing smaller number of objects since as the number grows message exchanges become complex and not visually readable. Collaboration diagram shows object organization compared to sequence diagram. These interaction diagrams are very helpful in both forward and reverse engineering.

#### 4.2.5 Activity Diagram

An activity diagram shows the flow of control from activity to activity. An activity is an ongoing execution within a state machine. It is essentially a flowchart modelling the dynamic aspects of the system.

The purpose of activity diagram is:

- Draw activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

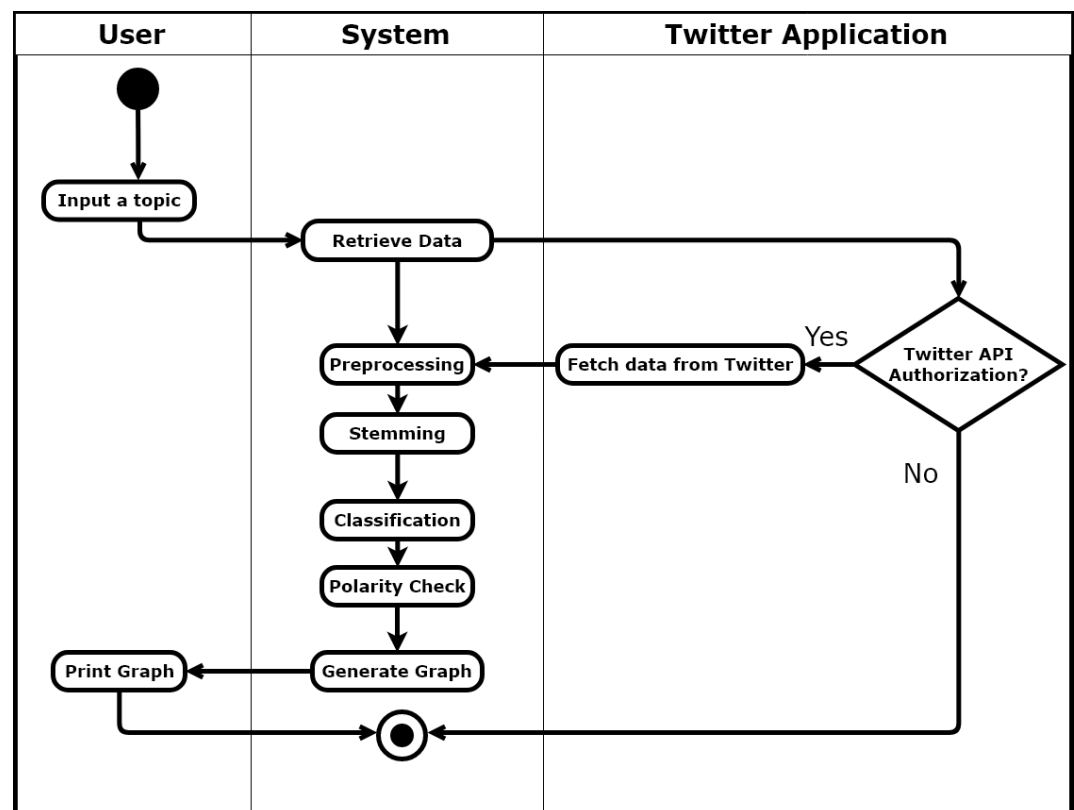


Fig. 4.2.5 Activity Diagram

The activity diagram explains the flow of how the data is processed in sequence and explains where the decisions are taken for the flow to change. The swimlanes here describe where each of the activities is taking place i.e. the user end or the system end or the twitter application end.

#### 4.2.6 Component Diagram

Component diagrams are used to model the static implementation view of a system. This involves modelling the physical things that reside on a node, such as executables, libraries, tables, files and documents. Component diagrams can be used to:

- Model the components of system.
- Model the database schema.
- Model the executables of an application.
- Model the system's source code.

The organization of these components help the developers to better understand the semantics of the system and flow of resources and data.

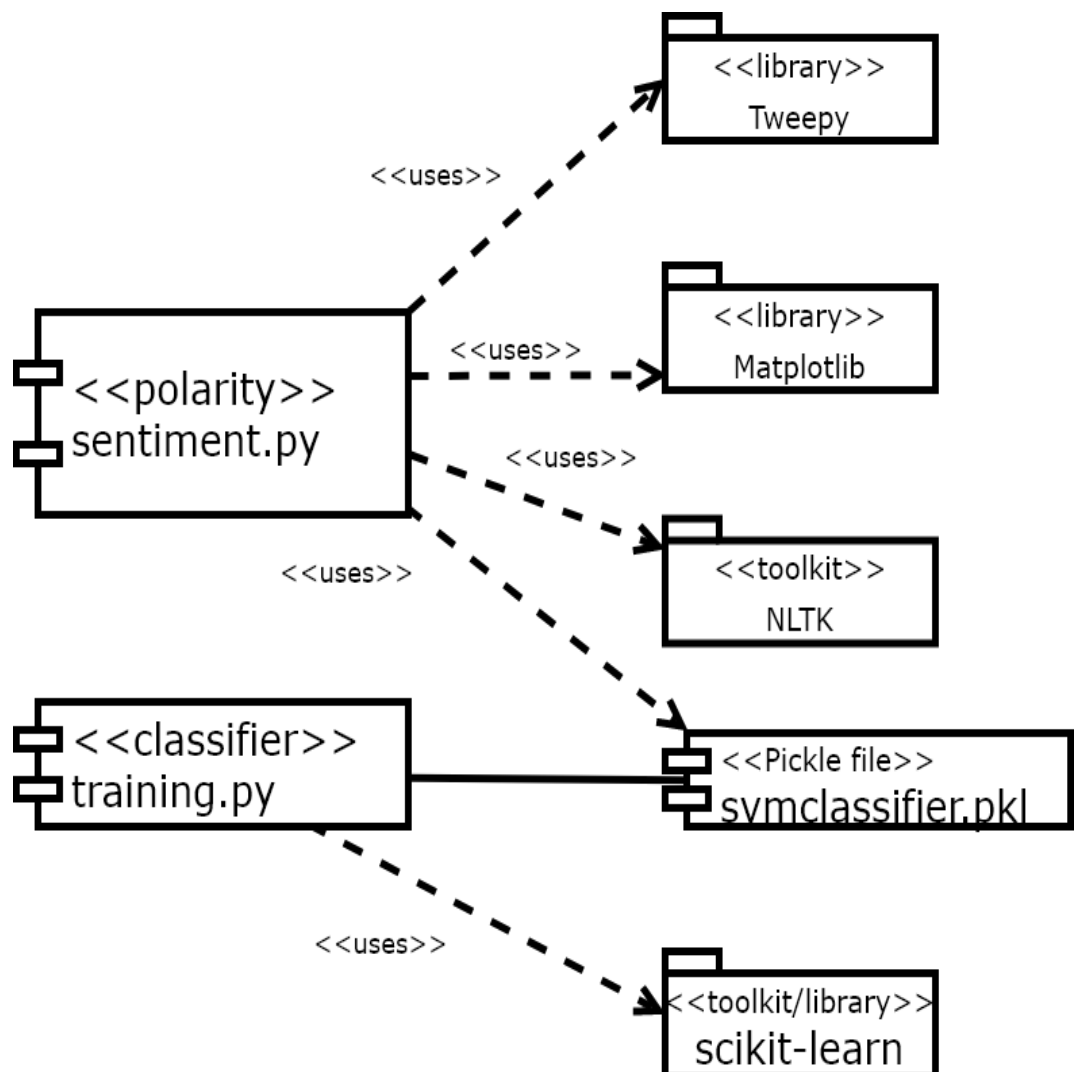


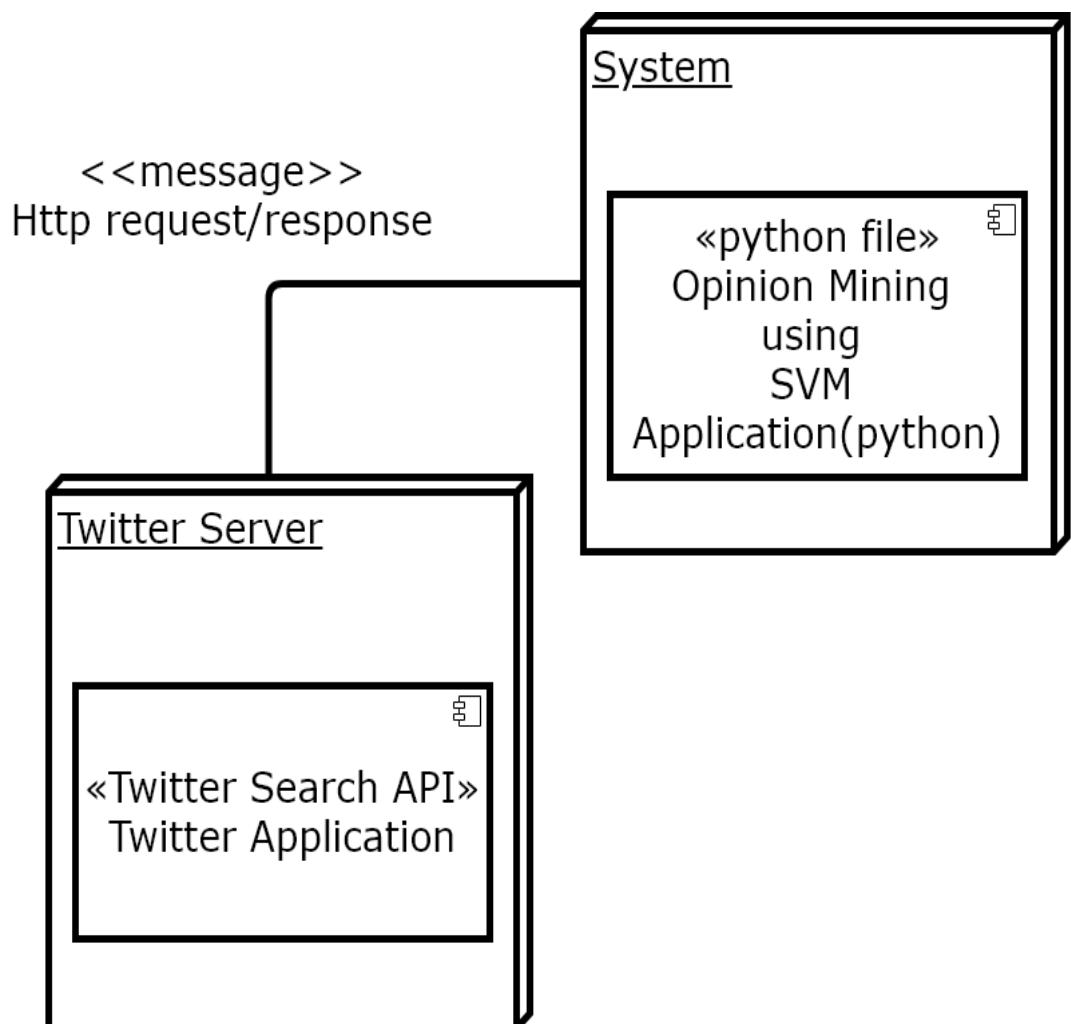
Fig. 4.2.6 Component Diagram

#### 4.2.7 Deployment Diagram

Deployment diagrams show the configuration of runtime processing nodes and the components that live on them. They model the topology of the hardware on which your system executes. The purpose of deployment diagrams is:

- Visualize the hardware topology of a system.
- Describe the hardware components used to deploy software components.
- Describe the runtime processing nodes.

They are mainly used by system engineers. To meet software applications, need of hardware resources, the hardware components must be designed efficiently and in a cost effective way.



**Fig 4.2.7 Deployment diagram**

The Fig. 4.1.7 shows how the two systems are connected to through a HTTP protocol.

## **5. Technology**

### **5.1 Opinion Mining**

Opinion mining (sometimes known as sentiment analysis or emotion AI) refers to the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine. Generally speaking, sentiment analysis aims to determine the attitude of a speaker, writer, or other subject with respect to some topic or the overall contextual polarity or emotional reaction to a document, interaction, or event. The attitude may be a judgement or evaluation, affective state (that is to say, the emotional state of the author or speaker), or the intended emotional communication (that is to say, the emotional effect intended by the author or interlocutor).

#### **5.1.1 Types**

A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level—whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, ‘beyond polarity’ sentiment classification looks, for instance, at emotional states such as “angry”, “sad”, and “happy”. Precursors to sentimental analysis include the General Inquirer, which provided hints toward quantifying patterns in text and, separately, psychological research that examined a person's psychological state based on analysis of their verbal behaviour. Subsequently, the method that looked specifically at sentiment and identified individual words and phrases in text with respect to different emotional scales.

A current system based on their work, called Effect Check, presents synonyms that can be used to increase or decrease the level of evoked emotion in each scale. Many other subsequent efforts were less sophisticated, using a mere polar view of sentiment, from positive to negative when applied with different methods for detecting the polarity of product reviews and movie reviews respectively. This work is at the document level.



One can also classify a document's polarity on a multi-way scale, expanded the basic task of classifying a movie review as either positive or negative to predict star ratings on either a 3- or a 4-star scale.

First steps to bringing together various approaches—learning, lexical, knowledge-based, etc.—were taken in the 2004 AAAI Spring Symposium where linguists, computer scientists, and other interested researchers first aligned interests and proposed shared tasks and benchmark data sets for the systematic computational research on affect, appeal, subjectivity, and sentiment in text.

Even though in most statistical classification methods, the neutral class is ignored under the assumption that neutral texts lie near the boundary of the binary classifier, several researchers suggest that, as in every polarity problem, three categories must be identified. Moreover, it can be proven that specific classifiers such as the Max Entropy and the SVMs can benefit from the introduction of a neutral class and improve the overall accuracy of the classification. There are in principle two ways for operating with a neutral class. Either, the algorithm proceeds by first identifying the neutral language, filtering it out and then assessing the rest in terms of positive and negative sentiments, or it builds a three-way classification in one step.

This second approach often involves estimating a probability distribution over all categories (e.g. naive Bayes classifiers as implemented by the NLTK). Whether and how to use a neutral class depends on the nature of the data: if the data is clearly clustered into neutral, negative and positive language, it makes sense to filter the neutral language out and focus on the polarity between positive and negative sentiments. If, in contrast, the data are mostly neutral with small deviations towards positive and negative affect, this strategy would make it harder to clearly distinguish between the two poles.

A different method for determining sentiment is the use of a scaling system whereby words commonly associated with having a negative, neutral, or positive sentiment with them are given an associated number on a  $-10$  to  $+10$  scale (most negative up to most positive) or simply from  $0$  to a positive upper limit such as  $+4$ . This makes it possible to adjust the sentiment of a given term relative to its environment (usually on the level of the sentence).

When a piece of unstructured text is analysed using natural language processing, each concept in the specified environment is given a score based on the way sentiment words relate to the concept and its associated score.

This allows movement to a more sophisticated understanding of sentiment because it is now possible to adjust the sentiment value of a concept relative to modifications that may surround it. Words, for example, that intensify, relax or negate the sentiment expressed by the concept can affect its score. Alternatively, texts can be given a positive and negative sentiment strength score if the goal is to determine the sentiment in a text rather than the overall polarity and strength of the text.

#### **5.1.1.1 Subjectivity/Objectivity Identification**

This task is commonly defined as classifying a given text (usually a sentence) into one of two classes: objective or subjective. This problem can sometimes be more difficult than polarity classification. The subjectivity of words and phrases may depend on their context and an objective document may contain subjective sentences (e.g., a news article quoting people's opinions). Moreover, results are largely dependent on the definition of subjectivity used when annotating texts.

#### **5.1.1.2 Feature/Aspect-based**

It refers to determining the opinions or sentiments expressed on different features or aspects of entities, e.g., of a cell phone, a digital camera, or a bank. A feature or aspect is an attribute or component of an entity, e.g., the screen of a cell phone, the service for a restaurant, or the picture quality of a camera. The advantage of feature-based sentiment analysis is the possibility to capture nuances about objects of interest. Different features can generate different sentiment responses, for example a hotel can have a convenient location, but mediocre food. This problem involves several sub-problems, e.g., identifying relevant entities, extracting their features/aspects, and determining whether an opinion expressed on each feature/aspect is positive, negative or neutral. The automatic identification of features can be performed with syntactic methods, with topic modelling, or with deep learning.

#### **5.1.2 Methods and Features**

Existing approaches to sentiment analysis can be grouped into three main categories: knowledge-based techniques, statistical methods, and hybrid approaches. Knowledge-based techniques classify text by affect categories based

on the presence of unambiguous affect words such as happy, sad, afraid, and bored. Some knowledge bases not only list obvious affect words, but also assign arbitrary words a probable “affinity” to particular emotions. Statistical methods leverage on elements from machine learning such as latent semantic analysis, support vector machines, “bag of words” and Semantic Orientation — Point wise Mutual Information. More sophisticated methods try to detect the holder of a sentiment (i.e., the person who maintains that affective state) and the target (i.e., the entity about which the affect is felt).

To mine the opinion in context and get the feature about which the speaker has opined, the grammatical relationships of words are used. Grammatical dependency relations are obtained by deep parsing of the text. Hybrid approaches leverage on both machine learning and elements from knowledge representation such as ontologies and semantic networks in order to detect semantics that are expressed in a subtle manner, e.g., through the analysis of concepts that do not explicitly convey relevant information, but which are implicitly linked to other concepts that do so.

Open source software tools deploy machine learning, statistics, and natural language processing techniques to automate sentiment analysis on large collections of texts, including web pages, online news, internet discussion groups, online reviews, web blogs, and social media. Knowledge-based systems, on the other hand, make use of publicly available resources, to extract the semantic and affective information associated with natural language concepts. Sentiment analysis can also be performed on visual content, i.e., images and videos. One of the first approach in this direction is SentiBank utilizing an adjective noun pair representation of visual content. In addition, the clear majority of sentiment classification approaches rely on the bag-of-words model, which disregards context, grammar and even word order. Approaches that analyses the sentiment based on how words compose the meaning of longer phrases have shown better result, but they incur an additional annotation overhead.

A human analysis component is required in sentiment analysis, as automated systems are not able to analyse historical tendencies of the individual commenter, or the platform and are often classified incorrectly in their expressed sentiment. Automation impacts approximately 23% of comments that are correctly classified by humans. However, humans often disagree, and it is argued that the inter-human

agreement provides an upper bound that automated sentiment classifiers can eventually reach.

Sometimes, the structure of sentiments and topics are fairly complex. Also, the problem of sentiment analysis is non-monotonic in respect to sentence extension and stop-word substitution (compare THEY would not let my dog stay in this hotel vs I would not let my dog stay in this hotel). To address this issue several rule-based and reasoning-based approaches have been applied to sentiment analysis, including defeasible logic programming. Also, there are several tree traversal rules applied to syntactic parse tree to extract the topicality of sentiment in open domain setting.

### **5.1.3 Evaluation**

The accuracy of a sentiment analysis system is, in principle, how well it agrees with human judgements. This is usually measured by variant measures based on precision and recall over the two target categories of negative and positive texts. However, according to research human raters typically only agree about 80% of the time (see Inter-rater reliability). Thus, a program which achieves 70% accuracy in classifying sentiment is doing nearly as well as humans, even though such accuracy may not sound impressive. If a program were “right” 100% of the time, humans would still disagree with it about 20% of the time, since they disagree that much about any answer. On the other hand, computer systems will make very different errors than human assessors, and thus the figures are not entirely comparable. For instance, a computer system will have trouble with negations, exaggerations, jokes, or sarcasm, which typically are easy to handle for a human reader: some errors a computer system makes will seem overly naive to a human.

In general, the utility for practical commercial tasks of sentiment analysis as it is defined in academic research has been called into question, mostly since the simple one-dimensional model of sentiment from negative to positive yields rather little actionable information for a client worrying about the effect of public discourse on e.g. brand or corporate reputation.

In recent years, to better fit market needs, evaluation of sentiment analysis has moved to more task-based measures, formulated together with representatives from PR agencies and market research professionals.

#### **5.1.4 Web 2.0**

The rise of social media such as blogs and social networks has fueled interest in sentiment analysis. With the proliferation of reviews, ratings, recommendations and other forms of online expression, online opinion has turned into a kind of virtual currency for businesses looking to market their products, identify new opportunities and manage their reputations. As businesses look to automate the process of filtering out the noise, understanding the conversations, identifying the relevant content and actioning it appropriately, many are now looking to the field of sentiment analysis. Further complicating the matter, is the rise of anonymous social media platforms such as 4chan and Reddit. If web 2.0 was all about democratizing publishing, then the next stage of the web may well be based on democratizing data mining of all the content that is getting published.

One step towards this aim is accomplished in research. Several research teams in universities around the world currently focus on understanding the dynamics of sentiment in e-communities through sentiment analysis. The CyberEmotions project, for instance, recently identified the role of negative emotions in driving social networks discussions.

The problem is that most sentiment analysis algorithms use simple terms to express sentiment about a product or service. However, cultural factors, linguistic nuances and differing contexts make it extremely difficult to turn a string of written text into a simple pro or con sentiment. The fact that humans often disagree on the sentiment of text illustrates how big a task it is for computers to get this right. The shorter the string of text, the harder it becomes.

Even though short text strings might be a problem, sentiment analysis within microblogging has shown that Twitter can be seen as a valid online indicator of political sentiment. Tweets' political sentiment demonstrates close correspondence to parties' and politicians' political positions, indicating that the content of Twitter messages plausibly reflects the offline political landscape.

#### **5.1.5 Application in Recommender System**

For a recommender system, sentiment analysis has been proven to be a valuable technique. A recommender system aims to predict the preference to an item of a target user. Mainstream recommender systems work on explicit data set. For example, collaborative filtering works on the rating matrix, and content-based filtering works on the meta-data of the items.

In many social networking services or e-commerce websites, users can provide text review, comment or feedback to the items. These user-generated text provide a rich source of user's sentiment opinions about numerous products and items. Potentially, for an item, such text can reveal both the related feature/aspects of the item and the users' sentiments on each feature. The item's feature/aspects described in the text play the same role with the meta-data in content-based filtering, but the former are more valuable for the recommender system.

Since these features are broadly mentioned by users in their reviews, they can be seen as the most crucial features that can significantly influence the user's experience on the item, while the meta-data of the item (usually provided by the producers instead of consumers) may ignore features that are concerned by the users. For different items with common features, a user may give different sentiments. Also, a feature of the same item may receive different sentiments from different users. Users' sentiments on the features can be regarded as a multi-dimensional rating score, reflecting their preference on the items.

Based on the feature/aspects and the sentiments extracted from the user-generated text, a hybrid recommender system can be constructed. There are two types of motivation to recommend a candidate item to a user. The first motivation is the candidate item have numerous common features with the user's preferred items, while the second motivation is that the candidate item receives a high sentiment on its features.

For a preferred item, it is reasonable to believe that items with the same features will have a similar function or utility. So, these items will also likely to be preferred by the user. On the other hand, for a shared feature of two candidate items, other users may give positive sentiment to one of them while give negative sentiment to another. Clearly, the high evaluated item should be recommended to the user. Based on these two motivations, a combination ranking score of similarity and sentiment rating can be constructed for each candidate item.

Except the difficulty of the sentiment analysis itself, applying sentiment analysis on reviews or feedback also face the challenge of spam and biased reviews. One direction of work is focused on evaluating the helpfulness of each review. Review or feedback poorly written are hardly helpful for recommender system. Besides, a review can be designed to hinder sales of a target product, thus be harmful to the recommender system even it is well written.

Researchers also found that long and short form of user-generated text should be treated differently. An interesting result shows that short form reviews are sometimes more helpful than long form, because it is easier to filter out the noise in a short form text. For the long form text, the growing length of the text does not always bring a proportionate increase of the number of features or sentiments in the text.

## **5.2 Machine Learning**

Machine learning is a field of computer science that uses statistical techniques to give computer systems the ability to “learn” (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed.

The name machine learning was coined in 1959 by Arthur Samuel. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or infeasible; example applications include email filtering, detection of network intruders or malicious insiders working towards a data breach, optical character recognition (OCR), learning to rank, and computer vision.

Machine learning is closely related to (and often overlaps with) computational statistics, which also focuses on prediction-making through the use of computers. It has strong ties to mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is sometimes conflated with data mining, where the latter subfield focuses more on exploratory data analysis and is known as unsupervised learning. Machine learning can also be unsupervised and be used to learn and establish baseline behavioural profiles for various entities and then used to find meaningful anomalies.

Within the field of data analytics, machine learning is a method used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to “produce reliable,

repeatable decisions and results” and uncover “hidden insights” through learning from historical relationships and trends in the data.

Effective machine learning is difficult because finding patterns is hard and often not enough training data are available; as a result, machine-learning programs often fail to deliver.

Tom M. Mitchell provided a widely quoted, more formal definition of the algorithms studied in the machine learning field: “A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$  if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ .” This definition of the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing's proposal in his paper “Computing Machinery and Intelligence”, in which the question “Can machines think?” is replaced with the question “Can machines do what we (as thinking entities) can do?”. In Turing's proposal the various characteristics that could be possessed by a thinking machine and the various implications in constructing one are exposed.

Machine learning tasks are typically classified into two broad categories, depending on whether there is a learning “signal” or “feedback” available to a learning system:

- 1) Supervised learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs. As special cases, the input signal can be only partially available, or restricted to special feedback:
  - a) Semi-supervised learning: the computer is given only an incomplete training signal: a training set with some (often many) of the target outputs missing.
  - b) Active learning: the computer can only obtain training labels for a limited set of instances (based on a budget), and also has to optimize its choice of objects to acquire labels for. When used interactively, these can be presented to the user for labelling.
  - c) Reinforcement learning: training data (in form of rewards and punishments) is given only as feedback to the program's actions in a dynamic environment, such as driving a vehicle or playing a game against an opponent.
- 2) Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in



itself (discovering hidden patterns in data) or a means towards an end (feature learning).

Applications of machine learning include classification, regression, clustering, density estimation, dimensionality reduction, etc. The current project requires machine learning in the form of classification which will be able to use approaches such as Decision trees, Association rule-based, Artificial Neural networks, Deep learning, Inductive logic programming, Naive Bayes, Support Vector Machines, etc. The proposed system needs text-based classification and hence Support Vector Machines are more reliable in the current context.

### **5.2.1 Machine Learning Applications**

A support vector machine is a classifier that divides its input space into two regions, separated by a linear boundary. Here, it has learned to distinguish black and white circles.

Another categorization of machine learning tasks arises when one considers the desired output of a machine-learned system:

- In classification, inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised way. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are “spam” and “not spam”.
- In regression, also a supervised problem, the outputs are continuous rather than discrete.
- In clustering, a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.
- Density estimation finds the distribution of inputs in some space.
- Dimensionality reduction simplifies inputs by mapping them into a lower-dimensional space. Topic modelling is a related problem, where a program is given a list of human language documents and is tasked to find out which documents cover similar topics.

- Among other categories of machine learning problems, learning to learn learns its own inductive bias based on previous experience. Developmental learning, elaborated for robot learning, generates its own sequences (also called curriculum) of learning situations to cumulatively acquire repertoires of novel skills through autonomous self-exploration and social interaction with human teachers and using guidance mechanisms such as active learning, maturation, motor synergies, and imitation.

### 5.2.2 Relation to Statistics

Machine learning and statistics are closely related fields. According to Michael I. Jordan, the ideas of machine learning, from methodological principles to theoretical tools, have had a long pre-history in statistics. He also suggested the term data science as a placeholder to call the overall field.

Leo Breiman distinguished two statistical modelling paradigms: data model and algorithmic model, wherein "algorithmic model" means more or less the machine learning algorithms like Random forest.

Some statisticians have adopted methods from machine learning, leading to a combined field that they call statistical learning.

A core objective of a learner is to generalize from its experience. Generalization in this context is the ability of a learning machine to perform accurately on new, unseen examples/tasks after having experienced a learning data set. The training examples come from some generally unknown probability distribution (considered representative of the space of occurrences) and the learner has to build a general model about this space that enables it to produce sufficiently accurate predictions in new cases.

The computational analysis of machine learning algorithms and their performance is a branch of theoretical computer science known as computational learning theory. Because training sets are finite and the future is uncertain, learning theory usually does not yield guarantees of the performance of algorithms. Instead, probabilistic bounds on the performance are quite common. The bias–variance decomposition is one way to quantify generalization error.

For the best performance in the context of generalization, the complexity of the hypothesis should match the complexity of the function underlying the data. If the hypothesis is less complex than the function, then the model has underfit the data. If the complexity of the model is increased in response, then the training error

decreases. But if the hypothesis is too complex, then the model is subject to overfitting and generalization will be poorer.

In addition to performance bounds, computational learning theorists study the time complexity and feasibility of learning. In computational learning theory, a computation is considered feasible if it can be done in polynomial time. There are two kinds of time complexity results. Positive results show that a certain class of functions can be learned in polynomial time. Negative results show that certain classes cannot be learned in polynomial time.

### **5.2.3 Approaches**

#### **1) Decision tree learning:**

Decision tree learning uses a decision tree as a predictive model, which maps observations about an item to conclusions about the item's target value.

#### **2) Association rule learning:**

Association rule learning is a method for discovering interesting relations between variables in large databases.

#### **3) Artificial neural networks:**

An artificial neural network (ANN) learning algorithm, usually called “neural network” (NN), is a learning algorithm that is vaguely inspired by biological neural networks. Computations are structured in terms of an interconnected group of artificial neurons, processing information using a connectionist approach to computation. Modern neural networks are non-linear statistical data modelling tools. They are usually used to model complex relationships between inputs and outputs, to find patterns in data, or to capture the statistical structure in an unknown joint probability distribution between observed variables.

#### **4) Deep learning:**

Falling hardware prices and the development of GPUs for personal use in the last few years have contributed to the development of the concept of deep learning which consists of multiple hidden layers in an artificial neural network. This approach tries to model the way the human brain processes light and sound into vision and hearing. Some successful applications of deep learning are computer vision and speech recognition.

5) Inductive logic programming:

Inductive logic programming (ILP) is an approach to rule learning using logic programming as a uniform representation for input examples, background knowledge, and hypotheses. Given an encoding of the known background knowledge and a set of examples represented as a logical database of facts, an ILP system will derive a hypothesized logic program that entails all positive and no negative examples. Inductive programming is a related field that considers any kind of programming languages for representing hypotheses (and not only logic programming), such as functional programs.

6) Support vector machines:

Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other.

7) Clustering:

Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to some predesignated criterion or criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some similarity metric and evaluated for example by internal compactness (similarity between members of the same cluster) and separation between different clusters. Other methods are based on estimated density and graph connectivity. Clustering is a method of unsupervised learning, and a common technique for statistical data analysis.

8) Bayesian networks:

A Bayesian network, belief network or directed acyclic graphical model is a probabilistic graphical model that represents a set of random variables and their conditional independencies via a directed acyclic graph (DAG). For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the

probabilities of the presence of various diseases. Efficient algorithms exist that perform inference and learning.

9) Reinforcement learning:

Reinforcement learning is concerned with how an agent ought to take actions in an environment so as to maximize some notion of long-term reward. Reinforcement learning algorithms attempt to find a policy that maps states of the world to the actions the agent ought to take in those states. Reinforcement learning differs from the supervised learning problem in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected.

10) Representation learning:

Several learning algorithms, mostly unsupervised learning algorithms, aim at discovering better representations of the inputs provided during training. Classical examples include principal components analysis and cluster analysis. Representation learning algorithms often attempt to preserve the information in their input but transform it in a way that makes it useful, often as a pre-processing step before performing classification or predictions, allowing reconstruction of the inputs coming from the unknown data generating distribution, while not being necessarily faithful for configurations that are implausible under that distribution.

Manifold learning algorithms attempt to do so under the constraint that the learned representation is low-dimensional. Sparse coding algorithms attempt to do so under the constraint that the learned representation is sparse (has many zeros). Multilinear subspace learning algorithms aim to learn low-dimensional representations directly from tensor representations for multidimensional data, without reshaping them into (high-dimensional) vectors. Deep learning algorithms discover multiple levels of representation, or a hierarchy of features, with higher-level, more abstract features defined in terms of (or generating) lower-level features. It has been argued that an intelligent machine is one that learns a representation that disentangles the underlying factors of variation that explain the observed data.

11) Similarity and metric learning:

In this problem, the learning machine is given pairs of examples that are considered similar and pairs of less similar objects. It then needs to learn a similarity

function (or a distance metric function) that can predict if new objects are similar. It is sometimes used in Recommendation systems.

#### 12) Sparse dictionary learning:

In this method, a datum is represented as a linear combination of basic functions, and the coefficients are assumed to be sparse. Let  $x$  be a  $d$ -dimensional datum,  $D$  be a  $d$  by  $n$  matrix, where each column of  $D$  represents a basis function.  $r$  is the coefficient to represent  $x$  using  $D$ . Generally speaking,  $n$  is assumed to be larger than  $d$  to allow the freedom for a sparse representation.

Learning a dictionary along with sparse representations is strongly NP-hard and also difficult to solve approximately. A popular heuristic method for sparse dictionary learning is K-SVD.

Sparse dictionary learning has been applied in several contexts. In classification, the problem is to determine which classes a previously unseen datum belongs to. Suppose a dictionary for each class has already been built. Then a new datum is associated with the class such that it's best sparsely represented by the corresponding dictionary. Sparse dictionary learning has also been applied in image de-noising. The key idea is that a clean image patch can be sparsely represented by an image dictionary, but the noise cannot.

#### 13) Genetic algorithms:

A genetic algorithm (GA) is a search heuristic that mimics the process of natural selection and uses methods such as mutation and crossover to generate new genotype in the hope of finding good solutions to a given problem. In machine learning, genetic algorithms found some uses in the 1980s and 1990s. Conversely, machine learning techniques have been used to improve the performance of genetic and evolutionary algorithms.

#### 14) Rule-based machine learning:

Rule-based machine learning is a general term for any machine learning method that identifies, learns, or evolves rules to store, manipulate or apply, knowledge. The defining characteristic of a rule-based machine learner is the identification and utilization of a set of relational rules that collectively represent the knowledge captured by the system. This is in contrast to other machine learners that commonly identify a singular model that can be universally applied to any

instance in order to make a prediction. Rule-based machine learning approaches include learning classifier systems, association rule learning, and artificial immune systems.

#### 15) Learning classifier systems:

Learning classifier systems (LCS) are a family of rule-based machine learning algorithms that combine a discovery component (e.g. typically a genetic algorithm) with a learning component (performing either supervised learning, reinforcement learning, or unsupervised learning). They seek to identify a set of context-dependent rules that collectively store and apply knowledge in a piecewise manner in order to make predictions.

### 5.2.4 Model Assessments

Classification machine learning models can be validated by accuracy estimation techniques like the Holdout method, which splits the data in a training and test set (conventionally 2/3 training set and 1/3 test set designation) and evaluates the performance of the training model on the test set.

In comparison, the N-fold-cross-validation method randomly splits the data in k subsets where the k-1 instances of the data are used to train the model while the kth instance is used to test the predictive ability of the training model. In addition to the holdout and cross-validation methods, bootstrap, which samples n instances with replacement from the dataset, can be used to assess model accuracy.

In addition to overall accuracy, investigators frequently report sensitivity and specificity meaning True Positive Rate (TPR) and True Negative Rate (TNR) respectively.

Similarly, investigators sometimes report the False Positive Rate (FPR) as well as the False Negative Rate (FNR). However, these rates are ratios that fail to reveal their numerators and denominators. The Total Operating Characteristic (TOC) is an effective method to express a model's diagnostic ability.

TOC shows the numerators and denominators of the previously mentioned rates; thus TOC provides more information than the commonly used Receiver operating characteristic (ROC) and ROC's associated Area Under the Curve (AUC).

### 5.2.5 Ethics

Machine learning poses a host of ethical questions. Systems which are trained on datasets collected with biases may exhibit these biases upon use (algorithmic bias), thus digitizing cultural prejudices.

For example, using job hiring data from a firm with racist hiring policies may lead to a machine learning system duplicating the bias by scoring job applicants against similarity to previous successful applicants.

Responsible collection of data and documentation of algorithmic rules used by a system thus is a critical part of machine learning. Because language contains biases, machines trained on language corpora will necessarily also learn bias.

### 5.2.6 Software Suites

Software suites containing a variety of machine learning algorithms include the following:

- 1) Free and open-source software
  - Mahout
  - MLPACK
  - OpenNN
  - scikit-learn
  - Spark MLlib
  - TensorFlow
  - Torch / PyTorch
  - Weka / MOA
- 2) Proprietary software with free and open-source editions
  - KNIME
  - RapidMiner
- 3) Proprietary software
  - Amazon Machine Learning
  - IBM Data Science Experience
  - Google Prediction API
  - IBM SPSS Modeler
  - MATLAB
  - Microsoft Azure Machine Learning
  - Oracle Data Mining



### 5.2.7 Support Vector Machines

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyse data used for classification and regression analysis.

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting).

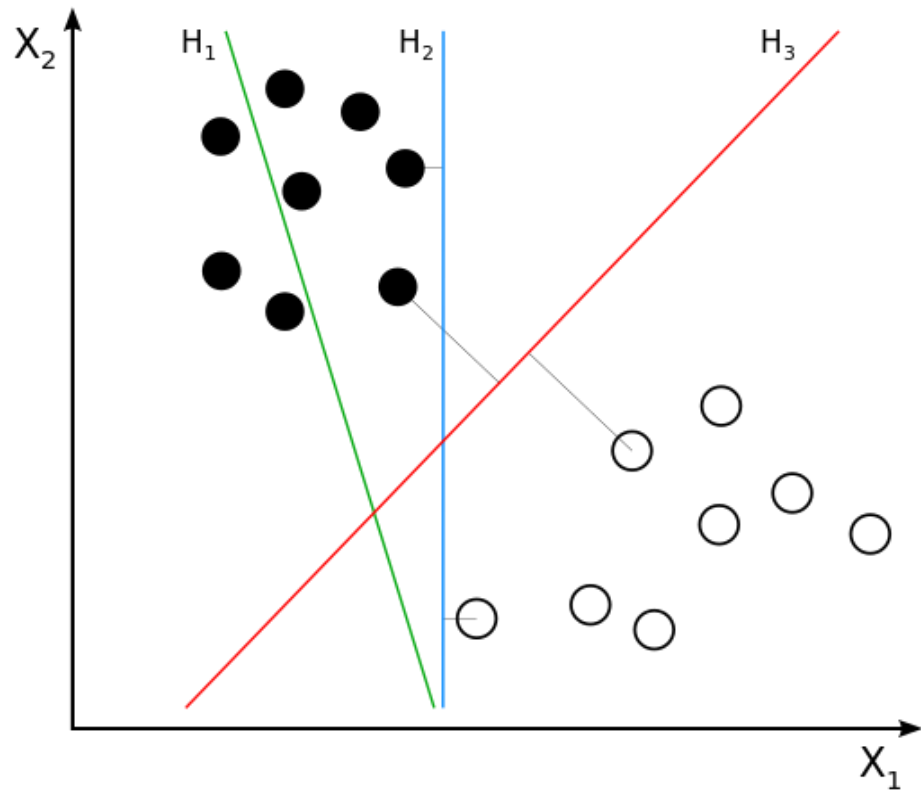
An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.

New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

When data are not labelled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups.

Classifying data is a common task in machine learning. Suppose some given data points each belong to one of two classes, and the goal is to decide which class a new Data point will be in. In the case of support vector machines, a data point is viewed as a  $p$ -dimensional vector (a list of  $p$  numbers), and we want to know whether we can separate such points with a  $(p-1)$ -dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes.

So, we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum margin classifier; or equivalently, the perceptron of optimal stability. H1 does not separate the classes. H2 does, but only with a small margin. H3 separates them with the maximum margin as seen in the Fig. 5.2.7.1.



**Fig. 5.2.7.1 Hyperplanes**

More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks like outlier's detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

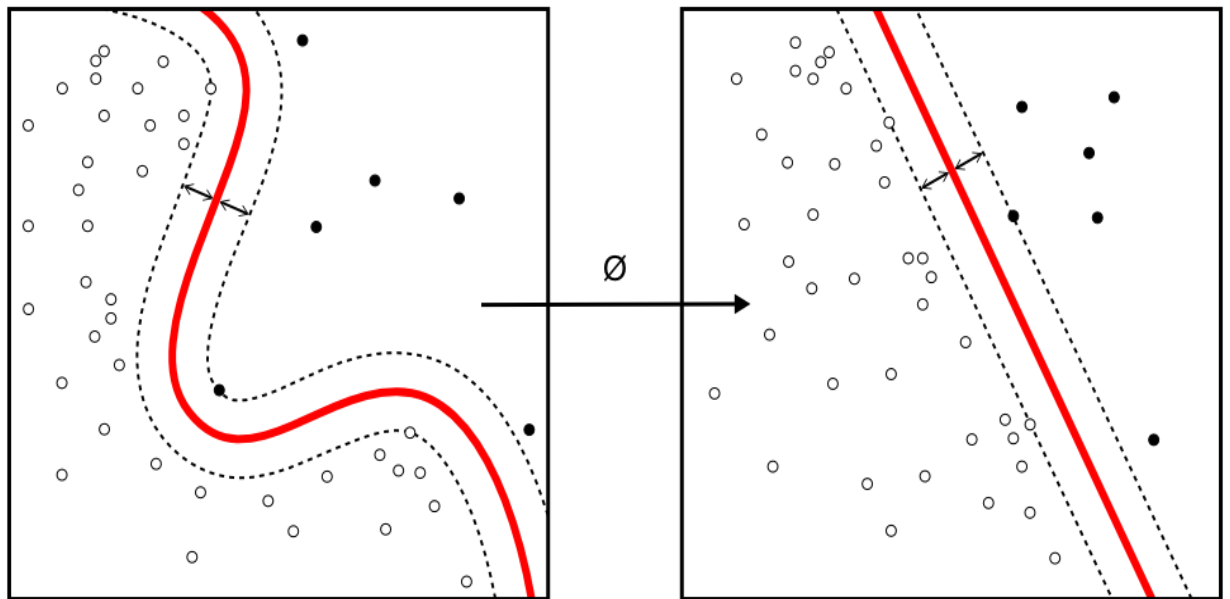
Whereas the original problem may be stated in a finite dimensional space, it often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, presumably making the separation easier in that space. To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function  $k(x,y)$  selected to suit the problem.

The hyperplanes in the higher-dimensional space are defined as the set of points whose dot product with a vector in that space is constant. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters  $a_i$  of images of feature vectors  $x_i$  that occur in the data base.

With this choice of a hyperplane, the points  $x$  in the feature space that are mapped into the hyperplane are defined by the relation:  $\sum_i a_i k(x_i, x)$ . Note that if  $k(x, y)$  becomes small as  $y$  grows further away from  $x$ , each term in the sum measures the degree of closeness of the test point  $x$  to the corresponding data base point  $x_i$ .

In this way, the sum of kernels above can be used to measure the relative nearness of each test point to the data points originating in one or the other of the sets to be discriminated. Note the fact that the set of points  $x$  mapped into any hyperplane can be quite convoluted as a result, allowing much more complex discrimination between sets which are not convex at all in the original space. SVM are of two types Linear and Non-linear, here we use linear since we a bipolar data set to classify.

The use of linear support vector machines due to its simplified way of creating hyperplanes and classifying bipolar data with a good accuracy. Also, there are several other support vector machines which are used for multi-level classification which have their importance in deep learning.



**Fig. 5.2.7.2 Kernel Machine**

### 5.2.7.1 Linear Support Vector Machines

We are given a training dataset of  $n$  points of the form

$$(\text{vec\_}x_1, y_1), \dots, (\text{vec\_}x_n, y_n)$$

where the  $y_i$  are either 1 or  $-1$ , each indicating the class to which the point  $\text{vec\_}x_i$  belongs. Each  $\text{vec\_}x_i$  is a  $p$ -dimensional real vector. We want to find the “maximum-margin hyperplane” that divides the group of points  $\text{vec\_}x_i$  for which  $y_i=1$  from the group of points for which  $y_i$ , which is defined so that the distance between the hyperplane and the nearest point  $\text{vec\_}x_i$  from either group is maximized.

Any hyperplane can be written as the set of points  $\text{vec\_}x$  satisfying

$$\text{vec\_}w \cdot \text{vec\_}x = b = 0,$$

where  $\text{vec\_}w$  is the (not necessarily normalized) normal vector to the hyperplane. This is much like Hesse normal form, except that  $\text{vec\_}w$  is not necessarily a unit vector. The parameter  $b / \|\text{vec\_}w\|$  determines the offset of the hyperplane from the origin along the normal vector  $\text{vec\_}w$ .

Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors as shown in the Fig. 5.2.7.3. Here  $\text{vec}$  indicates a vector symbol in equations.

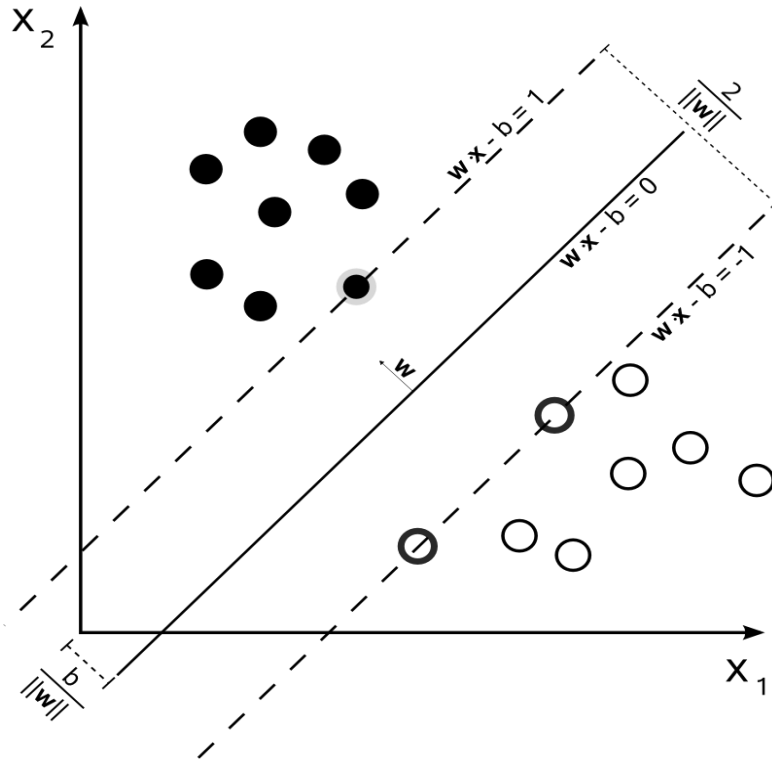


Fig. 5.2.7.3 Linear SVM

### 5.2.7.2 Implementation of SVM

The parameters of the maximum-margin hyperplane are derived by solving the optimization. There exist several specialized algorithms for quickly solving the QP problem that arises from SVMs, mostly relying on heuristics for breaking the problem down into smaller, more-manageable chunks.

Another approach is to use an interior point method that uses Newton-like iterations to find a solution of the Karush–Kuhn–Tucker conditions of the primal and dual problems. Instead of solving a sequence of broken down problems, this approach directly solves the problem altogether. To avoid solving a linear system involving the large kernel matrix, a low rank approximation to the matrix is often used in the kernel trick.

Another common method is Platt's sequential minimal optimization (SMO) algorithm, which breaks the problem down into 2-dimensional sub-problems that are solved analytically, eliminating the need for a numerical optimization algorithm and matrix storage. This algorithm is conceptually simple, easy to implement, generally faster, and has better scaling properties for difficult SVM problems.

The special case of linear support vector machines can be solved more efficiently by the same kind of algorithms used to optimize its close cousin, logistic regression; this class of algorithms includes sub-gradient descent (e.g., PEGASOS) and coordinate descent (e.g., LIBLINEAR). LIBLINEAR has some attractive training time properties. Each convergence iteration takes time linear in the time taken to read the train data and the iterations also have a Q-Linear Convergence property, making the algorithm extremely fast.

The general kernel SVMs can also be solved more efficiently using sub-gradient descent (e.g. P-packSVM]), especially when parallelization is allowed.

Kernel SVMs are available in many machine learning toolkits, including LIBSVM, MATLAB, SAS, SVMlight, kernlab, scikit-learn, Shogun, Weka, Shark, JKernelMachines, OpenCV and others.

## **6. Implementation**

In the implementation phase the software is built from ground up with the understanding acquired during the analysis and design phases.

### **6.1 Modules**

A module is block of code with unique functionality and here there are 6 main modules. They are:

- Fetch Tweets Module
- Pre-process Module
- Stemming Module
- Classifier Module
- Polarity Generation Module
- Visualizing Analytics Module

#### **6.1.1 Module Description**

##### **6.1.1.1 Fetch Tweets Module**

A query is made to retrieve tweets on specific topic using HTTP protocol using twitter search API which return with required tweets.

##### **6.1.1.2 Pre-process Module**

Pre-processing is way for making the raw data more understandable by the system for computation. It involves cleaning, editing, reduction, wrangling, etc.

##### **6.1.1.3 Stemming Module**

Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form generally a written word form. The stem need not be identical to the morphological root of the word.

##### **6.1.1.4 Classifier Module**

Classification is a general process related to categorization, the process in which ideas and objects are recognized, differentiated, and understood. A classifier system is an approach to accomplishing classification. Here Support vector machines are used for classification.

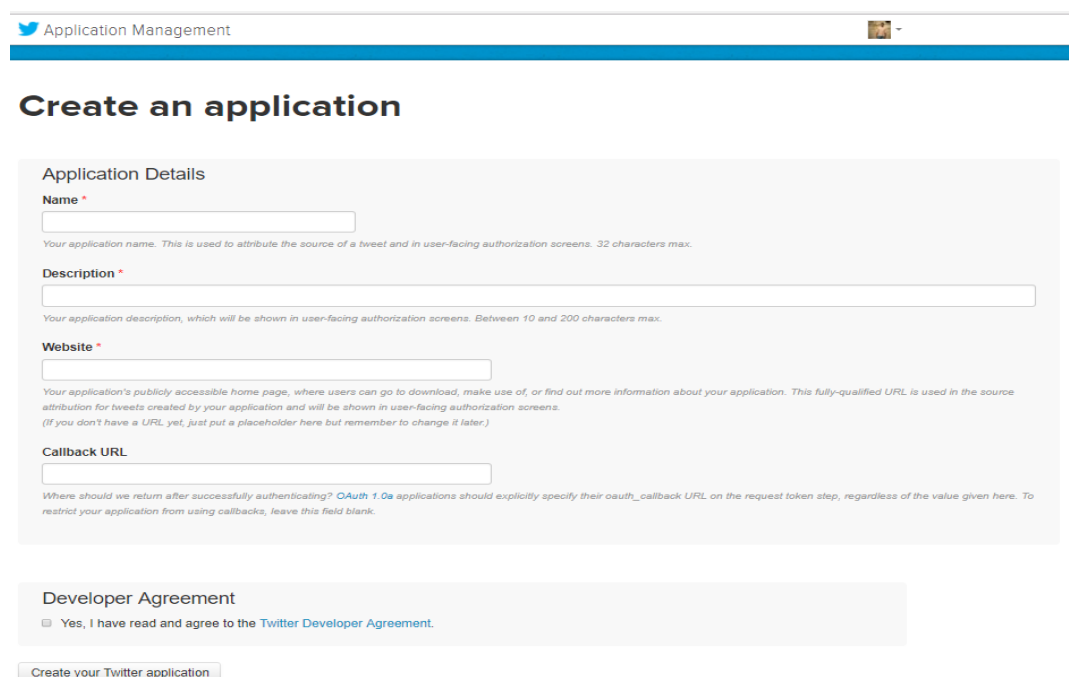
### 6.1.1.5 Polarity Generation Module

Polarity refers as sentiment in general. It tells the emotion of the statement i.e.: good, bad or neutral. Here the sentiment is usually good or bad which is bipolar, and text is classified as positive or negative.

### 6.1.1.6 Visualizing Analytics Module

The sentiment is displayed in a more readable and visual format in the form of graphs.

## 6.2 Sample Codes



The screenshot shows the 'Create an application' page in the Twitter Developer Console. At the top, there's a header with the Twitter logo and 'Application Management'. Below this is a blue bar with the text 'Create an application'. The main form is titled 'Application Details' and contains several fields: 'Name' (required, 32 characters max), 'Description' (required, 10-200 characters), 'Website' (required, fully-qualified URL), and 'Callback URL' (optional). Below the form is a 'Developer Agreement' section with a checkbox for 'Yes, I have read and agree to the Twitter Developer Agreement'. At the bottom is a button labeled 'Create your Twitter application'.

**Screenshot 6.2 Creating Twitter Developer Application**

Process of developing the Opinion system includes the following steps:

- First create an account in twitter.
- Then go to twitter developers console and create an application to use Twitter Search API Standard edition which retrieves tweets for past 7 days as shown in Fig. 6.2.1
- Once the application is created you will be provided with access tokens and consumer keys.
- Now Create a project in Spyder and add the code as in 6.2.1 to create access to twitter's tweets and retrieve them and store them in a list.

- Now the tweets are sent for pre-processing which cleans and edits the raw data using the code in 6.2.2.
- The processed tweets are then undergone stemming which is done in the code in 6.2.3.
- Before it goes through the sentiment generation, we need to create a classification module which implements support vector machines and classifies the pre-opinionated data and dumps it into a pickle file with the code in 6.2.4.
- Now after the trained data is classified, we send the pickle file along with the tweets for classification and polarity determination with the help of code in 6.2.5.
- Once the tweets are classified and polarity generated we need to display it in the form of a graph for easy understanding and in numeric. This is done using the code in 6.2.6.
- Once you execute the application you will be prompted to enter a topic to get overall opinion on it.
- The tweets of the topic are retrieved and processed and output is displayed in the form of graph and numeric. This is shown in the Fig. 5.2.2.
- The classifiers precision support and precision recall can also be noted when the classifier code as in 6.2.4 is executed. This is displayed as showed in Fig. 5.2.3.

### 6.2.1 Fetch Tweets Module

```
api = TwitterClient()
# calling function to get tweets
query = input("Enter the Topic for Opinion Mining: ")
tweets = api.get_tweets(classifier, query, count = 1000)

def get_tweets(self, classifier, query, count = 1000):
    """
    Main function to fetch tweets and parse them.
    """
    # empty list to store parsed tweets
    tweets = []
```



```

try:
    # call twitter api to fetch tweets
    fetched_tweets = self.api.search(q = query, count = count)

    # parsing tweets one by one
    for tweet in fetched_tweets:
        # empty dictionary to store required params of a tweet
        parsed_tweet = {}

        # saving text of tweet
        parsed_tweet['text'] = tweet.text
        # saving sentiment of tweet
        parsed_tweet['sentiment'] = self.predict(tweet.text, classifier)
        # appending parsed tweet to tweets list
        if tweet.retweet_count > 0:
            # if tweet has retweets, ensure that it is appended only
once
            if parsed_tweet not in tweets:
                tweets.append(parsed_tweet)
            else:
                tweets.append(parsed_tweet)

        # return parsed tweets
    return tweets

except tweepy.TweepError as e:
    # print error (if any)
    print("Error : " + str(e))

```

### 6.2.2 Pre-process Module

```

def preprocessTweets(self, tweet):

    #Convert www.* or https?://* to URL
    tweet = re.sub('((www\.[^\s]+)|(https?://[^\s]+))', 'URL', tweet)

```

```

#Convert @username to __HANDLE
tweet = re.sub('@[^\s]+','__HANDLE',tweet)
#Replace #word with word
tweet = re.sub(r'#([^\s]+)', r'\1', tweet)
#trim
tweet = tweet.strip('\n')
# Repeating words like happyyyyyyyyyy
rpt_regex = re.compile(r"(\1{1,})", re.IGNORECASE)
tweet = rpt_regex.sub(r"\1\1", tweet)

#Emoticons
emoticons = \
[
    ('__positive__',[':-)',' :)', '(:', '(-:', ':-D', ':D', 'X-D', 'XD', 'xD', '<3', ':\*',
';-)', ';)', ';-D', ';D', '(;', '(-;', ]),\
    ('__negative__',[':-(', ':(', '(:', '(-:', ':(', ':\(', ':"(', ':((', ]),\
]

def replace_parenth(arr):
    return [text.replace(')', '[]}\[]').replace('(', '[(\[]') for text in arr]

def regex_join(arr):
    return '(' + '|'.join( arr ) + ')'

emoticons_regex = [ (repl,
re.compile(regex_join(replace_parenth(regx))) ) for (repl, regx) in
emoticons ]

for (repl, regx) in emoticons_regex :
    tweet = re.sub(regx, ' '+repl+' ', tweet)

#Convert to lower case
tweet = tweet.lower()

return tweet

```

### 6.2.3 Stemming Module

#Stemming of Tweets

```
def stem(self,tweet):
    stemmer = nltk.stem.PorterStemmer()
    tweet_stem = ""
    words = [word if(word[0:2]!='__') else word.lower() \
              for word in tweet.split() \
              if len(word) >= 3]
    words = [stemmer.stem(w) for w in words]
    tweet_stem = ' '.join(words)
    return tweet_stem
```

### 6.2.4 Classifier Module

```
def classifier(X_train,y_train):

    vec = TfidfVectorizer(min_df=5, max_df=0.95, sublinear_tf =
True,use_idf = True,ngram_range=(1, 2))

    svm_clf =svm.LinearSVC(C=0.1)

    vec_clf = Pipeline([('vectorizer', vec), ('pac', svm_clf)])

    vec_clf.fit(X_train,y_train)

    joblib.dump(vec_clf, 'svmClassifier.pkl', compress=3)
    return vec_clf

# Main function

def main():

    X_train, X_test, y_train, y_test = getTrainingAndTestData()

    X_train, X_test = processTweets(X_train, X_test)

    vec_clf = classifier(X_train,y_train)

    y_pred = vec_clf.predict(X_test)
    print(sklearn.metrics.classification_report(y_test, y_pred))

if __name__ == "__main__":
    main()
```

### 6.2.5 Polarity Generation Module

```
def predict(self, tweet, classifier):

    #Utility function to classify sentiment of passed tweet

    tweet_processed = self.stem(self.preprocessTweets(tweet))

    if ( '__positive__' in (tweet_processed)):
        sentiment = 1
        return sentiment

    elif ( '__negative__' in (tweet_processed)):
        sentiment = 0
        return sentiment
    else:
        X = [tweet_processed]
        sentiment = classifier.predict(X)
        return (sentiment[0])
```

### 6.2.6 Visualizing Analytics Module

```
ntweets = [tweet for tweet in tweets if tweet['sentiment'] == 0]
neg=(100*len(ntweets)/len(tweets))
pos=(100-(100*len(ntweets)/len(tweets)))

# console output of sentiment
print("Opinion Mining on ",query)

# plotting graph
ax1 = plt.axes()
ax1.clear()
xar = []
yar = []
x = 0
y = 0
for tweet in tweets:
    x += 1
    if tweet['sentiment'] == 1 :
        y += 1
    elif tweet['sentiment'] == 0 :
        y -= 1
    xar.append(x)
    yar.append(y)
```

```

ax1.plot(xar,yar)
ax1.arrow(x, y, 0.5, 0.5, head_width=1.5, head_length=4, fc='k', ec='k')
plt.title('Graph')
plt.xlabel('Time')
plt.ylabel('Opinion')
plt.show()

# plotting piechart
labels = 'Positive Tweets', 'Negative Tweets'
sizes = [pos,neg]
# exploding Negative
explode = (0, 0.1)
fig1, ax2 = plt.subplots()
ax2.pie(sizes, explode=explode, labels=labels, autopct='%2.3f%%',
shadow=False, startangle=180)
# Equal aspect ratio ensures that pie is drawn as a circle.
ax2.axis('equal')
plt.title('Pie Chart')
plt.show()

# percentage of negative tweets
print("Negative tweets percentage: ",neg)
# percentage of positive tweets
print("Positive tweets percentage: ",pos)

now = datetime.datetime.now()
print ("Date and Time analysed: ",str(now))

```

Implementing the above sample code, the project has been developed into a fully functioning system.

## **7. Testing**

Testing your documents should be an integral part of your plain language writing process. It should not just be something you do after the final version to see if it worked. This is especially important if you're writing to hundreds, thousands or even millions of codes. The information gained in testing can save time in answering questions about your document later.

### **7.1 Software Testing**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs and whether software is fit for use.

### **7.2 Testing Methodologies**

#### **7.2.1 Verification**

It is also called static testing and involves reviews, walk throughs and inspections. It is often implicit as proofreading. Among the techniques for static analysis, mutation testing can be used to ensure the test cases will detect errors that are introduced by mutating the source code.

#### **7.2.2 Validation**

It is known as dynamic testing and is explicitly done since it is carried before complete code is written where only parts of code is tested. It involves execution of source code with given test cases. Together both verification and validation help improve software quality.

#### **7.2.3 White Box Testing**

. White Box Testing tests internal structures or workings of a program. In white-box testing, an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to train paths to determine appropriate outputs.

### **7.2.4 Black Box Testing**

Black-box testing treats the software as a “black box”, examining functionality without any knowledge of internal implementation, without seeing the source code.

## **7.3 Testing Levels**

### **7.3.1 Unit Testing**

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

### **7.3.2 Integration Testing**

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

## **7.4 Test Cases**

A test case is a condition which accepts inputs, processes it and gives expected output. A test case in software engineering normally consists of a unique identifier, requirement references from a design specification, preconditions, events, a series of steps (also known as actions) to follow, input, output and it validates one or more system requirements and generates a pass or fail.

The mechanism for determining whether a software program or system has passed or failed such a test is known as a test oracle. Test cases are often referred to as test scripts, particularly when written. Written test cases are usually collected into test suites. The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product.

Characteristics of a good test case:

- Accurate: Exacts the purpose.
- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.

- Repeatable: Can be used to perform the test over and over.
- Reusable: Can be reused if necessary.

The below Table 7.4 describes the various test cases done.

Test Case	Result
Verification	
Syntax and Walkthroughs	Pass
Validation	
UML Diagrams and Structures	Pass
Unit Testing	
Output Validation from each module	Pass
Exceptions control	Pass
Integration Testing	
Modules compatibility	Pass
Output Verification	Pass

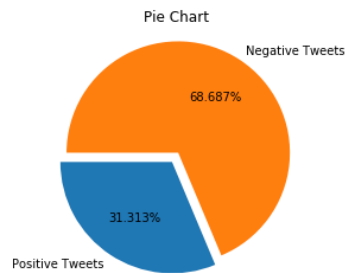
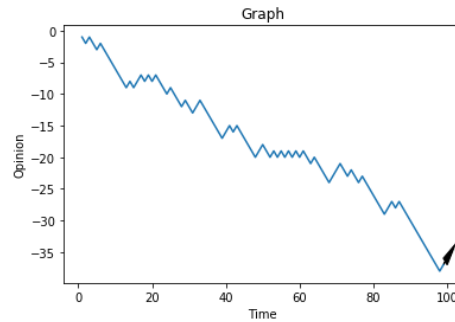
**Table 7.4 Test Cases**

## 7.5 Result and Analytics

The output of the system is the sentiment on various topics ranging from reviews on products to opinions about people. The system displays a graph which describes the opinions measured with time and a pie chart that shows the percentage of positive and negative tweets in overall number of tweets over a topic as shown in the Screenshot 7.5.1 and Screenshot 7.5.2. The screenshots tell the analytics over topics that are either trending topics that use hashtags or personal profile of an enterprise or a person.



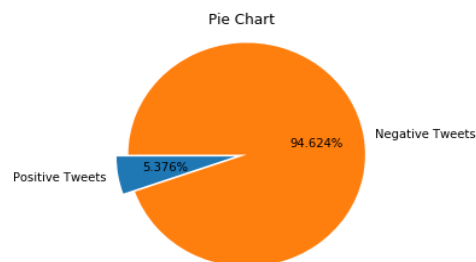
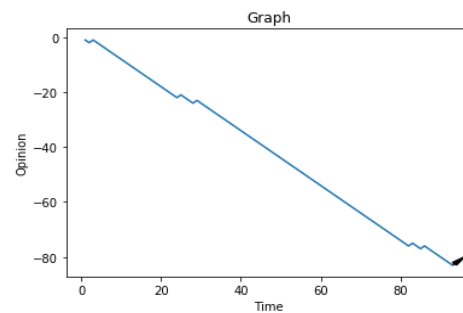
Enter the Topic for Opinion Mining: #JeepTime  
Opinion Mining on #JeepTime



Negative tweets percentage: 68.68686868686869  
Positive tweets percentage: 31.313131313131308  
Date and Time analysed: 2018-04-13

### Screenshot 7.5.1 Sentiment Analytics on Jeep Car

Enter the Topic for Opinion Mining: #Nifty  
Opinion Mining on #Nifty



Negative tweets percentage: 94.6236559139785  
Positive tweets percentage: 5.376344086021504  
Date and Time analysed: 2018-04-13

### Screenshot 7.5.2 Sentiment Analytics on Nifty Stock Exchange

The analytics that are acquired from the tweets are normalised and fitted into the classification of cluster vectors. The performance of the classification algorithm can be improved by using more accurate and well-defined training data sets. The precision of the sentiment can also be noted with the help of classification matrix which tells us the precision support over dimensions. This can be seen from the below Screenshot 7.5.3.

PROJECT/Code/SentimentAnalysis')					
	precision	recall	f1-score	support	
0	0.94	0.95	0.95	879	
1	0.96	0.95	0.95	1038	
avg / total	0.95	0.95	0.95	1917	

**Screenshot 7.5.3 Precision Support**

## 8. Advantages and Applications

The current system proposed has 3 major influencers, opinion mining, machine learning and SVM which is a part of supervised machine learning theory. Each of these individually have many advantages and also together they can give deeper insights in various fields.

### 8.1 Advantages and Applications of SVM

There are four main advantages:

- Firstly, it has a regularisation parameter, which makes the user think about avoiding over-fitting.
- Secondly it uses the kernel trick, so you can build in expert knowledge about the problem via engineering the kernel.
- Thirdly an SVM is defined by a convex optimisation problem (no local minima) for which there are efficient methods (e.g. SMO).
- Lastly, it approximates a bound on the test error rate, and there is a substantial body of theory behind it which suggests it should be a good idea.

SVMs can be used to solve various real-world problems:

- SVMs are helpful in text and hypertext categorization as their application can significantly reduce the need for labelled training instances in both the standard inductive and transductive settings.
- Classification of images can also be performed using SVMs. Experimental results show that SVMs achieve significantly higher search accuracy than traditional query refinement schemes after just three to four rounds of relevance feedback.
- This is also true of image segmentation systems, including those using a modified version SVM that uses the privileged approach as suggested by Vapnik. Hand-written characters can be recognized using SVM.
- The SVM algorithm has been widely applied in the biological and other sciences. They have been used to classify proteins with up to 90% of the compounds classified correctly.

- Permutation tests based on SVM weights have been suggested as a mechanism for interpretation of SVM models. Support vector machine weights have also been used to interpret SVM models in the past.
- Posthoc interpretation of support vector machine models to identify features used by the model to make predictions is a relatively new area of research with special significance in the biological sciences.

## **8.2 Disadvantages of SVM**

- The disadvantages are that the theory only really covers the determination of the parameters for a given value of the regularisation and kernel parameters and choice of kernel. In a way the SVM moves the problem of over-fitting from optimising the parameters to model selection. Sadly, kernel models can be quite sensitive to over-fitting the model selection criterion.

## **8.3 Advantages and Applications of Opinion Mining**

- Sentiment analysis has many applications and benefits to your business and organization. It can be used to give your business valuable insights into how people feel about your product brand or service.
- When applied to social media channels, it can be used to identify spikes in sentiment, thereby allowing you to identify potential product advocates or social media influencers.
- It can be used to identify when potential negative threads are emerging online regarding your business, thereby allowing you to be proactive in dealing with it more quickly.
- Opinion Mining can even be used for marketing and brand strategies to target the right audiences.
- It can be used to know about the public opinion about governmental policies.

## **8.4 Disadvantages of Opinion Mining**

- Sentiment analysis can be applied to many areas but arriving at whether a statement is positive or negative can be difficult. The categorization is mainly split into two types: facts and opinion.
- Facts are expressed about entities, whereas opinions are about their properties.
- Furthermore, opinions are completely subjective and describe people's sentiments, appraisals or general feeling towards entities and their properties.

- The human language can be complex for machine-based learning systems to interpret. For example, opinions can be expressed with sarcasm or irony, and the order of words can add even more confusion.

### **8.5 Advantages and Applications of Machine Learning**

- Machine Learning (ML) is expected to bring heavy changes to the world of technology.
- Machine learning is a subfield of artificial intelligence and computer science that allows software applications to be more accurate in predicting results.
- The prime objective of machine learning technology is to build algorithms that can get input data and leverage statistical analysis to predict an acceptable output value.
- Machine learning can build machines and enable them to work on their own. In fact, machine learning has already been in use for several years and marketers are gradually building some more digital strategies around machine learning available today.
- It is used in so many industries of applications such as banking and financial sector, healthcare, retail, publishing and social media, etc.
- It is used by Google and Facebook to push relevant advertisements based on users search history.
- It allows time cycle reduction and efficient utilization of resources.
- Due to machine learning there are tools available to provide continuous quality improvement in large and complex process environments.

Some popular Applications for machine learning include:

- Adaptive websites
- Bioinformatics
- Brain–machine interfaces
- Cheminformatics
- Classifying DNA sequences
- Computational anatomy
- Computer Networks
- Computer vision, including object recognition
- General game playing

- Internet fraud detection
- Linguistics
- Marketing
- Machine learning control
- Machine perception
- Medical diagnosis
- Economics
- Natural language processing
- Natural language understanding
- Optimization and metaheuristic
- Online advertising
- Recommender systems
- Robot locomotion
- Search engines
- Sentiment analysis (or opinion mining)
- Sequence mining
- Software engineering
- Speech and handwriting recognition
- Financial market analysis
- Structural health monitoring
- Syntactic pattern recognition
- Time series forecasting
- User behaviour analytics
- Translation

From the above we can see how all of these together can provide a way of giving deep predictive analytics for developing strategies in fields like marketing, brand management, administrative studies, academic researches, etc.

## **9. Conclusion**

We have proposed and built an application on data science for Opinion Mining on Twitter data using Machine Learning. This application mainly helps us to judge or know about various topics form products and services to people and companies to administration and governments within few seconds. Here instead of spending your valuable time reading whole text either tweets data content and analyse about it we can quickly conclude ourselves by looking at the graph. It is very advantageous especially for business people.

Sentiment analysis of the user generated data is very useful in knowing the opinion of the crowd. Despite all the challenges and potential problems that threatens Sentiment analysis, one cannot ignore the value that it adds to the industry. Because Sentiment analysis bases its results on factors that are so inherently humane, it is bound to become one the major drivers of many business decisions in future. Improved accuracy and consistency in text mining techniques can help overcome some current problems faced in Sentiment analysis. Looking ahead, what we can see is a true social democracy that will be created using Sentiment analysis, where we can harness the wisdom of the crowd rather than a select few experts. A democracy where every opinion counts, and every sentiment affects decision making.

## 10. Future Enhancements

Sentiment analysis methods till now have been used to detect the polarity in the thoughts and opinions of all the users that access social media. Researchers and Businesses are very interested to understand the thoughts of people and how they respond to everything happening around them. Companies use this to evaluate their advertisement campaigns and to improve their products. There is too much potential in machine learning, overtaking some of the manual labour of some lexicon-based tasks that are labour intensive. For example, lexicon sentiment creation is labour intensive and there are already unsupervised methods to create them. This is where machine learning will play a crucial role. Such algorithms will also have to understand and analyse natural text concept-wise and context-wise. Time will also be a crucial element looking at the amount of data that is being generated on the Web today. Collecting opinions on the web will still requires processing that can filter out unopinionated user-generated content and also to test the trustworthiness of the opinion and its source.

There is a lot of scope in analysing the video and images on the web. Now days, with the advent of Facebook, Instagram and Video vines people are expressing their thoughts with pictures and videos along with text. Sentiment analysis will have to pace up with this change. Tools which are helping companies to change strategies based on Facebook and Twitter will also have to accommodate the number of likes and re-tweets that the thought is generating on the Social media. People follow and unfollow people and comments on Social Media but never comment so there is scope in analysing these aspects of the Web as well. The use of punctuation is an obstacle in Sentiment Analysis which is under research as well. Sentiment Analysis has started helping us to predict events just like in the case of Obama vs Romney but is still naïve in most cases. A sentiment analysis tool Televiewer had predicted the winner of the show X factor but eventually that person came second.

So, improvements on the analysis is one scope which is under way by many tools available on the web. As new text types appear on the Social Web, the techniques to pre-process, as well as to tackle their informal style must be adapted, so as to obtain acceptable levels of performance of the sentiment analysis systems. The field will have to combine with effective computing, psychology and neuroscience to converge on a unified approach to understanding the sentiments better.



## 11. References

### **Textbooks:**

1. The Unified Modeling Language User Guide by Grady Booch, James Rumbaugh, Ivar Jacobson – Pearson
2. Sentiment Analysis: Mining Opinions, Sentiment and Emotions by Bing Liu- Cambridge University Press
3. Python Data Science Essentials by Alberto Boschetti and Luca Massaron- Packt

### **Research Journals and Publications:**

- [1] Sentiment Analysis and Text Summarization of Online Reviews: A Survey  
Pankaj Gupta, Ritu Tiwari, Nirmal Robert- ICCSP
- [2] Automated sentiment analysis system using machine learning algorithms by I. Hemalatha, Dr. G.P.S. Varma, Dr. A. Govardhan- IJRCCT
- [3] Sentiment analysis of twitter data using machine learning approaches and semantic analysis by Geetika Gautam, Divakar Yadav- IEEE