Emotion Detection Using Twitter Datasets Using Spacy Algorithm

A project Report

Submitted in the partial fulfillment of the requirements for the award of

The degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

By

SK. MAHAMMAD BIN ALI	**208A1A05A7**
K. HARSHITHA	**208A1A0575**
G. VENKATA SAI NIKITHA	**208A1A0568**
M. SUMA MADHURI	**208A1A0580**

Under the Esteemed Guidance of

Mr. R. V. SUBBAIAH M.Tech

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS

Valluru - 523272

(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi)

Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015

2023-2024

RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS

Valluru - 523272

(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi)

Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(2020-2024)

CERTIFICATE

This is to certify that this project work entitled **"Emotion Detection Using Twitter Datasets Using Spacy Algorithm"** is being submitted by **SK. Mahammad Bin Ali (208A1A05A7), K. Harshitha (208A1A0575), G. Venkata Sai Nikitha (208A1A0568), M. Suma Madhuri (208A1A0580)** in partial fulfillment of the requirements for degree of Bachelor of Technology in COMPUTER SCIENCE & ENGINEERING from JNTUK, Kakinada during the period of 2023-24 is a record of bonafide work carried out by them under our esteemed guidance and supervision.

PROJECT GUIDE	**HEAD OF THE DEPARTMENT**
Mr. R. V. Subbaiah	**Dr. K. Narayana Rao**
Associate Professor	**Professor & HOD**

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the work is being presented in this dissertation entitled **"Emotion Detection Using Twitter Datasets Using Spacy Algorithm"** submitted towards the partial fulfillment of requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering, work carried out under the supervision of **Mr. R. V. Subbaiah, Associate Professor**, **RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS**, **Valluru, Ongole.** The results embodied in this dissertation report have not been submitted to any other University for the award of any other degree. Furthermore, the technical details furnished in various chapters of this report are purely relevant to the above project and there is no deviation from the theoretical point of view for design, development and implementation.

SK. MAHAMMAD BIN ALI	**208A1A05A7**
K. HARSHITHA	**208A1A0575**
G. VENKATA SAI NIKITHA	**208A1A0568**
M. SUMA MADHURI	**208A1A0580**

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our guide **Mr. R. V. Subbaiah ,** Associate Professor, for providing his valuable guidance, comments, suggestions and support throughout the course of the project.

We express our heartful thanks to **Dr. K. Narayana Rao ,** Head of the Department, Computer Science and Engineering, for his valuable support to bring out this project in time.

We pay our profound sense of gratitude to our Principal **Prof. Dr A.V. Bhaskara Rao** for providing an excellent environment in our college and helping us at all points for achieving our task.

We would like to express our sincere thanks to the **Management of RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS, Valluru** for providing a good environment and infrastructure.

Finally, We thank all our faculty members, supporting staff of CSE Department and friends for their kind co-operation and valuable help for the completion of the project.

Project Associates

SK. MAHAMMAD BIN ALI	**208A1A05A7**
K. HARSHITHA	**208A1A0575**
G. VENKATA SAI NIKITHA	**208A1A0568**
M. SUMA MADHURI	**208A1A0580**

RISE KRISHNA SAI PRAKSAM GROUP OF INSTITUTIONS
Valluru - 523272
(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi)
Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision of the Institute	To be a premier institution in technical education by creating professionals of global standards with ethics and social responsibility for the development of the nation and the mankind.
Mission of the Institute	Impart Outcome Based Education through well qualified and dedicated faculty.
	Provide state-of-the-art infrastructure and facilities for application oriented research.
	Reinforce technical skills with life skills and entrepreneurship skills.
	Promote cutting-edge technologies to produce industry- ready professionals.
	Facilitate interaction with all stakeholders to foster ideas and innovation.
	Inculcate moral values, professional ethics and social responsibility
Vision of the Department	To be a center of excellence in computer science and engineering for value based education to serve humanity and contribute for socio-economic development.
Mission of the Department	Provide professional knowledge by student centric teaching-learn in process to contribute to software industry.
	Inculcate training on cutting edge technologies for industry needs.
	Create an academic ambience leading to research.
	Promote industry institute interaction for real time problem solving.

Program Educational Objectives (PEO's)

Graduates of the Program will be able to

PEO1	Develop software solutions for real world problems by applying Mathematics, Science and engineering principles.
PEO2	Function as members of multi-disciplinary teams and to communicate effectively using modern tools.
PEO3	Pursue career in software industry or higher studies with continuous learning and apply software knowledge.
PEO4	Practice the profession with ethics, integrity, leadership and social responsibility.

Program Outcomes (PO's)

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/Development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct Investigations of Complex Problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern Tool Usage:** Create, Select and apply appropriate techniques, resources,and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	**The Engineer and Society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent Responsibilities relevant to the professional engineering practice.
PO7	**Environmental and Sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	**Individual and Team Work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project Management and Finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSO's):

A student of the Computer Science and Engineering Program will be able to

PSO1	**Domain Knowledge:** Apply the Knowledge of Programming Languages, Networks and Databases for design and development of Software Applications.
PSO2	**Computing Paradigms:** Understand the evolutionary changes in computing possess knowledge of context aware responsibility of paradigms and meet the challenges of the future.

RISE KRISHNA SAI PRAKSAM GROUP OF INSTITUTIONS

Valluru - 523272

(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi)

Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Name of the Course	: Project - II	Year & Semester	: IV Year II
Academic Year	: 2023-2024	Regulation	: R20

Course Outcomes

Course Outcomes (CO's)		Taxonomy Level
C424.1	Develop application for community needs	Applying
C424.2	Extend skills for analysis and synthesis of Practical systems	Understanding
C424.3	Get hold of the use of new tools efficiently and creatively.	Analyzing
C424.4	Work in team to carry out analysis and cost-effective,environmentally friendly designs of engineering systems.	Analyzing
C424.5	Write Technical/Project reports and oral presentation of the work done to an audience.	Applying
C424.6	Demonstrate a product developed.	Understanding

Project Coordinator

HEAD OF THE DEPARTMENT

CO Vs PO Mapping

Course Outcomes (CO's)	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
C424.1	2	2	3	3	3	3	2	2	3	2	2	2
C424.2	2	2	3	3	3	3	2	2	3	3	3	3
C424.3	2	2	3	3	3	3	2	2	3	2	2	2
C424.4	2	2	3	3	3	3	3	3	3	3	3	3
C424.5	2	2	3	2	3	3	2	3	3	3	3	3
C424.6	2	2	2	2	3	3	3	3	3	3	3	3
C424	**2.00**	**2.00**	**2.83**	**2.67**	**3.00**	**3.00**	**2.33**	**2.50**	**3.00**	**2.67**	**2.67**	**2.67**

CO Vs PSO Mapping

Course out Comes (Co's)	PSO 1	PSO 2
C424.1	3	3
C424.2	3	3
C424.3	3	3
C424.4	3	3
C424.5	2	2
C424.6	3	3
	2.83	2.83

1: Low 2: Medium 3: High

Guide Signature

Emotion Detection Using Twitter Datasets Using Spacy Algorithm

INDEX

LIST OF FIGURES

ABSTRACT

People show emotions for everyday communication. Emotions are identified by facial expressions, behavior, writing, speaking, gestures and physical actions. Emotion plays a vital role in the interaction between two people. The detection of emotions through text is a challenge for researchers. Emotion detection from the text can be useful for real-world application. Automatic emotion detection in the original text aims to recognize emotions in any digital medium by using natural language processing techniques and different approaches. Enabling machines with the ability to recognize emotions in a particular kind of text such as twitter's tweet has important applications in sentiment analysis and affective computing. We have worked on the newly published gold data set (AIT-2018) and propose a model consisting of lexical based using Word Net-Affect and EmoSentic Net with supervised classifiers for detecting emotions in a tweet text.

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

Language is known to be a powerful instrument for communicating and conveying information and for expressing emotions. Currently, neuroscience, psychology, cognitive sciences, computer sciences, and computational sciences are studying emotional identification widely. The integration into our everyday life of several interactive online diaries, journals, and individual blogs helps meet important social-interaction needs .

In the world of today's social networks, users share their opinions and emotions in their way through different medium like Twitter, Instagram, Facebook, and many more. Where millions, in their everyday lives express their views and opinions and also their emotions on or about a particular thing through social networks . This gave the researchers an excellent opportunity to analyze the emotions of social networking users' activities. These large numbers of data, generated by social networks contain feelings, opinions, and emotions of people from day to day. Different emotional analytical research on the social platform has been underway over the years. As the public have different thoughts, it becomes a challenge to analyze the correct emotion from social data. This makes it clear about the need to work on these problems and it offers many possibilities for future research into the hidden identification of emotions of users in general or emotions of users on a specific topic, etc.

Here we will study and analyze previous works done in this area, identify research scope, understand the process, methods used and finally propose a model that will help us to detect an emotion which is expressed in tweets. We will work on AIT-2018 data set and our proposed methodology consists of different phases, a basic idea of the whole model is shown in figure 1 and detailed worked is described in the following sections.

CHAPTER 2
LITERATURE SURVEY

2 LITERATURE SURVEY

2.1 A Survey On Emotion Detection Techniques using Text in Blogposts :

Emotion can be expressed in many ways that can be seen such as facial expression and gestures, speech and by written text. Emotion Detection in text documents is essentially a content - based classification problem involving concepts from the domains of Natural Language Processing as well as Machine Learning. In this paper emotion recognition based on textual data and the techniques used in emotion detection are discussed.

2.2 The Impact of Social Media on Intercultural Adaptation :

Social media has become increasingly popular components of our everyday life in today's globalizing society. It provides a context where people across the world can communicate, exchange messages, share knowledge, and interact with each other regardless of the distance that separates them. Intercultural adaptation involves the process of promoting understanding through interaction to increase the level of fitness so that the demands of a new cultural environment can be met. Research shows that people tend to use social media to become more integrated into the host culture during their adaptation and to maintain connections to their home countries. This paper attempts to investigate the impact of using social media on the intercultural adaptation process. In-depth interviews of international students of a U.S. university are conducted. Based on the results of the analysis, directions for future studies in this line of research are also discussed.

2.3 Use of Word Clustering to Improve Emotion Recognition from Short Tex:

Emotion recognition is an important component of affective computing, and is significant in the implementation of natural and friendly human-computer interaction. An effective approach to recognizing emotion from text is based on a machine learning technique, which deals with emotion recognition as a classification problem. However, in emotion recognition, the texts involved are usually very short, leaving a very large, sparse feature space, which decreases the performance of emotion

classification. This paper proposes to resolve the problem of feature sparseness, and largely improve the emotion recognition performance from short texts by doing the following: representing short texts with word cluster features, offering a novel word clustering algorithm, and using a new feature weighting scheme. Emotion classification experiments were performed with different features and weighting schemes on a publicly available dataset. The experimental results suggest that the word cluster features and the proposed weighting scheme can partly resolve problems with feature sparseness and emotion recognition performance. © 2016. The Korean Institute of Information Scientists and Engineers.

2.4 Multiclass Emotion Extraction from Sentences :

This paper aims to investigate the extraction of different classes of emotion from sentences using supervised machine learning technique, Multinomial Naïve Bayes (MNB). Here a bag of word approach is used to capture the emotions. The unigrams are mainly used for this and the bigrams and trigrams are used to capture lower order dependencies. The work is done on the ISEAR dataset. The experiments with different feature sets selected using Weighted log-likelihood score (WLLS) shows that the MNB classifier provides good results when the unigram feature set size is 450 which provides an average accuracy of 76.96% across all emotion classes.

2.5 A Hybrid Model for Automatic Emotion Recognition in Suicide Notes :

We describe the Open University team's submission to the 2011 i2b2/VA/Cincinnati Medical Natural Language Processing Challenge, Track 2 Shared Task for sentiment analysis in suicide notes. This Shared Task focused on the development of automatic systems that identify, at the sentence level, affective text of 15 specific emotions from suicide notes. We propose a hybrid model that incorporates a number of natural language processing techniques, including lexicon-based keyword spotting, CRF-based emotion cue identification, and machine learning-based emotion classification. The results generated by different techniques are integrated using different vote-based merging strategies. The automated system performed well against the manually-annotated gold standard, and achieved encouraging results with a micro-averaged F-measure score of 61.39% in textual emotion recognition, which was ranked 1st place out of 24 participant teams in this challenge. The results demonstrate that effective emotion recognition by an automated system is possible when a large annotated corpus is available.

2.6 Computational approaches for emotion detection in text :

Emotions are part and parcel of human life and among other things, highly influence decision making. Computers have been used for decision making for quite some time now but have traditionally relied on factual information. Recently, interest has been growing among researchers to find ways of detecting subjective information used in blogs and other online social media. This paper presents emotion theories that provide a basis for emotion models. It shows how these models have been used by discussing computational approaches to emotion detection. We propose a hybrid based architecture for emotion detection. The SVM algorithm is used for validating the proposed architecture and achieves a prediction accuracy of 96.43% on web blog data.

CHAPTER 3
SYSTEM ANALYSIS

3 SYSTEM ANALYSIS

3.1 Existing System

There has been many works in this area for the last couple of years. In this section we are going to view some of the previous works done by different authors. In the authors created a corpus of Twitter tweets and used corpus annotation study to prepare an annotated corpus. Multi-class SVM kernels were used for learning model. For features selection Unigrams, Bigrams, Personal, pronouns, adjectives, Word-net Affect emotion lexicon, Word-net Affect emotion and Dependency–parsing features. In the authors first fetched the tweets from Twitter to create a dataset. Then they obtain target based extended features model. They trained four different supervised classifiers, Naïve Bayes (NB), Support Vector Machine (SVM), Maximum Entropy (MaxEn), Artificial Neural Networks (ANN). SVM combined with Principal Component Analysis (PCA) obtains the maximum accuracy. In at first, the authors preprocessed the training dataset and took similarity measurements amongst the data. Then using semantic similarity all the emotion labeled corpus are clustered. In the training phase, the authors represented each text as a feature vector and the SVM learning algorithm is applied to train an emotion classifier. In the authors focus on identifying seven different classes of emotions - Anger, Disgust, Fear, Guilt, Joy, Sadness, Shame. To extract features, preprocessed data is tokenized and then stemmed using porter stemming algorithm. Authors used Unigram, Bigram and Trigram feature. WLLS (Weighted Log-likelihood Score) scheme is applied to score n-grams with respect to each emotion resulting in a feature vector table. In the method, authors used MNB (Multinomial Naïve Bayes) as a classifier which is trained by the top scored n-grams and accuracy tested with different feature sets. In the author showed a hybrid model for emotion detection. In this model, it contains lexicon keyword spotting, CRF based emotion detection using NB, MaxEn, and SVM. In the authors have used a Hidden Markov Model which determines the emotion of the text. They considered each sentence contains many sub ideas and each idea is considered as an event that might cause a transition of a state. In the author created an automatic emotion detection system which can identify emotions in tweets streams. His approach included two-part, training an offline emotion classifier model which is based on his work from and in the second part he performed a two-step classification to identify tweets containing emotions and classifying these tweets into a more fine-grained category using soft classification techniques. In the authors tried to classify comments regarding a specific crisis on social media. They used the emotion of anger considering the fact that this same technique can be applied for other emotions as well. They performed

a short survey collecting 1192 responses in which the people are requested to comment under a news headline using social media. Using this as training set they obtain an accuracy of 90% in classifying anger in their dataset. They used logistic regression coefficients to select their features and random forest as their main classifier.

3.1.1 EMOTION DETECTION FROM TWEET :

For emotion detection, there are four kinds of text-based techniques as follows keyword spotting method, lexical affinity method, learning based method and hybrid methods. For detecting emotions from tweets, we have used the lexical affinity method combined with learning-based methods to automatically classify multi-class emotions from our dataset. We have used WordNet-Affect and EmoSenticNet.

emotion lexicon to extract the emotion containing words as features from the tweet separately. WordNet-Affect returns the emotion representing words from the tweets which is then considered as features but in most cases, it is unable to take the words which may not be an emotion word but do represent an emotion. For a small set of words WordNetAffect can determine if the word represents one of the six basic emotions. The main drawback of WordNet-Affect is that it can not give an intensity for the words as some words though they are a synonym of each other may represent different type of emotion with respect to text. On the other hand EmoSenticNet is an extension of WordNet-Affect which also apply the SenticNet rules. It then also finds the features which are not contained in WordNet-Affect.

Then we have used term frequency and inverse document frequency on the features to give the emotion features a better score, after that, we have used some different supervised algorithm for emotion classification. We have used Naïve Bayesian, Decision Tree and Support Vector Machine for emotion classification, all of these are supervised machine learning algorithm.

We have tried to experiment ourselves with detecting the emotion from a text document. Below we are presenting our proposed methodology in figure 1. The following sections describe each process in details.

3.1.2 DATASET

For our dataset we have taken SemEval-2018 Affect in Tweets Distant Supervision Corpus (AIT-2018 Dataset). Using twitter API these tweets are crawled from twitter from tweets that included emotion-related words such as '#angry', '#annoyed', '#panic', '#happy', '#love', '#surprised', etc. To

create a dataset of tweets rich in a particular emotion, they have used the following methodology. For each emotion X, they selected 50 to 100 terms that were associated with that emotion at different intensity levels. For example, the angry dataset used these terms as follows mad, frustrated, annoyed, peeved, irritated, miffed, fury, antagonism, and so on. This dataset consists of 4 emotion class anger, fear, joy and sadness, they have represented anger and disgust as anger and happiness and sadness as joy.

The dataset of the task was divided into 3 languages as follows English, Arabic and Spanish. In each language there are 5 sub-task datasets. We only work with EI-oc subtask dataset. In which for each tweet there is an emotion alongside the corresponding intensity of that tweet. An initial distribution of the dataset's task EI-oc can be found in figure 2.

3.2 Proposed System

Raw tweets scraped from twitter usually results in a noisy dataset with a lot of useless value. This is due to the nature of the user's usage of Twitter in their own way. Tweets have certain exceptional characteristics such as website URL, short form words, retweets, emoticons, person mentions, etc. which have to be suitably extracted. Therefore, raw twitter data has to be pre-processed to create a dataset which will be easy for different classifiers to generate good results.Using this as training set they obtain an accuracy of 95% in classifying anger in their dataset We have utilized a great variety of pre-processing steps to standardize the dataset and reduce its size. We do the pre-processing on tweets which are as follows.

- Removing tweets that are not in English.
- Converting the tweet to lower case.
- Removing URL from the tweet.
- Removal of mentions, retweet mentions, and unnecessary numbers.
- Separation of hashtags as they can play a vital role in emotion analysis.
- Changing short form words to their full form. E.g. between stands for by the way.
- Changing the emoticons with their meaning. E.g. (":D") stands for laugh/joy with respect to table I .
- Word tokenizing
- Striping punctuations ['"?!,.():;] from the tokenized words.

• Stop words removal from the tokenized words.

• Stemming and lemmatizing the tokenized words.

• In the end, making parts of speech tag for those tokenized words.

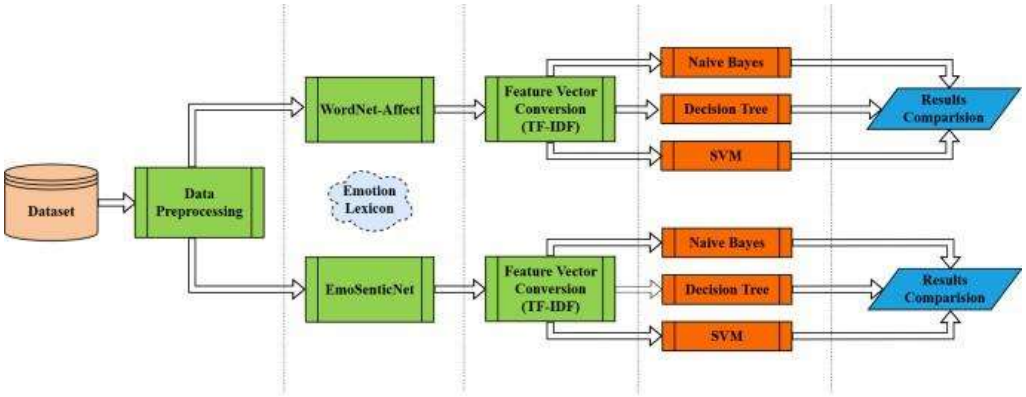


Fig. 1. Proposed Methodology

Fig. 2. Visualization of the initial number of tweets in the dataset

TABLE I
EMOJI TO EMOTION KEYWORDS

Emoji	Emotion Keywords
:), :), :-), (:, (:, (-:, :")	Smile
:D, : D, :-D, xD, x-D, XD, X-D	Laugh
<3, :*	Love
;-), ;), ;-D, ;D, (;, (-;	Affection
:-(, : (, :(,):,)-:	Sad
:'(, :'(, :"(	Cry

Fig 3

TABLE II
DATA PREPROCESSING OF TWITTER TWEET

Comment	Tweet
Real Tweet	@SatisfyingTaste @TheAnimalVines I used to make the peanut butter energy balls all the time. My famjam loved them! Btw my cats keep loving them as well :D :D #recipes #yummy 100 âðŸ https://bit.ly/2orTQLP
Preprocessed Tweet	i used to make the peanut butter energy balls all the time my famjam loved them by the way my cats keep loving them as well recipes yummy joy joy

Fig 4

After this, we have removed the stop words, stemmed using the Porter stemmer which works well for tweet and lemmatized the words and finally POS tagged the words. We can see it's quiet challenging for us to pre-process the tweet as Twitter itself has its own native way of representation. The most difficult part was to detect the misspelled words and finding the full form of short forms like ASAP is "as soon as possible". We have used a slang dictionary provided by and using SymSpell, we have corrected the misspelled words and compound words.

3.2.1 WORDNET-AFFECT AND EMOSENTICNET

After the data is preprocessed we have used WordNetAffect which is a subset of WordNet with only emotion words. We have mapped those words with our tweet's words and retrieve the emotion words only (words which represent the emotion of any kind). Then we took those emotions words as features.

EmoSenticNet is another lexical resource that assigns six WordNet Affect emotion labels to SenticNet concept. It can also be thought of as an expansion of WordNet Affect emotion labels to a larger vocabulary. We created a list of emotions words with syntactic relations. After that, we took these as features also.

As the dataset is a labeled one, each tweet has its own emotion with it. But as we are focusing on emotion words in tweets, we filtered out the emotion words from each tweet and stored them into a Pandas DataFrame, which was later used in training and testing the model. Here it should be noticed that we are assuming each tweet has only one emotion words. One example is given below:

Tweet: "Today I'm feeling loved."

Emotion: "joy"

So we filtered out the emotion word from this tweet.

Emotion word: loved

And store it in the DataFrame with the emotion, Joy. Such as

'loved' => 'Joy'.

The whole process of filtering the tweet with respect WordNet-Affect and EmoSenticNet is shown in Algorithm 1.

3.2.2 FEATURE VECTOR CONVERSION

We represent each tweet into a vector of features for the training of a classifier from labeled data. We must capture features describing each tweet's emotion. Selection of features plays an important role in the classification process ' effectiveness.

ess. We employed two well-known techniques to create the feature vector from the 2 features set. These are term frequency (TF) and term frequency and inverse document frequency (TFIDF).

Term Frequency, which measures how frequently a term occurs in a document. Inverse Document Frequency, which measures how important a term is.

$$T F - IDF(T, d, D) = T F(T, d) \times IDF(T, D)$$

Here T is number of terms appeared in a document, d is total number of term in each document, D is the total number of documents.

As after the feature selections we have only the features which bear emotions towards the classification, we have used TF and TF-IDF to increase the importance of those features.

3.2.3 SUPERVISED CLASSIFIER

We have used Naïve Bayesian, Decision Tree and Support Vector Machine in our model.

Bayesian Classifiers build a probabilistic model based on the word features in different classes. For our multi-class emotion classification problem we have taken Multinomial Naïve Bayes Classifier. We have trained our model on MultinomialNB provided by the sk-learn package. In Naive Bayes, texts are classified based on posterior probabilities generated based on the presence of different classes of

words in texts. This assumptions makes the computations resources needed for a naïve bayes classifier is far more efficient than non-naïve bayes approaches which is exponential complexity. Naïve Bayes has been widely used for classifying text because it is simple and fast.

hey accept high dimensional feature spaces and sparse feature vectors. Also, text classification using SVMs is very robust to outliers and does not require any parameter tuning. It finds a maximum margin separating hyper plane between two classes of data. For multi-class classification SVM maximize the margin for one vs all classes of data. For our classification problem we have used Linear SVM of sk-learn package. It has been shown that linear kernels based SVM performs a lot better on non-linear SVM in terms of text classification. Decision trees are slow and sometimes suffer from overfitting. However, its accuracy competes with well-known text classification algorithms such as SVM. For our model we took sk-learn DicisionTreeClassifier. We defined the criterion as entropy as we want the function to measure the quality of a split in text for the information gain. Information gain measures how much organized the input features became after we divide them up using a given feature. Also we have given a static random_state value for our classifier. As for text classification a decision tree takes the features as input values. The decision nodes checks the feature values and leaf nodes which assign one of the classes from our multi class emotion. To choose a class for our input text the model start with the initial node as root node which contains a condition on the input features, it than selects a branch based on that feature which leads to a new condition and it makes a new decision based on it. The flow continues until it arrives at a leaf node which provides an emotion class for the input value.

CHAPTER 4
FEASIBILITY STUDY

4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

4.1 ECONOMICAL 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. The expenditures must be justified. 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.

4.2 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. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

4.3 SOCIAL 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. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER 5
SYSTEM REQUIREMENTS

5 SYSTEM REQUIREMENTS

5.1 Hardware Requirements :

- System : Lenovo 14s Intel® Core™ i7 2.30 GHz
- Hard Disk : 40 GB.
- Monitor : 15 VGA(Video Graphics Array) Colour.
- Mouse : Lenovo.
- Ram : 512 MB.

5.2 Software Requirements :

- Operating system : Windows 8 Professional.
- Coding Language : python

CHAPTER 6
SYSTEM ENVIRONMENT

6 SOFTWARE ENVIRONMENT

What is Python :-

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python :-

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it *contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more.* So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities** to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

When working with Java, you may have to create a class to print **'Hello World'**. But in Python, just a print statement will do. It is also quite **easy to learn, understand,** and **code.** This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory.** This further aids the readability of the code.

8. Object-Oriented

This language supports both the **procedural and object-oriented** programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

9. Free and Open-Source

Like we said earlier, Python is **freely available.** But not only can you **download Python** for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to **code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages :

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and **machine learning**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in **slow execution**. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **client-side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don't need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can **raise run-time errors**.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

History of Python : -

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python.Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners[1], Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it."Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

What is Machine Learning : -

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of *building models of data*.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these

models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain.Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Learning :-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into *classification* tasks and *regression* tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as *clustering* and *dimensionality reduction.* Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning :-

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven’t surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can’t do without human intelligence,

but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Challenges in Machines Learning :-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

Quality of data − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Time-Consuming task − Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

Lack of specialist persons − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems − Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting − If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality − Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Difficulty in deployment − Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning :-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML −

- Emotion analysis
- Sentiment analysis

- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

How to Start Learning Machine Learning?

Arthur Samuel coined the term **"Machine Learning"** in 1959 and defined it as a **"Field of study that gives computers the capability to learn without being explicitly programmed".**

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is The Best Job of 2019 with a *344%* growth and an average base salary of **$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let's get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!!
Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as **Fork Python** available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model –** A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature –** A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label) –** A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training –** The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction –** Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

Supervised Learning – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.

Unsupervised Learning – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.

Semi-supervised Learning – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.

Reinforcement Learning – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning :-

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As **ML algorithms** gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning :-

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

Python Development Steps : -

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system.

Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked.Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode.Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and

"Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

- Print is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose :-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

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 is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have

to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Modules Used in Project :-

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the

origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

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 is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development,

the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Install Python Step-by-Step in Windows and Mac :

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here.The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

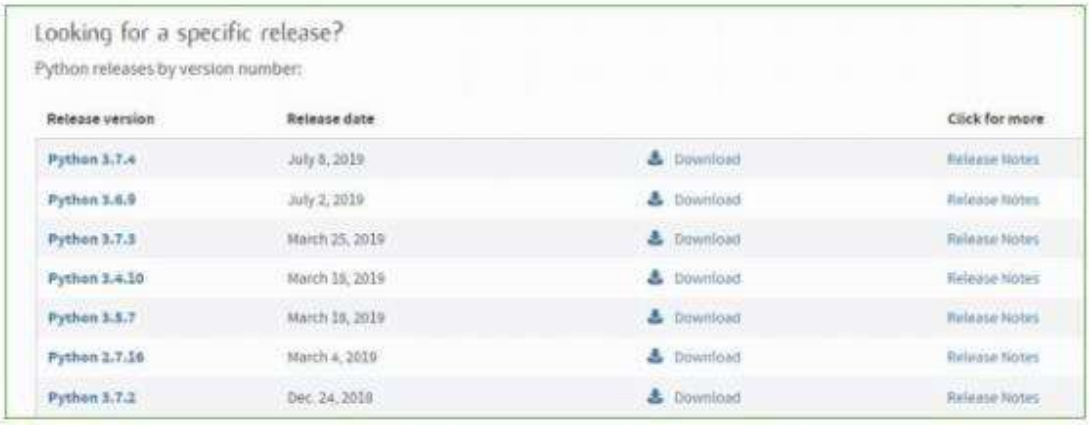
Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4



Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		6811167e5b2dbb4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c1eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.5 and later	6428bd4a7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5ddd60c3821fa45773bf5e4a936b243f	28982845	SIG
Windows help file	Windows		d6399957 3a2c06b2ec56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	5b03c3cf6d5ec0b9abe82184a497f28a2	7504291	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76d4b6d0304a3d83e563400	26883968	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28c31c9088bd73ae8e51a3bd35164bd2	1362504	SIG
Windows x86 embeddable zip file	Windows		9fab3b419864187f9da94135574139d8	6741628	SIG
Windows x86 executable installer	Windows		33c1022942d54446a3d8451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa6d317df82c30983ea371d87c	1324608	SIG

• To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.

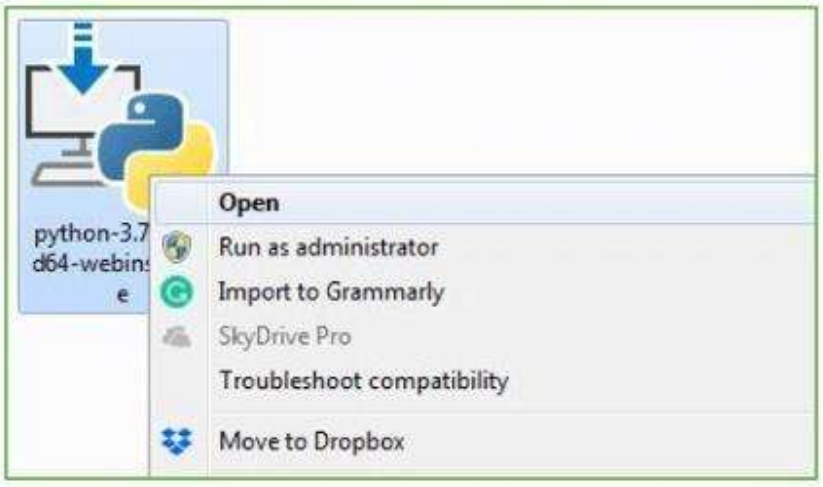
•To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.

Step 3: Click on Install NOW After the installation is successful. Click on Close.

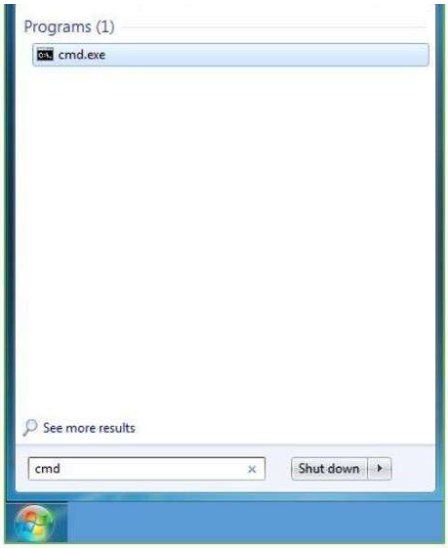
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

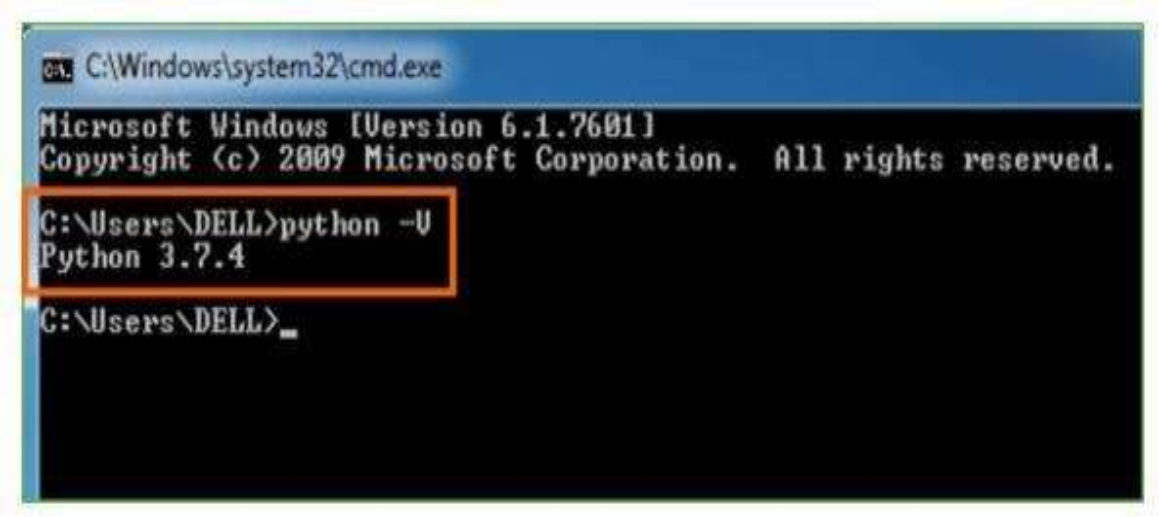
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type **python –V** and press Enter.



Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

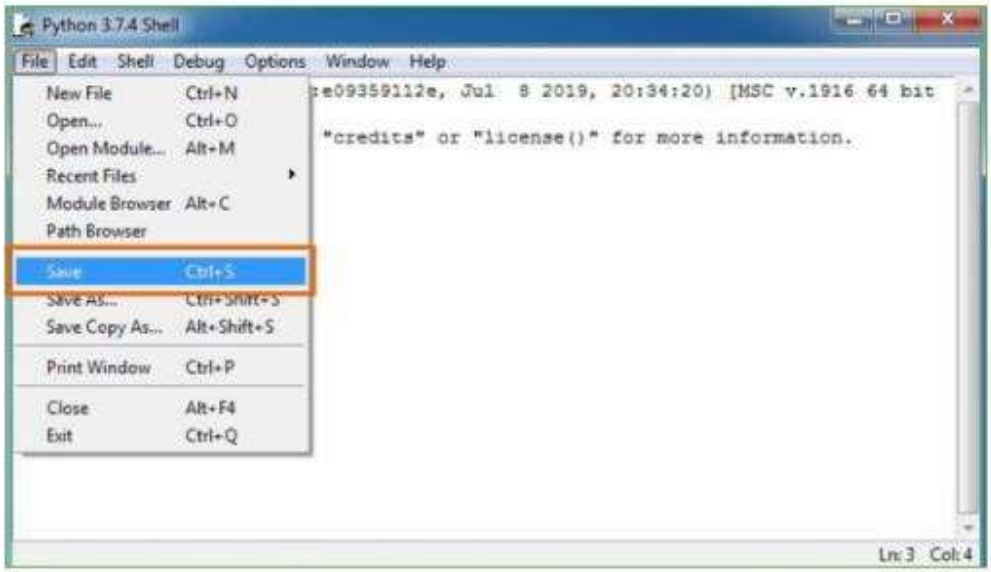
Check how the Python IDLE works

Step 1: Click on Start

Step 2: In the Windows Run command, type "python idle".

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. **enter print**

CHAPTER 7
SYSTEM DESIGN

7 SYSTEM DESIGN

7.1 UML DIAGRAMS

:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.

Support higher level development concepts such as collaborations, frameworks, patterns and components.

6. Integrate best practices.

USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

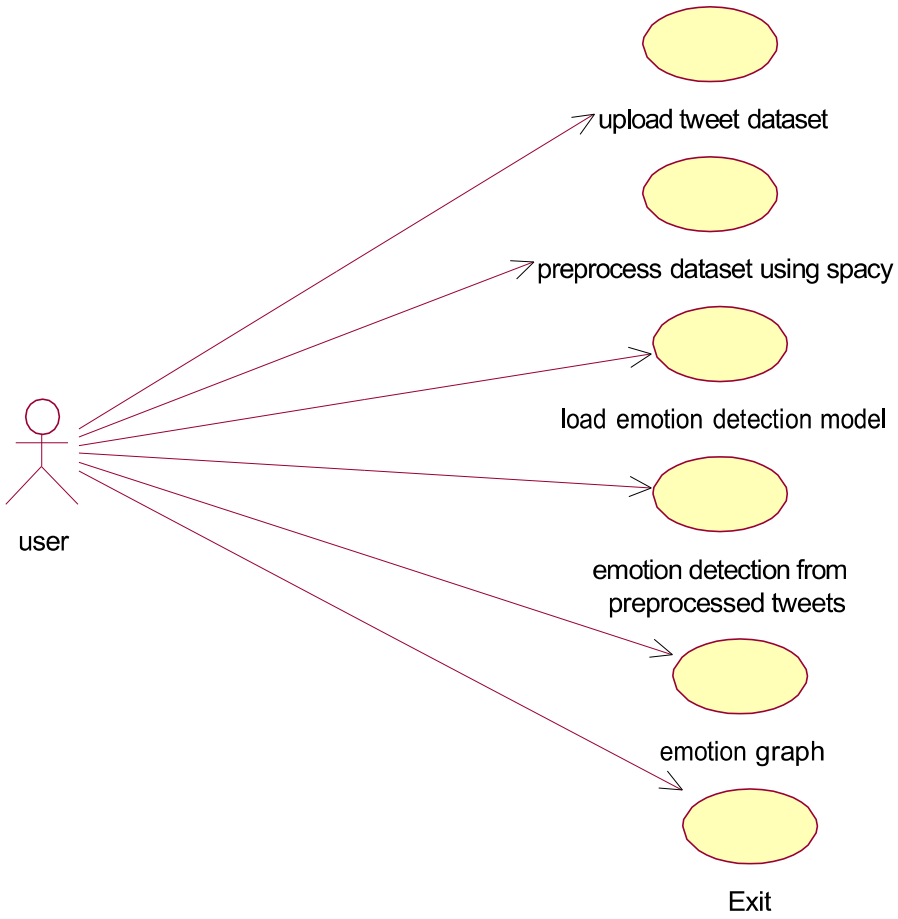


Fig 5

CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

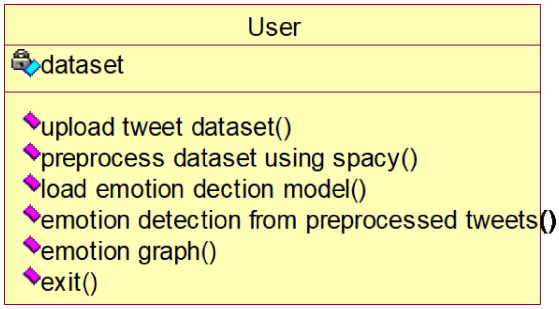
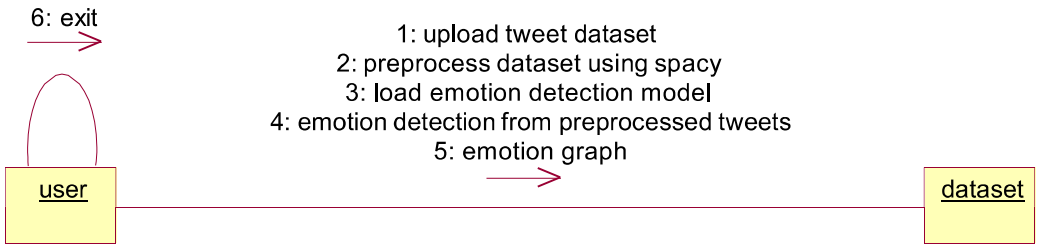


Fig 6

COLLABRATION DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

Fig 7



SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

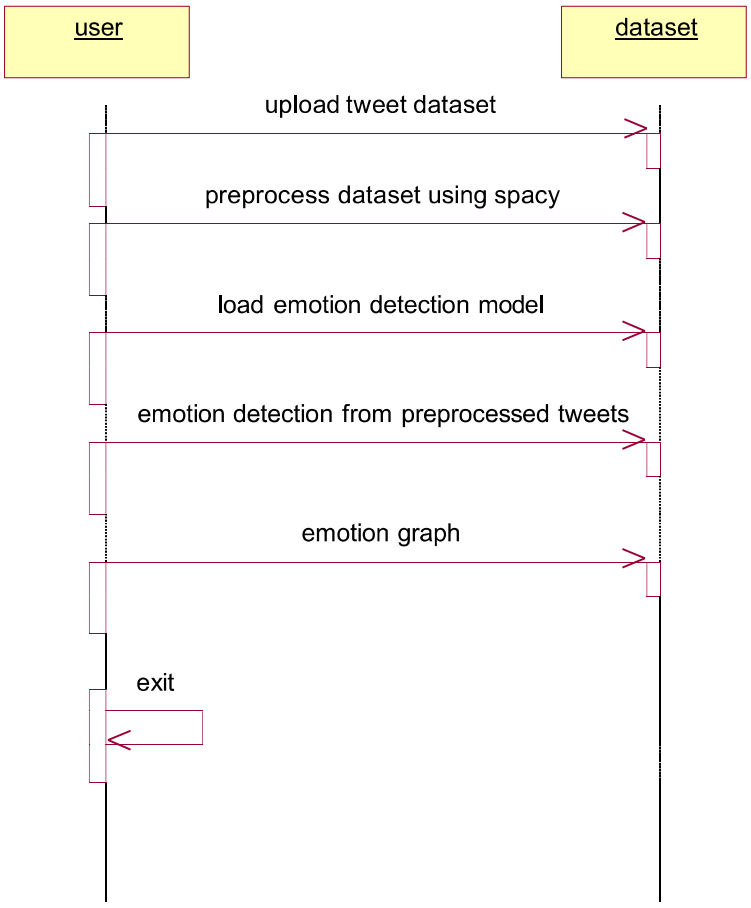


Fig 8

CHAPTER 8
IMPLEMENTATION

8 IMPLEMENTATION

8.1 Module :-

To implement this project we have designed following modules

1) **Upload Tweets Dataset**: using this module we will upload tweets messages to application
2) **Preprocess Dataset using Spacy**: using this module we will read each tweets and then apply spacy algorithm to clean and processed tweets
3) **Load Emotion Detection Model**: using this module we will load emotion detection machine learning algorithm
4) **Emotion Detection from Processed Tweets**: using this module we will apply each processed tweet on machine learning model which will predict emotion from given tweet
5) **Emotion Graph**: using this module we will plot emotion graph from all tweets

8.2 Source Code :-

```
from tkinter import messagebox

from tkinter import *

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

from tkinter.filedialog import askopenfilename

from tkinter.filedialog import askdirectory

import numpy as np

import os

import pandas as pd

import spacy
```

```
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
import matplotlib.pyplot as plt


main = tkinter.Tk()
main.title("Emotion Detection using Twitter Datasets and Spacy Algorithm") #designing main screen
main.geometry("1300x1200")


spacy_model = spacy.load('en_core_web_sm') #loading SPACY with english language model and dictionary



global emotion_model, dataset, tweets
global neutral, positive, negative


def uploadDataset():
    global dataset
    filename = filedialog.askopenfilename(initialdir="TweetsDataset")
    dataset = pd.read_csv(filename, encoding='utf-8',nrows=200)
    text.delete('1.0', END)
    text.insert(END,filename+" loaded\n\n")
    text.insert(END,str(dataset.head()))


def Preprocessing():
```

```
    text.delete('1.0', END)
    global tweets, dataset
    tweets = []
    dataset = dataset.values
    for i in range(len(dataset)):
        msg = dataset[i,1]
        msg = re.sub('[^A-Za-z]+', ' ', msg)
        msg = msg.strip("\n").strip()
        msg = spacy_model(msg)
        msg = msg.text
        tweets.append(msg)
        text.insert(END,msg+"\n\n")
        text.update_idletasks()
    messagebox.showinfo("Preprocessing Task Completed", "Preprocessing Task Completed")


def loadModel():
    global emotion_model
    text.delete('1.0', END)
    emotion_model = SentimentIntensityAnalyzer()
    text.insert(END,"Emotion Detection Model Loaded")



def detectEmotion():
```

```
    text.delete('1.0', END)
    global neutral, positive, negative, tweets, emotion_model
    neutral = 0
    positive = 0
    negative = 0
    for i in range(len(tweets)):
        sentiment_dict = emotion_model.polarity_scores(tweets[i].strip())
        compound = sentiment_dict['compound']
        if compound >= 0.05 :
            result = 'Positive'
            positive = positive + 1
        elif compound <= - 0.05 :
            result = 'Negative'
            negative = negative + 1
        else :
            result = 'Neutral'
            neutral = neutral + 1
        text.insert(END,str(tweets[i])+" ====> EMOTION DETECTED AS : "+result+"\n\n")


def emotionGraph():
    global neutral, positive, negative
    text.delete('1.0', END)
```

```
    plt.pie([positive, negative, neutral],labels=['Positive Tweets','Negative Tweets', 'Neutral Tweets'],autopct='%1.1f%%')
    plt.title('Tweets Emotion Graph')
    plt.axis('equal')
    plt.show()


def close():
    main.destroy()


font = ('times', 16, 'bold')
title = Label(main, text='Emotion Detection using Twitter Datasets and Spacy Algorithm')
title.config(bg='deep sky blue', fg='white')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)


font1 = ('times', 12, 'bold')
text=Text(main,height=20,width=150)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=50,y=120)
text.config(font=font1)
```

```
font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Tweets Dataset", command=uploadDataset)

uploadButton.place(x=50,y=550)

uploadButton.config(font=font1)


processButton = Button(main, text="Preprocess Dataset using Spacy", command=Preprocessing)

processButton.place(x=400,y=550)

processButton.config(font=font1)


emotionModelButton = Button(main, text="Load Emotion Detection Model", command=loadModel)

emotionModelButton.place(x=750,y=550)

emotionModelButton.config(font=font1)


emotionDetectionButton = Button(main, text="Emotion Detection from Processed Tweets",
command=detectEmotion)

emotionDetectionButton.place(x=50,y=600)

emotionDetectionButton.config(font=font1)


graphButton = Button(main, text="Emotion Graph", command=emotionGraph)

graphButton.place(x=400,y=600)

graphButton.config(font=font1)
```

```
exitButton = Button(main, text="Exit", command=close)

exitButton.place(x=750,y=600)

exitButton.config(font=font1)



main.config(bg='LightSteelBlue3')

main.mainloop()
```

CHAPTER 9
SYSTEM TESTING

9 SYSTEM TESTING

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 It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

9.1 TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

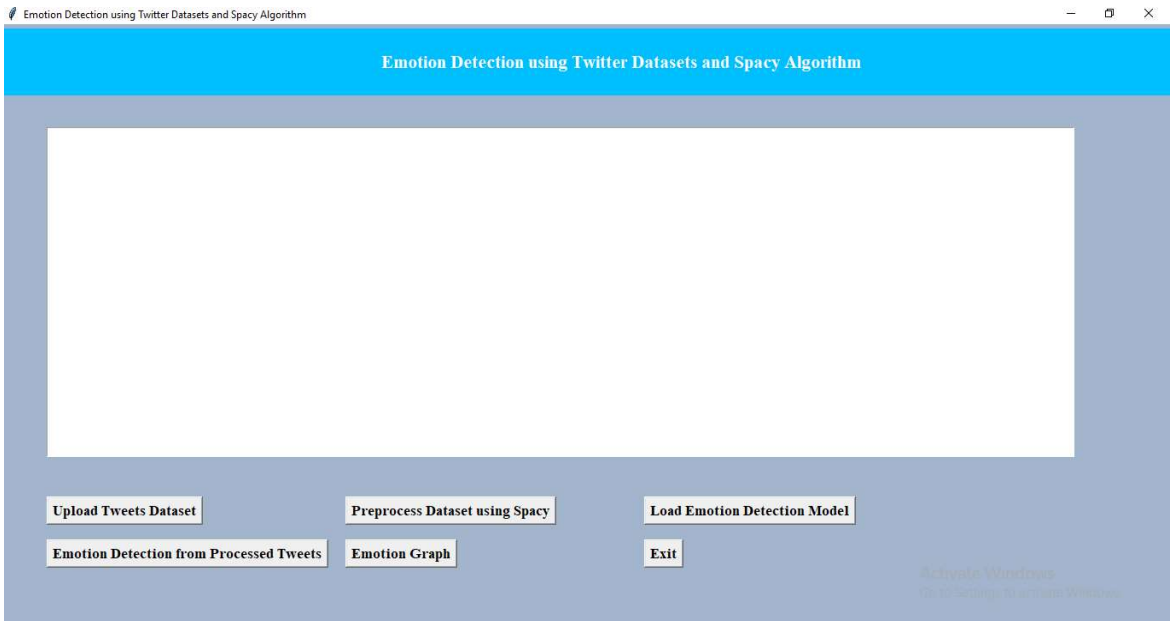
9.2 TEST CASES

Module	Functionality	Text Case	Expected Result	Actual Result	Result	Priority
Upload Tweets Dataset	Upload tweets messages to application	Click the upload tweets Dataset	Path of the dataset must be visible in the Dataset Location field	A validation has be populated as expected	pass	High
Preprocess Dataset Using Spacy	Filter and processed tweets	Click on Preprocess Dataset using spacy alogithm to clean tweets	The desired dataset is prepossed	Total tweets Unique words And hashtags are removed	pass	High
Load Emotion Detection	Load emotion detection machine algorithm	After click Ok button to click on Load Emotion detection model to load Ml models	Features extraction process completed	Feature extraction process completed	pass	High
Emotion Detection From Processed Tweets	Pridict emotion from given tweets	To predict emotion as Positive,Negative or Neutral	Moods Detected: Positive,Negative or Neutral	Moods are detected respectiviley	pass	High
Emotion Graph	Plot emotion graph from all tweets	To show the tweets percentage on Emotion Graph	Entire process is shown pie chat with averages of moods	The entire process is represented as pie chat respectively	pass	High

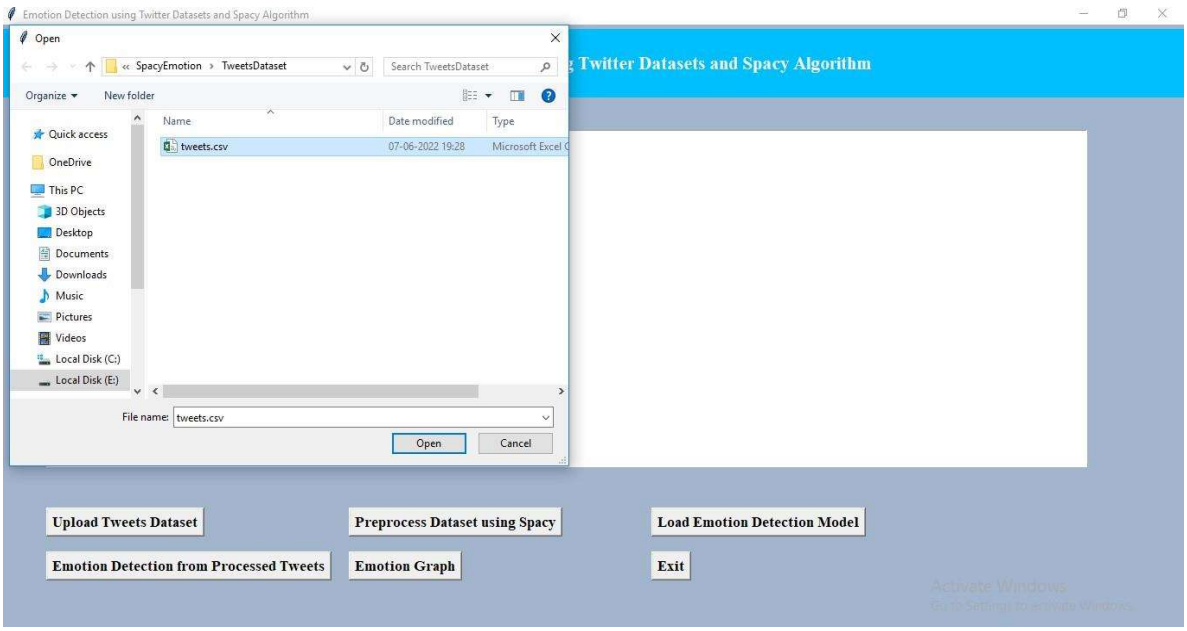
CHAPTER 10
SCREENSHOTS

10 SCREENSHOTS

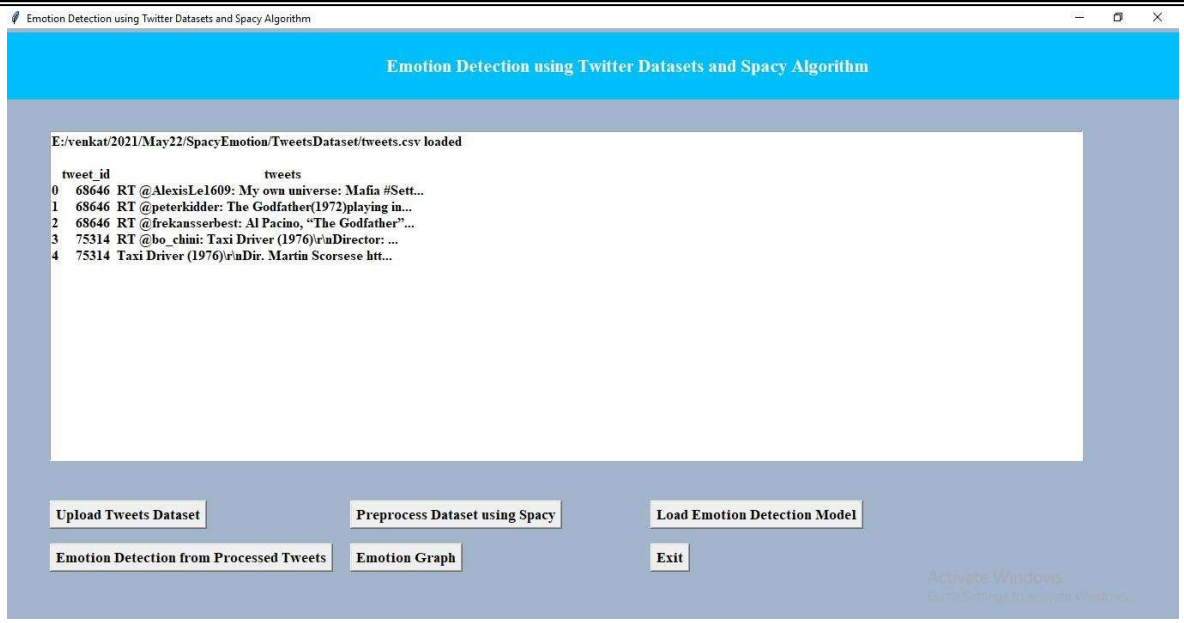
To run project double click on 'run.bat' file to get below screen



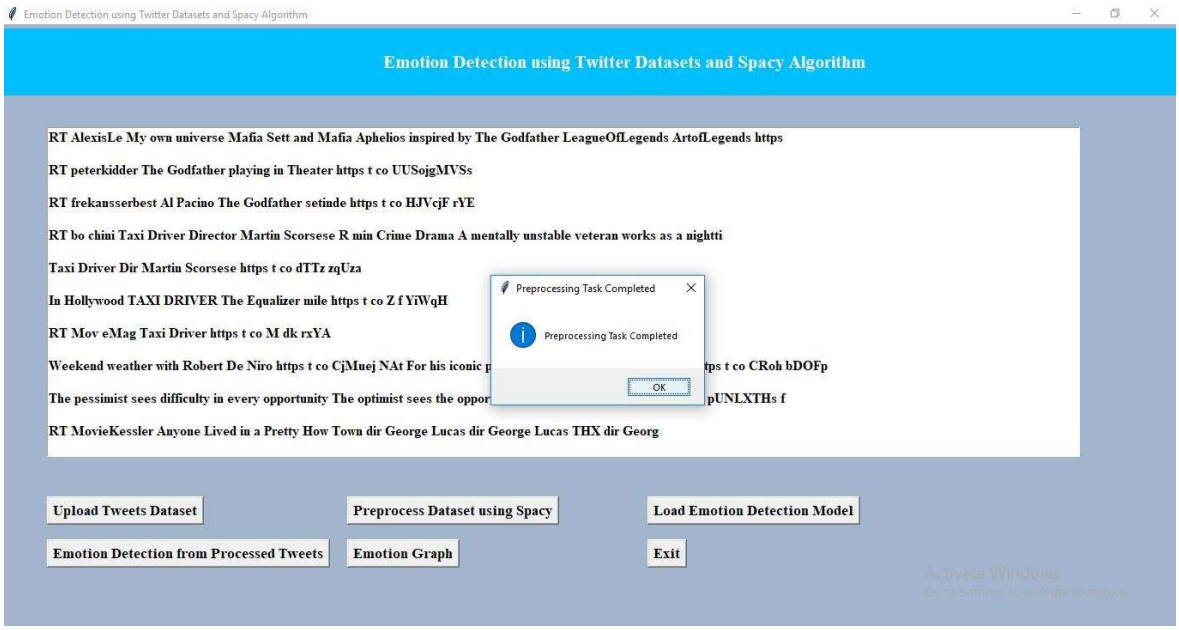
In above screen click on 'Upload Tweets Dataset' button to load tweets and get below output



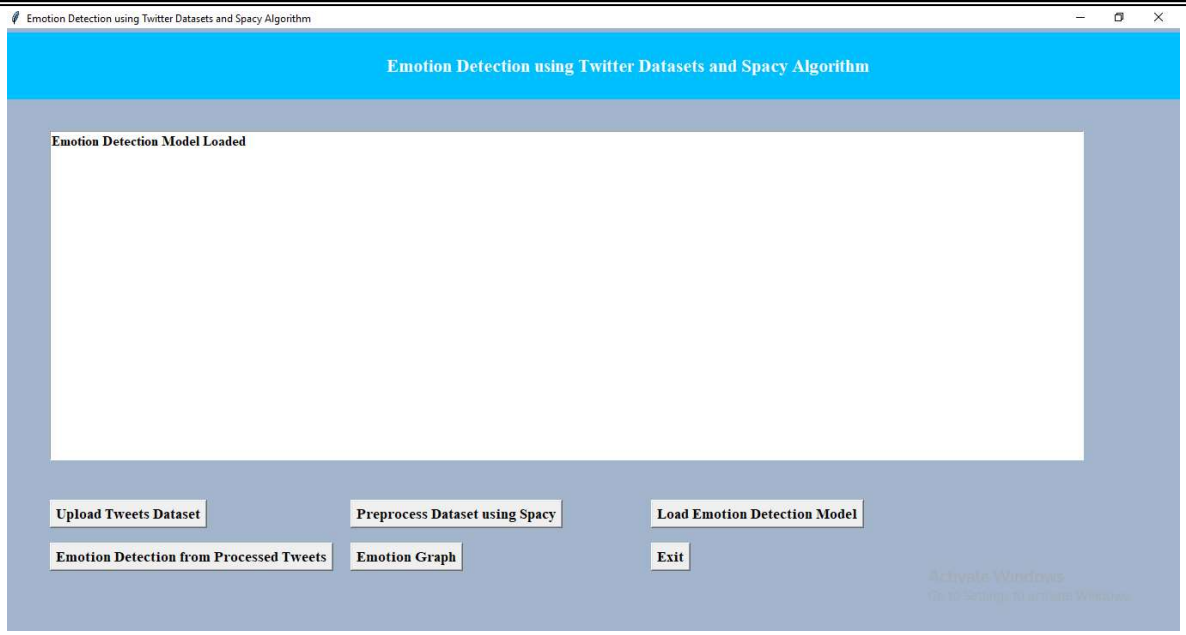
In above screen selecting and uploading tweets dataset and then click on 'Open' button to get below output



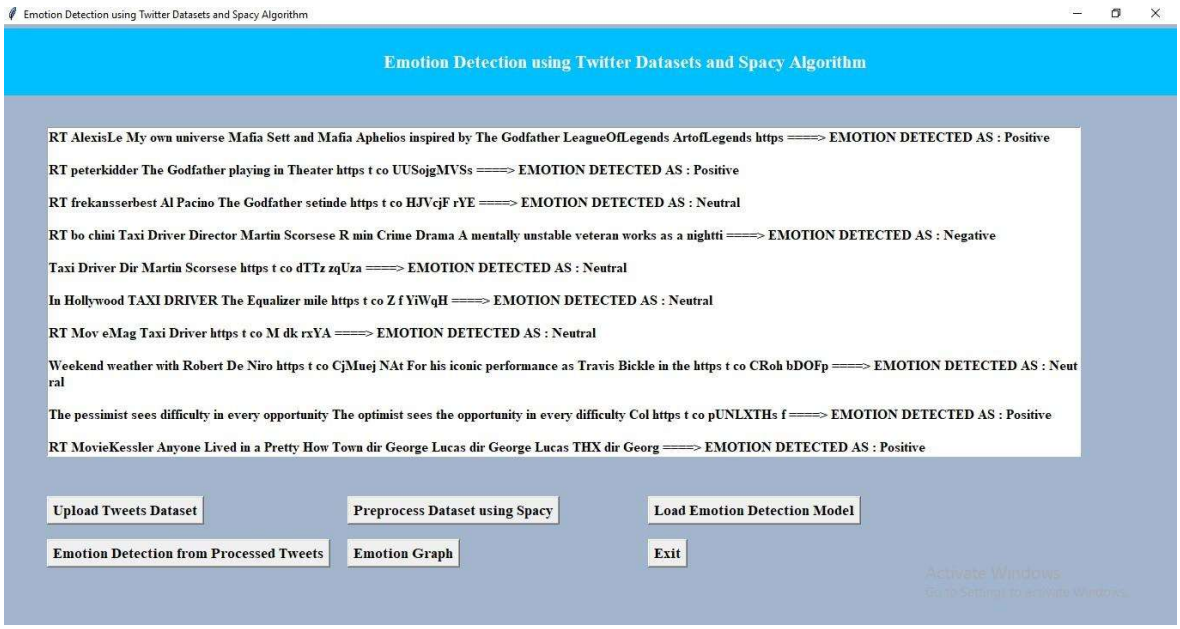
In above screen we can see dataset loaded and tweets contains total unstructured text with stop words and special symbols and now click on 'Preprocess Dataset using Spacy' to clean tweets and get below output



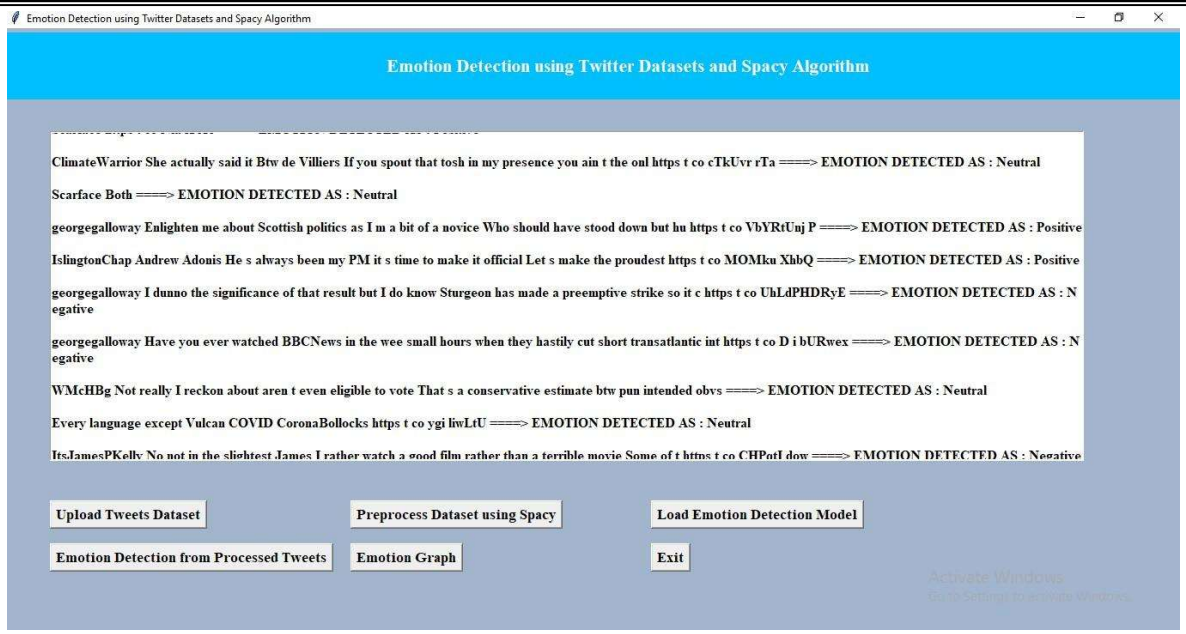
In above screen Preprocessing completed and we can see all tweets contains only text with clean words and now click 'Ok' button and then click on 'Load Emotion Detection Model' button to load machine learning model for emotion detection and get below output



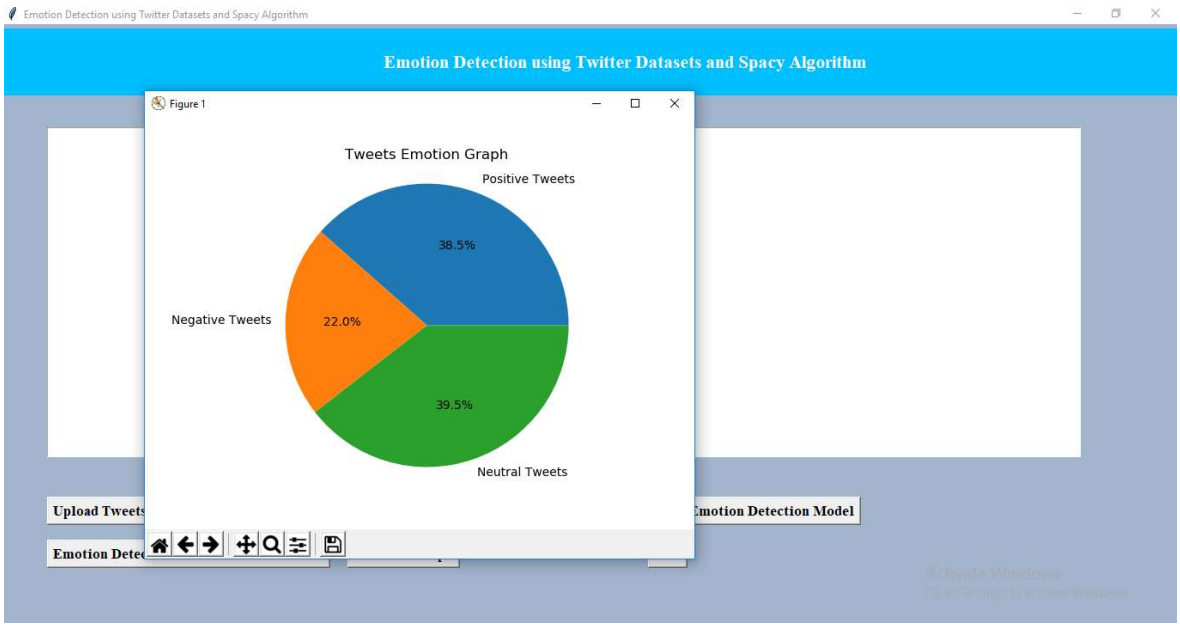
In above screen model is loaded and now click on 'Emotion Detection from Processed Tweets' button to detect emotion and get below output



In above screen before arrow symbol =➔ we can see clean tweet messages and after arrow symbol we can see predicted emotion as 'Positive, Negative or Neutral' and scroll down above screen to view all messages



In above screen we can see all tweets with emotion and now click on 'Emotion Graph' to know tweets percentage in each emotion



In above graph 38.5% peoples are giving positive tweets and 22% gave negative tweets and 39.5% gave neutral tweets so by using this application we can easily extract useful knowledge from peoples reviews weather they are satisfy or not on any topics tweets.

CHAPTER 11
CONCLUSION
&
FUTURE ENHANCEMENT

11 CONCLUSION & FUTURE ENHANCEMENT

11.1CONCLUSION :

Emotion detection is one of the toughest problems to solve. Detecting emotion from text is a challenging work and most of the research works have some kind limitations most importantly, language ambiguity, multiple emotion bearing text, text which does not contain any emotion words etc. Yet we have tried several approaches to detect emotion from twitter. We can say after using EmoSenticNet lexicon, the model performs better than using only WordNet-Affect. It can be also said that our model has performed well but still better results are achievable. As for accuracy, the EmoSenticNet outperforms WordNet-Affect by a great margin. Our limitations are that we have used a small sample as our dataset and there are still language ambiguity problems as we have not been able to address texts which represent multiple emotion at the same time. In the future, we will introduce Deep Learning techniques to identify emotion detection on this dataset.

TABLE VI
ACCURACY OF DIFFERENT CLASSIFIERS

Work	Feature Selection	Classifier	Result
[12]	Logistic Regression	Random Forest	90% on Anger Class
Our Model	EmoSenticNet	SVM	89.28% on Anger Class

11.2 Future Enhancement :

Future enhancements for the proposed model for emotion detection in tweet texts could include:

1. **Incorporating Contextual Embeddings:** Utilize pre-trained contextual embeddings such as BERT or GPT to capture richer semantic and contextual information from tweet texts. These embeddings can capture nuanced meanings and relationships between words, improving the model's ability to detect emotions accurately.

2. **Ensemble Learning**: Explore ensemble learning techniques to combine predictions from multiple models trained on different features or using different algorithms. Ensemble methods often lead to improved performance by leveraging the strengths of individual models and mitigating their weaknesses.

3. **Fine-tuning on Domain-Specific Data:** Fine-tune the model on domain-specific data related to tweets, such as Twitter datasets with labeled emotional content. Fine-tuning can help adapt the model to the specific characteristics and language conventions of tweets, enhancing its performance in real-world applications.

4. **Multi-modal Emotion Detection:** Integrate information from multiple modalities, such as text, images, and user metadata (e.g., user profiles, engagement patterns), for a more comprehensive understanding of emotional content in tweets. This multi-modal approach can capture emotions expressed not only through text but also through accompanying images, emojis, and user interactions.

5. **User-Level Emotion Analysis:** Extend the model to analyze emotions at the user level, considering the historical context and patterns of emotional expression of individual users on Twitter. This personalized approach can provide deeper insights into user behavior and emotional dynamics on social media platforms.

6. **Real-Time Emotion Detection:** Develop mechanisms for real-time emotion detection to enable timely response and intervention in online interactions. This could involve optimizing the model's efficiency for fast inference on streaming data or deploying it as part of a larger system for real-time sentiment analysis and monitoring.

7. **Addressing Bias and Fairness:** Conduct thorough analysis and mitigation of biases present in the training data and model predictions to ensure fair and equitable emotion detection across different demographic groups. This includes addressing biases related to gender, race, culture, and language that may affect the model's performance and downstream applications.

8. **Interpretability and Explainability:** Enhance the interpretability and explainability of the model by incorporating techniques such as attention mechanisms or generating explanations for model predictions. This can help users understand why certain emotions are detected in tweet texts and foster trust in the model's decisions.

By incorporating these future enhancements, the proposed model can become more robust, accurate, and applicable to various real-world scenarios involving emotion detection in tweet texts.

CHAPTER 12
BIBLIOGRAPHY

12 BIBLIOGRAPHY

12.1 REFERENCES :

1. R. Hirat and N. Mittal, “A Survey On Emotion Detection Techniques using Text in Blogposts,” International Bulletin of Mathematical Research, vol. 2, no. 1, pp. 180–187, 2015.

2. R. Sawyer and G.-M. Chen, “The Impact of Social Media on Intercultural Adaptation,” 2012.

3. S. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, “Semeval-2018 Task 1: Affect in Tweets,” in Proceedings of The 12th International Workshop on Semantic Evaluation, 2018, pp. 1–17.

4. R. C. Balabantaray, M. Mohammad, and N. Sharma, “Multi-class Twitter Emotion Classification: A New Approach,” International Journal of Applied Information Systems, vol. 4, no. 1, pp. 48–53, 2012.

5. M. Anjaria and R. M. R. Guddeti, “Influence factor based opinion mining of Twitter data using supervised learning,” in 2014 Sixth International Conference on Communication Systems and Networks (COMSNETS), 2014, pp. 1–8.

6. S. Yuan, H. Huang, and L. Wu, “Use of Word Clustering to Improve Emotion Recognition from Short Text,” Journal of Computing Science and Engineering, vol. 10, no. 4, pp. 103–110, 2016.

7. B. Thomas, P. Vinod, and K. A. Dhanya, “Multiclass Emotion Extraction from Sentences,” International Journal of Scientific & Engineering Research, vol. 5, no. 2, 2014.

8. H. Yang, A. Willis, A. D. Roeck, and B. Nuseibeh, “A Hybrid Model for Automatic Emotion Recognition in Suicide Notes,” Biomedical informatics insights, vol. 5, p. 8948, 2012.

9. D. T. Ho and T. H. Cao, “A High-order Hidden Markov Model for Emotion Detection from Textual Data,” in Pacific Rim Knowledge Acquisition Workshop, 2012, pp. 94–105.

10. M. Hasan, E. Rundensteiner, and E. Agu, “Automatic emotion detection in text streams by analyzing Twitter data,” International Journal of Data Science and Analytics, vol. 7, no. 1, pp. 35–51, 2019.

11. M. Hasan and E. Rundensteiner and E. Agu, “Emotex: Detecting Emotions in Twitter Messages,” 2014.

12. A. Seyeditabari, S. Levens, C. D. Maestas, S. Shaikh, J. I. Walsh, W. Zadrozny, C. Danis, and O. P. Thompson, "Cross Corpus Emotion Classification Using Survey Data," This paper was presented at AISB, 2017.

13. H. Binali, C. Wu, and V. Potdar, "Computational approaches for emotion detection in text," in 4th IEEE International Conference on Digital Ecosystems and Technologies, 2010, pp. 172–177.

14. C. Strapparava, A. Valitutti et al., "Wordnet-Affect: an Affective Extension of WordNet," in Lrec, vol. 4, 2004, p. 40.

15. S. Poria, A. Gelbukh, A. Hussain, N. Howard, D. Das, and S. Bandyopadhyay, "Enhanced SenticNet with Affective Labels for ConceptBased Opinion Mining," IEEE Intelligent Systems, vol. 28, no. 2, pp. 31–38, 2013.

16. E. Cambria, D. Olsher, and D. Rajagopal, "SenticNet 3: A Common and Common-Sense Knowledge Base for Cognition-Driven Sentiment Analysis," in Twenty-eighth AAAI conference on artificial intelligence, 2014.

17. D. Effrosynidis, S. Symeonidis, and A. Arampatzis, "A Comparison of Pre-processing Techniques for Twitter Sentiment Analysis," in International Conference on Theory and Practice of Digital Libraries, 2017, pp. 394–406.

18. A. G. Jivani et al., "A Comparative Study of Stemming Algorithms," Int. J. Comp. Tech. Appl, vol. 2, no. 6, pp. 1930–1938, 2011.

19. "Slang_Dict," https://floatcode.files.wordpress.com/2015/11/slang_dict. doc, accessed: 2019-05-17.

20. "Gthub - wolfgarbe/SymSpell," https://github.com/wolfgarbe/SymSpell, accessed: 2019-05-17.

21. Y. Yang, X. Liu et al., "A re-examination of text categorization methods," in Sigir, vol. 99, no. 8, 1999, p. 99.

22. H. Zhang, "The optimality of naive bayes," AA, vol. 1, no. 2, p. 3, 2004.

23. G. Forman and I. Cohen, "Learning from little: Comparison of classifiers given little training," in European Conference on Principles of Data Mining and Knowledge Discovery. Springer, 2004, pp. 161–172.

24. A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in Advances in neural information processing systems, 2002, pp. 841–848.

25. S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python, 1st ed. O'Reilly Media, Inc.