Analyzing Developer Sentiment throughout Development Process

by

Aman Kumar
201911018

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

MASTER OF TECHNOLOGY
in
INFORMATION AND COMMUNICATION TECHNOLOGY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

June, 2021

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Master of Technology in Information and Communication Technology at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

Aman Kumar

Certificate

This is to certify that the thesis work entitled *"Analyzing Developer Sentiment throughout Development Process"* has been carried out by Aman Kumar (201911018) for the degree of Master of Technology in Information and Communication Technology at *Dhirubhai Ambani Institute of Information and Communication Technology* under our supervision.

Dr. Manish Khare	Dr. Saurabh Tiwari
Thesis Supervisor	Thesis Co-Supervisor

Acknowledgments

My sincere gratitude to Dr. Manish Khare and Dr. Saurabh Tiwari, who expertly guided me throughout my MTech Thesis. This thesis couldn't be completed without the effort and contribution of my supervisor and co-supervisor. I would also like to thank DAIICT for providing me with the opportunity to work on my thesis and for providing supportive environment.

I would also like to express my gratitude to my panel members for their extended discussions and valuable suggestions which helped greatly in the completion of the thesis.

I would like to thank my family for their words of encouragement and valuable support throughout. I would also like to thank my friends for their much needed help and suggestions that went behind the completion of the thesis. Lastly, I would like to thank God Almighty for his blessings.

Contents

Abstract **v**

List of Principal Symbols and Acronyms **v**

List of Tables **vi**

List of Figures **vii**

1 Introduction **1**

2 Literature Survey **4**

2.1 **Sentiment Analysis** 4

2.1.1 **Related Work** 5

2.2 **GitHub** 6

2.2.1 **Related Work** 6

3 Dataset Used **8**

3.1 **Data Collection for Sentiment Analysis on GitHub** 8

3.2 **DataSet for Comparison of Sentiment Analysis Algorithms** 10

4 Sentiment Analysis Process **11**

4.1 **Data Pre-Processing** 11

4.2 **Data Processing** 12

4.2.1 **For Sentiment Analysis on GitHub** 12

4.2.2 **For Comparison of Sentiment Analysis Algorithms** 13

5 Experimental Results and Analysis **21**

5.1 **Sentiment of Python Developer related to day of week** 21

5.2 **Sentiment of Contributors throughout the course of the project.** . . 25

5.3 **Sentiment of developers across different programming languages.** 29

5.4 **Comparison of Sentiment Analysis Algorithms** 30

6 Discussions 32

7 Conclusions and Future Work 33

References 34

Abstract

Emotions plays a major role in productivity, quality and job satisfaction. In this thesis, I have analyzed sentiments of developer on GitHub based on the comments of different open source projects and analyze their relationship with different factors such as day of week in which comment was made, time duration from which project was created and used programming language. I analyzed sentiment of those developers using comments written on issues, pull requests and commits. My results show that projects developed on Monday tend to a more negative emotion. Additionally, comments written in issues tend to have higher negative polarity in their emotional content and projects developed in java and python have more positive comments as compared to C and C++. There are many Sentiment Analysis Techniques available. I have also discussed about the Sentiment Analysis algorithms which can be used to improve accuracy and generate more accurate results. This thesis provides an overall approach to perform sentiment analysis and talks about the conclusions which can drawn from this thesis.

Keywords: Sentiment Analysis, GitHub, Issues, Pull Request, Commit.

List of Tables

2.1 Example of Sentiment in texts. 4

3.1 GitHub Python Repositories for RQ1 and RQ2 9
3.2 GitHub Repositories for RQ3 10

5.1 Sentiment of Developer with respect to Week Days. 22
5.2 Sentiment of Developer with respect to Week Days in Issue Comments. 23
5.3 Sentiment of Developer with respect to Week Days in Pull Request Comments. 23
5.4 Sentiment of Developer with respect to Week Days in Commit Comments. 23
5.5 Sentiment of Developer throughout the course of the project. 26
5.6 Sentiment of Developer throughout the course of the project in Issue Comments. 27
5.7 Sentiment of Developer throughout the course of the project in Pull Request Comments. 27
5.8 Sentiment of Developer throughout the course of the project in Commit Comments. 27
5.9 Sentiment of Developer with respect to Programming Language. . 30

List of Figures

3.1 Proposed Methodology for Sentiment Analysis 8

4.1 Steps for Pre-Processing. 11
4.2 Data processing using *SentiStrength*. 14
4.3 Convolution 3 Layer Model. 17
4.4 Convolution 4 Layer Model. 18
4.5 LSTM Model. 19
4.6 Basic Components of LSTM Model. 19
4.7 Ensemble Model. 20

5.1 Sentiment of Developer with respect to Week Days. 22
5.2 Sentiment of Developer with respect to Week Days in Issue Comments. 24
5.3 Sentiment of Developer with respect to Week Days in Pull Request Comments. 24
5.4 Sentiment of Developer with respect to Week Days in Commit Comments. 25
5.5 Sentiment of Developer throughout the course of the project. 26
5.6 Sentiment of Developer throughout the course of the project in Issue Comments. 28
5.7 Sentiment of Developer throughout the course of the project in Pull Request Comments. 28
5.8 Sentiment of Developer throughout the course of the project in Commit Comments. 29
5.9 Sentiment of Developer with respect to Programming Language. . . 30
5.10 Comparison of Sentiment Analysis Algorithms. 31

CHAPTER 1
Introduction

The Process of analyzing emotions from texts is called Sentiment Analysis. Sentiment Analysis is one of the most famous and useful problem for quantifying the sentiments and emotions from neutral texts. In this thesis, I applied Sentiment Analysis on GitHub comments to gain the knowledge of emotions expressed in the comments, also to gain knowledge about the different Sentiment Analysis Algorithms.

Software development is a collaborative activity that involves humans (or developers) in different stages of the development cycle. GitHub[1] is a popular source code hosting and collaboration platform. The collaboration among developers requires communication channels to express their emotions w.r.to issues, discussions and logs. Their emotions with the development may affect productivity and the quality of the project [3]. The project managers need to be aware of the developers' emotions while working on the project to ensure that emotions would not affect project quality. In this thesis, I investigated developer emotions working on open-source projects and collaborations through GitHub Repositories. Additionally, studied about the relationship between their emotions, the day of the week comments were written, the time duration the project was created and the programming language used to develop the project [13]. The Research questions that guide the thesis are:

*RQ*1 **Sentiment of Python Developer related to day of week.**

*RQ*2 **Sentiment of Python Project Contributors throughout the course of the project.**

*RQ*3 **Sentiment of Developers across different programming languages in which project is developed.**

[1] https://github.com/

*RQ*4 **Comparison of Different Sentiment Analysis Algorithms.**

To answer the formulated RQs, I have analysed issue comments, pull request comments, and commit comments posted by developers in the repositories w.r.to their emotions to the day of the week and throughout the course of the project. To determine the emotion of developers related to the day of the week and throughout the course of the project, I have selected the top 10 Python GitHub repositories fetched based on the number of stars. The comments posted by the developers on the day of the week are analysed using Sentistrength [8][2]. The Sentistrength is a lexical extraction tool that estimates the strength of positive and negative sentiment in short texts. The analysis of the issue comments, pull request comments and commit comments posted by the developers w.r.to the day of the week revealed that comments written on Monday have more negative polarity emotions as compared to other days of the week. Next, I have analysed the emotion of developers throughout the course of the project by grouping the comments posted by the developers. The comments were then analysed based on time duration (i.e., days/months/years) from the date of project creation. I have used the Sentistrength tool to determine the sentiments and found that comments written in issues and pull requests are more negative in polarity as compared to comments written in commit comments. To answer RQ3, i.e., emotions of the developers based on the programming languages used for development. Here, I have used 3 GitHub repositories for each of the programming languages (such as Python, Java, C, C++ and Dot Net) based on the number of stars. I have observed that comments written in Python and Java programming languages have positive sentiments compared with other languages.

Next, I am comparing various algorithms which are used to classify sentiments in text. To compare different sentiment analysis algorithm I have used Sentiment140 [11][3] dataset and compared them on the basis of accuracy. These algorithms are [10]

1. NLTK
2. Decision Trees
3. Random Forest
4. Support Vector Machine

[2]http://sentistrength.wlv.ac.uk
[3]http://help.sentiment140.com/

(a) Linear Kernel
(b) Polynomial Kernel
(c) Gaussian Radial Basis Kernel
(d) Sigmoid Kernel

5. Convolutional Neural Network
6. Long Short Term Memory
7. Ensemble Techniques

I observed that Deep Learning algorithms are giving more accurate results as compared to other algorithms.This thesis first talk about the process of sentiment analysis then the different types of Sentiment Analysis algorithm and compare the results.

The organisation of the thesis is as follows. Chapter II contains the discussion about the literature survey which talks about sentiment analysis and the various research papers related to sentiment analysis, also sentiment analysis in the GitHub repository. Chapter III presents the dataset creation for the Research questions. Chapter IV discuss the process to perform sentiment analysis. Chapter V highlight the answers for RQs. Section VI discuss and presents the results of the analysis. Section VII presents the conclusions and future work of the study.

CHAPTER 2

Literature Survey

In this section, I have discussed about the different methodologies used in the thesis and their related work.

2.1 Sentiment Analysis

Sentiment Analysis is the management of sentiments, subjective texts and opinions. It provides the information related to public views, as it analyzes different comments and opinions. The Opinions can be categorized into negative, positive and neutral. The purpose of sentiment analysis is to automatically determine emotion of the user. The demand of sentiment analysis is raised due to increase in requirement of analyzing hidden information from social media in the form of unstructured data. Sentiment analysis is an important research area due to huge number of daily posts on social media, extracting people's opinion is a challenging task. Sentiment Analysis allows us to sort large sets of data and detect polarity of each text automatically which eventually saves time and resources [1].
Some Examples of Sentiment Analysis

Table 2.1: Example of Sentiment in texts.

Text	**Sentiment**
I am very happy today.	Positive
I have to complete this report.	Neutral
The weather is terrible.	Negative
Dataset is mixture of words.	Neutral

Advantage of Sentiment Analysis are [9]

1. Scalability: It can analyze huge amount of data within a short period of time which can be a hassle doing manually.

2. Real-Time Analysis: Industries use it to monitor real time data and make changes or improvements wherever needed.

3. Consistent Criteria: Analyzing sentiment is a subjective task which can be perceived differently when done by two member and results will probably be biased.

Applications of Sentiment Analysis [4]

1. Review – Related Websites – Movie reviews, Product reviews, etc.

2. Sub-Component Technology – Spam detection, Context sensitive information detection, etc.

3. Business and Government Intelligence – Gaining knowledge of consumer attitude and trends.

4. Across Different Domains – Gaining knowledge of political leaders to their notions about rules and regulation in place, etc.

5. Predicting stock prices based on opinions that people have about the companies and resources.

2.1.1 Related Work

Several studies have been conducted by several authors on sentiment analysis. An important aspect in approaching Sentiment analysis is represented by the work of Feldman [9]. The author discussed the techniques to perform sentiment analysis and the various applications of sentiment analysis that help us in solving real-world problems. The author discussed how sentiment analysis techniques can be used to solve complex problems which some industries face and how they can simplify their problems using sentiment analysis.

Beigi et al. [2] discussed the application of sentiment analysis on disaster relief and the overview of sentiment analysis in social media. The author talks about how sentiment analysis is not limited to business intelligence, politics, sociology, etc. The author studied the reaction of the local crowd and used such information to improve disaster management.

Dolianiti et al. [7] presented the applications of sentiment analysis in education. The author talked about how cognition and emotion are involved in every learning process and how student profiles can be enhanced with information regarding

the effective state. The author explored many different ways in which sentiment analysis can be applied in the educational domain.
Das et al. [5] discussed real-time prediction of stock by analysing sentiment of Twitter streaming data in their study. The author attempts to make financial decisions such as stock market prediction, to predict the potential prices of a company's stock and to serve the need of this, Twitter streaming data has been considered for scoring the impression that is carried for a particular firm.

2.2 GitHub

GitHub is a hosting platform for software development and version control using Git. It enables us to work with others on projects from anywhere around the globe. GitHub has repositories to organize single project. A Repository contains folders and files that project needs. User can also use branching to work indifferent versions of a repository at one time. On GitHub saved changes are called commit which has an associated commit message, which is a description explaining why particular changes were made. Pull Requests are used to propose your changes and requesting to review and merge the branches.Issues are used to collect user feedback, report bugs and organize tasks. In this Thesis I have used Comments from Issues, pull requests and commits to perform Sentiment Analysis and solve the problem statements [6][14].

2.2.1 Related Work

Sentiment Analysis on GitHub has many significant studies. One of them is done by Pletea et al. [23]. The authors detected the relationship between security-related topics and sentiment analysis on comments. As application security is becoming increasingly prevalent during software development, the authors studied that the majority of security-related comments have a negative sentiment. These findings confirm the importance of proper training of developers to address security concerns in their software.
Huq et al. [16] discussed the relationship between developer sentiments and their relationship to software bugs. The author correlated several patterns by conducting sentiment analysis on software collaborative artefacts like commits and aims to find the impact of those patterns by finding a relation between developer sentiment and software bugs. It is statistically observed by the author that Commits that introduce, precede or fix bugs are significantly more negative than regular

Commits, with a higher proportion of emotional (non-neutral) messages. It is also found that a distinction between buggy and correct fixes exists based on the message's neutrality.

Skriptsova et al. [28] presented the activity of newcomers in communicative posts on GitHub. The author aims to investigate relations towards newcomers through sentiment analysis of comments they receive in issues and pull requests in repositories on GitHub. By applying sentiment analysis, the author focuses on differences between reactions to contributions of 'old' and 'new' developers and finds that while the majority of comments is rated as neutral, the amount of negativity is slightly higher for newcomers.

Huq et al. [15] described how developer sentiment is related to bugs, by analysing the difference of sentiment between regular and Fix-Inducing Changes (FIC) - changes to code that introduce bugs in the system. The author analysed pull requests from GitHub repositories that contain contributor discussions and analyzed the sentiment of those discussions. FICs are detected from Commits by filtering the ones that fix bugs and tracking the origin of the code this remove.

CHAPTER 3

Dataset Used

3.1 Data Collection for Sentiment Analysis on GitHub

This study performs sentiment analysis on GitHub comments posted by the developers for the issues, pull requests and commits to assess the developer sentiments throughout the project duration, on the specific day of the week, and with a programming language. An overview of the methodology is provided in Fig. 3.1.

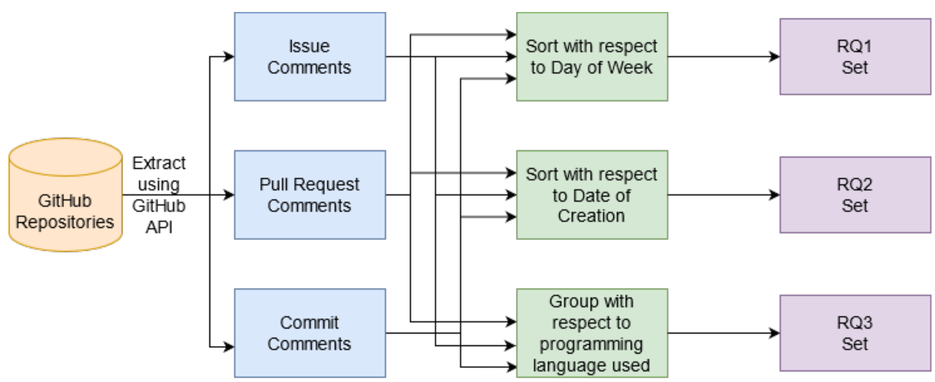

Figure 3.1: Proposed Methodology for Sentiment Analysis

The data collection from GitHub is done using pyGitHub[1], a library in python. pyGitHub is a library written in python to make use of GitHub API. Using pyGitHub, I have retrieved issue comments, pull request comments and commit comments posted by developers in the GitHub repositories. The pyGitHub contains functions and variables to communicate with the GitHub API and fetch the desired results from GitHub repositories.

To answer RQ1 and RQ2, I have considered the top 10 python repositories from GitHub based on the number of stars. In RQ1, I grouped the developer comments w.r.to the date, comments are written to determine the day of the week for getting

[1]https://pygithub.readthedocs.io/en/latest/

the date of the creation of the comment. In RQ2, I have considered only comments posted by the contributors. Then, I sorted those comments based on the date of posting those comments. Next, I grouped those comments based on time duration from the date of the creation of the comment (e.g., 1 month, 2 months, 3 months, 6 months, 1 year etc.).These repositories contains approximately 240,000 Issue comments, 150,000 pull requests comments and 14,613 commit comments. These Repositories are given in Table 3.1.

Table 3.1: GitHub Python Repositories for RQ1 and RQ2.

Repository	Issues	Pull Requests	Commits
pytorch/fairseq	2382	46	1752
keras-team/keras	10414	11	5398
scrapy/scrapy	2372	307	8604
AdguardTeam/AdguardFilters	70788	43	79988
freeCodeCamp/freeCodeCamp	15097	31	26899
scikit-learn/scikit-learn	8249	738	26502
home-assistant/core	18842	318	33036
ageitgey/face_recognition	1095	12	211
apache/superset	5653	87	6732
tensorflow/tensorflow	32931	188	112600

For RQ3, I have used the comments posted on the top 3 GitHub repositories for each of the programming languages (such as Python, Java, C, C++ and Dot Net) based on the number of stars. After extracting issue, pull request and commit comments, I grouped each of these comments based on the programming languages used to develop the projects. I have approximately 64,000 issue comments, 23,000 pull request comments and 170,000 commit comments. These Repositories are given in Table 3.2.

The Dataset contains 5 fields

1. repository: the name of repository.
2. user: the user that written the comment.
3. text: the text of the comment.
4. cdate: the date of creation of comment.
5. edate: the date of closing of issue (only in closed issues).

Table 3.2: GitHub Python Repositories for RQ3.

Language	Repository	Issues	Pull Requests	Commits
Python	pytorch/fairseq	2382	46	1752
Python	keras-team/keras	10414	11	5398
Python	scrapy/scrapy	2372	307	8604
Java	ReactiveX/RxJava	70788	43	79988
Java	google/guava	15097	31	26899
Java	spring-projects/spring-boot	8249	738	26502
C	Genymobile/scrcpy	18842	318	33036
C	obsproject/obs-studio	1095	12	211
C	git/git	5653	87	6732
C++	x64dbg/x64dbg	1974	629	4651
C++	tensorflow/tensorflow	484	593	1393
C++	tensorflow/tensorflow	728	755	649
DotNet	tensorflow/tensorflow	7179	2108	26814
DotNet	tensorflow/tensorflow	4110	1350	3186
DotNet	tensorflow/tensorflow	75	923	1450

3.2 DataSet for Comparison of Sentiment Analysis Algorithms

For RQ4, comparison of Sentiment Analysis Algorithms, I have used Sentiment140 [11] dataset. Sentiment140 was created by Computer Science graduate students at Stanford University. It contains 1,600,000 tweets extracted using twitter API. The tweets are annotated (0 = negative, 4 = positive) and they can be used to detect sentiment. It contains 6 fields

- target: the polarity of tweet.
- ids: the id of the tweet.
- date: the date of the tweet.
- flag: the query.
- user: the user that tweeted.
- text: the text of the tweet.

CHAPTER 4
Sentiment Analysis Process

Steps performed in Sentiment Analysis are

4.1 Data Pre-Processing

In this section, I have described the steps for performing sentiment analysis. The first step of sentiment analysis is data pre-processing. The data obtained from GitHub is in raw format. As the texts with special characters such as emoticons, punctuations, etc do not contribute to Sentiment Analysis, hence need to remove those before processing the data. The data pre-processing can be done in steps as shown in Fig. 4.1.

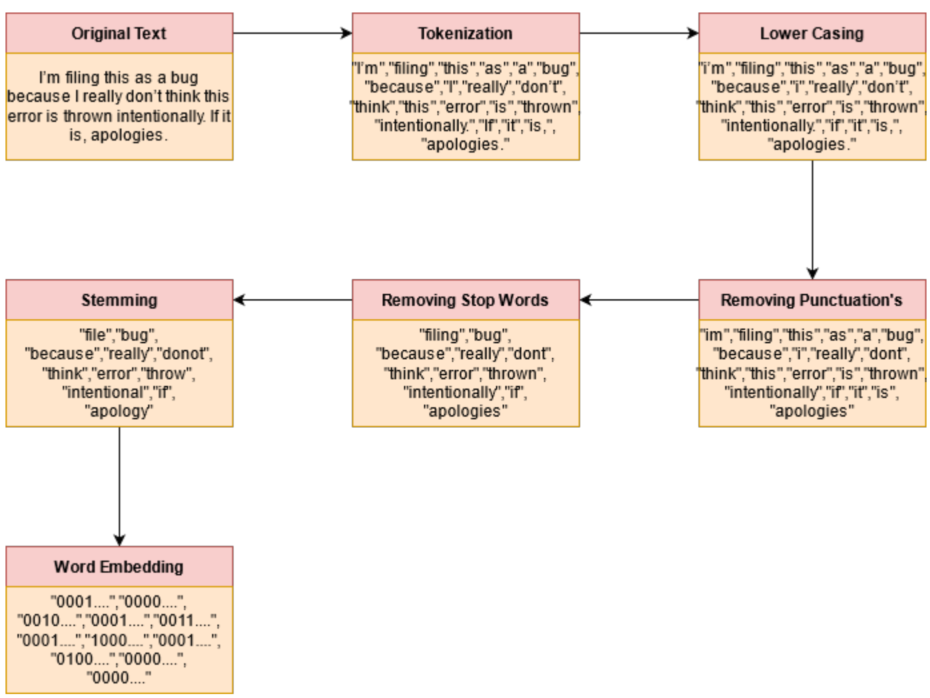

Figure 4.1: Steps for Pre-Processing.

1. Tokenization - It is the process of converting a piece of text into tokens. It is the task of chopping it up into pieces, called tokens. Tokens are the building blocks of Natural Language common ways of processing raw text happen at token level. I have used functions of list to tokenise the text [22].

2. Lower Casing - Converting the tokens into lower case letters. I converted all texts into lowercase so that the processing algorithm do not recognize capital letters and lowercase letters separately. I have used lower method of string in python for lower casing the text [19].

3. Removing punctuation's - It help us to optimize text so that processing algorithm works more efficiently because punctuation's do not add more value to sentiment data. I have removed punctuation's from text by using punctuation variable from string in python [19].

4. Removing Stop Words - Stop words are the words that does not contribute in sentiment analysis. These stop words are set of commonly used words that carry very little information. Stop words such as I, am, the, etc. I have used nltk.corpus[1] library to remove stop words from text [19].

5. Stemming - It is process of reducing a word to its root word by removing affixes and suffixes. Some examples of stemming are like for word "confirmed" stemmed word is "confirm". I have used WordNetLemmatizer from nltk.stem[2] library to perform word stemming [19].

6. Word Embedding - It is the representation of words in form of a vector that encodes the value of the word. I have used Word2Vec [32] for Word Embedding which allows us to represent each word in unique vectors and represent relationship between them. I have used Word Embedding for Machine Learning Algorithms such as SVM, CNN and LSTM.

4.2 Data Processing

4.2.1 For Sentiment Analysis on GitHub

After pre-processing the data and making it fit for our sentiment analysis algorithm, I have used different classifiers and an algorithm to classify our text into

[1]https://www.nltk.org/api/nltk.corpus.html
[2]https://www.nltk.org/api/nltk.stem.html

sentiments.

Classification of text in RQ1, RQ2 and RQ3 into sentiments is done using *SentiStrength*[3] a lexical extraction tool implemented in python to classify our text into sentiment scores. *SentiStrength* is a sentiment analysis tool used to convert texts into sentiment scores. I have chosen this tool because it has high accuracy rates as reported by previous studies [13][34][6]. *SentiStrength* assigns a score to each word that conveys an emotion. Words with negative emotions are given scores between -1 and -5 whereas words with positive emotions are given scores between 1 and 5. Then the word scores are summarised to generate a pair of values known as the senti-score. The first value of the senti-score is a positive score and the second one is a negative score. To find the final sentiment score, I take the sum of both values of the senti-score. A positive-sum represents positive sentiment, 0 represents neutral sentiment and a negative value represents negative sentiment.

Consider the following example of a commit log: "I'm filing this as a bug because I really don't think this error is thrown intentionally. If it is, apologies". *SentiStrength* provides us scores for each word and the sentence as:

"`file[0] bug[-1] because[0] really[0] donot[0] think[0] error[-1] throw[0] intentional[0] If[0] apology[0] [[Sentence = -3,1 = word max 1-5]]`".

The positive rating is 1 on the scale of 1 to 5 and the negative rating is -2 on the scale of -1 to -5. So the sentence comes out to be negative as when I take sum 1 and -2, it comes out as -1, which signifies negative sentiment. Here, I have taken the maximum positive and negative scores. This signifies the polarity of the commit log. The steps for analysing data using *SentiStrength* is shown in Fig. 4.2 [13][34][26].

4.2.2 For Comparison of Sentiment Analysis Algorithms

After pre-processing of data and making it fit for our sentiment analysis algorithm I have used different classifiers and algorithm to classify our text into sentiments [18] [25] [31].

[3] http://sentistrength.wlv.ac.uk/

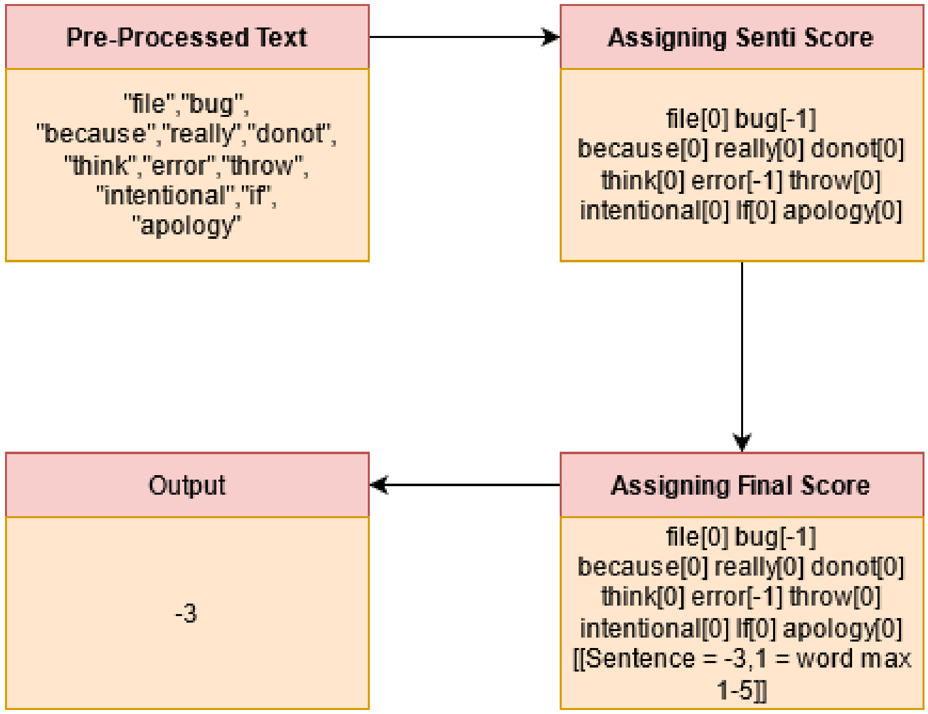

Figure 4.2: Data processing using *SentiStrength*.

Natural Language Toolkit (NLTK)

The main concept behind NLP is to analyze the dataset by accepting the dataset and reading the data. I have to read the data and compare it with dictionary of words to check if word is already defined in dictionary or not. NLP has two main approaches [23].

- Rule-Based Approach – It uses supervised learning algorithm which is based on dictionary of words and rules.
- Machine Learning Based Approach – It uses unsupervised data analysis technique. It uses training data as trained data to generate our model.

Decision Tree

It is a classifier model in which each node of the tree represents a test on attribute and its children represents the outcome. It is a supervised classification model that uses labels to form decision tree. The division of data is done recursively until the leaf node contain certain minimum numbers of records which are used for classification [24].

Random Forest

Random Forest is an ensemble technique which is used for classification and regression problems. It is a Bagging Problem. It includes Bootstrapping and Aggregation. Bootstrapping is selecting random samples and creating Decision Trees. Aggregation is the voting of classifier by checking each Decision Tree and noting the maximum frequency result [24].

Support Vector Machine (SVM)

SVM is a non-probabilistic binary linear classifier. It is sparse in nature so it is ideally suited for text classification. For a training set of points (xi,yi) where x is the feature vector and y is the class, I want to find the maximum-margin hyper-plane that divides the points with yi = 1 and yi = - 1. The equation of hyper-plane is

$$w^T.x + b = 0 \tag{4.1}$$

where $x, w \in R^n, b \in R$.
I need to maximize the margin[24]

$$min_{w,b,\xi_i} \frac{1}{2} w^T w + \sum_{i=1}^{l} \xi_i \tag{4.2}$$

Subject to,

$$y_i(w^T x_i + b) \geq 1 = \xi_i, \tag{4.3}$$

$\xi_i \geq 0, i = 1, 2, 3, ..., l$
Here l is the number of training point
Kernels used in SVM [24] [12] are

- Linear Kernel

$$K(x_i, x_j) = x_i^T x_j \tag{4.4}$$

- Polynomial Kernel

$$k(x_i, x_j) = (x_i^T * x_j + 1)^d \tag{4.5}$$

- Gaussian Radial Basis Kernel (RBF)

$$k(x_i, x_j) = exp(-\gamma||x_i - x_j||^2) \tag{4.6}$$

- Sigmoid Kernel

$$k(x,y) = tanh(\alpha x^T y + c) \tag{4.7}$$

Convolutional Neural Network

A Convolutional Neural Network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance to various aspects and be able to differentiate one from other. A Convolutional layer has a number of filters or kernels which it learns to extract specific features from the data. I used temporal convolution which is suitable for analyzing sequential data like text. I have used keras with Tensorflow to implement CNN model. From Word Embedding I received vocabulary of top 90000 tweets represented by a 200 dimensional vector. I have trained 2 CNN based models [8] [35] [20].

- 3 Layer Convolutional Neural Network - First Layer is a Dropout Layer with parameter 0.4 which drops out 40% of the data from Word Embedding Layer. Then I have used a CNN Layer with 600 filters of kernel size 3 with zero padding with relu activation function. After Convolutional layer I have applied a max pool layer. Similarly 2 more CNN Layers with 300 and 150 filters respectively. Then I have used Flatten Layer to flatten the vectors. Then I have used a dense layer with parameter 600 which have a dropout layer with parameter 0.5 which drops out 50% of the output. Then I have used dense layer with sigmoid activation function to get the output of the model. 3-Layer Convolutional model is given in Figure 4.3.

- 4 Layer Convolutional Neural Network - First Layer is a Dropout Layer with parameter 0.4 which drops out 40% of the data from Word Embedding Layer. Then I have used a CNN Layer with 600 filters of kernel size 3 with zero padding with relu activation function. After Convolutional layer I have applied a max pool layer. Similarly 3 more CNN Layers with 300, 150 and 75 filters respectively. Then I have used Flatten Layer to flatten the vectors. Then I have used a dense layer with parameter 600 which have a dropout layer with parameter 0.5 which drops out 50% of the output. Then I have used dense layer with sigmoid activation function to get the output of the model. And 4-Layer Convolution model is given in Figure 4.4.

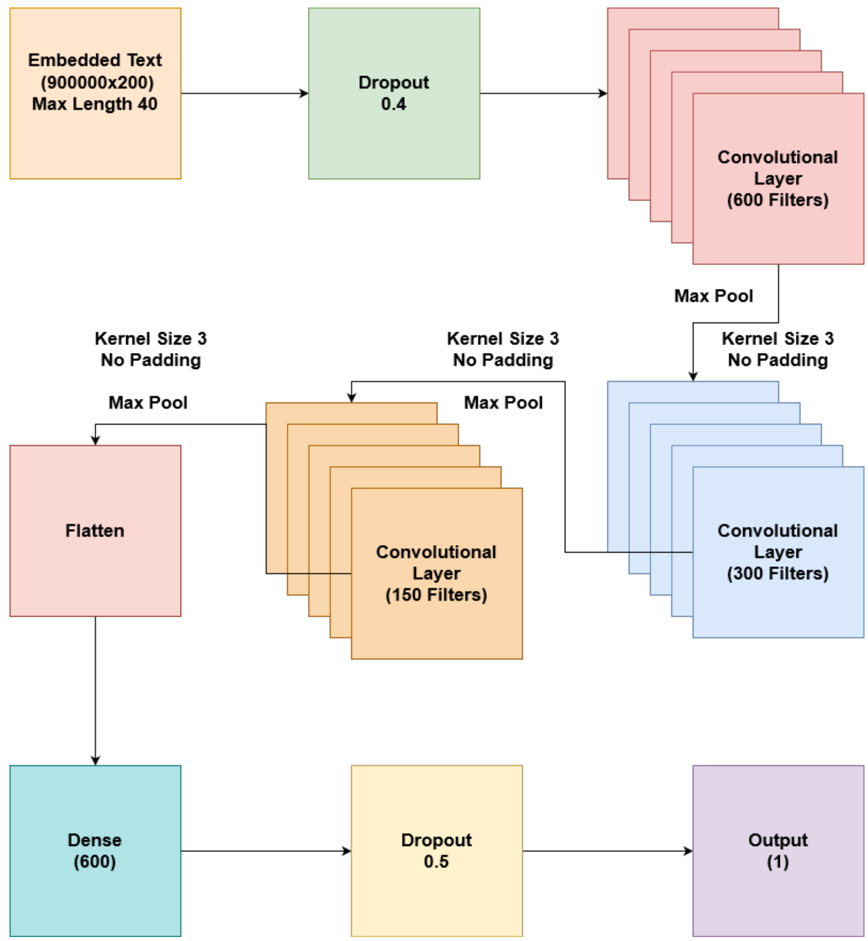

Figure 4.3: Convolution 3 Layer Model.

Long Short Term Memory

It is an improved RNN architecture used in field of deep learning. LSTM has feedback connections. It can also process entire sequence of data points. The core concept of LSTM's are the cell states and its various gates. The cell states work like a conveyer belt. It runs straight chain with only some minor linear interactions. The LSTM has ability to remove or add information to cell states. Gates are the way to let information through. They are composed of a sigmoid layer and a point wise multiplication operation. The gates can learn what information to keep and what to forget during training [29] [30] [21].

I have used keras with Tensorflow to implement LSTM model. From Word Embedding I received vocabulary of top 90000 tweets represented be a 200 dimentional vector. First Layer is a Dropout Layer with parameter 0.4 which drops out 40% of the data from Word Embedding Layer. Then I have used a LSTM layer with parameter 128 with relu activation function. Then I have used Flatten Layer

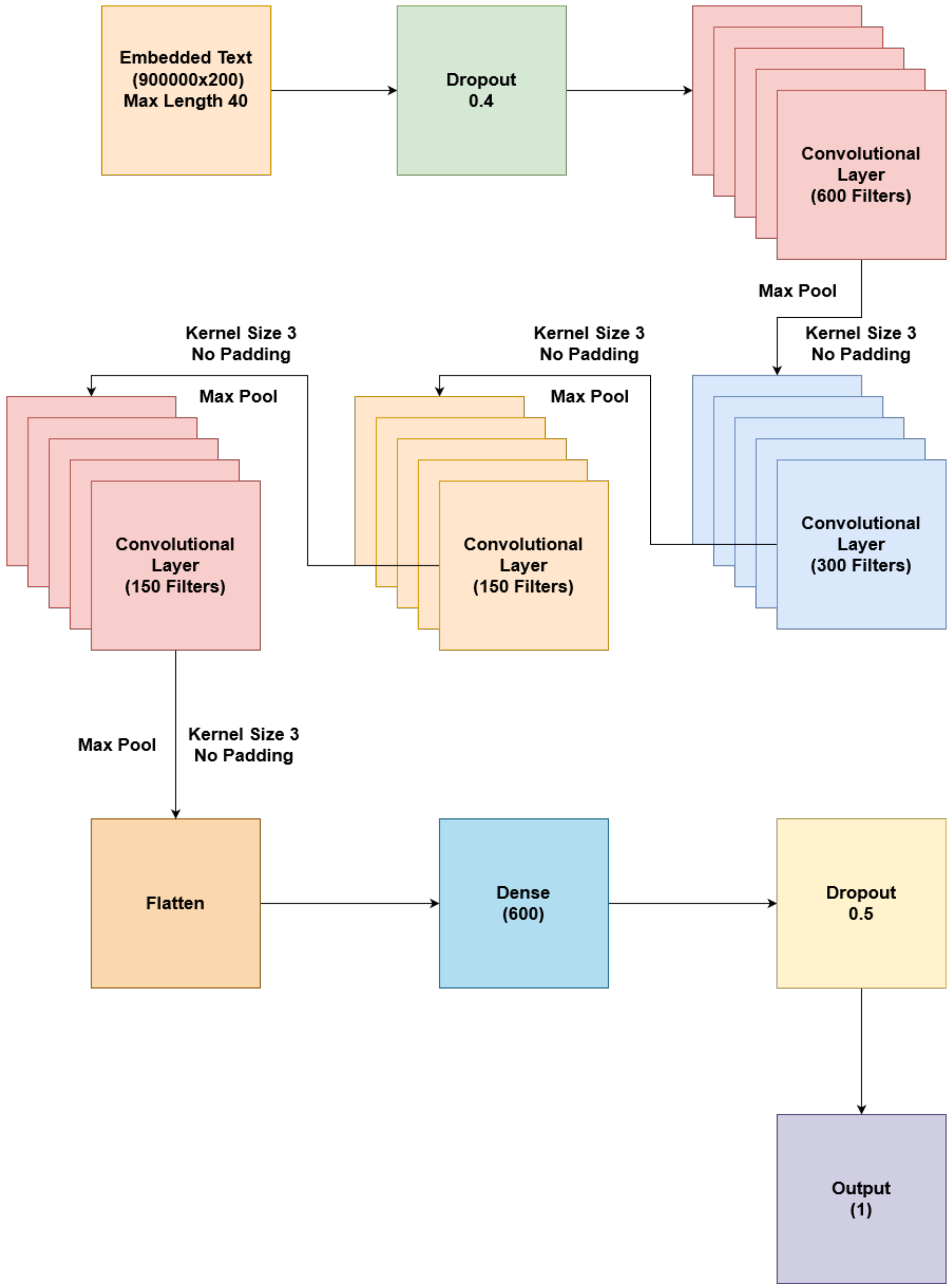

Figure 4.4: Convolution 4 Layer Model.

to flatten the vectors. Then I have used a dense layer with parameter 600 which have a dropout layer with parameter 0.5 which drops out 50% of the output. Then I have used dense layer with sigmoid activation function to get the output of the model. The LSTM Model is given in Figure 4.5. Basic components of LSTM network model are given in figure 4.6.

- Sigmoid – It is the activation function which gives values ranging from 0 to

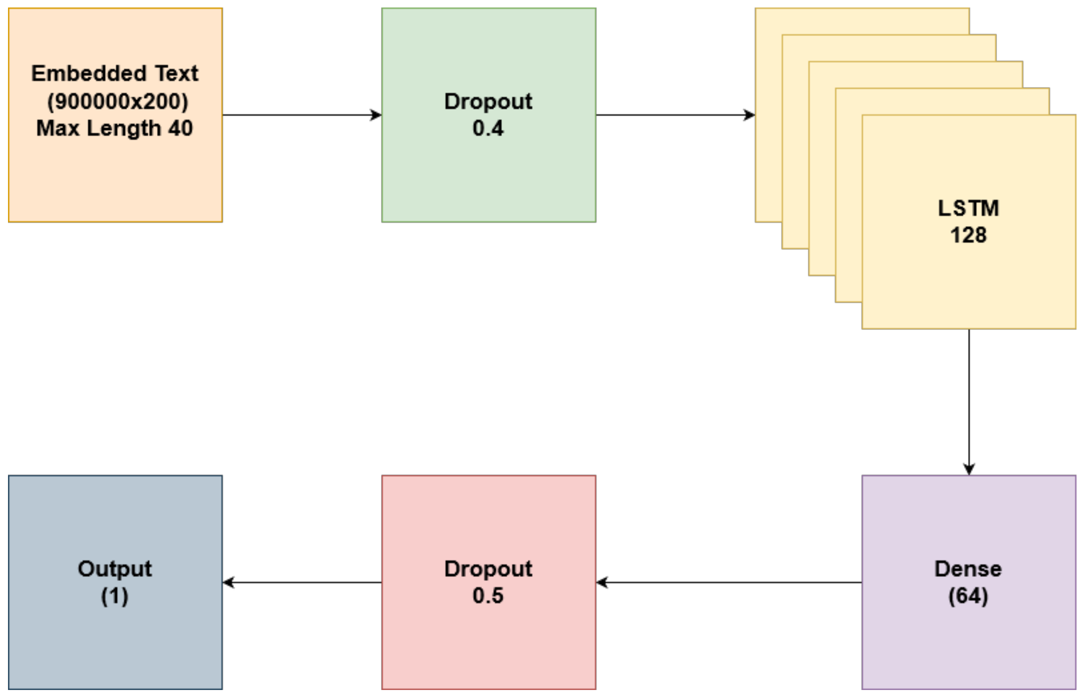

Figure 4.5: LSTM Model.

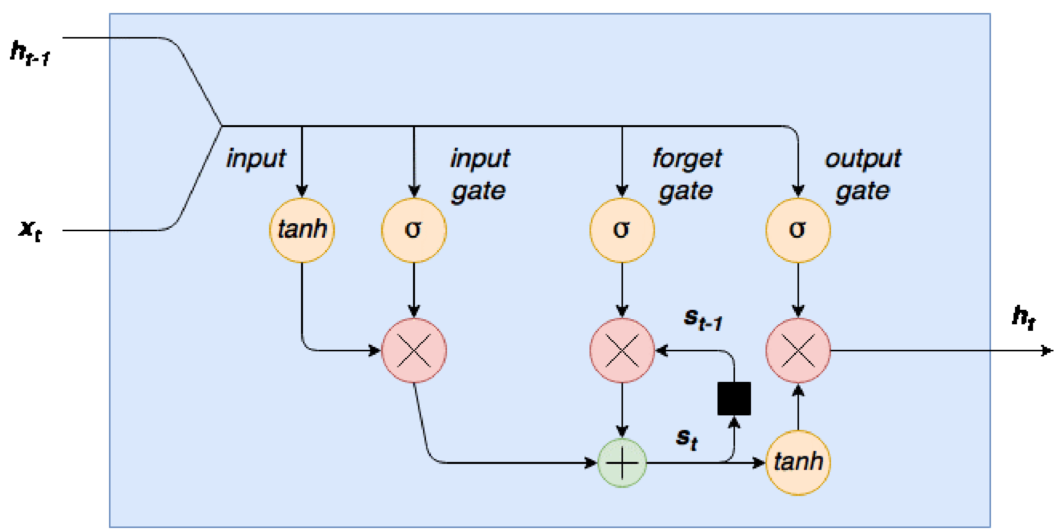

Figure 4.6: Basic Components of LSTM Model.

1.

- Forget gate – It decides whether to keep the information or to forget the information based on output of sigmoid function.
- Input gate – It is used to update cell state. It generates values between 0 and 1 and provides importance of data.
- Cell State – I can calculate cell state by multiplying cell state by forget vector.

- Output gate – It determines the next hidden state for prediction.

Ensemble Techniques

In the quest to improve accuracy I have developed a simple ensemble model. I have developed several models and used those models with Majority vote Ensemble to get the result which is favored by most of the models. I have developed 3 models for ensemble techniques. They are as follows [33].

- LSTM
- 3-Convolutional Neural Network
- 4-Convolutional Neural Network

I have used a majority vote ensemble to get the final sentimental score. In Majority vote I can give preference to the option which is coming from maximum number of model. For example if I have 3 model ensemble technique in which 2 models are giving True and one model is giving False. So I choose True as majority of the models are giving True as an answer. Ensemble model is given in Figure 4.7.

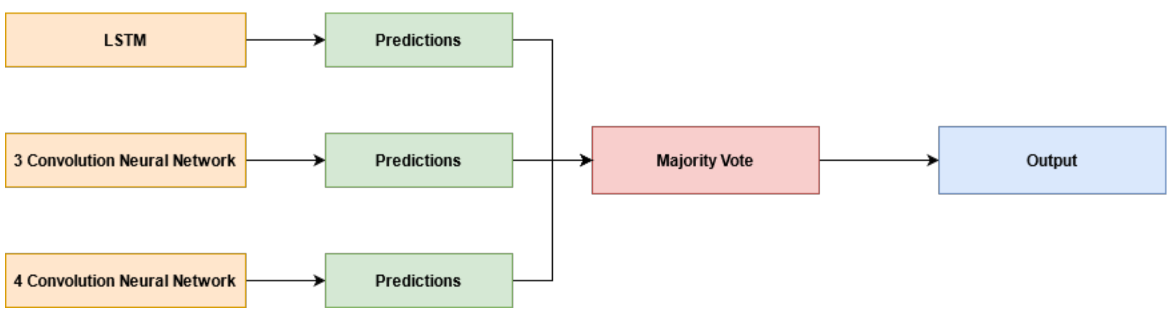

Figure 4.7: Ensemble Model.

CHAPTER 5

Experimental Results and Analysis

In the following sections I present an analysis of the emotions expressed in issue comments, pull request comments and commit comments relation with additional factors such as day of the week, throughout the course of project, for different programming language. Also, comparison of different Sentiment Analysis algorithms.

5.1 Sentiment of Python Developer related to day of week

I have analyzed emotions present in issue comments, pull request comments as well as commit comments. I have analyzed a total of 10 projects which are written in python. The average score tended between -1 to 1 as most of the comments only describe technical aspects and no emotions or have a slight positive or negative factor in them. Fig. 5.1 and table 5.1 shows the sentiment of a python developer related to the day of the week on which they write the comment. Most comments, 85%, were written during the week, while 15% during the weekends. Developer's shows more negative sentiments at the starting of the week as I can see that comments written on Monday are 42.58% negative. As the week progresses the sentiment of developers shifts from negative to positive but as the weekend approaches the sentiment of developers shifts to negative as I can see the sentiment on Friday and Saturday are 45.75 % and 43.86% negative. The reason behind this can be as Monday is the first day of the week, developers experience more negative emotions as they have to work after the weekend also like the weekend approaches developers reflects more negative emotions [27][17].

I observe the same trend in the case of issue comments, the comments were writ-

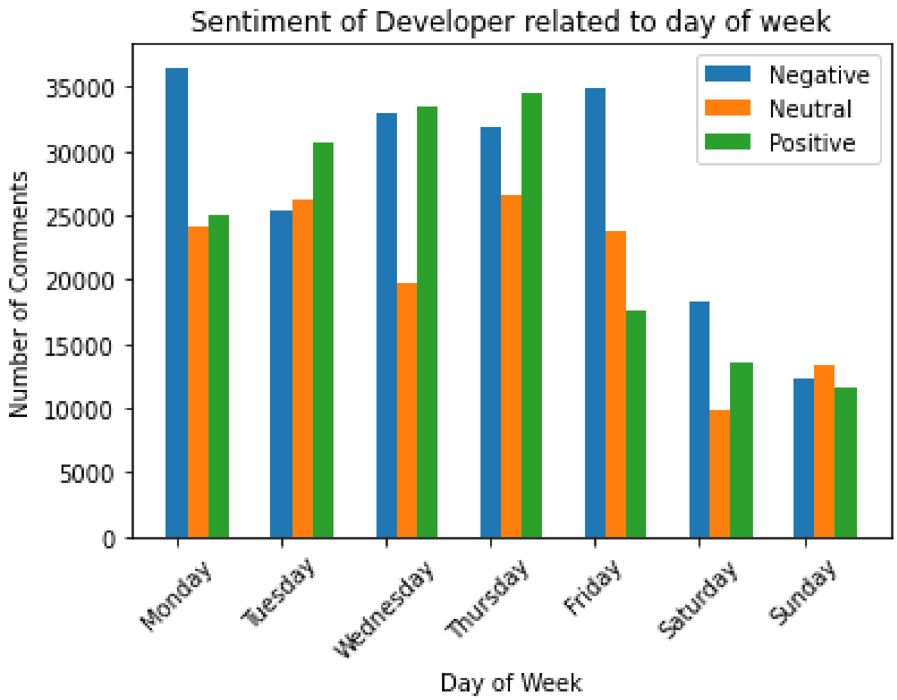

Figure 5.1: Sentiment of Developer with respect to Week Days.

Table 5.1: Sentiment of Developer with respect to Week Days.

Day of Week	**Negative**	**Neutral**	**Positive**
Monday	42.58	28.23	29.19
Tuesday	30.84	31.85	37.31
Wednesday	38.23	22.94	38.83
Thursday	34.29	28.60	37.11
Friday	45.78	31.09	23.13
Saturday	43.86	23.58	32.56
Sunday	33.03	35.99	30.98

ten on Monday, Friday and Saturday reflect more negative sentiments of developers as compared to other weekdays as shown in fig. 5.2 and table 5.2. Whereas comments written in pull requests have negative sentiments consistently throughout the weekdays as shown in fig. 5.3 and table 5.3. But in the case of comments written in commits, the sentiment of developers are positive throughout the week as shown in fig. 5.4 and table 5.4. The reason behind this can be the comments written in issues and pull requests signify errors or changes in source code, so comments written in issues and pull requests are more negative compared to comments written in commits.

Table 5.2: Sentiment of Developer with respect to Week Days in Issue Comments.

Day of Week	Negative	Neutral	Positive
Monday	40.46	31.68	27.86
Tuesday	27.12	37.81	35.07
Wednesday	36.58	25.92	37.50
Thursday	33.16	25.36	41.48
Friday	44.99	30.52	24.49
Saturday	42.00	24.90	33.10
Sunday	48.62	28.65	22.73

Table 5.3: Sentiment of Developer with respect to Week Days in Pull Request Comments.

Day of Week	Negative	Neutral	Positive
Monday	46.29	23.52	30.19
Tuesday	35.11	25.67	39.22
Wednesday	42.22	16.92	40.86
Thursday	37.53	32.99	29.48
Friday	47.76	31.25	20.99
Saturday	46.11	23.32	30.57
Sunday	27.70	40.31	31.99

Table 5.4: Sentiment of Developer with respect to Week Days in Commit Comments.

Day of Week	Negative	Neutral	Positive
Monday	27.48	23.66	48.86
Tuesday	25.99	28.15	45.86
Wednesday	16.29	36.03	47.68
Thursday	28.91	33.97	37.12
Friday	23.88	37.82	38.30
Saturday	19.40	26.87	53.73
Sunday	18.32	25.0	56.68

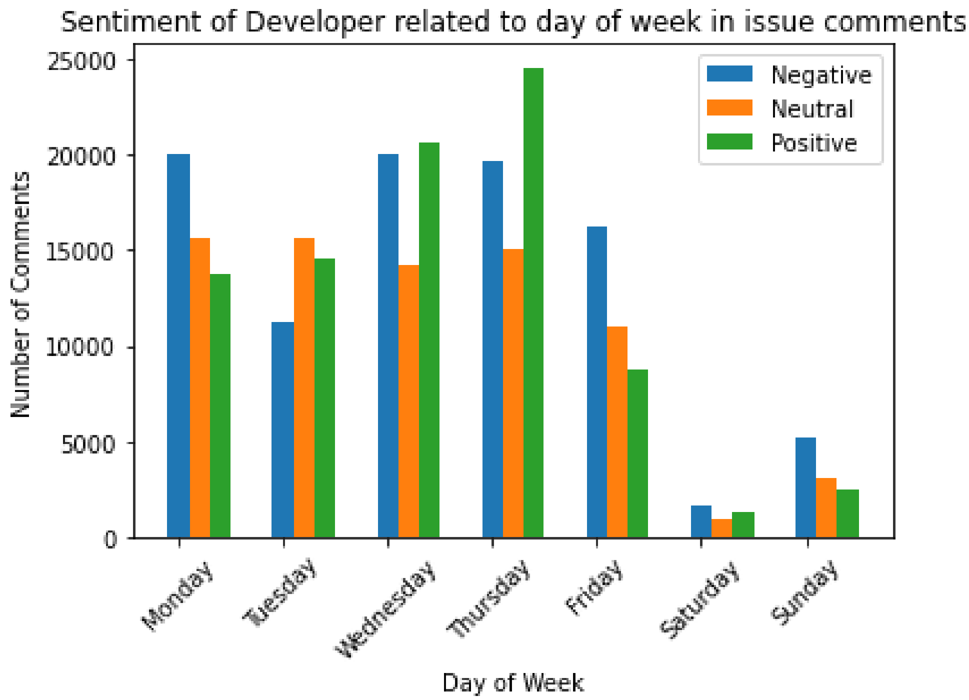

Figure 5.2: Sentiment of Developer with respect to Week Days in Issue Comments.

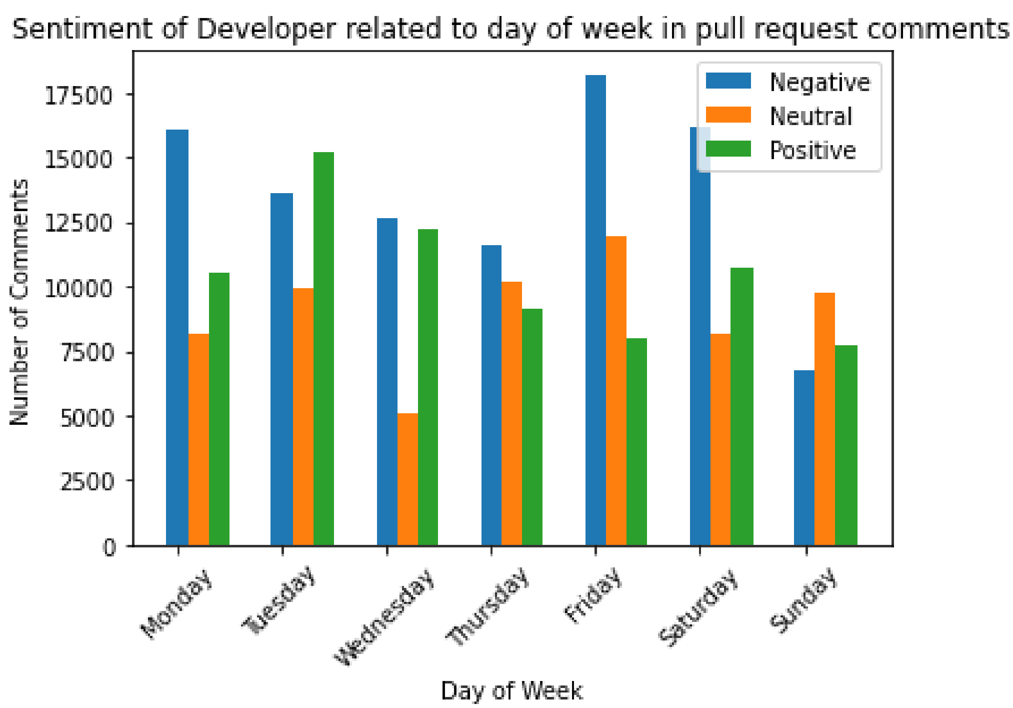

Figure 5.3: Sentiment of Developer with respect to Week Days in Pull Request Comments.

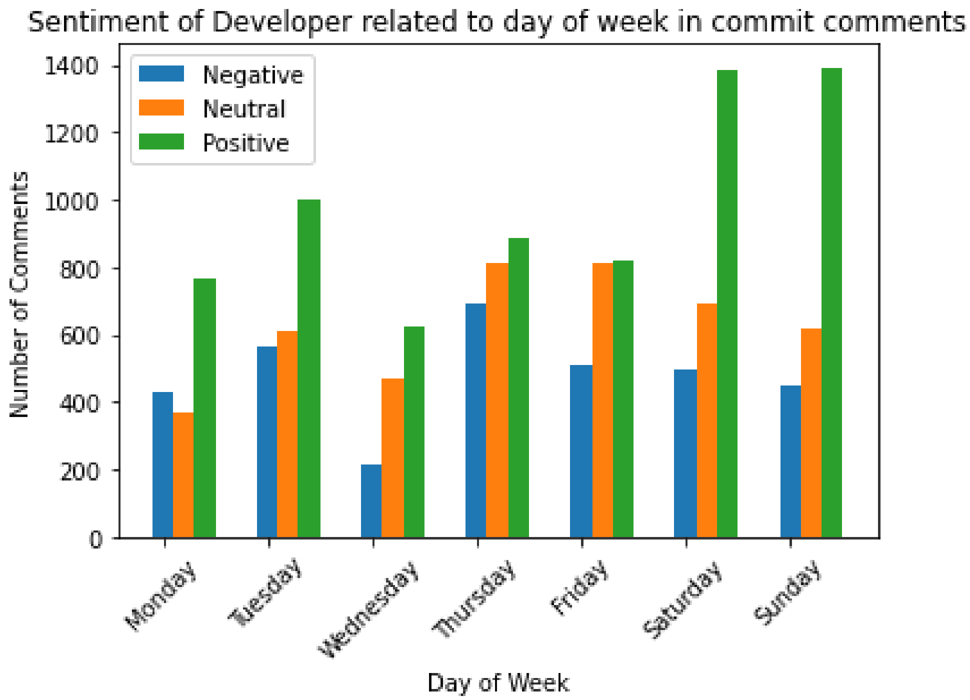

Figure 5.4: Sentiment of Developer with respect to Week Days in Commit Comments.

5.2 Sentiment of Contributors throughout the course of the project.

To study the relationship between the sentiments expressed by contributors in comments written in issues, pull requests and commits, and the time spent on the project I analyzed the data of 10 python projects. I manually sorted the comments by the date on which the comment is posted and grouped the data with respect to time spent from the creation of a project. I have analyzed the data for 1 month, 3 months, 6 months, 9 months, 1 year, 2 years and 5 years from the creation of the project. As shown in Fig. 5.5 and table 5.5 the sentiment of the developer is mostly negative throughout the project. The overall sentiment becomes negative as there are more issue comments and pull request comments which are negative in nature as compared to commit comments.

I observed that the sentiment of the contributor in issue comments are negative throughout the project as shown in fig 5.6 and table 5.6. As comments written in issues represent bugs or errors in code, hence these comments are more negative in nature. Also, comments written in pull requests have negative sentiment throughout the project as shown in fig 5.7 and table 5.7 as they signify

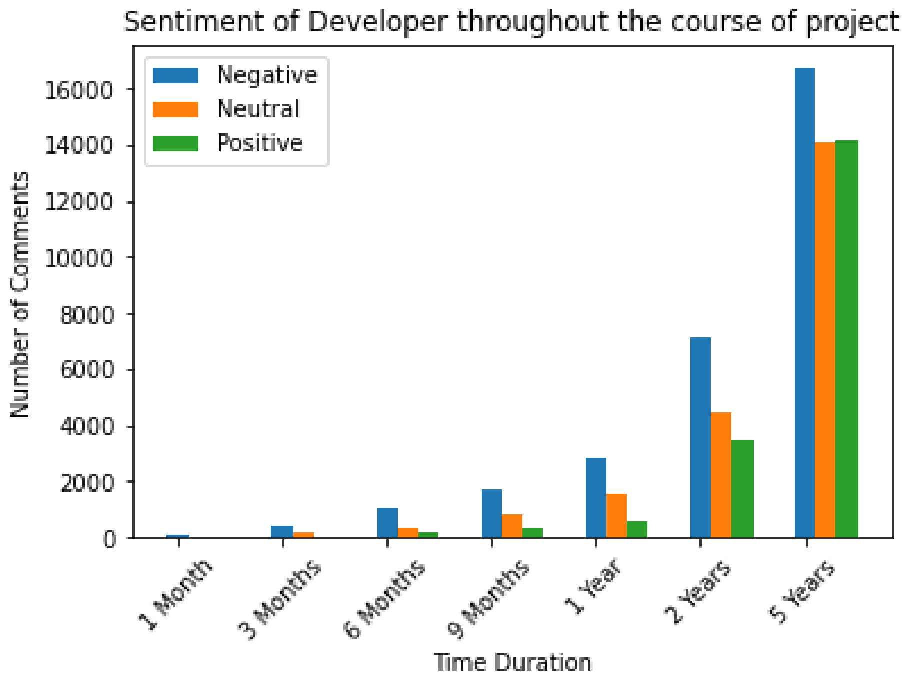

Figure 5.5: Sentiment of Developer throughout the course of the project.

Table 5.5: Sentiment of Developer throughout the course of the project.

Day of Week	Negative	Neutral	Positive
1 Month	52.94	36.02	11.04
3 Months	65.92	28.02	6.06
6 Months	66.49	21.31	12.20
9 Months	59.74	28.42	11.84
1 Year	57.49	31.53	10.98
2 Years	47.22	29.71	23.07
5 Years	37.17	31.32	31.51

code changes, other developers suggest changes in code through pull requests. But comments written in commits are positive in nature throughout the project as shown in fig 5.8 and table 5.8. Commit comments are written when a user commits the repository, i.e., they save the state of the repository so that a user can get versions of the project. So it consists of more positive comments as compared to issues and pulls requests.

Table 5.6: Sentiment of Developer throughout the course of the project in Issue Comments.

Day of Week	Negative	Neutral	Positive
1 Month	57.95	38.63	3.42
3 Months	73.60	23.59	2.81
6 Months	77.07	18.95	3.98
9 Months	64.89	28.86	6.25
1 Year	57.69	36.28	6.03
2 Years	43.84	34.19	21.97
5 Years	45.82	31.09	23.09

Table 5.7: Sentiment of Developer throughout the course of the project in Pull Request Comments.

Day of Week	Negative	Neutral	Positive
1 Month	47.50	32.50	20.00
3 Months	46.62	36.19	17.19
6 Months	46.49	26.49	27.02
9 Months	54.50	27.56	17.94
1 Year	60.57	25.68	13.75
2 Years	51.57	25.81	22.62
5 Years	40.01	26.90	33.09

Table 5.8: Sentiment of Developer throughout the course of the project in Commit Comments.

Day of Week	Negative	Neutral	Positive
1 Month	14.28	28.57	57.15
3 Months	12.00	52.00	36.00
6 Months	12.16	29.72	58.12
9 Months	15.68	28.43	55.89
1 Year	12.41	22.87	64.72
2 Years	15.52	28.05	56.43
5 Years	19.00	23.16	57.84

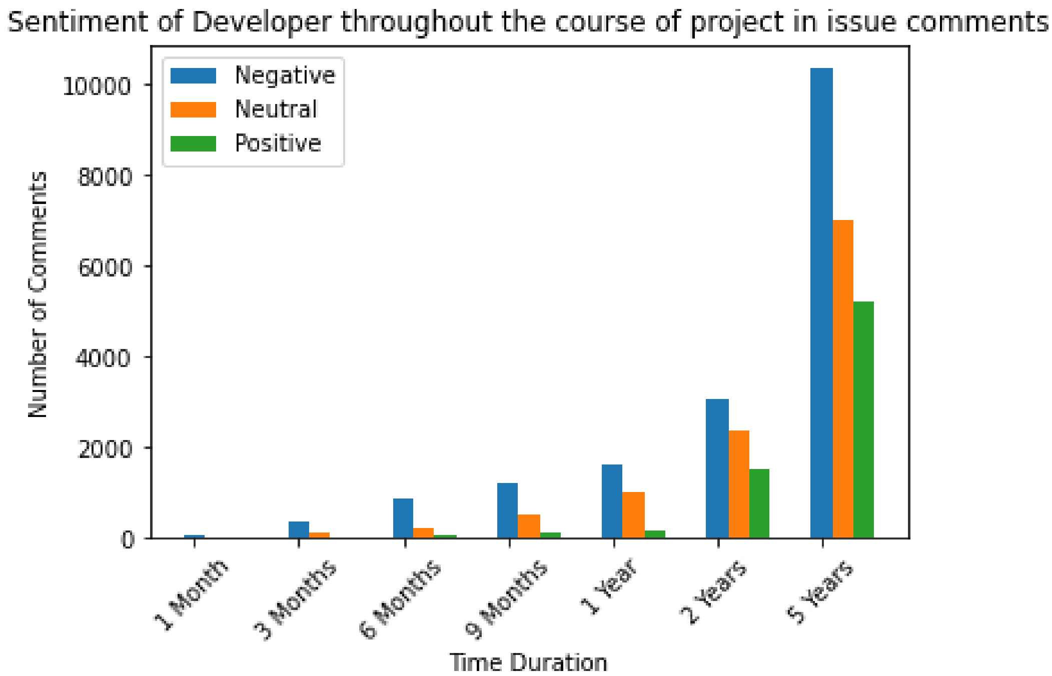

Figure 5.6: Sentiment of Developer throughout the course of the project in Issue Comments.

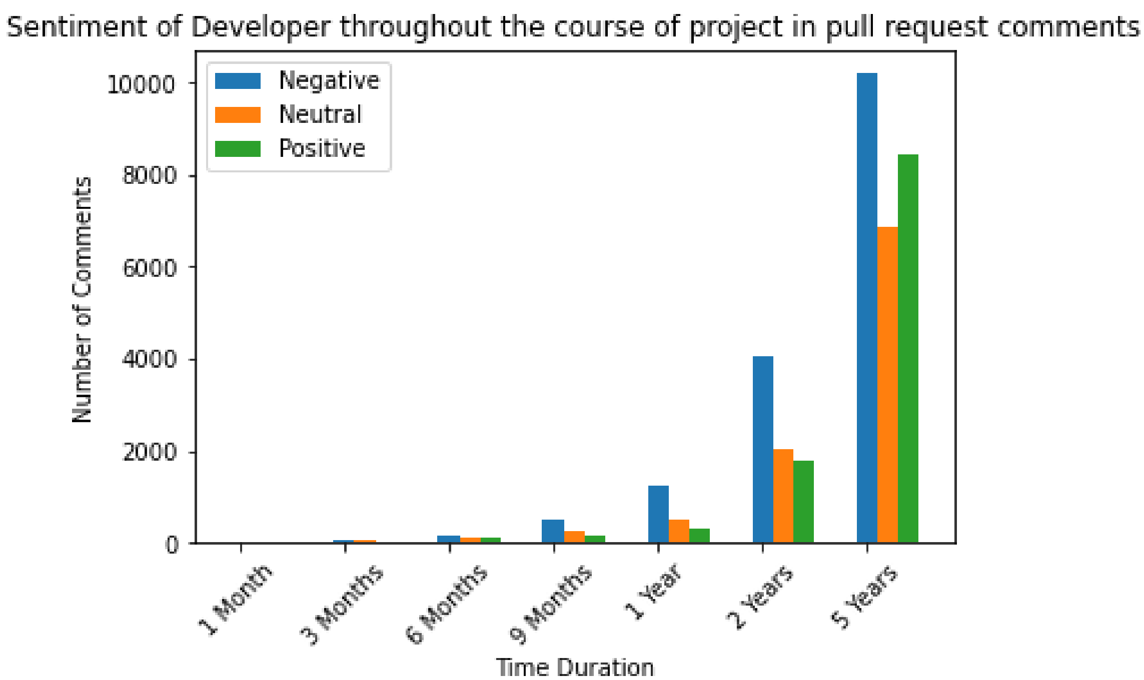

Figure 5.7: Sentiment of Developer throughout the course of the project in Pull Request Comments.

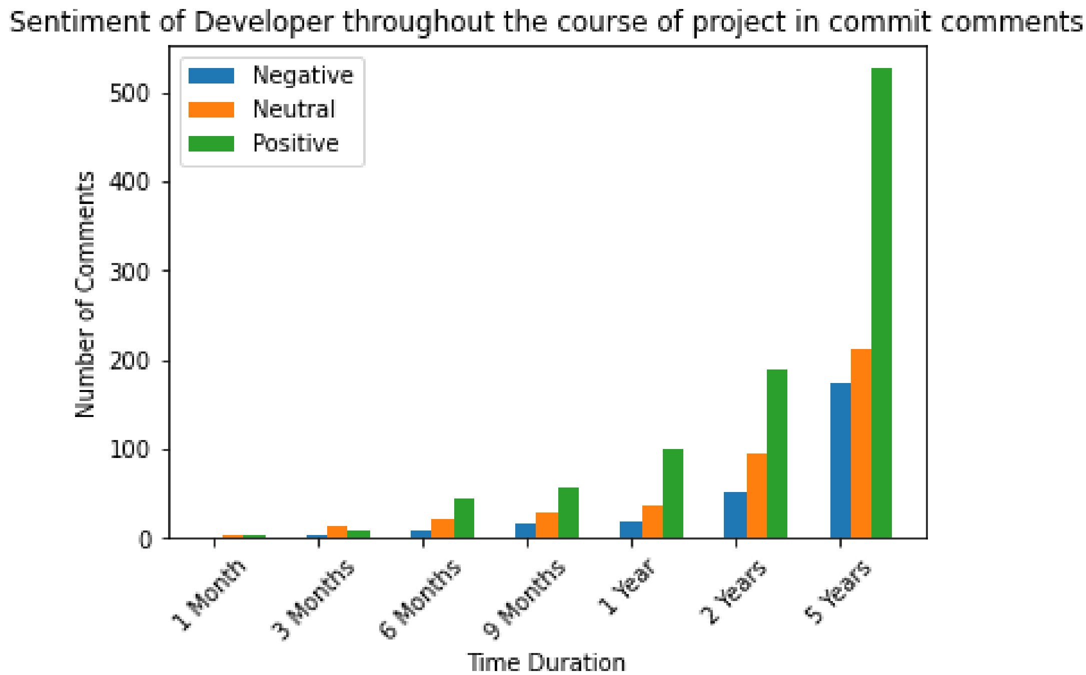

Figure 5.8: Sentiment of Developer throughout the course of the project in Commit Comments.

5.3 Sentiment of developers across different programming languages.

The repositories in this thesis were implemented in the 5 most used programming languages. These are Python, Java, C, C++ and DotNet. I have chosen the top 3 repositories for each language. This study is performed on the issue comments, pull request comments and the commit comments of the particular libraries. Fig. 5.9 and table 5.9 shows the comparison of different sentiment scores for each language. I have observed that the Sentiment of python and java developers have more positive sentiments 37.97% and 36.85% respectively as compared to C and C++ developers 29.8% and 27.55% respectively. Whereas DotNet developers have neutral sentiments having an almost equal number of positive and negative comments. I observe these trends as C and C++ are the base of languages and are harder to learn and require a deeper understanding to program and implement, whereas python is easy to learn and implement.

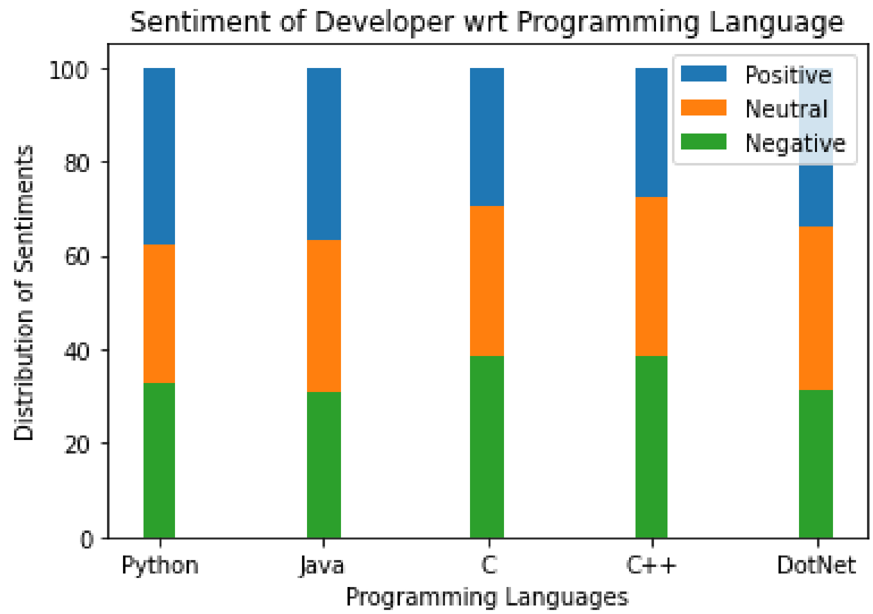

Figure 5.9: Sentiment of Developer with respect to Programming Language.

Table 5.9: Sentiment of Developer with respect to Programming Language.

Programming Language	**Negative**	**Neutral**	**Positive**
Python	32.72	29.31	37.97
Java	31.02	32.13	36.85
C	38.75	31.45	29.80
C++	38.68	33.76	27.56
DotNet	31.18	34.98	33.84

5.4 Comparison of Sentiment Analysis Algorithms

The provided data was a mixture of words, emoticons, URLs, hashtags, user mentions and symbols. Before training I need to pre-process the data to make it suitable for feeding it into our models. I have implemented several Machine Learning Algorithms like NLTK, Decision Trees, Random Forest, Support Vector Machine, Convolutional Neural Network, Long Short Term Memory and Ensemble Techniques. My dataset contains 800,000 positive and 800,000 negative tweets in text format. So for machine learning algorithms I have converted them in suitable format by using Word2Vec as a Word Embedding Technique [7]. I have Found that NLTK has the minimum accuracy of 61.2% whereas ensemble model has highest accuracy of 83.88%. Also in SVM, RBF Kernel has the most accuracy among all the Kernels as shown in figure 5.10.

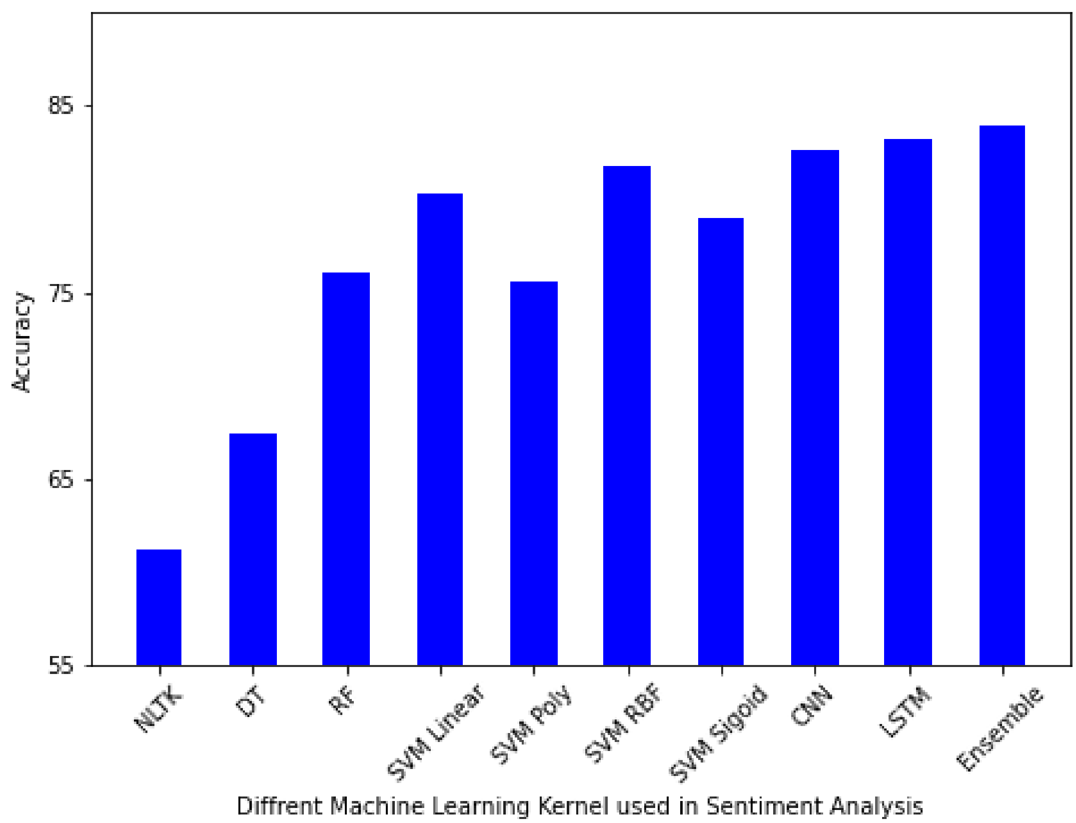

Figure 5.10: Comparison of Sentiment Analysis Algorithms.

CHAPTER 6

Discussions

This study suggests that comments logs that are submitted on Monday, Friday and Saturday have more negative sentiment than comments submitted on any other day of the week. This is also reported by E. Guzman, et al [14] in their study during sentiment analysis of commit logs. They report that commit logs submitted on Monday have a more negative emotion than commits submitted on any other working day of the week. From this analysis, this thesis can help developers to get a more positive response from contributors. My results also suggest that comments written in issue comments and pull request comments are more negative as compared to comments written in commit comments. These results provide a strong correlation between the course of the project and the sentiment carried by the comments. However, more work is needed in this area to clearly understand how this relationship impacts the project as a whole. I have also found that comments written in python and java repositories are more positive as compared to programming languages such as C and C++. My study was conducted on the top 3 GitHub repositories based on a number of stars. However, the projects were not related in terms of the function they are performing so we cannot say that particular programming language contains more positive sentiment comments.

This thesis also suggests that deep learning algorithms such as CNN, LSTM give more accurate results than non machine learning algorithms for sentiment analysis. Also if we apply ensemble techniques we can get even more accurate results but it takes more time to train.

CHAPTER 7

Conclusions and Future Work

This study deals with the sentiment of developer with respect to many factors such as sentiment of developer on the days of week. I have also analyzed the behavior of developer throughout the course of the project. On the basis of my research, I can conclude that developer sentiment on Mondays are much more negative then other week days because it is the first day of the week. Also, comments written on weekends are comparatively low as most people see weekends as resting days so there is comparatively low comments on weekends. I can also conclude that sentiment in issues are mostly negative as the developer reacts negatively to bugs whereas commit comments are mostly positive in nature. This trend can also be seen in the project's life cycle. The issue comments are more negative and commit comments are positive throughout the course of the project. This Thesis also talks about the sentiment of developers with respect to the programming languages they are using. I can conclude from our results, developers comment on python repositories are mostly positive and comments on programming languages like C and C++ are more negative in comparison. I have also compared different sentiment analysis algorithms which one can use to classify comments. From my results I can conclude that the ensemble techniques give us more accuracy but they need more time to train as it have to train many models for applying majority vote ensemble.

For future work I can extend the application of my sentiment analysis algorithms for example sentiment analysis algorithms to gain knowledge about reviews from twitter, I can also use it for research purpose like developing a project which can get the sentiment of any topic from twitter and can review the topic. For Example, COVID-19, I can gain knowledge about the sentiment of the world by analyzing tweets. I can also implement Sentiment Analysis algorithms so that they can work with different languages.

References

[1] Q. T. Ain, M. Ali, A. Riaz, A. Noureen, M. Kamran, B. Hayat, and A. Rehman. Sentiment analysis using deep learning techniques: A review. *International Journal of Advanced Computer Science and Applications*, 8(6), 2017.

[2] G. Beigi, X. Hu, R. Maciejewski, and H. Liu. *An Overview of Sentiment Analysis in Social Media and Its Applications in Disaster Relief*, pages 313–340. Springer International Publishing, Cham, 2016.

[3] Z. Chen, Y. Cao, X. Lu, Q. Mei, and X. Liu. Sentimoji: An emoji-powered learning approach for sentiment analysis in software engineering. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2019, page 841–852, New York, NY, USA, 2019. Association for Computing Machinery.

[4] F. Chiarello, A. Bonaccorsi, and G. Fantoni. Technical sentiment analysis. measuring advantages and drawbacks of new products using social media. *Computers in Industry*, 123:103299, 2020.

[5] S. Das, R. K. Behera, M. kumar, and S. K. Rath. Real-time sentiment analysis of twitter streaming data for stock prediction. *Procedia Computer Science*, 132:956–964, 2018. International Conference on Computational Intelligence and Data Science.

[6] G. Destefanis, M. Ortu, D. Bowes, M. Marchesi, and R. Tonelli. On measuring affects of github issues' commenters. In *2018 IEEE/ACM 3rd International Workshop on Emotion Awareness in Software Engineering (SEmotion)*, pages 14–19, 2018.

[7] F. S. Dolianiti, D. Iakovakis, S. B. Dias, S. Hadjileontiadou, J. A. Diniz, and L. Hadjileontiadis. Sentiment analysis techniques and applications in education: A survey. In M. Tsitouridou, J. A. Diniz, and T. A. Mikropoulos, editors,

Technology and Innovation in Learning, Teaching and Education, pages 412–427, Cham, 2019. Springer International Publishing.

[8] C. dos Santos and M. Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78, Dublin, Ireland, Aug. 2014. Dublin City University and Association for Computational Linguistics.

[9] R. Feldman. Techniques and applications for sentiment analysis. *Commun. ACM*, 56(4):82–89, Apr. 2013.

[10] V. G and D. Chandrasekaran. Sentiment analysis and opinion mining: A survey. *Int J Adv Res Comput Sci Technol*, 2, 06 2012.

[11] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *Processing*, 150, 01 2009.

[12] M. Goyal, N. Gupta, A. Jain, and D. Kumari. Smart government e-services for indian railways using twitter. In D. K. Sharma, V. E. Balas, L. H. Son, R. Sharma, and K. Cengiz, editors, *Micro-Electronics and Telecommunication Engineering*, pages 721–731, Singapore, 2020. Springer Singapore.

[13] E. Guzman, D. Azócar, and Y. Li. Sentiment analysis of commit comments in github: An empirical study. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, page 352–355, New York, NY, USA, 2014. Association for Computing Machinery.

[14] E. Guzman, D. Azócar, and Y. Li. Sentiment analysis of commit comments in github: An empirical study. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, page 352–355, New York, NY, USA, 2014. Association for Computing Machinery.

[15] S. F. Huq, A. Z. Sadiq, and K. Sakib. Understanding the effect of developer sentiment on fix-inducing changes: An exploratory study on github pull requests. In *2019 26th Asia-Pacific Software Engineering Conference (APSEC)*, pages 514–521, 2019.

[16] S. F. Huq, A. Z. Sadiq, and K. Sakib. Is developer sentiment related to software bugs: An exploratory study on github commits. In *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 527–531, 2020.

[17] N. Imtiaz, J. Middleton, P. Girouard, and E. Murphy-Hill. Sentiment and politeness analysis tools on developer discussions are unreliable, but so are people. In *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering*, SEmotion '18, page 55–61, New York, NY, USA, 2018. Association for Computing Machinery.

[18] R. Jongeling, S. Datta, and A. Serebrenik. Choosing your weapons: On sentiment analysis tools for software engineering research. In *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 531–535, 2015.

[19] D. Kawade. Sentiment analysis: Machine learning approach. *International Journal of Engineering and Technology*, 09:2183–2186, 06 2017.

[20] Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[21] L. Kurniasari and A. Setyanto. Sentiment analysis using recurrent neural network. *Journal of Physics: Conference Series*, 1471:012018, 02 2020.

[22] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[23] D. Pletea, B. Vasilescu, and A. Serebrenik. Security and emotion: Sentiment analysis of security discussions on github. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, page 348–351, New York, NY, USA, 2014. Association for Computing Machinery.

[24] M. A. Rahman and M. H. Seddiqui. Comparison of classical machine learning approaches on bangla textual emotion analysis, 2019.

[25] U. Ramakrishnan, R. Shankar, and G. Krishna. Sentiment analysis of twitter data: Based on user-behavior. 10:16291–16301, 01 2015.

[26] V. Sinha, A. Lazar, and B. Sharif. Analyzing developer sentiment in commit logs. In *Proceedings of the 13th International Conference on Mining Software*

Repositories, MSR '16, page 520–523, New York, NY, USA, 2016. Association for Computing Machinery.

[27] V. Sinha, A. Lazar, and B. Sharif. Analyzing developer sentiment in commit logs. In *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)*, pages 520–523, 2016.

[28] E. Skriptsova, E. Voronova, E. Danilova, and A. Bakhitova. Analysis of newcomers activity in communicative posts on github. In D. A. Alexandrov, A. V. Boukhanovsky, A. V. Chugunov, Y. Kabanov, O. Koltsova, and I. Musabirov, editors, *Digital Transformation and Global Society*, pages 452–460, Cham, 2019. Springer International Publishing.

[29] D. Tang, B. Qin, and T. Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics.

[30] A. Tholusuri, M. Anumala, B. Malapolu, and J. Lakshmi. Sentiment analysis using lstm. In *International Journal of Engineering and Advanced Technology (IJEAT)*, page 2249 –8958. Blue Eyes Intelligence Engineering & Sciences Publication, Sept. 2019.

[31] S. Varsha, S. Vijaya, and P. Apashabi. Sentiment analysis on twitter data. In *International Journal of Innovative Research in Advanced Engineering (IJIRAE), 2015)*, pages 2349–2163. International Journal of Innovative Research in Advanced Engineering (IJIRAE), Jan. 2015.

[32] J.-H. Wang, T.-W. Liu, X. Luo, and L. Wang. An LSTM approach to short text sentiment classification with word embeddings. In *Proceedings of the 30th Conference on Computational Linguistics and Speech Processing (ROCLING 2018)*, pages 214–223, Hsinchu, Taiwan, Oct. 2018. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).

[33] X. Wang, W. Jiang, and Z. Luo. Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2428–2437, Osaka, Japan, Dec. 2016. The COLING 2016 Organizing Committee.

[34] B. Yang, X. Wei, and C. Liu. Sentiments analysis in github repositories: An empirical study. In *2017 24th Asia-Pacific Software Engineering Conference Workshops (APSECW)*, pages 84–89, 2017.

[35] L. Zhang, S. Wang, and B. Liu. Deep learning for sentiment analysis: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4):e1253, 2018.