
Electronic Theses and Dissertations

2020

Comparative sentiment analysis of techniques for cyberbullying detection on twitter.

Kanam, Victor Otieno
Faculty of Information Technology
Strathmore University

Recommended Citation

Kanam, V. O. (2020). *Comparative sentiment analysis of techniques for cyberbullying detection on twitter* [Thesis, Strathmore University]. <http://hdl.handle.net/11071/12040>

Follow this and additional works at: <http://hdl.handle.net/11071/12040>

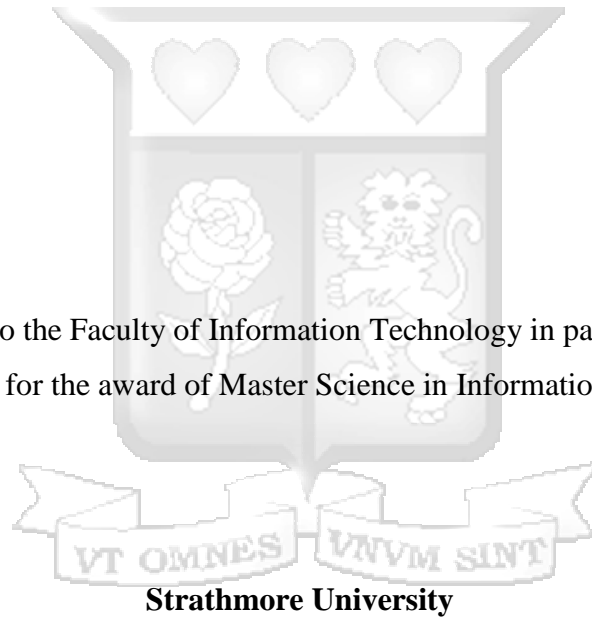
Comparative Sentiment Analysis of Techniques for Cyberbullying Detection on Twitter

By

VICTOR KANAM

113269

A Thesis Submitted to the Faculty of Information Technology in partial fulfillment of the requirements for the award of Master Science in Information Technology



April 2020

Declaration and Approval

I declare that the research has not been turned in to any academic institution or University for the award of a master's degree of science in IT.

Student's Name: **Victor Kanam**

Admission Number: 113269

Signature: _____

Date: _____

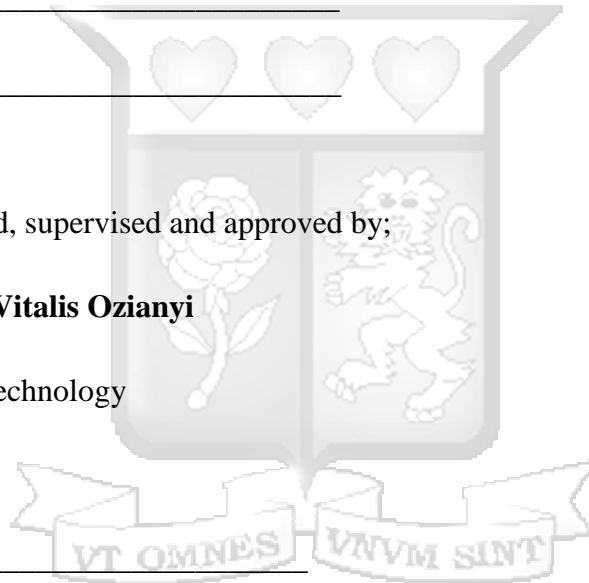
This Research was guided, supervised and approved by;

Supervisor's Name: **Dr. Vitalis Ozianyi**

Faculty of Information Technology

Signature: _____

Date: _____



Abstract

Cyberbullying has become a common vice on the social media platforms and is quickly running out of hand. The psychological researches conducted on its effect are showing dire trends on the victims, sometimes leading to suicides among the victims. Currently, the efforts by the social media sites in curbing cyberbullying is largely user centered. Twitter platform provides a series of reactionary measures of dealing with cyberbullying instances, including; blocking users, reporting users, deleting posts and tagging tweets with warning labels. However, these approaches are more of reactionary than preventive. This leaves a gap in the software systems design which should eliminate the human intervention, by implementing technological methods in curbing cyberbullying. This research implemented the application of machine learning techniques to build a text classifier to detect instances of cyberbullying as the tweets are being composed. The research collected data from Twitter which was processed and labelled appropriately. A Support Vector Machine model was developed, trained and validated based on labelled text data using bigram features and term frequency-inverse document frequency weighting. An experimental approach was taken in determining what combination of features provided the most desirable performance outcome on the data collected. A comparative analysis was then done between the text classification algorithms (including Naïve Bayes, K-Nearest Neighbor and Random Forest Classifier) coupled the different features. The SVM classifier coupled with the bi-gram feature emerged as the best classifier while using sentiment to classify texts documents, with an accuracy of 84.22%.

Acknowledgement

Special appreciation and gratitude to my supervisor, Dr. Vitalis Ozianyi, for giving direction and support during the study and Dr. Benard Shibwabo, who through the thesis seminars pointed us in the right direction for the research process.



Table of Contents

Declaration and Approval.....	ii
Abstract.....	iii
Acknowledgement	iv
List of Abbreviations	ix
List of Equations.....	xiii
Chapter 1: Introduction.....	1
1.1 Background	1
1.2 Problem Statement	3
1.3 Research Objectives	3
1.3.1 Main Objective	3
1.3.2 Specific Objectives.....	4
1.4 Research Questions	4
1.5 Justification	4
1.6 Scope and Limitation	4
Chapter 2: Literature Review.....	6
2.1 Introduction.....	6
2.2. Freedom of Expression and Censorship.....	6
2.3 Various Organizations on Cyberbullying	6
2.3.1 Kenya National Computer Incident Response Team (KE-CIRT).....	6
2.3.2 United Nations Children’s Fund - UNICEF.....	7
2.4 Cyberbullying on Twitter.....	8
2.4.1 Forms of Cyberbullying in Twitter	8
2.5 Machine Learning	8
2.5.1 Supervised Learning.....	9
2.5.2 Lexicon Based Approach	16
2.6 Sentiment Analysis	17
2.7 Twitter Data Analysis	18
2.8 N-gram Language Modeling.....	18

2.9 Data Analysis Tools and Libraries.....	20
2.9.1 Python Programming Language.....	20
2.9.2 SciKit-Learn.....	20
2.9.3 Joblib.....	20
2.9.4 Jefferson Henriques Tool.....	21
2.10 Related Studies.....	22
2.10.1 Hate speech detection in Social Media.....	22
2.10.2 Prediction of Aggressive Comments in Social Media.....	22
2.10.3 Sentiment Analysis for Social Media Data.....	22
2.11 Conceptual Framework.....	23
Chapter 3: Research Methodology.....	25
3.1 Introduction.....	25
3.2 Research Design.....	25
3.4 Data Collection.....	25
3.5 System Development Methodology.....	26
3.5.1 Rapid Application Development (RAD) Structure.....	26
3.6 Research Quality.....	27
Chapter 4: System Design and Architecture.....	31
4.1 Introduction.....	31
4.2 Requirement Analysis.....	31
4.2.1 Functional requirements.....	31
4.2.2 Non-Functional requirements.....	31
4.3 System Architecture.....	32
4.4 System Analysis.....	33
4.4.1 Use Case Diagram.....	33
4.4.2 Sequence Diagram.....	37
4.5 System Design.....	38
4.5.1 Context Diagram.....	38
4.5.2 Level 1 Data Flow Diagram.....	39
Chapter 5: System Implementation and Testing.....	41

5.1 Introduction.....	41
5.2 Sentiment Analysis	41
5.2.1 Sentiment Corpus Building	41
5.2.2 Tweets Pre-processing.....	42
5.2.3 Tweets Labeling	44
5.3 The Support Vector Machine Model training.....	45
5.3.1 Support Vector Machine Model n-gram Experiments	45
5.3.2 Testing the Model.....	46
5.4 Model Application	48
5.4.1 Application Integration.....	49
Chapter 6: Discussions	50
6.1 Introduction.....	50
6.2 Sentiment Analysis Experiments and Results.....	50
6.2.1 Use of Different Classifiers.....	50
6.2.2 Using Different Feature Types on the Classifiers	55
6.3 Comparative Analysis.....	56
6.3.1 Precision	56
6.3.2 Recall (Sensitivity).....	57
6.3.3 Accuracy.....	57
Chapter 7: Conclusions and Recommendations	58
7.1 Conclusion	58
7.2 Recommendations.....	58
7.3 Future Work	59
REFERENCES	60
APPENDIX I: CODE SNIPPETS	64
APPENDIX II Research Budget.....	66
APPENDIX III: Ethical Approval Letter	68
APPENDIX IV: Turnitin Originality Report.....	69



List of Abbreviations

ANN – Artificial Neural Network

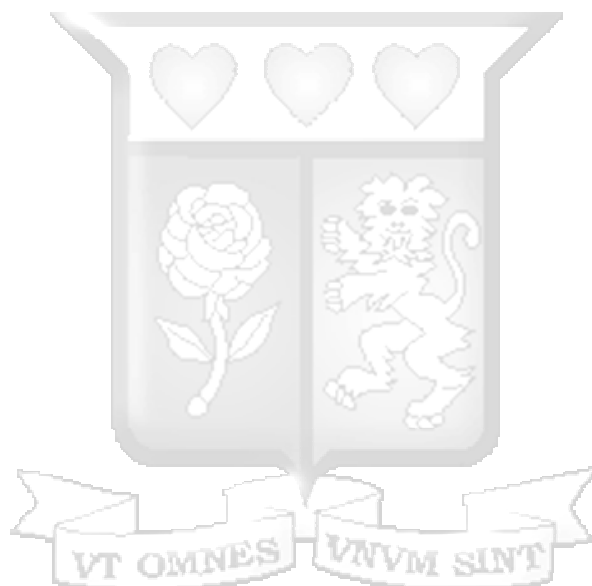
API – Application Programming Interface

CSV - comma-separated values file

NLP – Natural Language Processing

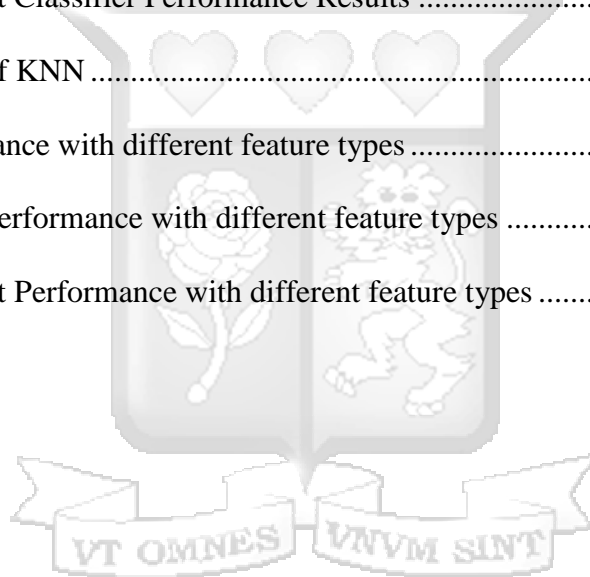
SVM – Support Vector Machine

NN – Neural Networks



List of Tables

Table 3. 1 Confusion Matrix.....	28
Table 5. 1 Confusion Matrix for the Trained Model.....	47
Table 5. 3 Derived Values from the Confusion Matrix.....	47
Table 6. 1 Classifiers' Performance Results Summary.....	50
Table 6. 2 KNN Performance Results.....	51
Table 6. 3 SVM Performance Results.....	52
Table 6. 4 Naïve Bayes Performance Results.....	53
Table 6. 5 Random Forest Classifier Performance Results.....	54
Table 6. 6 Performance of KNN.....	55
Table 6. 7 SVM Performance with different feature types.....	55
Table 6. 8 Naïve Bayes Performance with different feature types.....	56
Table 6. 9 Random Forest Performance with different feature types.....	56



List of Figures

Figure 2.1 SVM Classifier	10
Figure 2.2 Simple Decision Tree	13
Figure 2.3:Random Forest Model Prediction Making	14
Figure 2.4 Artificial Neural Network (Zupan, 1994).....	15
Figure 2.5 Steps in Sentiment Analysis	17
Figure 2.6 Conceptual Framework.....	24
Figure 3. 1: RAD Development Cycles (Araújo, 2017)	26
Figure 3. 2 AUC - ROC Curve (Schnober Carsten, 2018)	30
Figure 4. 1 System Architecture.....	33
Figure 4. 2 Use Case Diagram	34
Figure 4. 3 Sequence Diagram.....	38
Figure 4. 4 Context Diagram.....	39
Figure 4. 5 Level 1 Data Flow Diagram	40
Figure 5. 1 Jefferson Henrique’s Crawling Code	41
Figure 5. 2: Tweets Crawl and Download process from Twitter.....	42
Figure 5. 3 Snippet of Raw Tweets Crawled	42
Figure 5. 4: Tweets Pre-processor Code	43
Figure 5. 5 Snippet of cleaned Tweets	43
Figure 5. 6 Snippet of Labelled Tweets	44
Figure 5. 7 SVM Training.....	45
Figure 5. 8 N-gram SVM experiment.	46
Figure 5. 9 Printing SVM test Metrics	46
Figure 5. 10 SVM Test ROC Curve.....	48
Figure 5. 11 SVM Persistence Code Snippet.....	48

Figure 5. 12 Wireframe of Model Integration into Chat Application 49

Figure 6. 1 KNN ROC curve 51

Figure 6. 2 SVM ROC curve 52

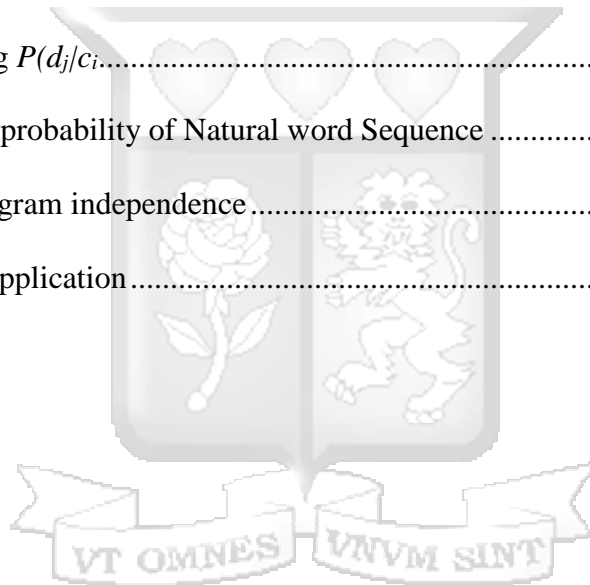
Figure 6. 3 Naïve Bayes ROC curve..... 53

Figure 6. 4 Random Forest ROC curve..... 54



List of Equations

Equation 1. 1 : Hyperplane Linear Function.....	10
Equation 1. 2: Assigning input vector.....	11
Equation 1. 3: Determining Decision Surface	11
Equation 1. 4: Calculating Posterior Probability	12
Equation 1. 5: Calculating $P(d/c_i)$	12
Equation 1. 6: Calculating $P(c_i)$	12
Equation 1. 7: Calculating $P(d_j/c_i)$	12
Equation 1. 8: Predicting probability of Natural word Sequence	19
Equation 1. 9: Markov n-gram independence.....	19
Equation 1. 10: N-gram application.....	19



Chapter 1: Introduction

1.1 Background

The period between 2008 and 2019 has seen the growth of the social networking sites in the internet. People around the globe are using forums, blogs, social networking sites for their networking. As the use of these sites is on the rise, a section of these platforms' users are violating the provisions of using them, which are unethical. This has led to other users within the sites being victims of cyberbullying (Edosomwan et al., 2011).

Globally, the definitions of cyberbullying vary depending on the context of usage. Aune (2009) defines cyberbullying as the use of information and communication technologies such as e-mail, cell phone text messages, instant messaging, personal websites and personal online polling websites to support deliberate, repeated and hostile behavior by an individual or groups, that is intended to harm others. Patchin and Hinduja (2006) on the other hand defined cyberbullying as “willful and repeated harm inflicted through the medium of electronic text”. This definition seems to be shallow as it does not paint the full picture of what cyberbullying is. The United States National Crime Prevention Council defines Cyberbullying as “when the Internet, cell phones or other devices are used to send or post text or images intended to hurt or embarrass another person” (npc, 2019).

Unlike traditional bullying, the severity of cyberbullying is intense. Cyberbullying gives the perpetrators the power to embarrass or hurt a victim before a global community, especially in the realms of social media (Edosomwan et al., 2011). This is propagated by the characteristics of social media, against traditional forms of communication; searchable; the characteristic that allows users to search and read previous posts; share-ability; this is the characteristic that allows users to share content gotten online with other users in their networks, also known as “viral nature” of the SNSs, ease of access, anonymity amongst others (Cohen-Almagor, 2011).

Previous research has established connections between cyberbullying and increased negative outcomes including; decreased performance in school, absenteeism, truancy, violent behaviors and psychological effects such as depression (Cowie, 2013), suicide and suicide ideation (Hosseinmardi et al., 2015). Given the grave consequences of cyberbullying and its growing trend, there is need to research on how to preempt and curb it. Van-Royen, Poels, Daelemans, and

Vandebosch (2015) report on the general agreement by experts in the field of cyberbullying that computational automatic monitoring of cyberbullying on social networking sites would enhance the strategies used in combating it.

Cyberbullying has proven to be a persistent version of the traditional forms of bullying. It goes beyond the regular confines of the usual societal bullying confines, with the victim often experiencing no respite from it. This is cemented by the anonymity, itinerancy permanency and cross-jurisdiction of online content. The social media networks provide bullies with global reach. This gives them the power to embarrass or hurt their victims at a global level amongst the online community (Dinakar, Reichart, & Lieberman, 2011).

Previous research into cyberbullying have looked into both psychological effects on the victims and ways of detecting the phenomenon on Twitter. Mody, Shah, Pimple and Shekokar (2018) in their study identified the need to curb cyberbullying using sentiment analysis. Progressively, more research is being conducted on detection using machine learning techniques (Sugandhi et al., 2016). Less work however is being done on how to stop an on-going attack, or even to prevent attacks before they happen (Hamiza Wan Ali et al., 2019).

The current efforts to curb the cyberbullying menace rely on the human aspect intervention. The measures in place by Twitter to contain the problem include; Blocking a user; this is where users can prevent interactions with other users. The functionality allows users to prevent other users from communicating with them, once they consider them as bullies, or generally do not ascribe to their line of thought. Reporting a user; this is feature by Twitter, which allows users to report other users to the organization based on the judgement of the content they have posted. Twitter then evaluates and takes the appropriate action. Deletion of posts; Twitter allows users to delete any posts they have made (Twitter, 2019). Though providing a way of dealing with cyberbullies already on the platforms and content already posted, these measures are still proving insufficient in the fight against cyberbullying (Nadali et al., 2013).

Different machine learning techniques have been researched with regards to detection of cyberbullying. Sentiment Analysis is considered based on the assumption that most cyberbullying tweets are aggressively negative messages (Mercado et al., 2018). Support Vector Machines

(SVM) was chosen as the algorithm of preference as it has been proven to provide desirable accuracy against high-skew text classification experiments, which resembles the ones under this study (Noviantho et al., 2018).

1.2 Problem Statement

Monitoring bullying in other forms of media for example newspapers, and radio is far much easier compared to the complexity of the social media sites. This is largely due to the checks and balances that exist in those environments to ensure what gets to the public meets the standards of journalism. However, unlike traditional media, in social media sites, the content is user generated and checks only occur once they are published.

Social media platforms contain features which allow cyberbullying to thrive. The ease with which victims are targeted, the anonymity aspect and lack of control, provide an enabling environment (Cheng et al., 2019) The current efforts to curb cyberbullying on the social media sites seem not to cope with keeping the phenomenon at bay. The measures taken by the social sites themselves are reactive in nature. The reactive measures are outlined in the users' policy document, where twitter has laid out instructions on how to block, flag, and report bullies or content deemed to contain cyberbullying.

This research proposes the development of a model which seeks to apply automatic techniques in identifying cyberbullying and aggression as the users are typing them and alerting the users to change their tweets composition, before posting. This model seeks to use Sentiment Analysis to decipher opinions being expressed in tweets, then categorizing them as either positive or negative. This automatic detection as the tweets are being composed would significantly improve the efforts in reducing and curbing cyberbullying. It is a preventive measure.

1.3 Research Objectives

1.3.1 Main Objective

The research designed a prototype prediction model which can detect cyberbullying in tweets as they are being composed before they are posted on to the public on twitter. This was achieved through sentiment analysis on data crawled from Twitter.

1.3.2 Specific Objectives

- i. To investigate the techniques used by social media platforms in mitigating cyberbullying
- ii. To develop a Sentiment Analysis model to counter cyberbullying on Twitter
- iii. To test and validate the model

1.4 Research Questions

- i. What are the forms of cyberbullying on Twitter?
- ii. What techniques are currently used by social media in managing cyberbullying?
- iii. How will the model be designed, tested and validated?

1.5 Justification

Currently, social media networks rely on users reporting cyberbullying incidences. Automation for this process is almost absent. Considering the massive big data set generated in social sites platforms daily, automating the detection and prevention of cyberbullying content would therefore be required to reduce cyberbullying effectively (Van Hee et al., 2016). To achieve both detection and prevention, sentiment analysis approach is applied; where a tweet can be analyzed and categorized as either constituting cyberbullying or not. This study sought to extend the use of sentiment analysis to include prevention of cyberbullying, not just detection. Little attention has been devoted to prevention, beyond regular expression-driven systems based on keywords. There has is a famine of computationally driven approaches for its prevention. Moreover, most of the researches conducted focus on categorizing already posted tweets, detecting them once out (Hollá, Fenyvesiová, & Hanuliaková, 2017).

1.6 Scope and Limitation

This study will limit its research on Twitter. This is because Twitter, unlike other social media platforms, allows researchers to use their platform to conduct researches. Secondly, Twitter posts reflect peoples' instantaneous opinions regarding events, products and even other people. This provides the rich data required in developing the model.

The study would be limited to analyzing tweets composed in English language. This is because the researcher is conversant with the language and most tweets are composed in the language, therefore will not need any translation.



Chapter 2: Literature Review

2.1 Introduction

This chapter reviews applicable literature for a deeper understanding of the research problem. The state of cyberbullying and its forms in Twitter will be reviewed. This should inform the forms of cyberbullying which could be curbed using the proposed model.

Previous significant research on sentiment analysis was further reviewed to understand the application of Sentiment analysis and relevant machine learning techniques in sentiment mining and text classification, as well as the different algorithms used in text classification. A conceptual framework will finally be presented at the end of the chapter.

2.2. Freedom of Expression and Censorship

Previously, in the traditional forms of media, gatekeepers mitigated and negotiated access to mass media platforms. Presently however, potentially anyone and any content can reach millions of users in an instant. This development bears great opportunities for the democratization of expression and the diversification of public discourse but has likewise broadened the impact of harm caused online. This raises the question how platforms and services can be regulated effectively to combat online harms without jeopardizing free and open discourse (Cowie, 2013).

As regulators the globe grapple with responses to the legal novelties and challenges that social media represents, striking the balance between freedom of expression and censorship has continuously proved a challenge (Milosevic, 2016). Questions about the scope of protections to be broader and allow more free speech online are being asked, or instead should there be recalibration to address the increases in hateful, violent and discriminatory content? More generally, how can human rights law be asserted and enforced in an online environment where the social media corporations like Facebook and Twitter seem to hold all the cards (Cohen-Almagor, 2011)?

2.3 Various Organizations on Cyberbullying

2.3.1 Kenya National Computer Incident Response Team (KE-CIRT)

The Kenya National Computer Incident Response Team (KE-CIRT) defines cyberbullying as “when a child, preteen or teen is tormented, threatened, harassed, humiliated, embarrassed or

otherwise targeted by another child, preteen or teen using the Internet, interactive and digital technologies or mobile phones”. According to the organization, it has to have a minor on both sides, or at least have been instigated by a minor against another minor (KE-CIRT/CC, 2019).

According to the commission, children are so often motivated by anger, revenge or frustration. Sometimes they do it for entertainment or because they are bored and have too much time on their hands and too many tech toys available to them. Many do it for fun or to get a reaction. Some do it by accident, and either send a message to the wrong recipient or didn't think before they did something. The power-hungry do it to torment others and for their ego.

To curb cyberbullying, the commission advises children to never share any information online which could be used against them. Cyberbullies often use pictures, status updates, and personal information they find online to harass their targets. It's fine to share a little information about yourself online, but never reveal something you don't want the whole world to know(KE-CIRT/CC, 2019).

2.3.2 United Nations Children's Fund - UNICEF

The UNICEF organization defines cyberbullying as bullying with the use of digital technologies. It can take place on social media, messaging platforms, gaming platforms and mobile phones. It is a repeated behaviour, aimed at scaring, angering or shaming those who are targeted. Examples include: spreading lies about or posting embarrassing photos of someone on social media; sending hurtful messages or threats via messaging platforms; impersonating someone and sending mean messages to others on their behalf(UNICEF, 2020).

Face-to-face bullying and cyberbullying, according to the organization, can often happen alongside each other. But cyberbullying leaves a digital footprint – a record that can prove useful and provide evidence to help stop the abuse. To help end the vice, UNICEF and its partners are calling for action in the following area;

- i. Implementation of policies to protect children and young people from cyberbullying and bullying.
- ii. Establishment and equipment of national helplines to support children and young people.

- iii. Advancement of ethical standards and practices of social network providers specifically in regards to the collection, information and management of data.
- iv. Collection of better, disaggregated evidence about children and young people's online behaviour to inform policy and guidance.
- v. Training for teachers and parents to prevent and respond to cyberbullying and bullying, particularly for vulnerable groups.

2.4 Cyberbullying on Twitter

Compared to detecting spam where same message is broadcasted to many recipients, detecting cyberbullying, which is more personalized and context oriented, is much more difficult. Most cyberbullying has been identified to revolve around selected topics including; race and ethnicity, sexuality and sexual identity, body shape and appearance, intelligence, and social inclusion and non-acceptance. Understanding if a text composition and its sentiment is aligned to these topics, and the intonation positive or negative, would be crucial in singling out possible cyberbullying messages (Lieberman, Dinakar, & Jones, 2011).

2.4.1 Forms of Cyberbullying in Twitter

Cyberbullying takes many forms including; flaming, pestering, denigration, impersonation, outing, boycott and cyberstalking. The most extreme prevalent cases of cyberbullying are flaming and the less severe is cyberstalking. Flaming frequently happens when a group of users argue about topics which include crude, aggressive and foul language and are exchanged using electronic messages. Flaming is the most extreme form of cyberbullying as it comes from online squabbles amongst internet's users. It is therefore difficult to identify the cyberbully and the victim during these altercations (Hamiza, Wan, Ali et al., 2019).

2.5 Machine Learning

These are methods that are extraordinary for AI solutions. They are founded on the proposition of learning as a result of continuous training and practices. The association models such as artificial neural networks are well fit for machine learning where new association weights are adjusted to progress the competence of a formed network.

In supervised machine learning, a classifier is developed by training it using the learning the properties of categories from a set of pre-classified training data sets. When using machine learning techniques four main issues need to be considered: classifications that will be used to group the instances, training data, features that will be used to represent each instance and the algorithm to be used for categorization (Feldman & Sanger, 2007). Since an experimental approach is taken, a combination of the categories is often used to ensure best achievable outcome. The following subsections discuss the different techniques of machine learning applied in text classification.

2.5.1 Supervised Learning

2.5.1.1 Rule-based Learning

The rule-based learning classifier uses the principle of the rule of incidences of sentiments of text composed messages. When any selected words contain positive emotions, the conclusion drawn is that it is positive. Else when the words contain negative emotions, the conclusion is that they are negative. The rule-based algorithm has similarities with fuzzy-logic systems which allow the median value to be well-defined between conventional evaluations like positive or negative, true or false and others (Bhardwaj, 2015).

2.5.1.2 Support Vector Machines (SVM) or Support Vector Networks (SVN)

These are classification and regression examination techniques. They are categorized as supervised machine learning models for information analysis as well as pattern recognition. The common application areas for the SVM algorithms include image processing, bioinformatics and text analysis. With the use of an SVM learning algorithm, it is probable to create a room that is transformable. The model signifies the examples as points in space, maps separate categories and divides them as much as possible. The main objective is to design a hyperplane which separates the training vectors into 2 distinct groups, where the ultimate choice is a hyperplane which leaves appropriate maximum margin for both classes (Platt, 1999). Recent research and state of the art approaches of Support Vector Machines show that using ensemble approaches can drastically reduce the training complexity while maintaining high predictive accuracy. This has been done by implementing the SVM without duplicate storage and evaluation of support vectors, which has been shared between consistent models (Marc et al, 2014).

Geometrically, binary SVMs are seen as hyper planes in the feature space separating points which represent negative instances. During training of the algorithm, a classifying hyper plane is identified. This is the unique plane that separates the negative instances from the known positive ones with the maximal margin. Support vectors are used in determining the SVM hyper planes as Figure 2.0 illustrates (Potha & Maragoudakis, 2015).

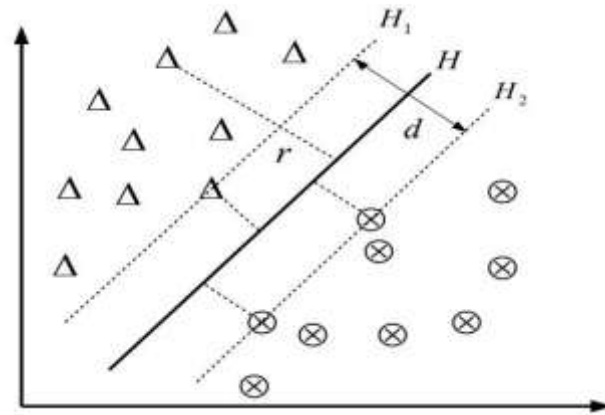


Figure 2.1: SVM Classifier

From the figure above, SVM builds binary classifiers. Let the training set of examples named D be;

$$\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\},$$

$x_i = (x_{i1}, x_{i2}, \dots, x_{ir})$ refers to the r -dimensional input vector in a real-valued space and represents the positive class as 1, and the negative class as -1 (Liu, 2011). To build the hyperplane, a linear function is built of the form;

$$f(x) = (w \cdot x) + b$$

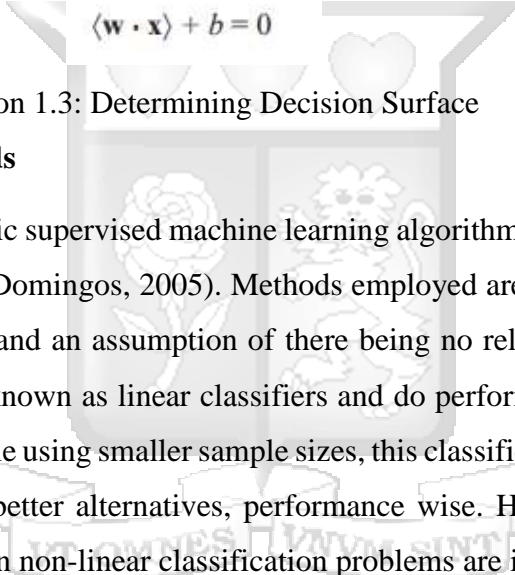
Equation 1.1 : Hyperplane Linear Function

Positive is assigned to the input vector x_i when $f(x_i) \geq 0$, or otherwise negative as shown by the following equation.

$$y_i = \begin{cases} 1 & \text{if } \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 0 \\ -1 & \text{if } \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b < 0 \end{cases}$$

Equation 1.2: Assigning input vector

SVM therefore determines the hyperplane by the equation below, also called the decision surface.


$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$$

Equation 1.3: Determining Decision Surface

2.5.1.3 Naive Bayes methods

These are a set of probabilistic supervised machine learning algorithms that are used for clustering and classification (Lowd & Domingos, 2005). Methods employed are founded on the application of Thomas Bayes' theorem and an assumption of there being no relation amongst every pair of selected features. They are known as linear classifiers and do perform well, simply and are very efficient (Zhang, 2004). While using smaller sample sizes, this classifier sometimes provides better outcomes compared to the better alternatives, performance wise. However, the classifier often gives poor performance when non-linear classification problems are involved. These methods are used in several fields including; diseases diagnostics, RNA sequences categorization in taxonomic studies and identifying spam emails e-mail clients (Raschka, 2014). Research of Naive Bayes has previously been proved to be an optimal method of clustering and classification, no matter how strong the attributes involved are related. Whether the dependencies are evenly distributed in the classes or when they cancel out each other, optimal performance is still achieved by Naive Bayes (Zhang 2004). Recently, Naive Bayes theorem has been applied to image classification algorithms, where the Local Naïve Bayes Nearest Neighbor algorithm improves the classification accuracy and the ability to exponentially tackle bigger numbers of object classes (Lowe, 2012).

Given a document \mathbf{d} and a set of classes $\{c_i\}$, the posterior probability that the \mathbf{d} belongs to the classes is calculated and the document is assigned to the class with the highest value of probability (Bai & Nie, 2004)

(Liu, 2011) opines the posterior probability should be calculated using the following equation;

$$P(c_i | d) = \frac{P(d | c_i)P(c_i)}{P(d)}$$

Equation 1.4: Calculating Posterior Probability

Assuming the conditional independence among the words in a class, then $P(d|c_i)$ is solved as;

$$P(d | c_i) = \prod_{j=1}^m P(d_j | c_i)$$

Equation 1.5: Calculating $P(d|c_i)$

$P(c_i)$ is determined as the share of training sets in class $P(c_i)$ using the following equation;

$$P(c_i) = \frac{N_i}{N}$$

Equation 1.6: Calculating $P(c_i)$

From the above equation, N refers to the total number of training documents and N_i to the number of training documents in the class c_i . $P(d_j|c_i)$ is thence calculated as;

$$P(d_j | c_i) = \frac{1 + \text{count}(d_j, c_i)}{|V| + N_i}$$

Equation 1.7: Calculating $P(d_j|c_i)$

Naïve Bayes algorithm combines its ease of use with efficiency and accuracy. Moreover, it works well with numerical and textual data (Swamy & Hanumanthappa, 2013).

2.5.1.4 Decision Tree Classifiers

These classifiers employ a hierarchical decomposition of the training data where certain conditions on an attribute value are used in classifying and dividing data (Quinlan, 1986). The predicate and conditions used implies the absence and presence of more than one word. In decision trees; the process of dividing data takes place recursively until all the leaf nodes have a minimum number of records that show a detailed classification. Figure 2.2 Shows a visualization of a simple decision tree.

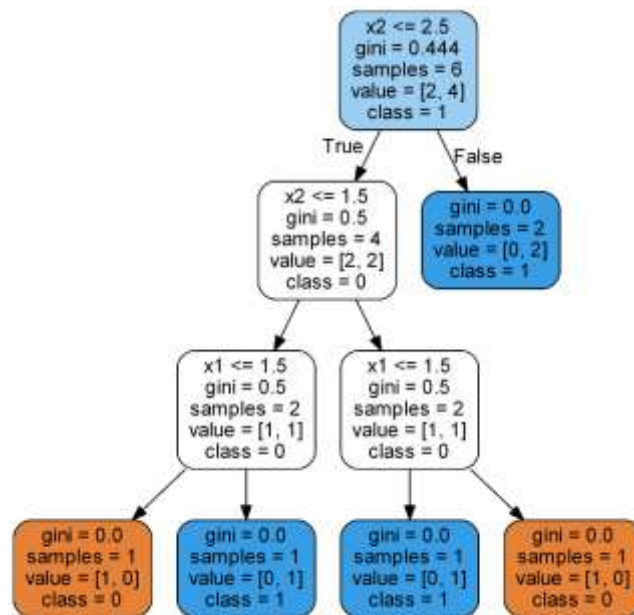


Figure 2.2: Simple Decision Tree

2.5.1.5 Random Forest

This is a supervised learning algorithm composed of several decision trees. It uses two fundamental principles; randomly sampling the training data points while coming up with the trees and random subsets of features considered when splitting the nodes (Dong et al., 2019).

The algorithm works in two phases, in the first phase, it creates the random forest creation and in the second it makes the prediction from the forest created. The following pseudocode depicts the whole process involve in the algorithm;

- i. Select randomly “**K**” features from total “**m**” features where $k \ll m$
- ii. Among the “**K**” features, calculate the node “**d**” using the best split point
- iii. Split the node into **daughter nodes** using the **best split**
- iv. Repeat the **i to iii** steps until “**T**” number of nodes has been reached
- v. Build forest by repeating steps **i to iv** for “**n**” number times to create “**n**” **number of trees**

Every individual tree in the random forest gives its own prediction. These are then collated and the prediction with the most tallies from the trees becomes the model’s prediction(Jain & Katkar, 2016). Figure 2.3 below shows the visualization.

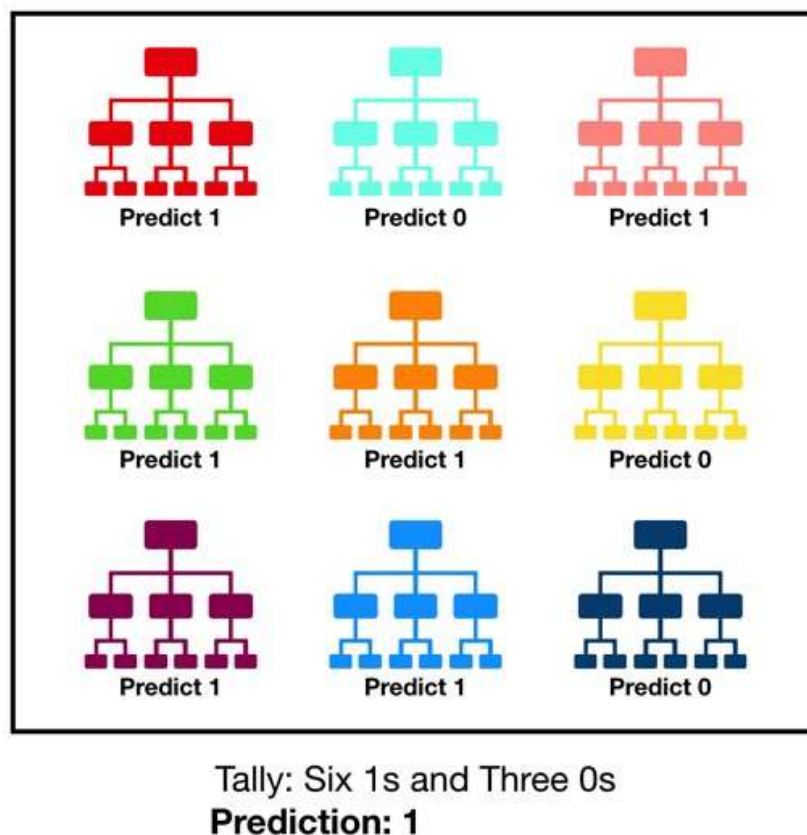


Figure 2.3: Random Forest Model Prediction Making

2.5.1.6 Artificial Neural Network

The bio-inspired machine learning model shows incredible success in its application and in the fields of artificial intelligence. Scholars have illustrated that using the bio-inspired algorithm has satisfactorily improved the results in the research domain. They include artificial neural networks

(ANN), artificial immune systems, evolutionary computation, fuzzy systems, as well as swarm intelligence (Andries, 2007).

An artificial neuron network takes the model of biological neurons. Artificial neuron accepts signals or inputs from other neurons or surrounding. The signal is fired and given certain conditions, thus, transferring the signal to all the connected neurons (Uhrig, 1995). Figure. 2.4 below shows an artificial neuron. Here, there is an association between the numerical positive and the negative value which is associated with each neuron such that they either inhibit or excite inputs with each connection made to the artificial neuron. The activation functions in ANN are used to regulate the firing taking place in the artificial neuron. The neuron then collects all incoming signals by computing their net input signals as a function with the associated or given weights. These net input signals then serve as input to the activation function which calculates the output signal of the artificial neurons (Zupan, 1994). An ANN is a layered system containing of one or many artificial neurons. ANN components include the input layer, hidden layer and the output layer. Based on the interconnection of the components; the ANN has been modelled with the ability to perform learning, generalize and map abilities to process information in parallel.

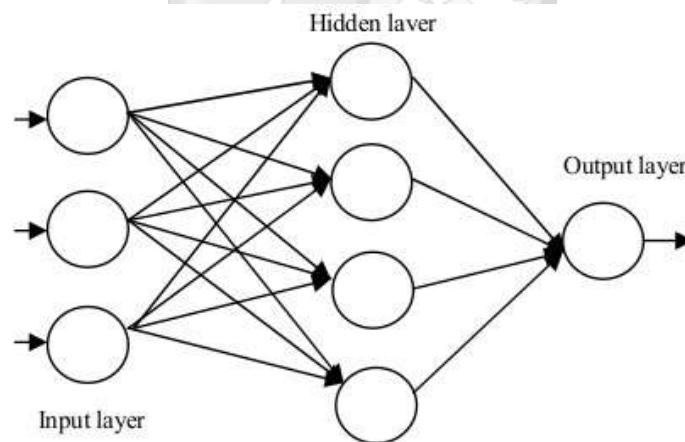


Figure 2.4: Artificial Neural Network (Zupan, 1994)

Several ANN architectures have been developed such as feedforward neural network, recurrent neural network, and spiking neural network. Also, there are also different types of neural network

such as single-layer neural network, multi-layer perceptron (MLP), temporal neural network, radial basis neural function network, self-organizing neural network (Peterson & Rögnavaldsson, 1992).

2.5.2 Lexicon Based Approach

This approach aims at attaining an effective cross-domain performance. The methodology works with the assumption that; total of the sentiment orientations of all words make contextual sentiment orientations. Here, words that are opinionated are used on the classification tasks. The positive opinions are employed in describing desired states while the negative opinions express the undesired states. There exists several opinion idioms and phrases that are known as lexicons. There exist different approaches when it comes to compiling and collecting the opinions used in the word list. Since the manual approach is time consuming; it is used together with other faster approaches that are automatic with the aim of checking out for any errors and mistakes. The common approaches are discussed in the section below (Bhardwaj, 2015).

2.5.2.1 Dictionary-Based Approach

In a dictionary-based approach; sets of words or opinions are collected manually based on set subjects. The same grows through searching for more words in a corpora WordNet or thesaurus for simple synonyms as well as antonyms. The words obtained are often added to a seed-list resulting in the formation of new iterations. The process of iteration stops when the system fails to find new words. At the end of the process; there is manual inspection done with the purpose of eliminating any existing errors. The key disadvantage of this approach is the inability to find any opinions or words that are context or domain specific orientations (Mohammad et al., 2009).

2.5.2.2 Corpus-based Approach

This method aids in resolving the difficulty of finding opinion words given context specific orientations. Its methods depend on syntactic patterns or patterns that occur together along with a seed list of opinion words to find other opinion words in a large corpus (Medhat et al., 2013). The approach tries to find the existence of co-occurrence patterns of words. This helps in determining their sentiments. It is founded on seeding a list of opinion words, then finding other opinion words which have a similar context. It is used in assigning the happiness factor of words, depending on the frequency of their occurrences in “happy” or “sad” posts (Bhardwaj et al., 2015).

2.6 Sentiment Analysis

According to Liu (2012) a sentiment is defined as a feeling, opinion expressed by a person towards something, an idea or someone. Computer studies define sentiment analysis as; attitude mining, opinion mining studying the sentiments of people towards certain ideologies (Fang & Zhan, 2015). Different approaches exist through which sentiment analysis can be achieved; machine learning approach, hybrid approach and lexicon-based approaches (Maynard & Funk, 2011). Machine learning approach employs the use of machine learning classifiers and simple linguistic features. In lexicon-based approaches, the algorithm relies on sentiment lexicons, a collection of precompiled known terms. These are further classified as dictionary-based approaches or corpus-based approaches, which utilize semantic statistical techniques to find sentiment polarity. The following Figure 2.1 shows the steps taken in sentiment analysis modelling.

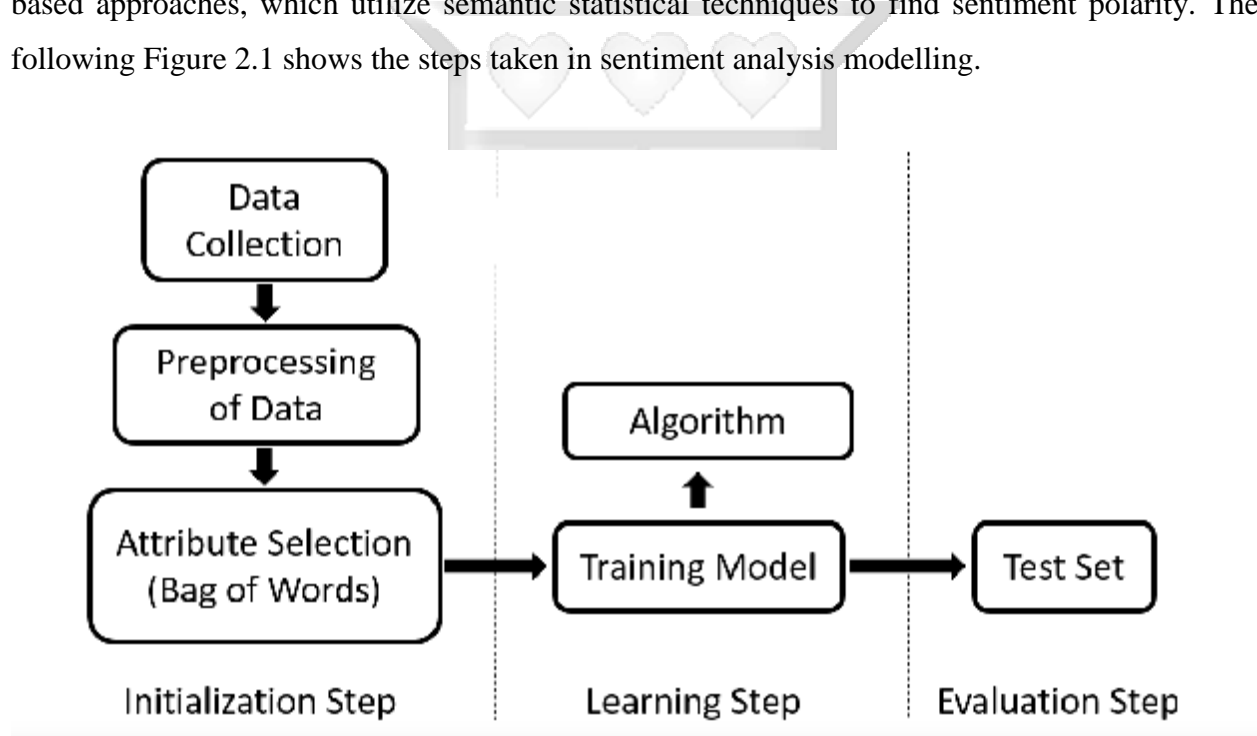


Figure 2.5: Steps in Sentiment Analysis

The process of sentiment analysis can be grouped into three stages; Initialization step, learning step and the evaluation step. The initialization step starts by data collection, then preprocessing the data and ends with attributes or features extraction from the data. The second next step is the model training, where the labeled data from the initialization step is used. The third and final step is

evaluation. This is where the model is reviewed for accuracy against test data (Angiani et al., 2016). The desired output of the model is correct predictions of tweets with cyberbullying content. This calls for the implementation of supervised learning on the SVM algorithm (Brownlee Jason, 2019).

2.7 Twitter Data Analysis

According to Sayce (2019), about 500 million tweets are composed each day. This forms a rich ground from which to harvest data that can be used in this research. The social networking microblogging service had about 330 million active monthly users in 2019. (Statista, 2019). To collect this data, there are available tools. First there's Twitter's APIs, provided by the Twitter platform. Twitter Search API allows for search up to seven days back and the Streaming API allows for real-time queries against tweets. (Twitter, 2019).

The second available tool for collecting the tweeter data is the GetOldTweets tool developed by Jefferson Henrique available on GitHub. This is the tool to be adopted for the research. The tool allows search of tweets without the timeline restriction of the Twitter APIs.(Henrique Jefferson, 2018). The data crawled by this tool is exported in csv format for further analysis.

First, the data undergoes preprocessing. This is where the undesired tweets components are removed from the tweets. These include dates, symbols, punctuation marks, emoticons, permalinks among others. The tweets are then processed to extract the features. In this instance, whether they contain cyberbullying or not. The processed tweets are then labelled as either positive or neutral. This is a manual process that involves reading the tweets one by one and assigning labels, 1 for tweets that contain cyberbullying and 0 for neutral ones. The resulting labeled data forms the corpus to be used on model training.(Jayasekara Dilan, 2019).

2.8 N-gram Language Modeling

Language modeling been successfully implemented in information retrieval, tracking, topic detection and text classification (Van Hee et al., 2018). It is largely used because it has a supporting foundation in statistics. Its main goal is predicting the probability of natural word sequences. Given

w_1, w_2, \dots, w_t as a word sequence, the probability of any word sequence is calculated using the below equation (Nurrahmi & Nurjanah, 2018).

$$P(w_1 w_2 \dots w_T) = \prod_{i=1}^T P(w_i | w_1 \dots w_{i-1})$$

Equation 1.8: Predicting probability of Natural word Sequence

N-gram models calculate the probability with the assumptions that only words relevant to predicting $P(w_i | w_1, \dots, w_{i-1})$ are the previous $n-1$. The models assume the Markov n-gram independence, depicted by the following equation (Shirakawa et al., 2013).

$$P(w_i | w_1 \dots w_{i-1}) = P(w_i | w_{i-n+1} \dots w_{i-1})$$

Equation 1.9: Markov n-gram independence

The maximum probability estimate of n-gram probabilities from a corpus is given by the observed frequency by the following equation. $\#(\cdot)$ represents the number of occurrences of a specified gram in the training corpus (Peng, 2003).

$$P(w_i | w_{i-n+1} \dots w_{i-1}) = \frac{\#(w_{i-n+1} \dots w_i)}{\#(w_{i-n+1} \dots w_{i-1})}$$

Equation 1.10: Probability estimate of n-gram probabilities

Various n-gram language models (uni-gram, bi-gram tri-gram, n-gram) can therefore be applied to text classification algorithms. A document \mathbf{d} is categorized under a category \mathbf{c} according to (Peng, 2003). This is illustrated by the following equation (Peng, 2003).

$$c^* = \arg \max_{c \in C} \{P(c | d)\}$$

Equation 1.101: N-gram application

2.9 Data Analysis Tools and Libraries

2.9.1 Python Programming Language

According to Guelton (2018), the Python programming language is described as a high-level, object-oriented programming language with dynamic semantics. It has high-level built in data structures, accompanied with dynamic typing and binding which makes it a favorite while using the Rapid Application Development methodology. The language is also extensively used in development of interfaces and extending already existing systems.

The language's syntax, which is easily comprehensible, insists on readability and therefore easier program maintenance. The language enables the use of modules and packages, this enhances modularity implementation in programs and code reuse (Schnober, 2018).

The language is supported by a number of integrated development environment software application for development and testing, including; Visual Studio Code, PyCharm, Atom, Jupiter Notebook, among others.

2.9.2 SciKit-Learn

This is an open source machine learning library developed for the python programming language. It is built on top of SciPy and is open source. The library includes various feature including; regression, clustering and regression algorithms, which include support vector machines, random forest, k-means, among others. The library is designed to interoperate with python numerical and scientific libraries like NumPy and SciPy (pydata.org, 2018).

Data scientists and Statisticians use it to handle standard machine learning and data mining tasks for example classification, regression, clustering, dimensionality reduction, and model selection. It comes with quality documentation and offers high performance therefore ideal for easy implementation (Schnober Carsten, 2018).

2.9.3 Joblib

These are a set of tools which provides lightweight pipelining in Python. It provides; transparent disk-caching of functions and lazy re-evaluation (memorize pattern) and easy, simple parallel

computing capability. The library is optimized to be fast and robust on large data. It also has specific optimizations for NumPy arrays mostly used in python (Burdi et al., 2017).

The library boasts a Transparent and fast disk-caching of output value: a make-like functionality for Python functions which works well for arbitrary Python objects, including large numpy arrays; Embarrassingly parallel helper: which makes it easy to write readable parallel code and debug it quickly; Fast compressed Persistence: a replacement for pickle to work efficiently on Python objects containing large data (pydata.org, 2018).

2.9.4 Jefferson Henriques Tool

Twitter has two APIS which allow developers to access the platform; the Representational State Transfer (REST) API and the Streaming API. They both use Open Authentication to allow applications gain access to the Twitter platform. When data is queried, the response is given in JSON format (JavaScript Object Notation). The Streaming API allows developers to process tweets in real time. This is done by steadily delivering responses in JSON format over HTTP connections. Read and write capabilities are granted by REST API on Twitter data. One of the limitations of the APIS is that users can only query up to 7 days old tweets(Twitter, 2019). This calls for the use of the Jefferson Henriques Tool. Also called GetOldTweets.

This tool imitates the working function of the search feature on Twitter through a browser to retrieve the older tweets. Its remarkable features such as counting retweets, searching over multiple users accounts, allows for the crawling a lot of data over a short period of time. Basically, when a Twitter page is entered to the tool, a scroll loader starts, it scrolls down to get more and more tweets, all through calling to a JSON provider (Jefferson, 2017). The tool allows collection of older tweets using the following parameters;

- i. Query search – which describes the text to be matched,
- ii. Username – Username of the given twitter account,
- iii. Bound dates (since and Until) which describes the period of interest,
- iv. Maxtweets – which refers to the total number of tweets to crawl,

2.10 Related Studies

2.10.1 Hate speech detection in Social Media

This research used unigram features and term frequency inverse document frequency (TF-IDF) weighting in labeling and model training. The model was used to predict hate or non-hate speech tweets posted on Twitter. In this study tweets are gathered using the Twitter APIs, then presented with the TF-IDF weighting and used on the SVM classifier (Mugambi, 2017).

This research however does not address the automation process in the prevention of hate speech. The research only looked into predicting what tweets contained hate speech and which ones did not. The desirable accuracy of SVM in this study influenced its adoption for this research.

2.10.2 Prediction of Aggressive Comments in Social Media

The research primarily looked into the detection of aggressive texts in social media platforms. The researchers further looked into the different computational techniques that can be used in tackling the phenomenon of cyberbullying including the different machine learning techniques. Their findings recommend the adoption of the identification of profane words as a key feature to be used in developing the models (Del Bosque & Garza, 2016).

The research however fails to come up with prevention techniques that can be used in overcoming the phenomenon of cyberbullying. The gap on prevention techniques is therefore identified as an area that needs tackling.

2.10.3 Sentiment Analysis for Social Media Data

Ramadhani and Goo (2017) in this study proposed to analyze social media data. The initial step in the process involved text mining which involved the following stages;

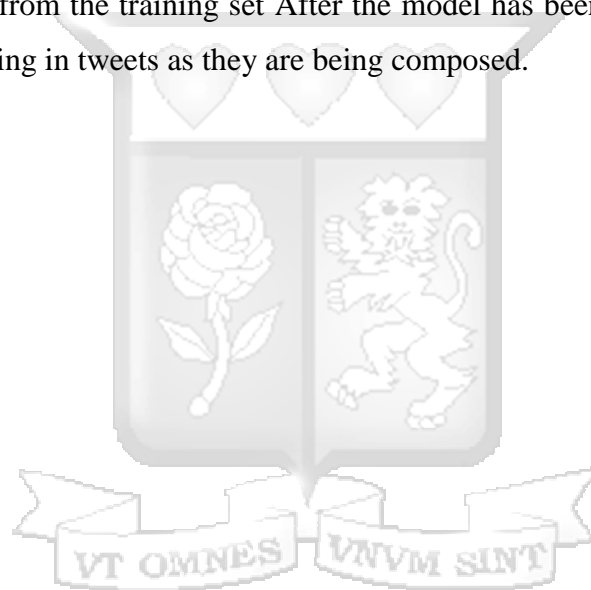
- i. Crawling data from the social media platforms
- ii. Performing fractional examination and recognizing any pertinent data points.
- iii. Digging into the data to retrieve knowledge.

A corpus of about 4000 tweets was utilized on a neural system which involved a hundred neurons, 3-layer design and a stochastic descent. The layers give a bit by bit technique for clearing words, sentences lastly, directing the prominence.

2.11 Conceptual Framework

The Figure 2.3 below shows the conceptual framework proposed to detect cyberbullying instances in tweets being composed for twitter platform. Cyberbullying relevant data will be crawled from Twitter and stored in csv formats. The data will then undergo text pre-processing. During this process, undesirable components of the tweets including symbols, stop words, punctuation marks and hyperlinks will be removed. The preprocessed tweets are then taken through feature extraction. The tweets will be annotated as containing cyberbullying content or not. The tweets are then labeled as positive or neutral. The labeled tweets form the corpus to be used in model training.

For its suitability in text classification problems, SVM algorithm is trained to learn a model for detecting cyberbullying, from the training set After the model has been persisted, it can then be used to detect cyberbullying in tweets as they are being composed.



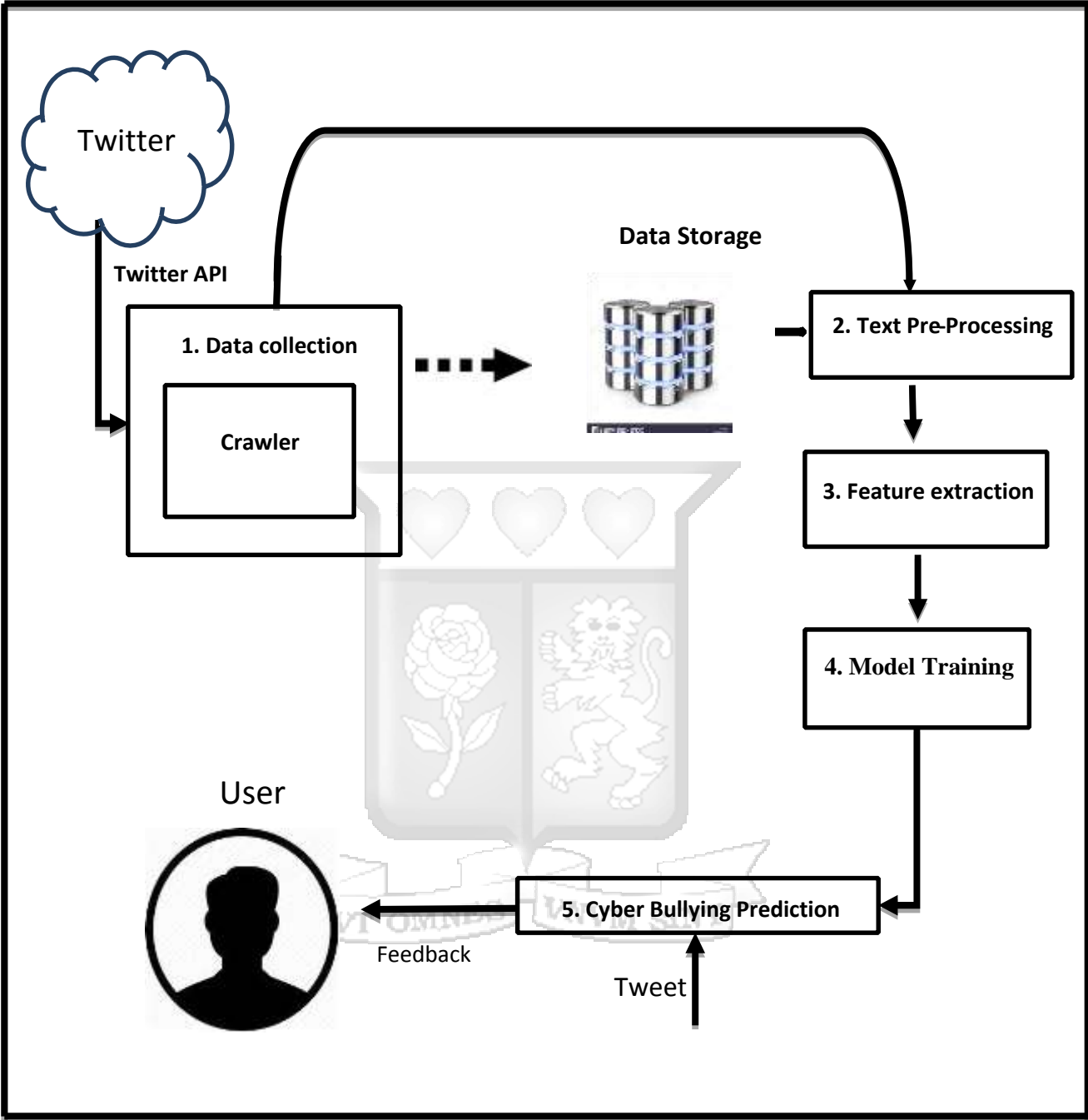


Figure 2.6: Conceptual Framework

Chapter 3: Research Methodology

3.1 Introduction

This chapter presents the various methods and procedures that were adopted in carrying out the research. The chapter begins with the research design which describes the approach taken, then the data and its collection. Here, the data to be collected and tools to be used are highlighted. The model learns from tweets identified to have cyberbullying composition, crawled from Twitter. The chapter then ends with system development methodology, which looked at the adopted methodology and what the different phases of development involve.

3.2 Research Design

A research design refers to the sequence of conditions for the collection and data analysis in a way that aims at combining relevance to the purpose of the research, with economy in procedure. This research seeks to take an experimental design approach, which involves identifying the research objectives, building of the model as a proof of concept and validating the model using a number of experiments to ensure best performance. To design and evaluate the model, the research crawled tweets as the data to be used from Twitter.

3.3 Target Population and Sampling

A population is defined by Bryman (2012) as the total number of units in a study environment from which a sample may be selected. In this study, twitter posts containing cyberbullying were used. Purposive sampling was applied in the research. This is where the sample is determined by the judgement of the researcher based on prior knowledge of characteristics of tweets that constitutes cyberbullying. This study crawled a total of 63,560 tweets. 6,990 tweets were used, identified to have instances of cyberbullying.

3.4 Data Collection

Data collection in sentiment analysis involves collecting of sentiment related data. The research focuses on twitter for the openness and platform availability to developers. To build a corpus, cyberbullying tweets were collected from Twitter. To build robust comprehensive corpora, the

Jefferson Henrique's open source code, made available through GitHub to crawl the required data from twitter. This tool allows tweets to be collected based on keywords, user handles and hashtags features of tweets.

Python programming language was used for the data analysis, primarily for its superiority in statistical computing and computer graphics. The language also has a large user community of data miners and statisticians. Coupled with its vast libraries for statistics and machine learning, the language provided the best option for the data analysis conducted.

3.5 System Development Methodology

The research developed the prototype following the Rapid Application Development (RAD) system development methodology. This methodology emphasizes the creation of applications in the minimum time possible, sometimes compromising on usability, features and execution speed (Naz & Khan, 2015). The methodology was developed by James Martin to accelerate the development of applications while emphasizing on quality and saving resources used in the development process (Beynon-Davies, Came, Mackay, & Tudhope, 1999).

3.5.1 Rapid Application Development (RAD) Structure

The Figure 3.1 shows the different cycles of the Rapid Application Development Methodology.

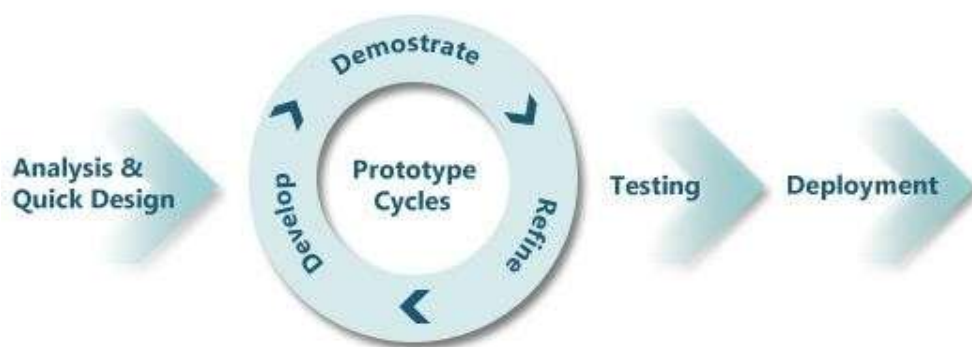


Figure 3.1: RAD Development Cycles (Araújo, 2017)

3.5.1.1 Planning Phase

This phase involves getting the system requirements and quick analysis. In this research, the structure and architecture of the prototype was designed. Unified Modelling Language (UML) diagrams were used in depicting the various components and aspects of the system including; use case diagrams, context diagrams, data flow diagrams and sequence diagrams.

3.5.2.2 Prototyping phase

This phase defines the designs and models for the prototype to be created. The phase was iterative in nature. The prototype was implemented using python as a development language. Python's scikit-learn library were applied in implementing the various machine learning algorithms (scikit-learn, 2019). Pandas library was used to provide the easy to use data structures used in data manipulation process.

3.5.2.3 Testing

In this phase, the created model is validated. Unit testing, integration and system testing are done. 30% of the data set was used for testing. Iteration continued to improve the model towards desirable accuracy. The output of the model was then compared with the actual labels.

3.5.2.4 Cut-over or Deployment Phase

This is the final phase. The model was persisted to maintain its learned state. It was then integrated with a chat application environment. This allowed the simulation of the chat environment to test the accuracy in such environment.

3.6 Research Quality

Validity refers to how much a concept, a measurement or a conclusion is accurately measured in a quantitative study. The second element in measuring the research quality is reliability or accuracy. This refers to the extent to which a research instrument persistently outputs the same results if it is used in the same situation on repeated occasions (Heale & Twycross, 2015).

The sources of information and data used in the study have been well cited and guaranteed in the research. In all the data sets used, there were no missing values. A confusion matrix was then

applied to the model. The matrix contains information about the actual classification and the predicted ones by the model.

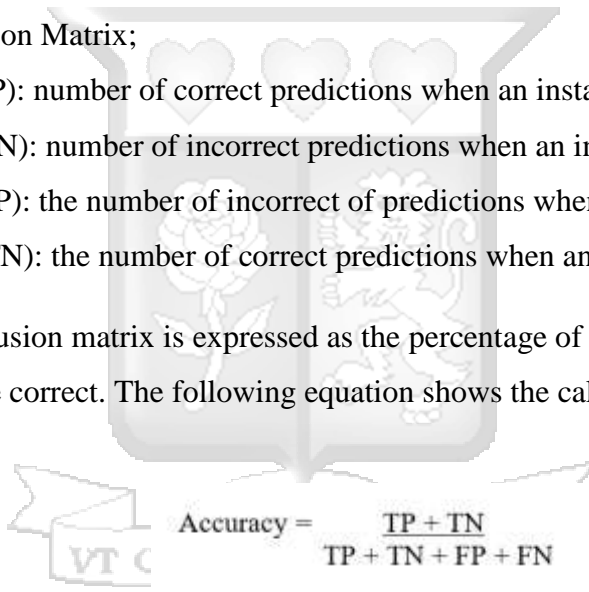
Table 3.1: Confusion Matrix

	Actual Positive Sentiment	Actual Negative Sentiment
Predicted Positive Sentiment	TP	FP
Predicted Negative Sentiment	FN	TN

Evaluation of the Confusion Matrix;

- True positives (TP): number of correct predictions when an instance is positive
- True negatives (TN): number of incorrect predictions when an instance is negative.
- False positives (FP): the number of incorrect of predictions when an instance is positive.
- False negatives (FN): the number of correct predictions when an instance is negative.

Accuracy: from the confusion matrix is expressed as the percentage of the total number of predictions found to be correct. The following equation shows the calculation;



$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Error Rate: Calculates how often the model is wrong.

$$\text{Error Rate} = \frac{\text{FP} + \text{FN}}{\text{Total}}$$

True Positive Rate or Recall: Shows the ratio of positive cases correctly predicted, calculated as:

$$\text{True Positive Rate} = \frac{\text{TP}}{\text{Actual Positive Values}}$$

False Positive Rate: This refers to the proportion of negatives cases incorrectly classified as positive. It is calculated as;

$$\text{False Positive Rate} = \frac{\text{FP}}{\text{Actual Negative Values}}$$

Precision: this is the proportion of the predicted positive cases that were correct. It is calculated using the following equation

$$\text{Precision} = \frac{\text{TP}}{\text{Predicted Positive Values}}$$

3.6.1 Receiver Operating Characteristics - ROC or Area Under the Curve - AUROC

Performance measurement is an element in machine learning. AUC - ROC curve is a tool used in measuring the classification problems at various thresholds settings. It is a probability curve and represents the degree or measure of separability. It tells how much model is capable of distinguishing between classes. The higher the value of the AUC, the more accurate the model is at predicting the negatives as negatives and the positives as so (Shirakawa et al., 2013).

A good model's AUC is close to 1, which shows it performs better and has a good measure of separability. A poor model has its AUC nearing 0 to means its measure of separability is worse. The model is predicting some negatives as positives and vice versa. When AUC is 0.5, it means model has no class separation capacity whatsoever (Shirakawa et al., 2013).

ROC curve is drawn by plotting with TPR against the FPR. TPR is plotted on the y-axis and FPR is plotted on the x-axis as shown in the Figure 3.2 below.

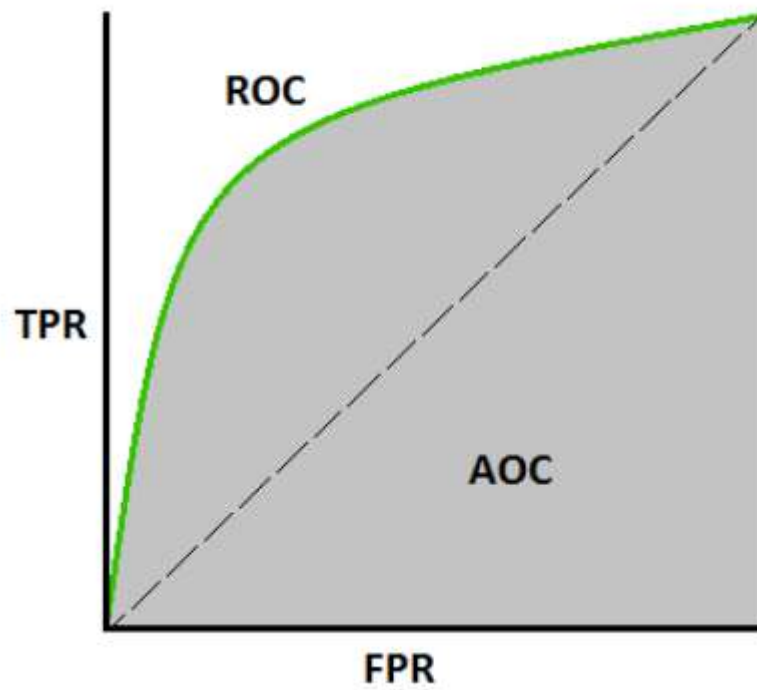
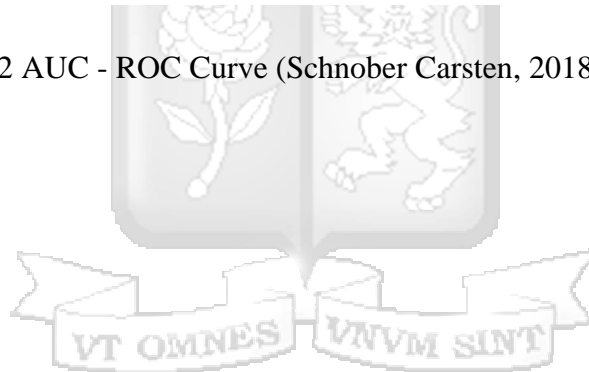


Figure 3. 2 AUC - ROC Curve (Schnober Carsten, 2018)



Chapter 4: System Design and Architecture

4.1 Introduction

This chapter reviews the proposed architecture, analysis and design of the cyber-bullying detection model. The system design and architecture were achieved through UML diagrams including; dataflow diagrams, use case diagrams and sequence diagrams. The diagrams give descriptions of the various components of the proposed system and their interactions at the various levels.

4.2 Requirement Analysis

This section outlines the various requirements that ought to be met by the study. The requirements are based on the research objectives and user requirements.

4.2.1 Functional requirements

The application should:

- i. Allow users to enter keywords to be used as search parameters in the retrieval of tweets,
- ii. Crawl tweets using the Twitter Search API, matching the keywords specified by the user,
- iii. preprocess and store retrieved tweets to a comma-separated values (csv) file,
- iv. perform feature weighting and represent the tweet in a document term matrix suitable for machine learning with weighting done in a similar manner to the one used in training the model,
- v. categorize tweets as containing cyberbullying or not,
- vi. give the user feedback and allow them to reclassify as appropriate

4.2.2 Non-Functional requirements

These are specifications which describe a system's operation properties and constraints that enhance functionality. They are classified based on the needs of the users and describe how well system achieves its functions. The non-functional requirements include;

- i. **Usability:** The proposed solution targets twitter users. The interaction with the model is near passive as the model takes the tweets being composed and analyses them, giving feedback. The model should therefore work seamlessly to enhance the user experience.

- ii. **Persistent Storage:** The system should provide permanent storage for tweets identified as hate speech. Such tweets may be used as required as evidence in prosecuting hate speech and as such must be retrievable as and when needed.
- iii. **Interoperability:** this refers to the degree to which the developed model will integration with the various social media platforms.
- iv. **Reliability:** The reliability of the model will highly depend on the accuracy of the data collected. The data is used to train the model, which will be used in the analysis of tweets being composed.
- v. **Response time:** defined as the amount of time from the moment a user sends a request until the request is completed or a response is received and the application indicates that the request is completed. For the proposed model, the response time for the model is between 0 and 3 seconds.

4.3 System Architecture

System architecture is the depiction of the system that aids in understanding its design and behavioral components. The proposed solution comprises of the crawler, the classifier, the machine learning predictor, pre-processor component and historical tweets data. The is initiated by user entering keywords to be used to retrieve matching tweets. The crawler captures the key words and collects tweets matching the keyword from Twitter Search API and stores them in a database. The tweets are then cleaned at the pre-processor.

From the prep-processor the tweets are converted to a document-term matrix suitable for machine learning algorithms using the feature processor. The tweets are classified as positive (1), neutral (0), or negative (-1). Based on this classification, tweets will be matched against the processed tweets in the database, reviewing them for cyber-bullying content. The result is presented to the user for review. Figure 4.1 illustrates the proposed architecture.

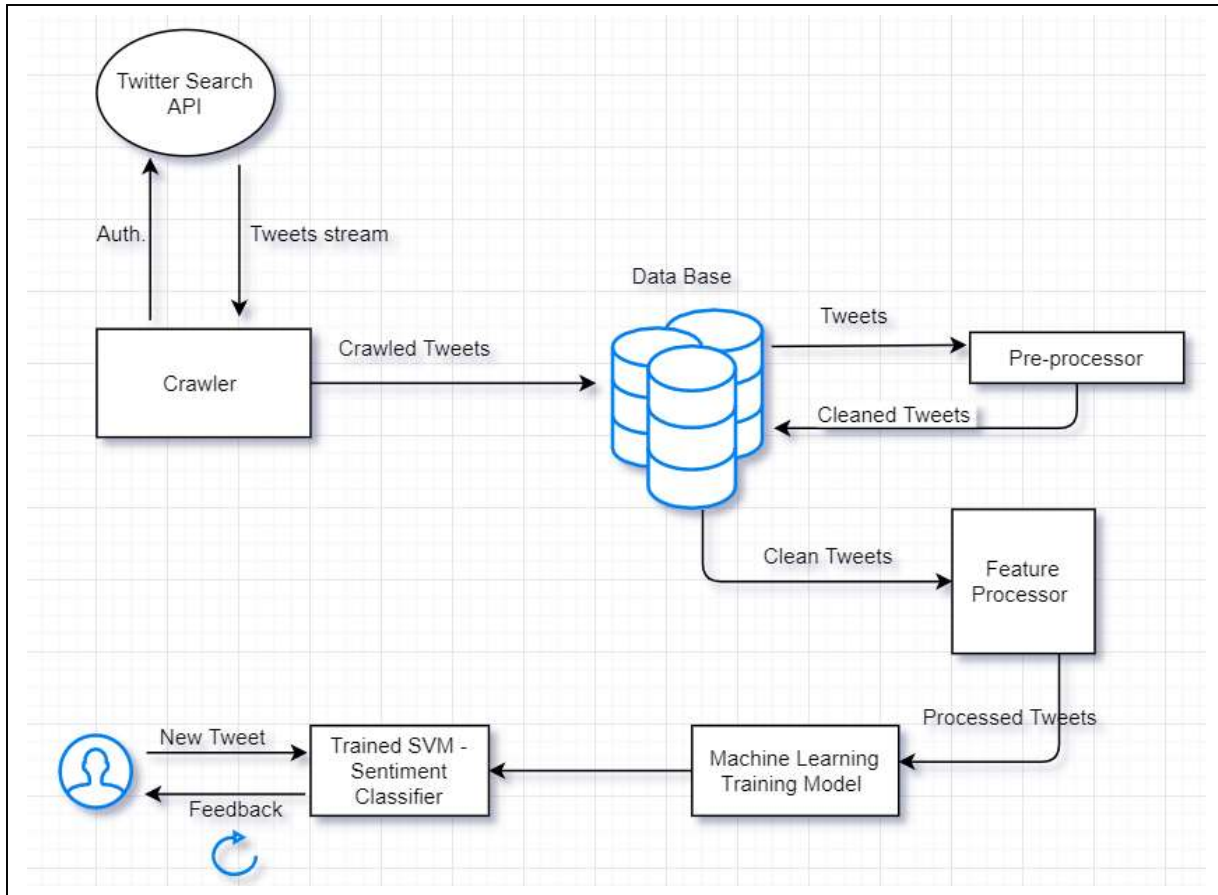


Figure 4.1: System Architecture

4.4 System Analysis

Refers to the process of decomposing the system into its components. It involves collecting and understanding the facts in order to achieve the set objectives. This is through the various tools discussed below.

4.4.1 Use Case Diagram

Use case diagrams, in the analysis phase, are used to depict the systems functionality and requirements, by showing the relations between actors and the system. This is done using the actors and the use cases. In the cyberbullying prevention model using sentiment analysis, the actors are; the twitter users, the prediction model and the Twitter Search API. Figure 4.2 illustrates how these actors and the cyberbullying prevention model will interact.

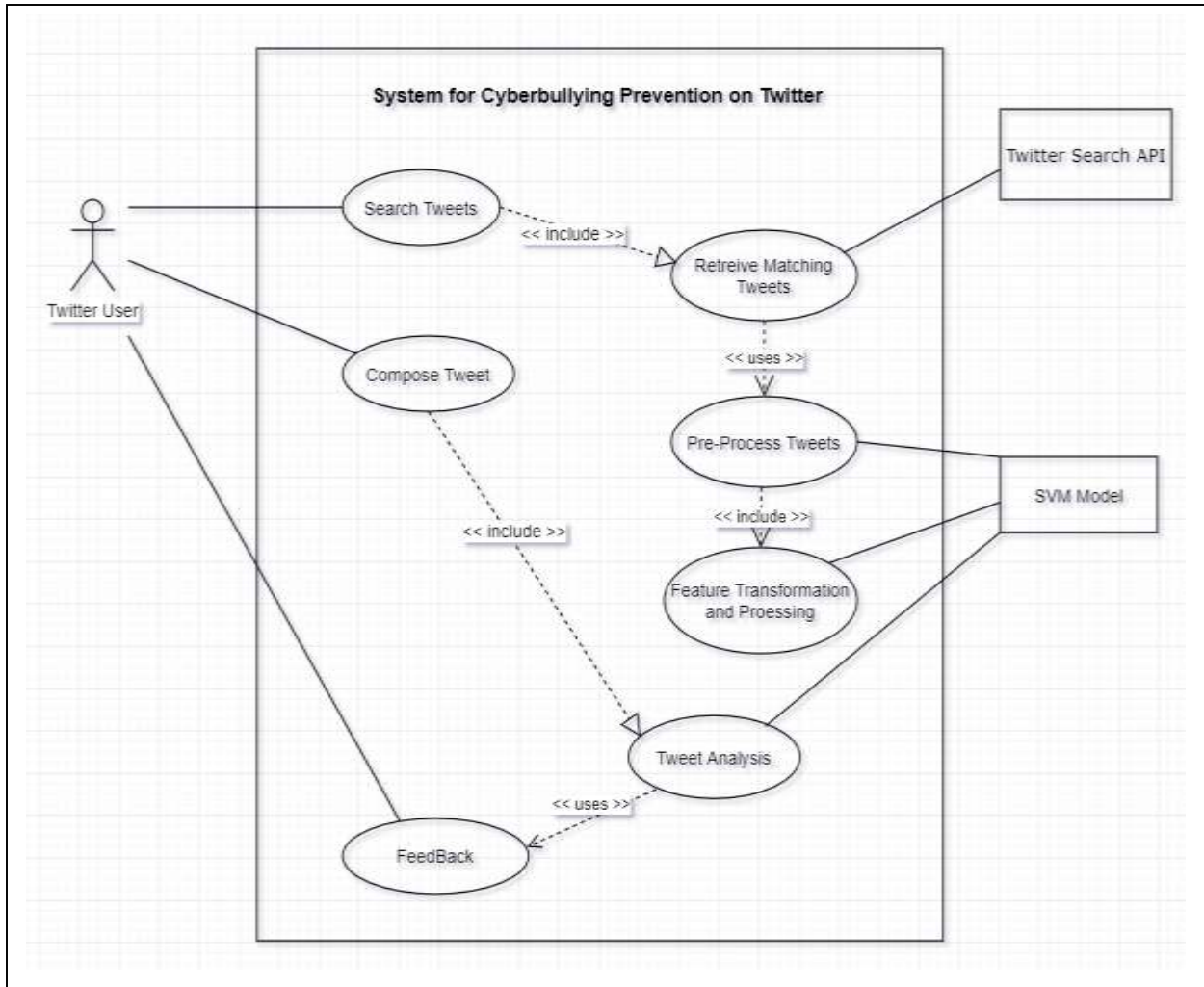


Figure 4.2: Use Case Diagram

4.4.1.1 Use Case Scenarios

These are the use case narratives. They are the step-by-step dialogues between actors and the system and are text-based. Use case scenarios comprehensively explain transaction processes. Detailed below is the use case scenario for the proposed model.

Use case Scenario 1 - Search and Crawl Tweets

Use case: Search and Crawl Tweets

Primary Actors:

User/Twitter user

Twitter APIs

Preconditions:

Tweets successfully collected and stored

Post conditions:

Accurately analyzed tweets of whether containing cyberbullying or not.

Main Success Scenario:

Actor Action System Responsibility

1. Enter keywords to be used for search
2. Capture keywords as parameters for tweets crawl
3. Fetch matching tweets from Search APIs using the entered parameters
4. Save tweets to a csv file
5. Access csv files with saved tweets

Extensions

When system fails to retrieve tweets: Confirm access to internet and restart the system.

Use case Scenario 2 – Pre-processing Tweets, Feature Transform processing, Tweets Analysis

Use case: Pre-processing Tweets, Feature Transform processing, Tweets Analysis

Primary Actors:

System

User/Twitter user

Preconditions:

Accurately analysis of tweets by the system

Post conditions:

Accurately analyzed tweets with matching feedback

Main Success Scenario:

Actor Action System Responsibility

1. User composes the tweets
2. The Model captures the tweet and analyses for cyberbullying content.
3. Transforms the tweet to a document term matrix, as per the SVM model
4. Analyses tweets for cyberbullying content
5. Return the content classification feedback to the user.
6. Exit the system

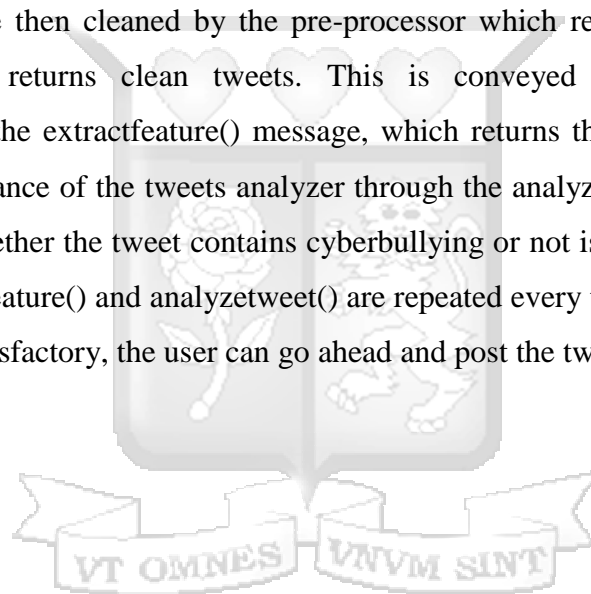
Extensions:

Anytime a user gets a negative review of their tweets, they should recompose the tweet to ensure cyberbullying is absent in the tweet.

4.4.2 Sequence Diagram

A Sequence diagram shows the interaction between objects of a system in a sequential manner. It also shows the relationship and interaction between the user and the system, in addition to interactions between the various components of the system. As illustrated in Figure 4.3, in the proposed system, the user will enter keywords, which form the search parameters used in crawling tweets containing cyberbullying.

The collected tweets are then cleaned by the pre-processor which receives the message from tweetscleaner(), which returns clean tweets. This is conveyed to an instance of the FeatureProcessor using the extractfeature() message, which returns the document term matrix. This is passed to an instance of the tweets analyzer through the analyzetweet() message and the analysis feedback of whether the tweet contains cyberbullying or not is returned. The messages; tweetscleaner(), extractfeature() and analyzetweet() are repeated every time a tweet is composed. Once the feedback is satisfactory, the user can go ahead and post the tweet.



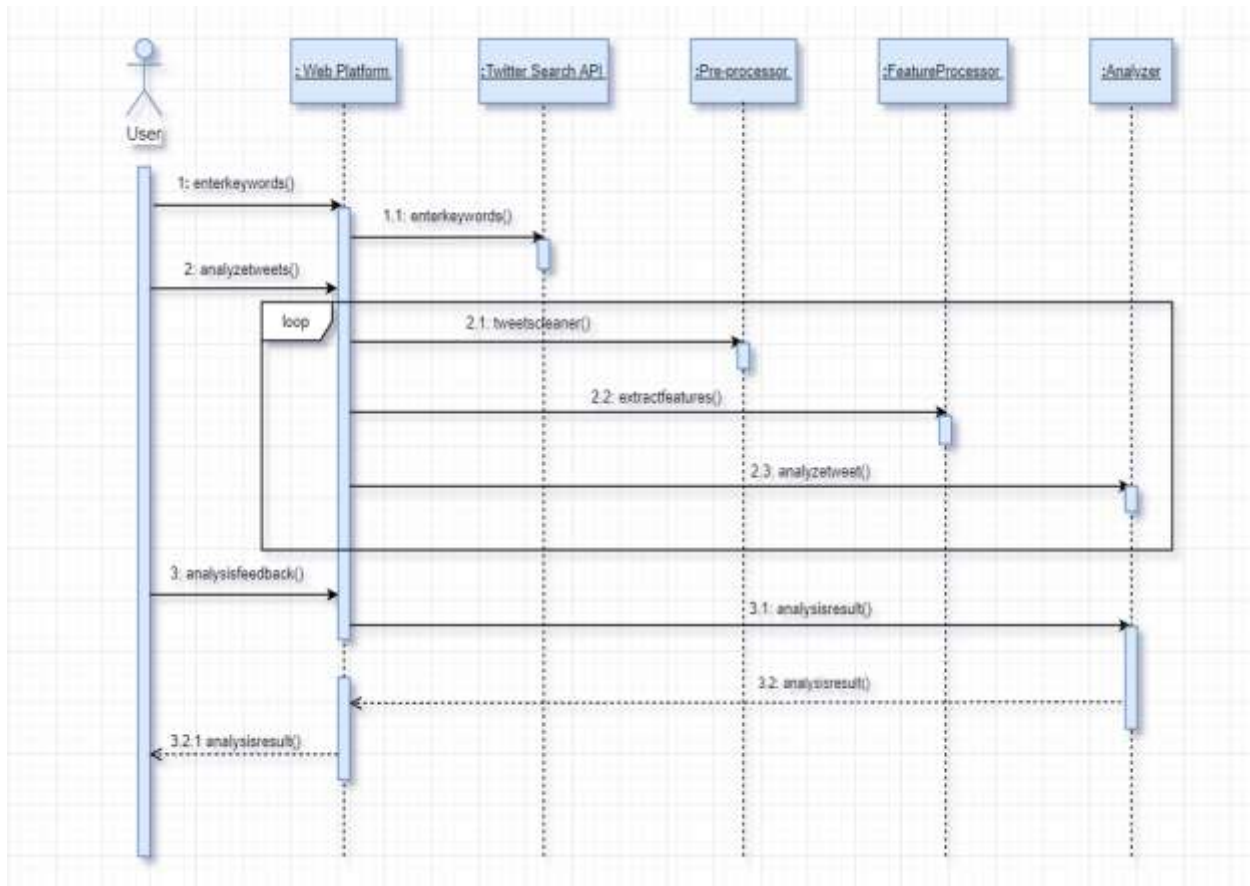


Figure 4.3: Sequence Diagram

4.5 System Design

System design comprises of the processes that define the architecture, the data and how it flows through the proposed model. The following tools designing the model.

4.5.1 Context Diagram

Figure 4.4 illustrates the context diagram showing the boundaries of the prototype model and the entities. The diagram graphically depicts what is inside the system and components interacting with the system from outside, and the relationships. The diagram shows the system as a single process, by indicating the flow of information. The main entities in the context diagram interacting with the system are, the user and the Twitter Search APIs. The user composes the tweets which go

through the twitter APIs. The model then picks the tweet and analyses it for cyberbullying content, then relays the feedback to the user. Once a tweet is cleared of containing any cyberbullying, the user can post.

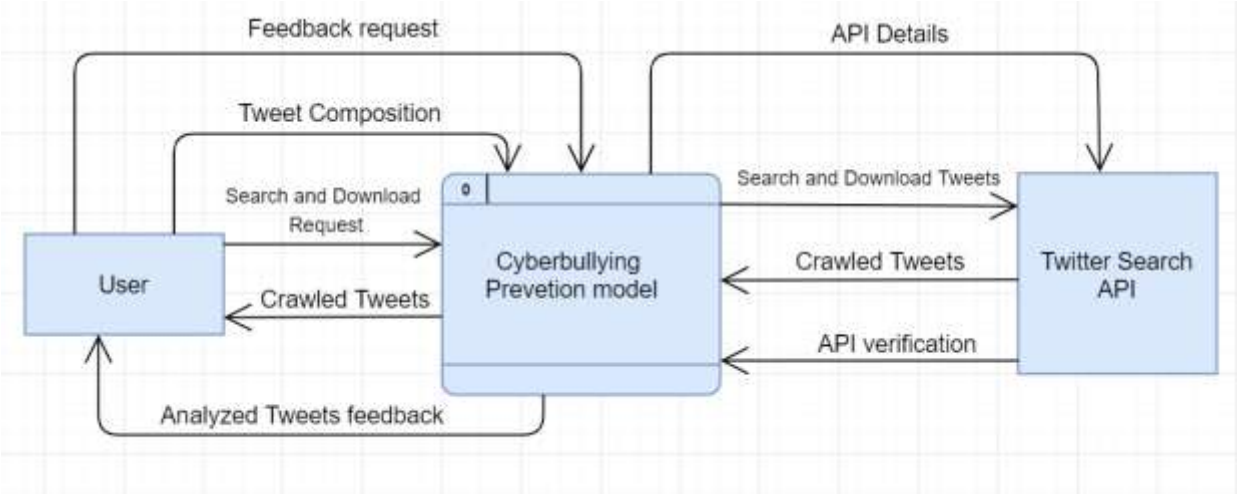


Figure 4.4: Context Diagram

4.5.2 Level 1 Data Flow Diagram

The Level 1 Data Flow Diagram in figure 4.5 is an expansion of the context diagram in figure 4.4. The diagram gives more details of the proposed model by showing the main processes within the model, the data stores and entities. It also shows the processes are related to each other. This allows for easier understanding of the system as both users and non-users easily understand how data is flowing through the system.

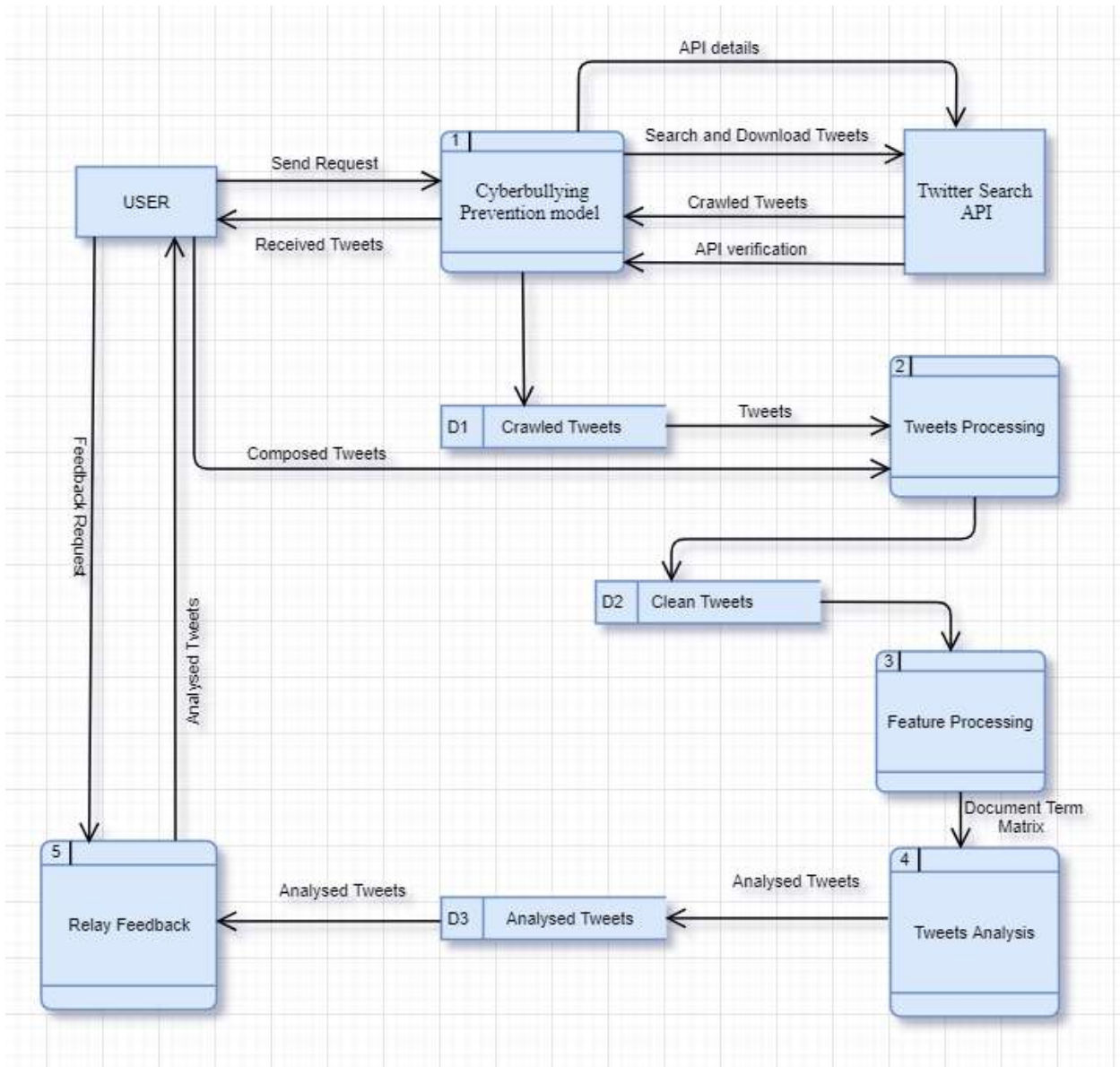


Figure 4.5: Level 1 Data Flow Diagram

Chapter 5: System Implementation and Testing

5.1 Introduction

This chapter explains the processes undertaken in the development of the model and its testing. It includes the sentiment analysis, context analysis, the data format and collection method. This was followed by the pre-processing and testing of the model. The model is tested with 20% of the data set to ensure the accuracy. Different experiments with various algorithms and features were carried out to come up with the best model.

5.2 Sentiment Analysis

5.2.1 Sentiment Corpus Building

Tweets were collected using key words identified to be mostly used in instances of cyberbullying. The tweets were exported to a comma-separated value (csv) file. Jefferson Henrique's open source code was used to collect the tweets. Figure 5.1 shows the code used in crawling the tweets and Figure 5.2 shows the tweets downloading process using the Jefferson Henrique's tool from the windows powershell terminal.



```
import codecs
import getopt
import sys

if sys.version_info[0] < 3:
    import got
else:
    import got3 as got

def main(argv):

    if len(argv) == 0:
        print('You must pass some parameters. Use \'-h\' to help.')
        return

    if len(argv) == 1 and argv[0] == '-h':
        f = open('exporter_help_text.txt', 'r')
        print(f.read())
        f.close()

        return

    try:
        opts, args = getopt.getopt(argv, "", ("username=", "near=", "within=", "since=", "until=", "querysearch=", "toptweets", "na

        tweetCriteria = got.manager.TweetCriteria()
        outputFileName = "output_got8.csv"

        for opt, arg in opts:
```

Figure 5.1: Jefferson Henrique's Crawling Code

```

Windows PowerShell
except arg:
TypeError: catching classes that do not inherit from BaseException is not allowed
PS C:\Users\User\Desktop\GetOldTweets-Thesis\GetOldTweets-python-master> python .\Exporter.py --username "aarpoc@uic.edu" --since 2018-01-01 --until 2016-12-31
Searching...

More 100 saved on file...
More 100 saved on file...
More 100 saved on file...
More 100 saved on file...
More 100 saved on file...
More 100 saved on file...
More 100 saved on file...
More 100 saved on file...
More 100 saved on file...

```

Figure 5.2: Tweets Crawl and Download process from Twitter

5.2.2 Tweets Pre-processing

The crawled data was in highly unstructured format, thus cannot not be used in machine learning modelling. The collected data contained username, date, retweets, favorites, mentions, hashtags, id and the permalink (the permanent tweet link). Figure 5.3 shows a snippet of the raw data collected.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	username;	date;	retweets;	favorites;	text;	mentions;	hashtags;	id;	permalink						
2	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
3	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
4	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
5	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
6	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
7	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
8	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
9	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
10	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
11	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
12	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
13	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
14	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
15	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
16	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
17	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
18	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;
19	;	;	;	;	;	;	;	;	;	;	;	;	;	;	;

Figure 5.3: Snippet of Raw Tweets Crawled

To build the corpus, the text bit of the tweets become the center of focus. The raw tweets are first harmonized to lowercase. This ensures the tweets classification process is not compromised. Other information in the tweets including; non-utf8 characters, URL links, Twitter signs and symbols,

usernames and mentions (@usernames), hashtags (#) and punctuations are then removed. The Figure 5.4 shows the tweets pre-processor code snippet, used in cleaning the data.

```

6 class Tcleaner:
7     def __init__(self):
8         self.remove_punctuations = str.maketrans('', '', string.punctuation)
9
10    def tweets_cleaner(self, tweet):
11        html_escaped = html.unescape(tweet)
12        comma_replacement = html_escaped.replace(';','')
13
14        lower_case_text = comma_replacement.lower() # Change all texts to lowercase
15
16        removed_url = re.sub(r'http\S+', '', lower_case_text) # Removing URL links
17
18        removed_hash_tag = re.sub(r'#\w+', '', removed_url) # Removing hashtags
19
20        removed_username = re.sub(r'@\w+\s?', '', removed_hash_tag) # Removing mentions and usernames
21
22        removed_retweet = removed_username.replace("rt", "", True) # Remove to retweet
23
24        removed_punctuation = removed_retweet.translate(self.remove_punctuations) # Removing punctuations marks
25
26        remove_g_t = removed_punctuation.replace("&gt;", "", True) # Removing spaces
27        remove_a_m_p = remove_g_t.replace("&#amp;", "", True)
28        final_text = remove_a_m_p
29        return final_text
30

```

Figure 5.4: Tweets Pre-processor Code

Figure 5.5 Shows a snippet of the resulting cleaned tweets.

```

1 Text
2 you bitches are so selfish and inconsiderate... u shouldve just left this one in the draft you oompa loompa w/box braids body based ass bitch
3 more bitches
4 you amazing beautiful talented gorgeous ass bitch i walked in love
5 im done reaching out to you bitches i'm done being the good friend fuck all that shit talk to yourself or them fake ass so called friends like gtf
6 them bitches that used to talk shit now are meth heads got ugly ass babies
7 while im on it plz get this to btw im fine oomfs dw i just want a hug cause im a clingy ass bitch
8 this bitch is literally insane or gives no fucks of your feelings and like kinda wanted to be nice cause you do shit for her and she is not giving all that up for your dumb ass questions
9 made a new twitter just for this shit cause dummy you really bitch made like you really came on here to talk about a whole female argue with a nigga my guy someone you actually can ducking fight weak ass bitch
10 oh ugly ass bitch
11 who tf are you
12 wack ass bitch
13 lmao come hang out with me in atlanta bitch i love yo ass
14 if you cheat you're a pussy ass bitch idgaf who you are
15 this is for rachel you big fat white nasty smelling fat bitch why you took me off the motherfuckin schedule with your triffin dirty white racist ass you big fat bitch oompa loompa body ass bitch
16 i saw a snake in my yard i threw a rake at it
17 oh nooooo luckily im a cheap ass bitch who does my own hair
18 me please college and roma be kicking me ass with bills
19 i cannot argue with no bum ass delusional bitch
20 hell naw they want long time girlfriends down ass bitches not wives
21 i seen my mama beat a racist white bitch ass once for calling me the n word i know she got hands
22 if a mfka sick as hell id be a dumb ass to stick around while they sick aint nun romantic about dying together bitch i love my life
23 die
24 but the bitch ass hoe who told me to calm down
25 the day you should have beat you boyfriend ass not tryna fight the next bitch then still got beat by her
26 i hate when tall ass bitches be tryna advertise themselves like they short knowing they tall big as madea ass pushing

```

Figure 5.5: Snippet of cleaned Tweets

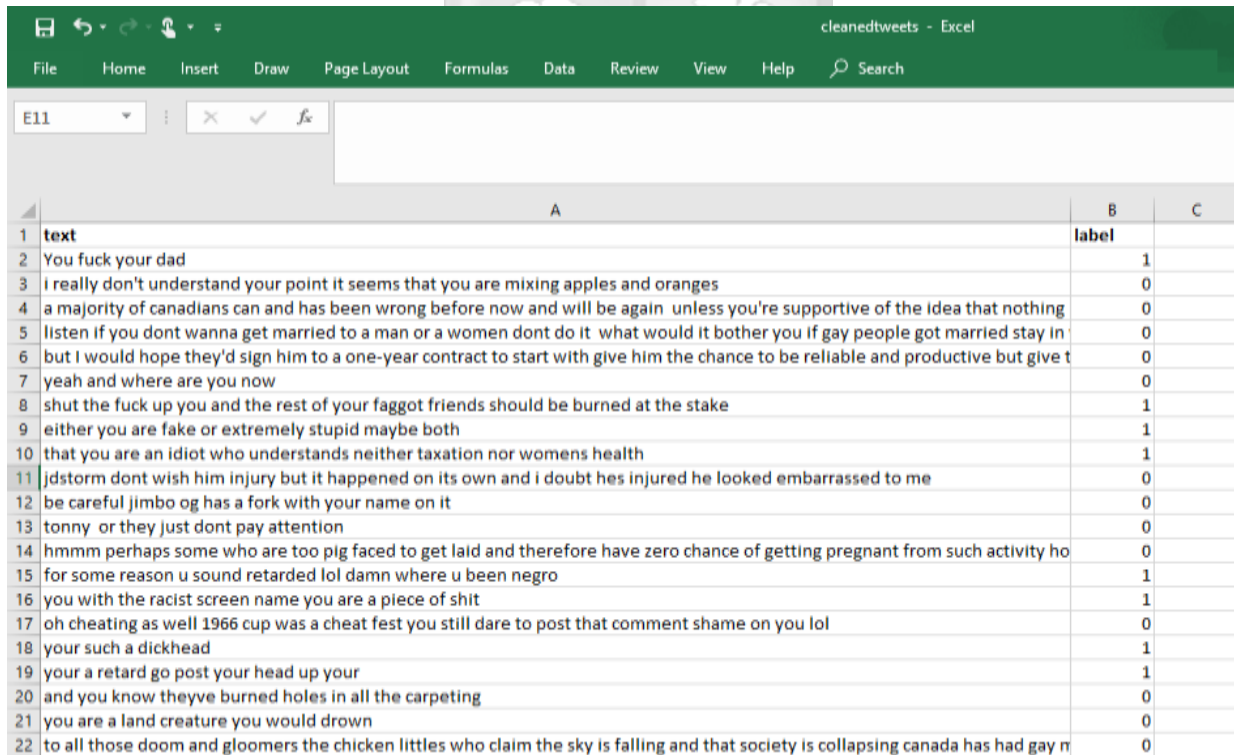
5.2.3 Tweets Labeling

Following pre-processing is the process of data labelling. The data is to be split into training and testing sets. This was a manual process which entailed reading the tweets one by one. The tweets were classified using the sentiment meaning of the tweets. The Table 5.1 below shows the labeling categories used for the data.

Table 5.1: Tweets Labelling Categories

Positive	1	Tweets contain cyberbullying
Negative	0	Tweets do not contain cyberbullying

The tweets are assigned either 1 for containing elements of cyberbullying, 0 for neutrality. Figure 5.6 shows a sample of the labelled tweets.



	A	B	C
1	text	label	
2	You fuck your dad	1	
3	I really don't understand your point it seems that you are mixing apples and oranges	0	
4	a majority of Canadians can and has been wrong before now and will be again unless you're supportive of the idea that nothing	0	
5	listen if you dont wanna get married to a man or a women dont do it what would it bother you if gay people got married stay in	0	
6	but I would hope they'd sign him to a one-year contract to start with give him the chance to be reliable and productive but give t	0	
7	yeah and where are you now	0	
8	shut the fuck up you and the rest of your faggot friends should be burned at the stake	1	
9	either you are fake or extremely stupid maybe both	1	
10	that you are an idiot who understands neither taxation nor womens health	1	
11	jdstorm dont wish him injury but it happened on its own and i doubt hes injured he looked embarrassed to me	0	
12	be careful jimbo og has a fork with your name on it	0	
13	tonny or they just dont pay attention	0	
14	hmmm perhaps some who are too pig faced to get laid and therefore have zero chance of getting pregnant from such activity ho	0	
15	for some reason u sound retarded lol damn where u been negro	1	
16	you with the racist screen name you are a piece of shit	1	
17	oh cheating as well 1966 cup was a cheat fest you still dare to post that comment shame on you lol	0	
18	your such a dickhead	1	
19	your a retard go post your head up your	1	
20	and you know theyve burned holes in all the carpeting	0	
21	you are a land creature you would drown	0	
22	to all those doom and gloomers the chicken littles who claim the sky is falling and that society is collapsing Canada has had gay n	0	

Figure 5.6: Snippet of Labelled Tweets

5.3 The Support Vector Machine Model training

From the pre-processing, the labelled dataset is then used in the creation of the model. The labeled set is first randomized then split into two; the training set which was 70% and the testing dataset, which is used in testing the model's accuracy, was 30%. The data was first converted to a document term matrix, since machine learning algorithms don't analyze texts in plain format. Figure 5.7 below shows the SVM training code.

```
def svm_trainer(X, y):  
  
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state = 1)  
  
    svm = Pipeline([('vect', CountVectorizer()), ('tfidf', TfidfTransformer()),  
                   ('svm', SVC(kernel="linear", C=1))])  
  
    svm = svm.fit(X_train, y_train)  
    ypred = svm.predict(X_test)  
  
    print("SVM Trained Metrics")  
    print(metrics.accuracy_score(y_test, ypred))  
    print(metrics.classification_report(y_test, ypred))  
    drawrocSVM(y_test, ypred)
```

Figure 5.7: SVM Model Training

5.3.1 Support Vector Machine Model n-gram Experiments

SVM experiments were then carried out using different n-gram features. Bigram feature provided the edge in accuracy compared to unigram, and trigram. This experiment was also carried out on the other classifiers including K-Nearest Neighbor, SVM, Naïve Bayes and Random Forest. The Figure 5.8 below shows a snippet of a section of the code used in this experiment.

```

def svm_ngram(X, y):
    """Different features with 13M"""
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
    svm = createSVM(X_train, y_train)
    y_pred = (svm.predict(X_test))
    print("Original Accuracy: 13M with tf-idf")
    print(metrics.confusion_matrix(y_test, y_pred))
    print(metrics.accuracy_score(y_test, y_pred))
    print(metrics.classification_report(y_test, y_pred))
    svm2 = Pipeline([('count', CountVectorizer()), ('svm', SVC(kernel="linear", C=1))])
    svm2 = svm2.fit(X_train, y_train)
    ypred2 = svm2.predict(X_test)
    print("Just unigram counts Accuracy")
    print(metrics.accuracy_score(y_test, ypred2))
    print(metrics.classification_report(y_test, ypred2))
    svm3 = Pipeline([('count', CountVectorizer(ngram_range=(1, 2))), ('svm', SVC(kernel="linear", C=1))])
    svm3 = svm3.fit(X_train, y_train)
    ypred3 = svm3.predict(X_test)
    print("Just bigram counts Accuracy")
    print(metrics.accuracy_score(y_test, ypred3))
    print(metrics.classification_report(y_test, ypred3))
    svm4 = Pipeline([('count', CountVectorizer(ngram_range=(1, 3))), ('svm', SVC(kernel="linear", C=1))])
    svm4 = svm4.fit(X_train, y_train)
    ypred4 = svm4.predict(X_test)
    print("Trigram counts Accuracy")
    print(metrics.accuracy_score(y_test, ypred4))
    print(metrics.classification_report(y_test, ypred4))
    svm5 = Pipeline([('count', CountVectorizer(ngram_range=(1, 2))), ('tfidf', TfidfTransformer()),
                    ('svm', SVC(kernel="linear", C=1))])

```

Figure 5. 8 N-gram SVM experiment.

The results of the experiment are discussed further in chapter six section 6.2 under Sentiment Analysis Experiments and Results.

5.3.2 Testing the Model

To test the model, 30% of the labelled dataset was used. The data was collected, cleaned, labelled and parsed through the model. The test data was ran against the trained model and the outcome is compared with the labeled set. Figure 5.9 shows the code printing out the SVM test metrics.

```

svm = svm.fit(X_train, y_train)
ypred = svm.predict(X_test)

print("SVM Model Metrics")
print(metrics.accuracy_score(y_test, ypred))
print(metrics.classification_report(y_test, ypred))
drawrocSVM(y_test, ypred)

```

Figure 5.9: Printing SVM test Metrics

The resulting confusion matrix is shown in the table below. Values for true positive, false positive, true negative and false negatives from the confusion matrix are shown in the Table 5.2 below.

Table 5.1: Confusion Matrix for the Trained Model

	Actual 0 - Negatives	Actual 1 – Positives
Predicted 0 (Negative)	241	301
Predicted 1 (Positive)	85	1470

Table 5.3 below shows the metrics; accuracy, recall, precision and the F-score calculated from the values in Table 5.2. The model's computed accuracy is 81%.

Table 5.2: Derived Values from the Confusion Matrix

True Positive - TP	1470
False Positive – FP	301
True Negative - TN	241
False Negative - FN	85

The Resulting ROC curve is shown in the Figure 5.10 below.

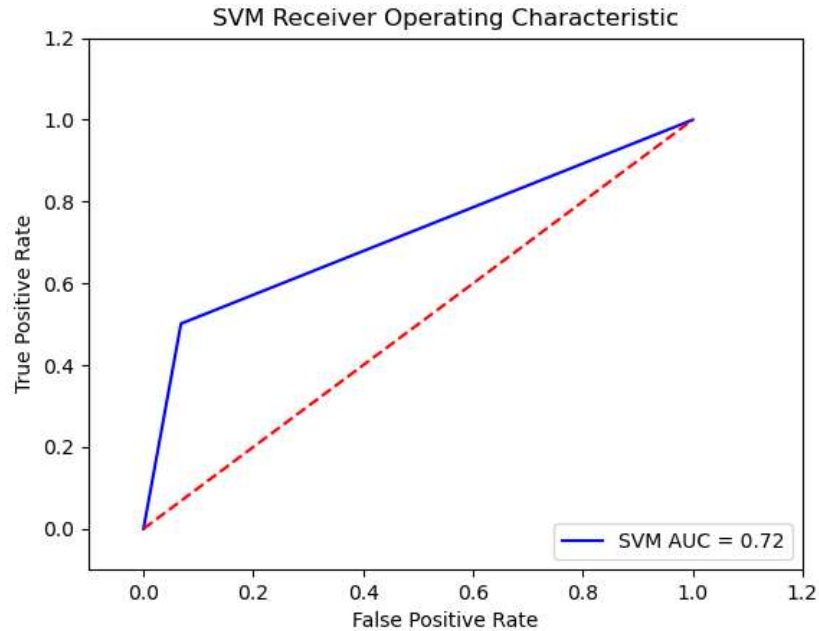


Figure 5.10: SVM Test ROC Curve

5.4 Model Application

The model was also persisted to ensure it is available for use every time. To achieve this, joblib library of sklearn was used. Figure 5.8 illustrates the code which was used.

```

54
55 from sklearn.externals import joblib
56 def svm_trainer(clf):
57     joblib.dump(clf, 'model.pk1')
58

```

Figure 5.11: SVM Persistence Code Snippet

5.4.1 Application Integration

To achieve its purpose, the model needs to capture tweets as they are being composed. To simulate this process, an application that simulates tweets composition is developed. The model is to be integrated with the application and sits right below the text composition action in the logic. A listener captures the text being composed and it is run through the model. When the model finds instance of cyberbullying in the text, a pop-up message appears alerting the user to recompose the tweet, when the result is otherwise, the user can proceed to post the message. Figure 5.12 shows the wireframe of the integration.

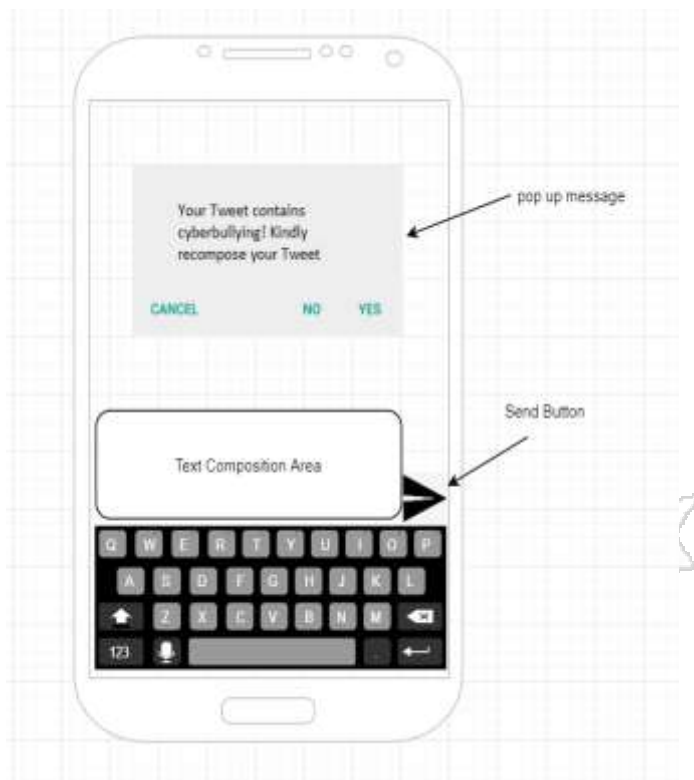


Figure 5.12: Wireframe of Model Integration into Chat Application

Chapter 6: Discussions

6.1 Introduction

This chapter delves into the experiments that were carried out during this research in order to achieve the set objectives. The principal objective was to come up with a prototype of a model, which can be used to check elements of cyberbullying in tweets as they are being composed, in order to arrest them, before they are posted to the public. As such an SVM model, using bigram features, emerged as the better for sentiment analysis, compared to other machine learning methods.

6.2 Sentiment Analysis Experiments and Results

6.2.1 Use of Different Classifiers

Table 6.1 shows a summary of the results, of the performance of the different classifiers for sentiment analysis. K-Nearest Neighbor, SVM, Naïve Bayes and Random Forest were the learning methods experimented with.

Table 6.1: Classifiers' Performance Results Summary

Classifier	Precision	Recall	F-Score	Accuracy
KNN	0.80	0.81	0.80	0.8051
SVM	0.80	0.81	0.80	0.8101
Naïve Bayes	0.81	0.73	0.64	0.7333
Random Forest	0.81	0.79	0.75	0.7873

Below are the individual results of each of the classifiers;

6.2.1.1 Using K-Nearest Neighbor (KNN)

Table 6.2 shows the results from using KNN.

Table 6.2: KNN Performance Results

```
KNN Metrics
0.8050632911392405
      precision    recall  f1-score
0      0.84      0.90      0.87
1      0.68      0.58      0.63

accuracy          0.81
macro avg         0.76      0.74      0.75
weighted avg      0.80      0.81      0.80
```

The Receiver Operating Characteristic (ROC) curve in figure 6.1 shows how the KNN classifier performed.

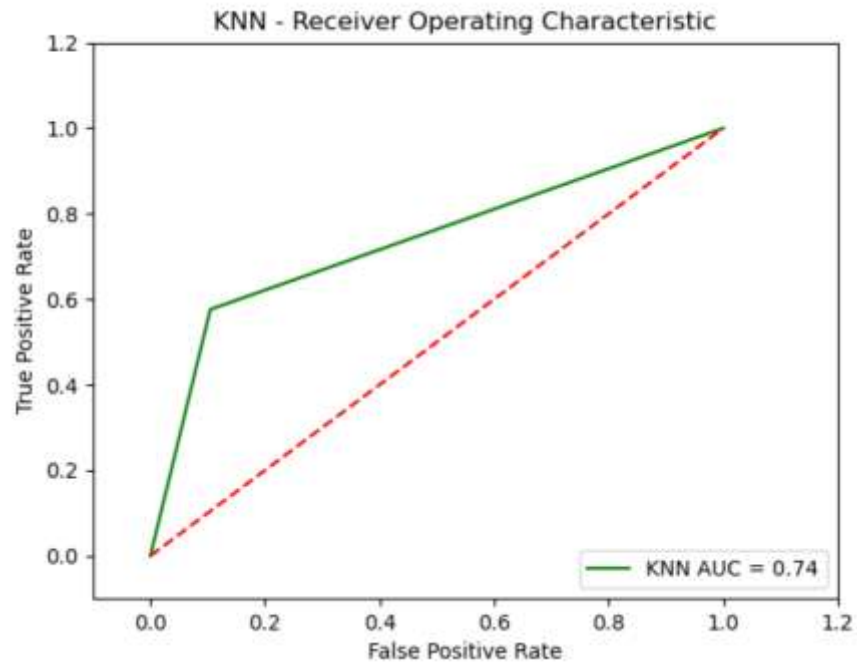


Figure 6.1: KNN ROC curve

6.2.1.2 Using SVM Classifier

Table 6.3 shows the Results while using SVM classifier.

Table 6.3: SVM Performance Results

```
SVM Model Metrics
0.810126582278481
      precision    recall  f1-score   support

0         0.83      0.93      0.88         10
1         0.74      0.50      0.60         10

accuracy          0.81
macro avg         0.78      0.72      0.74
weighted avg      0.80      0.81      0.80
```

The Receiver Operating Characteristic (ROC) curve in figure 6.2 shows the performance of the SVM classifier

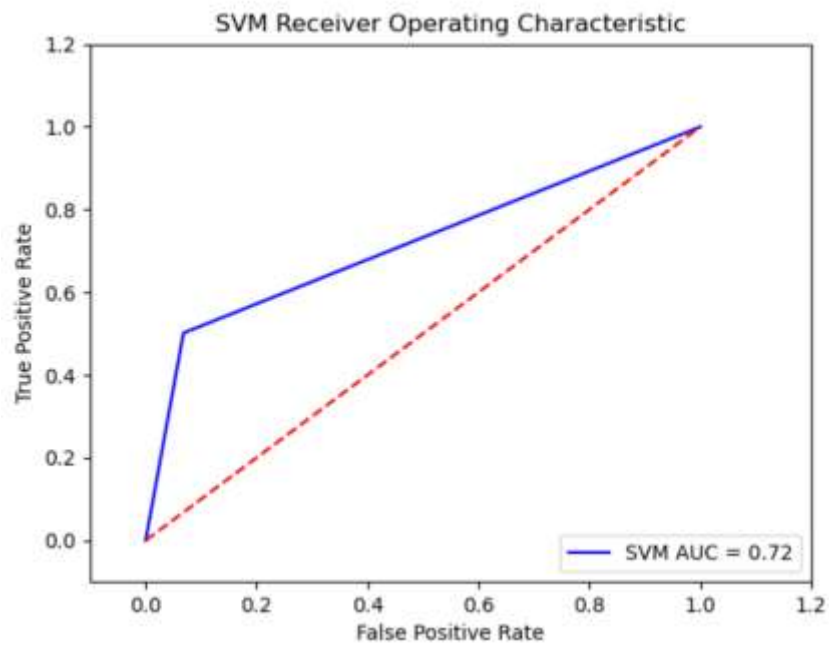


Figure 6.2: SVM ROC curve

6.2.1.3 Using Naïve Bayes Classifier

Table 6.4 shows the Results while using Naïve Bayes classifier.

Table 6.4: Naïve Bayes Performance Results

```
Naive Bayes Metrics
0.7333333333333333
      precision    recall  f1-score   \

0         0.73         1.00         0.84
1         1.00         0.06         0.11

 accuracy         0.73
 macro avg         0.86         0.53         0.48
 weighted avg         0.81         0.73         0.64
```

The Receiver Operating Characteristic (ROC) curve in figure 6.3 shows the performance of the Naïve Bayes classifier

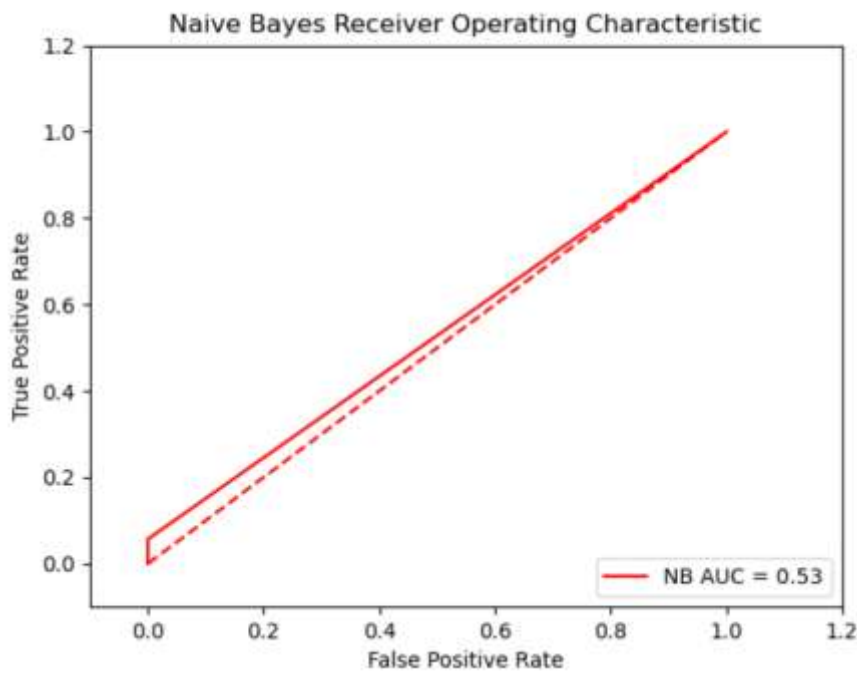


Figure 6.3: Naïve Bayes ROC curve

6.2.1.4 Using Random Forest Classifier

Table 6.5 shows the Results while using Random Forest Classifier

Table 6.5: Random Forest Classifier Performance Results

```
0.7873417721518987
      precision    recall  f1-score
0      0.78      0.99      0.87
1      0.89      0.28      0.43

 accuracy          0.79
 macro avg      0.83      0.63      0.65
 weighted avg   0.81      0.79      0.75
```

The Receiver Operating Characteristic (ROC) curve in figure 6.2 shows the performance of the Random Forest Classifier

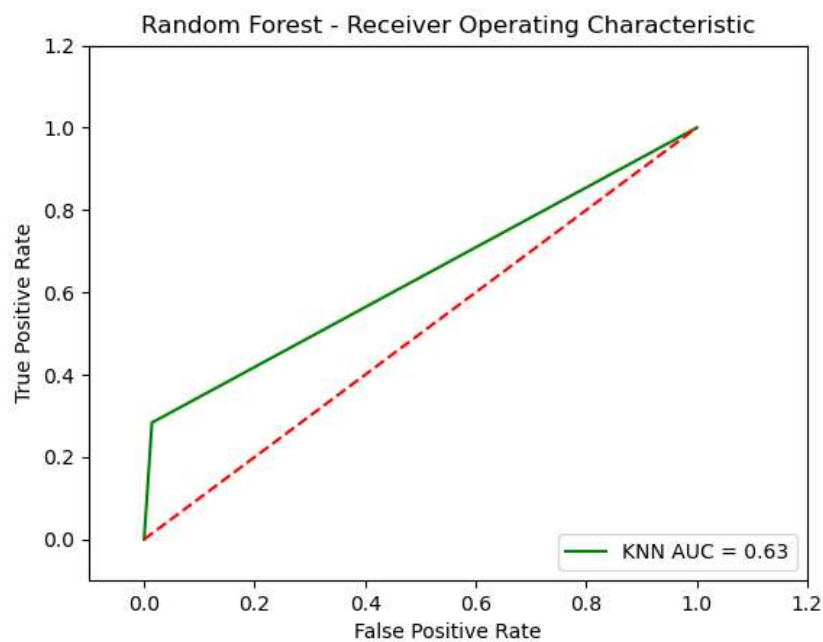


Figure 6.4: Random Forest ROC curve

6.2.2 Using Different Feature Types on the Classifiers

The motivation of the experiment was to find out the effects of using different feature types on the different machine learning methods, and the accuracy of the models. Features experimented with were unigrams, bigrams and trigrams.

6.2.2.1 KNN Experiment with Different Feature Types

Table 6.6 shows the results from the experiment. The best performance was achieved when using the bigram feature.

Table 6.6: Performance of KNN

Feature	Precision	Recall	F-Score	Accuracy
Unigram	0.72	0.73	0.72	0.7257
Bigram	0.71	0.73	0.72	0.7308
Trigram	0.70	0.73	0.71	0.7274

6.2.2.2 SVM Experiment with Different Feature Types

Table 6.7 shows the results achieved from the experiment. The best performance was achieved when using the bigram feature.

Table 6.7: SVM Performance with different feature types

Feature	Precision	Recall	F1-Score	Accuracy
Unigram	0.82	0.82	0.82	0.8245
Bigram	0.84	0.84	0.84	0.8422
Trigram	0.83	0.83	0.83	0.8329

6.2.2.3 Naïve Bayes Experiment with Different Feature Types

Table 6.8 shows the results while using feature types on Naïve Bayes. The best performance was achieved when using the unigram feature.

Table 6.8: Naïve Bayes Performance with different feature types

Feature	Precision	Recall	F-Score	Accuracy
Unigram	0.80	0.81	0.80	0.8110
Bigram	0.79	0.80	0.64	0.7992
Trigram	0.80	0.79	0.76	0.7941

6.2.2.4 Random Forest Experiment with Different Feature Types

The same experiment was conducted using Random Forest classifier and Feature Types. Table 6.9 shows the results while using feature types on Naïve Bayes. The best performance was achieved when using the unigram feature.

Table 6.9: Random Forest Performance with different feature types

Feature	Precision	Recall	F-Score	Accuracy
Unigram	0.81	0.81	0.78	0.8093
Bigram	0.81	0.80	0.76	0.7992
Trigram	0.81	0.79	0.74	0.7890

6.3 Comparative Analysis

6.3.1 Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. This metric looks the ratio to which the predicted values are actually correct. From the experiments carried out, Naïve Bayes and Random Forest algorithms give a precision score of about 0.81, which is pretty good. SVM and KNN score 0.81 on the same metric.

When the n-gram feature experiments were carried out, the results tilt in favor of SVM across the unigram, bi-gram and tri-gram features tested; with the results being 0.82, 0.84 and 0.83 respectively. This compared to the other algorithms, which had scores 0.81 and below, gives SVM the best score in precision metric.

6.3.2 Recall (Sensitivity)

This metric refers to the ratio of correctly predicted positive observations to the all observations in actual class. It helps answer the question, of all the messages predicted, how many were actually labelled? A recall score of above 0.5 is considered good.

Without any features implemented, both SVM and KNN had a recall of 0.81. Naïve Bayes had 0.73 while Random Forest scored 0.79. This sets SVM and KNN apart in this study. However, when the feature types are implemented, SVM again score better across all the feature types tested, the unigram, bi-gram and tri-gram; with the results being 0.82, 0.84, and 0.83 respectively. Naïve Bayes and Random Forest come second with a tied score of 0.81, 0.80 and 0.79 across the three feature types respectively. This gives SVM the best score in recall/sensitivity.

6.3.3 Accuracy

This metric is the most intuitive performance measure. It is the ratio of correctly predicted observation to the total observations. The algorithms posted different accuracy levels. SVM had an accuracy of 0.8101, followed by KNN with 0.8051, Random Forest with 0.7873 while Naïve Bayes had 0.733.

When the n-gram features are applied to the models, SVM posts the best accuracy across the implemented feature types; unigram, bi-gram and tri-gram, with the results being, 0.8242, 0.8422 and 0.8329 respectively. It is followed closely by Naïve Bayes which posted; 0.8110, 0.7992, and 0.7941 respectively. KNN was the least accurate algorithm with scores of 0.7257, 0.7308 and 0.7274 respectively. SVM again gave the best accuracy with the feature bi-gram implemented with a score of 0.8422, which translates to about 84% level of accuracy. This therefore informs the decision to settle on SVM as the classifier of choice in this study.

Chapter 7: Conclusions and Recommendations

7.1 Conclusion

The primary objective of this research was to develop a model which can be used to detect instances of cyberbullying in tweets as they are being composed to ensure they are rectified before those tweets are posted to public audiences or intended recipients. To successfully achieve this, it was imperative to understand the different forms of cyberbullying that manifest on social media especially Twitter, the platform in which the research was focused. Similar researches and literature were therefore studied to understand cyberbullying and the implementation of machine learning techniques in text classification and sentiment analysis.

To achieve the main objective, cyberbullying data was crawled from Twitter using the Henrique Jefferson code available on GitHub. The data was then preprocessed and labelled. The data was then split into the training set and the testing set. These were used to carry out different experiments using different classifiers to determine with machine learning methods would give the best performing model. SVM model provided the more desirable results using the bigram features, hence settled on. The model was then persisted before being integrated with a chat application to test prediction of new texts being composed.

7.2 Recommendations

This research showed that cyberbullying can be arrested at the point of origin, when the texts are being composed, before being posted to the public platforms. The model can therefore be adopted in the current campaign efforts against cyberbullying.

A total of 63,560 tweets were collected, only 6,990 tweets were used, a process curtailed by the laborious and expensive process of labelling the tweets. For better prediction, larger data sets should have been used. The researcher recommends increasing the size of both the training and testing data sets to improve the performance of the model.

7.3 Future Work

Future work list should include the following; more features., working with emoticons, using other forms of model evaluation. Expanding the model to cover other languages, as primarily this model concentrated on only English composed tweets.

Twitter allows up to 280 characters for the composition of tweets. This has led to constant evolution in the way people communicate. New word abbreviations and context usages arise. It is therefore imperative that tracking of these new forms is implemented to ensure detection of cyberbullying even in the new forms. This will require the implementation of continual learning.



REFERENCES

- Andries, P. E. (2007). *Computational Intelligence: An Introduction* (2 Edition). Wiley Publishing.
- Applications for Python | Python.org. (n.d.). Retrieved July 29, 2019, from <https://www.python.org/about/apps/>
- Araújo, M. (2017). What are RAD, Framework and IDE – concepts and applicability | Scriptcase Blog - Development, Web Design, Sales and Digital Marketing. Retrieved July 29, 2019, from <https://www.scriptcaseblog.net/development/rad-framework-ide-conceptsapplicability/>
- Beynon-Davies, P., Came, C., Mackay, H., & Tudhope, D. (1999). Rapid application development (Rad): An empirical review. *European Journal of Information Systems*, 8(3), 211–232. <https://doi.org/10.1057/palgrave.ejis.3000325>
- Bhatnagar, M., & Kumar Singh, P. (2013). Research Methodology as SDLC Process in Image Processing. *International Journal of Computer Applications*, 77(2), 43–45. <https://doi.org/10.5120/13369-0972>
- Burdi, F., Setianingrum, A. H., & Hakiem, N. (2017). *Application of the Naive Bayes Method to a Decision Support System to Provide Discounts (Case Study: PT. Bina Usaha Teknik)*. 281–285. <https://doi.org/10.1109/ict4m.2016.064>
- Cheng, L., Li, J., Silva, Y. N., Hall, D. L., & Liu, H. (2019). *XBully: Cyberbullying Detection within a Multi-Modal Context*. 19, 339–347. <https://doi.org/10.1145/3289600.3291037>
- Cohen-Almagor, R. (2011). Fighting Hate and Bigotry on the Internet. *Policy & Internet*, 3(3), 89–114. <https://doi.org/10.2202/1944-2866.1059>
- Cowie, H. (2013). Cyberbullying and its impact on young people’s emotional health and well-being. *Psychiatrist*, 37(5), 167–170. <https://doi.org/10.1192/pb.bp.112.040840>
- Cyberbullying: What is it and how to stop it* | UNICEF. (n.d.). Retrieved July 12, 2020, from <https://www.unicef.org/end-violence/how-to-stop-cyberbullying>
- Del Bosque, L. P., & Garza, S. E. (2016). Prediction of Aggressive Comments in Social Media:

An Exploratory Study. *IEEE Latin America Transactions*, 14(7), 3474–3480.

<https://doi.org/10.1109/TLA.2016.7587657>

Dong, P., Peng, H., Cheng, X., Xing, Y., Zhou, X., & Huang, D. (2019). A Random Forest Regression Model for Predicting Residual Stresses and Cutting Forces Introduced by Turning IN718 Alloy. *2019 IEEE International Conference on Computation, Communication and Engineering, ICCCE 2019*, 5–8.

<https://doi.org/10.1109/ICCCE48422.2019.9010767>

Guelton, S. (2018). Pythran: Crossing the python frontier. *Computing in Science and Engineering*, 20(2), 83–89. <https://doi.org/10.1109/MCSE.2018.021651342>

Hamiza Wan Ali, W. N., Mohd, M., & Fauzi, F. (2019). Cyberbullying Detection: An Overview. *Proceedings of the 2018 Cyber Resilience Conference, CRC 2018*, 1–3.

<https://doi.org/10.1109/CR.2018.8626869>

Henrique Jefferson. (2018). *GitHub - Jefferson-Henrique/GetOldTweets-python: A project written in Python to get old tweets, it bypass some limitations of Twitter Official API.*

<https://github.com/Jefferson-Henrique/GetOldTweets-python>

Jain, A. P., & Katkar, V. D. (2016). Sentiments analysis of Twitter data using data mining.

Proceedings - IEEE International Conference on Information Processing, IICIP 2015, 807–

810. <https://doi.org/10.1109/INFOP.2015.7489492>

Jayasekara Dilan. (2019). *Extracting Twitter Data, Pre-Processing and Sentiment Analysis using*

Python 3.0. <https://towardsdatascience.com/extracting-twitter-data-pre-processing-and-sentiment-analysis-using-python-3-0-7192bd8b47cf>

Mercado, R. N. M., Chuctaya, H. F. C., & Gutierrez, E. G. C. (2018). Automatic cyberbullying detection in spanish-language social networks using sentiment analysis techniques.

International Journal of Advanced Computer Science and Applications, 9(7), 228–235.

<https://doi.org/10.14569/IJACSA.2018.090733>

Mody, A., Shah, S., Pimple, R., & Shekokar, N. (2018). Identification of Potential Cyber Bullying Tweets using Hybrid Approach in Sentiment Analysis. *3rd International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques, ICEECCOT 2018*, 878–881.

<https://doi.org/10.1109/ICEECCOT43722.2018.9001476>

Nadali, S., Murad, M. A. A., Sharef, N. M., Mustapha, A., & Shojaee, S. (2013). A review of cyberbullying detection: An overview. *2013 13th International Conference on Intelligent Systems Design and Applications*, 325–330. <https://doi.org/10.1109/ISDA.2013.6920758>

National KE-CIRT/CC Stop Cyberbullying. (n.d.). Retrieved July 13, 2020, from

<https://www.ke-cirt.go.ke/index.php/stop-cyberbullying/>

Noviantho, Isa, S. M., & Ashianti, L. (2018). Cyberbullying classification using text mining. *Proceedings - 2017 1st International Conference on Informatics and Computational Sciences, ICICoS 2017, 2018-Janua*, 241–245.

<https://doi.org/10.1109/ICICOS.2017.8276369>

Nurrahmi, H., & Nurjanah, D. (2018). Indonesian Twitter Cyberbullying Detection using Text Classification and User Credibility. *2018 International Conference on Information and Communications Technology, ICOIACT 2018, 2018-Janua*, 543–548.

<https://doi.org/10.1109/ICOIACT.2018.8350758>

Potha, N., & Maragoudakis, M. (2015). Cyberbullying detection using time series modeling. *IEEE International Conference on Data Mining Workshops, ICDMW, 2015-Janua*(January), 373–382. <https://doi.org/10.1109/ICDMW.2014.170>

pydata.org. (2018). *Python Data Analysis Library — pandas: Python Data Analysis Library*.

<https://pandas.pydata.org/>

Schnober Carsten. (2018). *A Library for Many Jobs » Towards Data Science*.

<https://www.admin-magazine.com/HPC/Articles/Parallel-Python-with-Joblib>

Shirakawa, M., Nakayama, K., Hara, T., & Nishio, S. (2013). Probabilistic semantic similarity measurements for noisy short texts using wikipedia entities. *International Conference on Information and Knowledge Management, Proceedings*, 903–908.

<https://doi.org/10.1145/2505515.2505600>

Sugandhi, R., Pande, A., Chawla, S., Agrawal, A., & Bhagat, H. (2016). Methods for detection of cyberbullying: A survey. *International Conference on Intelligent Systems Design and Applications, ISDA, 2016-June*, 173–177. <https://doi.org/10.1109/ISDA.2015.7489220>

TIJANA MILOSEVIC. (2016). Social Media Companies' Cyberbullying Policies.: EBSCOhost. *International Journal of Communication*, 10(2016), 5164–5185.

[http://web.b.ebscohost.com.er.lib.k-](http://web.b.ebscohost.com.er.lib.k-state.edu/ehost/pdfviewer/pdfviewer?vid=0&sid=4880879f-024a-46e7-aa1a-21abed0a948e%40sessionmgr101)

[state.edu/ehost/pdfviewer/pdfviewer?vid=0&sid=4880879f-024a-46e7-aa1a-21abed0a948e%40sessionmgr101](http://web.b.ebscohost.com.er.lib.k-state.edu/ehost/pdfviewer/pdfviewer?vid=0&sid=4880879f-024a-46e7-aa1a-21abed0a948e%40sessionmgr101)

Twitter. (2018). Twitter Privacy Policy. *Privacy Policy*. <https://twitter.com/en/privacy>

Twitter. (2019). *Twitter Developers*. <https://developer.twitter.com/en/docs>

Van Hee, C., Jacobs, G., Emmery, C., DeSmet, B., Lefever, E., Verhoeven, B., De Pauw, G.,

Daelemans, W., & Hoste, V. (2018). Automatic detection of cyberbullying in social media text. *PLoS ONE*, 13(10). <https://doi.org/10.1371/journal.pone.0203794>

Van Hee, C., Lefever, E., Verhoeven, B., Mennes, J., Desmet, B., De Pauw, G., Daelemans, W., & Hoste, V. (n.d.). *Automatic Detection and Prevention of Cyberbullying*. Retrieved April



APPENDIX I: CODE SNIPPETS

SVM Model Trainer Code

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.svm import SVC
from sklearn.pipeline import Pipeline
from sklearn.metrics import roc_curve, auc
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer

def readcsv():
    df = pd.read_csv("../data/dataset/csv/train.csv", ) # read
    labelled tweets
    X = df.text
    y = df.label
    return X, y

def drawrocSVM(y_test, y_pred):
    fpr, tpr, threshold = roc_curve(y_test, y_pred)
    print("Drawing")
    roc_auc = auc(fpr, tpr)
    plt.title('SVM Receiver Operating Characteristic')
```

```

plt.plot(fpr, tpr, 'b', label='SVM AUC = %0.2f' % roc_auc,
color='b')
plt.legend(loc='lower right')
plt.plot([0, 1], [0, 1], 'r--')
plt.xlim([-0.1, 1.2])
plt.ylim([-0.1, 1.2])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()

def svm_trainer(X, y):

    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state = 1)

    svm = Pipeline([('vect', CountVectorizer()), ('tfidf',
TfidfTransformer()),
                    ('svm', SVC(kernel="linear", C=1))])

    svm = svm.fit(X_train, y_train)
    ypred = svm.predict(X_test)

    print("SVM Model Metrics")
    print(metrics.accuracy_score(y_test, ypred))
    print(metrics.classification_report(y_test, ypred))
    drawrocSVM(y_test, ypred)

def main():
    X, y = readcsv()
    svm_trainer(X, y)

if __name__ == "__main__":
    main()

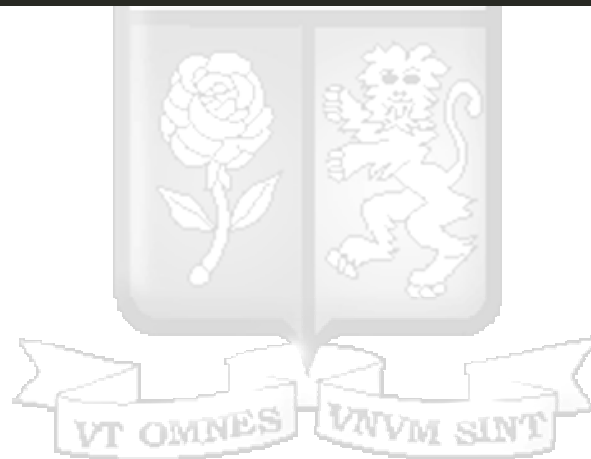
```

Twitter Data Cleaning Code

```

6 class Cleaner:
7     def __init__(self):
8         self.remove_punctuations = str.maketrans('', '', string.punctuation)
9
10    def tweets_cleaner(self, tweet):
11        html_escaped = html.unescape(tweet)
12        comma_replacement = html_escaped.replace('; ', '')
13
14        lower_case_text = comma_replacement.lower() # Change all texts to lowercase
15
16        removed_url = re.sub(r'http\S+', '', lower_case_text) # Removing URL links
17
18        removed_hash_tag = re.sub(r'#\w+', '', removed_url) # Removing hashtags
19
20        removed_username = re.sub(r'@\w+', '', removed_hash_tag) # Removing mentions and usernames
21
22        removed_retweet = removed_username.replace("rt", "", True) # Remove to retweet
23
24        removed_punctuation = removed_retweet.translate(self.remove_punctuations) # Removing punctuations marks
25
26        remove_g_t = removed_punctuation.replace(">", "", True) # Removing spaces
27        remove_a_m_p = remove_g_t.replace("&", "", True)
28        final_text = remove_a_m_p
29        return final_text
30

```



APPENDIX II Research Budget

Item	Unit Cost	Total
------	-----------	-------

1 laptop HP X360 8 GB RAM, 256 SSD, 3.2GH	Kshs 60,000	Kshs. 60,000
Zuku Internet Access	Kshs. 2,500 per month	Kshs 10,000
Data Labelling	Entire Data Set	Kshs. 5,000
TOTAL		Kshs. 75,000



APPENDIX III: Ethical Approval Letter



30th January 2020

Mr Kanam, Victor
victor.kanam@strathmore.edu

Dear Mr Kanam,

RE: Countering Social Media Cyberbullying using Sentiment Analysis on Twitter


This is to inform you that SU-IERC has reviewed and **approved** your above research proposal. Your application approval number is **SU-IERC0590/19**. The approval period is **30th January, 2020 to 29th January, 2021**.

This approval is subject to compliance with the following requirements:

- i. Only approved documents including (informed consents, study instruments, MTA) will be used
- ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-IERC.
- iii. Death and life threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-IERC within 72 hours of notification
- iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-IERC within 72 hours
- v. Clearance for export of biological specimens must be obtained from relevant institutions.
- vi. Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.
- vii. Submission of an executive summary report within 90 days upon completion of the study to SU-IERC.

Prior to commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology and Innovation (NACOSTI) <https://oris.nacosti.go.ke> and also obtain other clearances needed.

Yours sincerely,

for: 
Dr Virginia Gichuru,
Secretary; SU-IERC

Cc: Prof Fred Were,
Chairperson; SU-IERC



Ole Sangale Rd, Madaraka Estate. PO Box 59857-00200, Nairobi, Kenya. Tel +254 (0)703 034000
Email info@strathmore.edu www.strathmore.edu

APPENDIX IV: Turnitin Originality Report

4/17/2020
Turnitin

[Document Viewer](#)

Turnitin Originality Report

Processed on: 17-Apr-2020 2:12 PM EAT
 ID: 1299857479
 Word Count: 11954
 Submitted: 2

Countering Social Media Cyberbullying using S... By Victor Otieno Kanam

Similarity Index

22%

Similarity by Source

Internet Sources:	13%
Publications:	10%
Student Papers:	19%

[exclude quoted](#) [include bibliography](#) [exclude small matches](#)

mode: quickview (classic) report

Change mode
[print](#) [download](#)

<p>1% match (publications)</p> <p>Chang Sim Vuj, Gan Kim Soop, Chin Kim On, Bayner Alfred, Patricia Anthony, "A review of stock market prediction with Artificial neural network (ANN)", 2013 IEEE International Conference on Control System, Computing and Engineering, 2013</p>
<p>1% match (publications)</p> <p>Senthil Kumar A. V., Rathi M., "chapter 8 Keystroke Dynamics", IGI Global, 2019</p>
<p>1% match (Internet from 28-Jan-2018)</p> <p>http://nefy.org</p>
<p><1% match (student papers from 18-Apr-2019)</p> <p>Submitted to National College of Ireland on 2019-04-18</p>
<p><1% match (Internet from 28-Jan-2020)</p> <p>https://repositorj.foi.unizg.hr/islandora/object/foi:268/datastream/PDF</p>
<p><1% match (Internet from 02-May-2019)</p> <p>https://eodf.tips/web-data-mining-2nd-edition-exploring-hyperlinks-contents-and-usage-data.html</p>
<p><1% match (Internet from 08-Mar-2020)</p> <p>http://deshbhagatuniversity.in</p>
<p><1% match (Internet from 16-Apr-2020)</p> <p>https://rvpi.org/project/joblib/</p>
<p><1% match (student papers from 21-Mar-2020)</p>

69