SPAM DETECTION IN TWITTER BY USING K-NEAREST NEIGHBOR (KNN)

SHWE THA ZIN

M.C.Sc. **DECEMBER, 2022**

SPAM DETECTION IN TWITTER BY USING K-NEAREST NEIGHBOR (KNN)

By

SHWE THA ZIN

B.C.Sc.

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Computer Science

(M.C.Sc.)

University of Computer Studies, Yangon

DECEMBER, 2022

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my sincere thanks to those who helped me with various aspects of conducting research and writing this thesis. To complete this thesis, many things are needed like my hard work as well as the supporting of many people.

First and foremost, I would like to express my deepest gratitude and my thanks to **Dr. Mie Mie Khin**, Rector, University of Computer Studies, Yangon, for her kind permission to submit this thesis.

My thanks and regards to my supervisor, **Dr. Si Si Mar Win** and **Dr. Tin Zar Thaw**, Professor, Faculty of Computer Science, University of Computer Studies, Yangon, for her support, guidance, supervision, patience and encouragement during the period of study towards completion of this thesis.

I would like to express my appreciation to **Dr. Zin Thu Thu Myint**, Associate Professor, Faculty of Information Science, University of Computer Studies, Yangon, for her superior suggestion, administrative supports and encouragement during my academic study.

I also wish to express my deepest gratitude to **Daw Win Lai Lai Bo**, Assistant Lecturer, Department of English, University of Computer Studies, Yangon, for her editing this thesis from the language point of view.

Moreover, I would like to extend my thanks to all my teachers who taught me throughout the master's degree course and my friends for their cooperation.

Last but not least, I especially thank my parents, all of my colleagues, and friends for their encouragement and help during my thesis.

ABSTRACT

Since more than 300 million monthly users send 500 million tweets daily, Twitter is a popular social networking platform. This is the main reason why spammers use Twitter to spread malicious software that steals user personal information, tweets with faulty or fake URLs, assertively following or un-following users, trending fake tweets to attract users' attention, and spreading pornographic advertisements and among other reprehensible activities. The research clearly demonstrates that over 32 million people have engaged with the server for casual information on a daily basis. Twitter is said to have collected data on active users in previous years and studied their actions. Therefore, today's social media landscape, it is crucial to recognize and filter out the damaging or unwanted trends or malicious tweets. This technique suggests analyzing tweets and categorizing them as spam or ham based on the words they include. While there are several machine learning and deep learning techniques for categorizing and detecting spam tweets, this system will employ the clustering and binary detection model from KNN. This system is implemented using ASP.Net programming language with Microsoft SQL Server Database Engine.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

CHAPTER 1
INTRODUCTION

One of the most well-known social media platforms is Twitter, which enables a social network of users to broadcast information in "tweets" of around 140 characters. Twitter allows the chance for users to submit their messages about anything which is considered to be authentic, such as news, events, celebrities, political topics, etc. (Bravo-Marquez, 2013). According to Twitter, there are 313 million active monthly users on the platform who post 500 million tweets per day, or nearly 350,000 tweets per second (Bai, 2017). However, spammers are also drawn in by this notoriety and common sense. In or around April of 2014, a large number of spammed user's accounts posted a lot of malicious tweets, flooding Twitter (Chen et al., 2015). 23 million or 8.5% of its dynamic monthly users were accessing its servers on a regular basis for updates (Seward, 2014) which discovered in August, 2014.

Spammers send their contents to the user while portraying them as useful or relevant. The genuine users erroneously believe the spam to be essential information. Despite the fact that email providers like Gmail, Microsoft, and others have been successful in identifying spam messages, spam messages continue to circulate online. These administrations reported that 90 to 95 percent of all email transactions now involve email spam (Waters 2009). Spammers take advantage of the fact that companies cannot stop them once and they have successfully detected spams to persuade consumers to click on a spam interface (Perveen et al., 2016). With the growth of online social communities, the threat posed by spam become more serious, and Twitter stands out as one of the most popular online social communities that have been greatly impacted by spam (Perveen et al., 2016). The trending topics on Twitter are more the focus of Twitter spam, which is harder to hack due to its hash-tag administrator (Perveen et al., 2016).

Teachers, students, celebrities, public leaders, business clients, and advertising are just a few of the many types of people who utilize Twitter. Twitter is accessible to users of all ages, but the 55 to 64 age group uses it the most frequently (Perveen et al., 2016). About 60% of people accesses twitter on their mobile devices. A user encounters a variety of problems with query items as a result of this spamming worry, which leads to redundant and superfluous data.

Additionally, since a user must sift through all the information to obtain a broad understanding of the subject, this can be extremely stressful. Because of the usage of URLs, acronyms, colloquial language, and modern linguistic concepts, it is challenging to locate spam on the Twitter network (Stringhini et al., 2010). Here, outdated methods for detecting spam data fall short. There is currently literature on a number of techniques for spotting spam on Twitter. This technique suggests analyzing tweets and categorizing them as spam or ham based on the words they include. While there are several machine learning and deep learning techniques for categorizing and detecting spam tweets, this system will employ the clustering and binary detection model from KNN.

1.1 Aims and Objectives of the Thesis

The main aims and objectives of the thesis are as follows:

- To classify between spam and legitimate (ham) textual comments of tweets by using K-nearest Neighbor (KNN) algorithm.
- To provide better performance in real time spam detection.
- To detect spam and fraudulent tweets by using K-nearest Neighbor (KNN) algorithm.
- To analyze the effectiveness of KNN tweets' spam classifier by classifying tweets into different categories (Spam and Ham).
- To evaluate the accuracy of classified data by using K-nearest Neighbor (KNN) algorithm with confusion matrix.

1.2 Motivations

As a result of its widespread usage in daily life, Twitter has actually more spam information than other social networking sites. Twitter users have the option to "follow" other accounts that interest them. The connection between users is bi-directional as opposed to simplex links on other social media sites which could result in one person not following one of his followers. As a result, spammers have a chance to spread their spam. Spamming refers typically to send the user undesired information or data. This system's primary objective is to find tweets that are perceived as spam. A Twitter user is only identified by their username with their real

name being optional. It can be problematic to accept an Associate in nursing erroneous friend request from a stranger. Whether or whether the victim is aware of the aggressor in the actual world, the user will still click any link contained in communications. If the attacker uses information stolen from victims' friends in informal organizations to build their phishing messages. Therefore, this approach can assist in finding those spam tweets with reasonable accuracy.

1.3 Related Works

One of the main issues with electronic communication including massive email systems is spam. One important function of an email system is the classification of spam emails. Numerous variables including the number of features, feature selection methods, symbol representation and classifier show how effective this procedure is. This study focuses on the classification of spam and junk emails using a multilayer perceptron (MLP) technique. The term frequency and inverse document frequency (tf-idf) feature selection methods as well as fisher score are used by the system during preprocessing. These techniques enable the selection of pertinent features and the addition of advantages to the email classification system in terms of improvement in accuracy and decreased time complexity [1].

This objective of this system is to assess sentiment categorization performance in terms of recall, accuracy, and precision. This system contrasts the Naive Bayes and K-NN supervised machine learning methods for classifying the sentiment of movie and hotel reviews. The experimental results demonstrate that the classifiers produced improved predictions for the movie reviews with the Nave Bayes' technique surpassing the k-NN approach and providing above 80% accuracy [7].

Since social networks represent a system of trust unmistakably, abusing this trust could have severe consequences. 41% of the Facebook users who were surveyed in 2008 were able to distinguish a friend request from a random person according to the research [10]. According to L. Bilge et al. (2009), regardless of whether the victim is familiar with the hacker or not, once the hacker has gotten access to the victim's computer system and victim is likely to click on any links that make up the published messages. Another intriguing finding by experts is that phishing attempts will probably succeed if the hacker uses data acquired from the victims' acquaintances in social networks to generate their phishing messages (Jagatic et al., 2007). For

instance, phishing messages from the soppy bag were usually sent from a user's friend list; as a result, a user is frequently tricked into believing that communications come from friends who can trust and subsequently and readily provides login information for his or her email account.

The developers create a prominent hashtag on Twitter and observe how spammers started to use it in their posts according to Yardi et al. (2009). They discuss a few characteristics that can be utilized to tell spammers apart from real users like message repetition and hub degree. As long as there are certain youthful Twitter users or TV anchors who post a lot of messages, the utilization of fundamental components like message duplication and hub degree may not be adequate. Bigger spam research was discussed in Stringhini et al. (2010). In order to attract spammers, Stringhini et al. (2010)'s creators created nectar profiles. On well-known social networking sites like Twitter, Myspace, and Facebook, they each create 300 profiles. The 900 profiles had generated 4250 friend requests (mostly from Facebook). On Twitter, 361 of 397 buddy requests came from spammers. Later, they suggest employing indicators for identifying spam including the frequency of tweets containing URLs, message similarities, the total amount of messages sent, and the number of friends. They discover that their Twitter database's Random Forest classifier can provide an inaccurate positive ratio of 2.5% and an incorrect negative ratio of 3%. (Mccord, and Chuah, 2011).

The authors of Wang (2010) proposed using chart-established and content-established components to identify spammers. The number of followers, number of friends (the number of people a user is following), and notoriety score which is defined as the ratio between the number of followers over the sum and the number of user is following were the chart's established elements. The assumption is that if a user has fewer followers than the number of people they are following, their renown is low and there is a higher chance that the connected account is spam.

1.4 Organization of the Thesis

The thesis is organized into five chapters: an abstract, an acknowledgment, and references.

The introduction of the system, aims and objectives of the thesis, related works, and thesis organization are described in chapter one.

The background theory is presented in chapter two. The twitter social network and twitter spam detection methods are briefly explained in this chapter.

The proposed methodology is discussed in chapter three. First of all, the development of KNN, tasks of KNN, advantages of KNN, and disadvantages of KNN are described in this chapter.

In chapter four, the design and implementation of the proposed system are expressed. First of all, the TF-IDF method and KNN algorithm are briefly described in this chapter. The overview of the system flow and the implementation of programming modules for the proposed system are illustrated with a graphical user interface.

And then, the experimental results of the proposed system are shown in a graph. Finally, the results of accuracy, precision, recall, and f-measure for the K-nearest Neighbor classifier based on the spam detection system are described in this chapter.

Finally, the conclusions and benefits of the proposed system are described in this chapter. Further extensions that suggest some enhancements that could be completed are presented. The limitations of the system are also expressed in this chapter.

CHAPTER 2
BACKGROUND THEORY

The term "social media" refers to methods of communication in which individuals build, share, and exchange knowledge and concepts through online groups and networks. The primary accounts on Facebook, Twitter, Instagram, LinkedIn, and YouTube are managed by the Office of Communications and Marketing.

2.1 The Twitter Social Network

Twitter is a micro blogging platform that enables users to post brief messages, or tweets, which are visible on the pages of their friends. Additionally, it can be contrasted with other social networking websites like MySpace and Facebook (Kwak et al., 2010). A Twitter account can only be identified by its username, however sometimes real names are also used. A Twitter user can start "following" another user named "U". Thus, user U's tweets are visible to that user on her page. The "following" user U is free to follow back at any time. Using hash tags, which are well-known words beginning with the "#" symbol, tweets can be gathered. Users can search for tweets about topics that interest them using hash tags. When a user loves a tweet, she has the option to "retweet" it. Therefore, all of her supporters will see that information. A user has the option of protecting her profile. Any user wishing to follow that private user must obtain her permission in order to do so. With more than 328 million active monthly users, twitter has increased over time (Arun et al., 2017).

2.1.1 Features of Twitter

Unlike other social media platforms, Twitter enables accounts to "follow" other accounts that they find interesting. This link is bidirectional rather than unidirectional because a given user may not be following one of his followers. A tweet can be "liked" or "retweeted (RT)" by the user, which means he is sharing it with his "following" (Atefeh and Khreich, 2015). Figure 2.1 depicts the connections between individuals on Twitter. Users can submit tweets that mention other users by inserting their usernames beginning with "@," which is known as a "mention" on Twitter. Each user has a unique Twitter username (Atefeh and Khreich, 2015). Users

are immediately informed by notifications whenever he receives a mention, like, or RT on any of his tweets (Atefeh and Khreich, 2015).

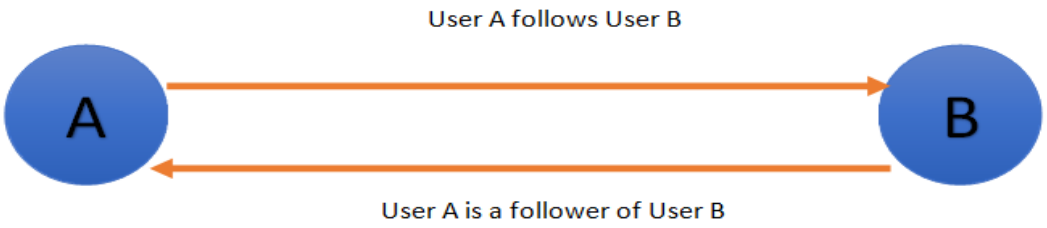

Figure 2.1 Relationship between Users in Twitter

Another feature of Twitter is that it allows users to convert their account types from public to private and vice versa, with the intention of organizing their interests by bringing together others with similar or comparable interests (Kim et al., 2010; Yamaguchi et al., 2011). User-subscribed lists are referred to as "subscribed to," whereas owner-included lists are referred to as "member of."

2.1.2 How Twitter Deals with Spam

Twitter fights spammers by using both manual and automated procedures. Twitter offers users an option to report spammers via the spammers' profile pages as part of the manual way. Figure 2.2 shows the user interface (UI) that Twitter offers for picking a cause to report an account.

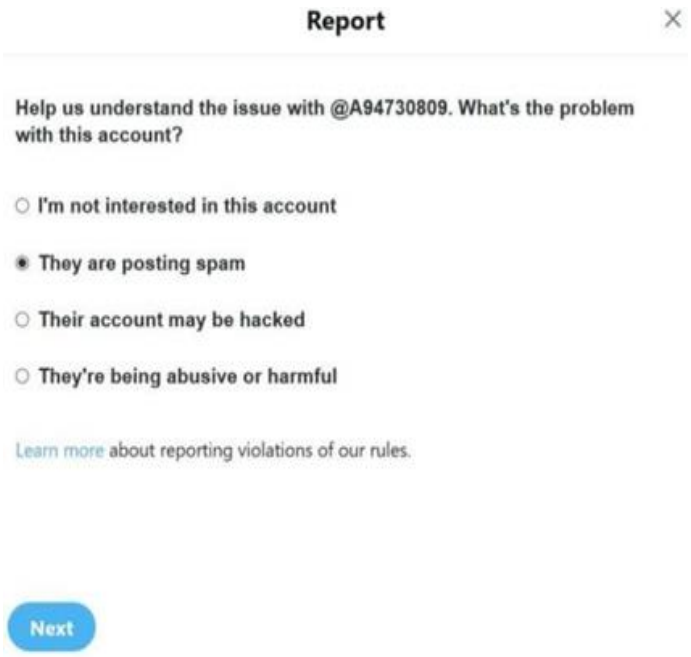

Figure 2.2 The User Interface of Twitter which is Used to Report an Account by Selecting the Motive

Another strategy described in the literature entails reporting spammers to the official "@spam" address (Song et al., 2011; Wang 2010; Kaur et al., 2016; Chen et al., 2016; Verma, and Sofat, 2014; Gee and The, 2010). However, as demonstrated by the most recent Twitter report, this method of identifying spam is no longer effective (Atefeh and Khreich, 2015). Wang claims that both the creators and spammers abuse this strategy (Wang, 2010). Due to the enormous number of users, these manual approaches are taxing and might not be able to discriminate between all spammers. (1) Publishing duplicate tweets across multiple accounts or multiple copies of the same tweet in one account,

(2) Following/Unfollowing a large number of accounts quickly,

(3) Having a large number of spam complaints against the account, (4) Forcibly liking, following, and retweeting,

(5) Publishing malicious connections,

(6) Publishing tweets that, for the most part, contain connections rath (Atefeh and Khreich, 2015).

(7) The display of irrelevant tweets to a hot topic to determine whether lead is considered to be spamming (Atefeh and Khreich, 2015).

2.2 Twitter Spam Detection Methods

Following preprocessing of the tweets, the retrieved features are categorised using a variety of classifiers, including Account-based Spam Detection Methods, Tweet-based Spam Detection Methods, Graph-based Spam Detection Methods, and Hybrid Spam Detection Methods.

2.2.1 Account-Based Spam Detection Methods

The elements (or their combination) of the Twitter account listed in Table 1 are used in account-based spam recognition algorithms. A honeypot-based approach to dealing with spam differentiation in social media platforms was proposed by Lee et al. (2010). They took into account the account's history on Twitter, the frequency of daily tweets, the ratio of followers to followers, the rate of bi-directional friends, the proportion of URLs in the 20 most recent tweets, the proportion of unique URLs in the 20 most recent tweets, the proportion of usernames in the 20 most recent tweets, and other factors when determining whether a tweet was spam. According to Lin and

Huang (2013), a method for spotting spam on Twitter is based on the following two premises:

(1) URL ratio, which describes the percentage of tweets with URLs in the total number of tweets, and

(2) Connection ratio, which describes the percentage of tweets working together in the total number of tweets.

A strategy for the account-based components, such as the ratio of followers to followers, the number of tweets to account lifetime ratio, the average time between posts, posting time variety, the maximum idle hours, and connection division, as described by Gee and Teh (2010). The manual way of spotting spam in Twitter, which has lately been highlighted as being out of date, is the focus of this article.

2.2.2 Tweet-Based Spam Detection Methods

The elements (or their combination) in a tweet are what are used in twitter-based spam recognition systems. Approaches for separating URLs use static or dynamic crawlers to look up recently visited URLs. They also take the source code of the landing page and URL or domain boycotting to identify unusual URL redirections into account (HTML). McGrath and Gupta presented a phishing discovery technique that takes into account the lexical elements of a URL (2008). They take into account the length of the URL and the domain name, the originality of the domain name, the proximity of brand names in the URLs, and the abuse of URL-connection and cheap web hosting services when determining whether anything is phishing. Ma et al. (2009) proposed a method of looking through URLs to find the harmful sites. They distinguish malicious websites with WHOIS resources by the components they employ. Examples of such information include the site's enlistment center, registrant, enrollment date, domain name resources (such as the time-to-live (TTL) value for DNS documents), and geographic resources. The speed of the uplink connection and neighboring lexical URL parts are included, as well as the country where the IP address is located. Static evaluation techniques are used by Canali et al. (2011) to identify the dangerous components of a website. The components of the website come from the following sources:

(1) the HTML component of the site, such as the proportion of components having a narrow range, the proportion of components made up of suspicious

components, the proportion of included URLs, and the proportion of examples known to be malicious,

(2) the related JavaScript code, such as the ratio of catchphrases to words, the quantity of long strings close to translating schedules, the likelihood of shell code close by, and the quantity of DOM-altering capacity, and

(3) the comparison of URL, such as the quantity of dubious URL designs, the proximity of subdomains or IP addresses in URLs, and the TTL value for DNS A and NS record.

Prophase uses static investigative techniques, hence it is unable to detect malicious URLs placed into an active component. Examples of it include Java applets, Flash, and a piece of JavaScript, the most popular programming language at the moment (Parveen et al., 2016) Strategies for dynamic inquiry methods, such as those by Whittaker et al. (2010), Wang et al. (2006), Thomas (2011), and Cova et al. (2010), use virtual machines and automated web applications like Selenium to perform in-depth component analysis. Chhabra et al. (2011) present a URL-based phishing location approach. Their method is specifically designed to be able to evaluate short URLs, which Twitter frequently uses to filter out spam tweets, as was recently described.

The amount of clicks, land spread, ephemeral spread, and web fame are the components of the suggested technique that use recognizing phishing via URL. A suspicious URL recognition structure for Twitter developed by Lee and Kim (2013) investigates the relationships between URL divert chains. The length of URL redirects, the number of different landing URLs, the relative numbers of various Twitter accounts, the similarity in the account creation dates, the similarity in the number of followers and following, the similarity in the follower following proportion, and the comparability of tweets are just a few of the 14 factors used by Lee and Kim (2013) to identify suspicious URLs. Martinez-Romo and Ajauro (2013) propose a tweet-based spam identification approach that focuses on the analysis of the language used as a component of the tweet.

(1) the tweets' linguistic prototype, shown by an inclining point,

(2) the tweet's original language, and

(3) the tweet's linked page's language prototype. Many Twitter spam discovery approaches use tweet-based aspects related to other spam identification to give a stronger spam location, much like the account-based spam detection techniques.

2.2.3 Graph-Based Spam Detection Methods

Techniques for finding spam using diagrams rely on the elements—or their combinations in a tweet. The distinction and availability between a tweet's sender and mentions are eliminated by Song et al. (2011). Separate describes the distance of the shortest path between the sender of a tweet and any mentions, whereas affiliation describes the nature of the relationship between users. The chart information patterns are used by diagram-based spam location algorithms to display Twitter's hubs and edges. In this way, social networks that are primarily built on users, subjects, and reciprocal interactions, like Facebook and Twitter, frequently use charts (Ugander et al., 2011; Weaver and Tarjan, 2013; Myers et al., 2014; Gabielkov and Legout, 2012). Despite this, the chart-based components offer the finest execution in terms of accuracy and capability to distinguish genuine users from spammers. Hybrid spam identification strategies, which combine existing spam localization strategies with other diagram-based spam recognition algorithms, are introduced.

2.2.4 Hybrid Spam Detection Methods

In order to provide more dynamic spam detection that more thoroughly investigates the likelihood of spam, hybrid spam identification approaches combine the spam location tactics shown in the preceding subsections. An approach is recommended by Stringing et al. (2010) for both accounts-based and tweet-based aspects. Both of them are inversely correlated with the user's friend count and the number of friend requests she has sent. They also depend on the user's total amount of tweets, the number of tweets that make up the URLs, the similarity of the user's tweets, the quantity of tweets the user sends, the number of friends the user has, and the possibility that an account utilized a list of names to choose its friends or not. Gao et al. (2012) proposed a tweet-based spam identification approach that is based on the sender's social standing, the communication's historical context, the size of the group, the regular time interval, the regular number of URL in tweets, and the unique number of URL in tweets.

According to a dataset of 6.5 million spam tweets, Chen et al. (2015) take out twelve irrelevant variables and present a continuous spam identification approach for Twitter. Age of the account, number of followers, number of those following, number of likes the account received, number of records that the account has, number of tweets in the account, number of retweets, number of hash tags used as a part of the tweet, number of users mentioned in the tweet, number of URLs used as a part of the tweet, and number of characters used as a part of the tweet are the elements they tried to use to identify spam on Twitter.

Wang (2010) proposed a hybrid diagram-based and tweet-based approach for the location of twitter spam. The number of followers, the number of those following, and a notoriety score—which is calculated as the ratio between the number of followers over the total number of followers and following—are the diagram-based elements taken into account in the suggested technique. The following tweet-based factors are taken into account in the suggested strategy: tweet comparability, number of tweets making up URLs in the most recent 20 tweets, number of tweets making up mentions in the most recent 20 tweets, and number of tweets containing hash tags. Yang et al. (2013) propose a combination of the diagram-based, tweet-based, and account-based components as a twitter spam identification method.

The suggested technique makes advantage of stronger components, such as the quantity of connections that go both ways, their proportion, the relationship between significance, and the amount of connections that group tweet- and account-based features together. The example includes the number of followers, number of following, number of tweets sent by the account, length of account existence, percentage of tweets that make up the URL, percentage of tweets that contain hash tags, percentage of copy tweets, percentage of spam words, percentage of tweets used to respond to others, and percentage of the quantity of retweets. A hybrid spam location method is suggested by Benevento et al. (2010) for the account-based components. The example shows the number of followers, the number of people who follow the account, the ratio of followers to people who follow, the number of tweets the account sent, the number of mentions it received, the number of responses it received, and the percentage of tweets that came from followers. The suggested strategy's tweet-based components include the word count per tweet, the number of URLs per word, the number of expressions per tweet, the character count per tweet,

the number of hash tags per tweet, the number of mentions per tweet, the number of URLs per tweet, and the number of retweets per tweet.

Chu et al. (2010) present a method that relies on both account-based and tweet-based factors to classify Twitter accounts as human, bot, and cyborg. The amount of the percentage of tweets that contain URLs, the makeup of the device, the ratio of followers to friends, the connection of interests, and if the account is approved are the factors they take into account when categorizing a Twitter account as human, bot, or cyborg. Given both account-based and tweet-based components, Amleshwaram et al. (2013) propose an account-based and tweet-based crossover Twitter spam location technique. They divide spammers into two groups:

(1) URL-driven and

(2) user-driven.

The number of notable mentions, spontaneous mentions, snatching trends, crossing points with approved patterns, variation in tweet intervals (VaTi), variation in tweet width (VaTw), proportion of VaTi and VaTw, tweet sources, copy of URLs, copy of domain names, IP/domain fluxing, uniqueness of tweet's language, relationship between tweets, similarity between URL and tweet, followers-to-following ratio, and profile description are the factors they take into consideration when.

Given account-based and tweet-based components that make use of some contemporary components, Chakraborty et al. (2012) propose a hybrid solution. In the example, the hash tag's usual compatibility, the appearance or absence of the profile picture, and the spam score of the description, name, and screen name all come into play.

To help with spam identification, McCord and Chuah (2011) present a combination technique involving account-based and tweet-based components. They use the following elements in their suggested method: the distribution of tweets over a 24-hour period, the number of URLs, the total number of responses/answers in the last 100 tweets, the total number of retweets in the last 20 to 100 tweets, and the total number of hash tags in the last 100 tweets. et al. (2015) suggest an approach for identifying spam that takes into account account-based, tweet-based, natural language processing (NLP), and hypothesis highlighting. The length of the profile name, naturally or artificially created assessment vocabularies, the number of exclamation

points, the number of question marks, the most extreme word length, the mean word length, the number of upper case words, the number of void areas, and part of speech (POS) labels per tweet are some unique elements that they use to distinguish spam.

CHAPTER 3
THE PROPOSED METHODOLOGY

The K-Nearest-Neighbor (KNN) classification is a non-parametric calculation; for example, it makes no presuppositions about the fundamental dataset. It is renowned for being simple and practical. This learning calculation is governed. In order to predict the class of the unlabeled information, a marked preparation dataset is provided in which the information's main points are arranged into several classes.

The class to which the unlabeled information belongs in classification is determined by a number of factors. Typically, KNN is used as a classifier. Information is grouped according to nearby or nearby preparing models in a particular area. This method is used because it is simple to use and takes little time to calculate. It uses the Euclidean distance to calculate its closest neighbors for unending information. Information's K nearest neighbors are identified, and the majority of the adjacent information chooses the new information's layout. Although this classifier is simple, the importance of "K" plays a significant role in the ranking of the unlabeled data. There are numerous ways of choosing the qualities for 'K', however we can basically run the classifier on various occasions with various qualities to see which worth gives the best outcome. The calculation cost is somewhat high since every one of the estimations is made while the preparation information is being arranged, not when it is experienced in the dataset.

It is a sluggish learning calculation because, other from storing the preparation data and keeping the dataset overall, little much is done when the dataset is being prepared. On the preparation dataset, speculation is not performed. Therefore, while testing is being done, the entire primary dataset needs to be prepared. KNN forecasts constant properties in relapse. This value represents the typical benefit of its K-nearest neighbor.

3.1 Improvement of KNN

When unambiguous parametric approximations of likelihood densities were ambiguous or difficult to decide, K-closest neighbor arrangement was developed to carry out trademark examination. Fix and Hodges introduced the K-closest neighbor rule, a non-parametric design grouping computation, in a 1951 US Air Force School of Aviation Medicine study that was never published.

3.2 Responsibilities of KNN

A classification algorithm is KNN. Generally speaking, classification involves two steps:

1. A classifier is built using the training data in the first learning step.

2. A classifier evaluation.

The new unlabeled data is organized, as suggested by the closest neighbor approach, by determining which classes its neighbors belong to. This concept is incorporated into the KNN computation. When using KNN, a certain value of K is fixed, this aids in organizing the cryptic tuple. KNN does two things when a new unlabeled tuple appears in the dataset:

The K closest neighbors the focuses that are the closest to the new data of interest are broken down first.

Second, KNN determines which class the new information should be organized into using the classes of the neighbors.

When some new information is added, it describes the information in a similar way. In a dataset that is typically divided into groups and has a location with a specific district of the information plot, it is more valuable. As a result, the calculation is more accurate and clearly separates the information inputs into several classifications. KNN groups the classes with the most concentrations that are closely spaced from the information manual that needs to be sorted. Therefore, the Euclidean distance between the exam and the planned preparation tests should be calculated.

After assembling K-Nearest Neighbors, we effectively take the majority of them to predict the preparation model's class. The value of K, the Euclidean distance, and the standardization of the boundaries are the three factors that affect how KNN is

presented. The following methods are used to understand the calculation's precise workings:

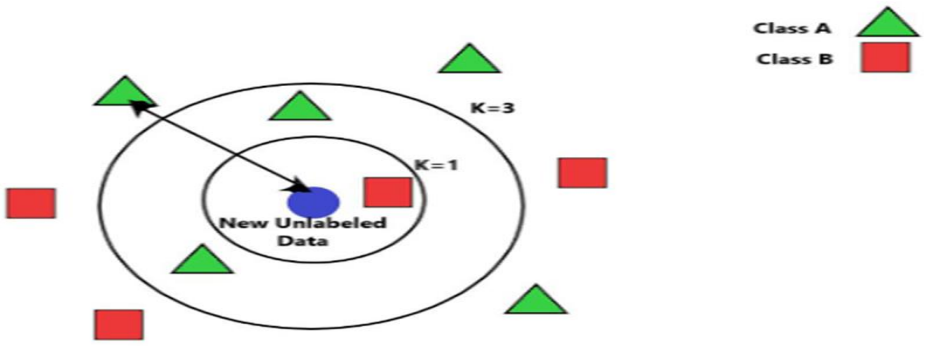

Figure 3.1 Sample Classification of KNN

Given the training dataset : { (x(1), y(1)) , (x(2), y(2)), ... , (x(m), y(m)) }

Step 1: Store the training set.

Step 2: For each new untagged data,

A. Calculate Euclidean distance with all training data points using the formula

$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}.$$

Equation 3.1

B. Find the k-nearest neighbors

C. Assign classes including the maximum number of nearest neighbors.

After putting away the preparation, all limits should be placed in a uniform manner in order to make estimations simpler. The grouping's outcome depends on how much "K" is worth. The number of neighbors that should be taken into consideration is determined by the information variable "K." The value of "K" has an impact on the calculation because it allows us to construct the bounds of each class.

TO DEFINE K: The ideal value of K is chosen by first reviewing the data; greater upsides of K are more accurate as they reduce net commotion, but this isn't guaranteed. Cross approval can also be used to resolve a respectable value of K. The information is essentially assigned to the class of its nearest neighbor on the off event that K=1. For the preparation information, the error rate is unquestionably 0 at K=1. This occurs because the nearest highlight to any preparation-related information item

is the preparation itself. As a result, K should equal 1 to achieve the optimal results. However, the bounds are over fitted when K=1.

The calculation is too delicate to even consider noise in the case of minute "k" upsides. The preparation and approval set needs to be separated from the underlying dataset in order to obtain a reliable value for K. The outcome is ambiguous if the two closest neighbors (K=2) have a location with two distinct classes. By doing this, we increase the number of closest neighbors to a higher value (say 5-closest neighbors). This will distinguish an earliest neighbor region and provide clarity.

Larger upsides of "K" smooth out class boundaries, which is probably not appealing because different classes' grades might then be recalled for the neighborhood. Even though the preparation-related information is dispersed, it can be difficult to assess K's value.

3.3 Advantages of KNN

KNN is noted for its effortlessness, intelligibility and adaptability. It is simple to understand. The estimation period is shorter. Similar to the clairvoyant power, which is extraordinarily high, it is fascinating and effective. KNN is incredibly persuasive for large preparation sets. The means that this calculation followed in the grouping are a little less confusing than those that other computations followed. The mathematical calculations are simple to understand and grasp. They exclude computations that seem to be difficult. As opposed to using other composite procedures like inclusion or separation, basic concepts like Euclidean distance estimation are used, which improve the calculation's simplicity. For knowledge that is indirect, it is useful. KNN is convincing in terms of both characterization and relapse.

3.4 Disadvantages of KNN

If the dataset is huge, KNN can declare K to be high. Instead of a strong classifier, it needs a more imposing storage. In KNN, the prospect stage is delayed for a bigger dataset. Similarly, calculation of accurate distances assumes a major part in the word of the calculations precision. Choosing the boundary K is one of the key steps in KNN. It is unclear at the moment the distance to use and which element will produce the best results. Due to the distance at which each arranging model must be

completed, the computation cost is extremely large. KNN is a slow growing computation since it offers to keep the preparation information and then uses it to describe the new information, rather than increasing from it.

3.5 KNN and its Variants

As previously discussed, altering the factors that control the calculation can increase its effectiveness. There are many variations of KNN that have been focused before to make this calculation more effective, some of them are:

(1) Locally Adaptive KNN:

It is suggested to use locally adjustable KNN calculations [7]. By considering the outcomes of cross-approval computations in the close-by vicinity of the untagged information, it determines the value of k that should be utilized to order an impact.

(2) Weight Adjusted KNN:

According to the computation in [7], the distances on which the early phase of the search for the closest neighbors is located must be converted into equivalent measurements that can be utilized as loads. The referenced loads establish the degree to which a quality affects the classification process. This classifier is very useful when a dataset has a large number of items, some of which can be viewed as optional but still having a high computational rate.

(3) Improved KNN for Text Categorization:

The research suggests a developed KNN calculation for script order that combines KNN text classification with a restricted one pass group calculation to create a description model. In the unlikely event that a stable value of K is chosen for each class, the class with the most attributes will have an advantage. In an improved KNN, transmission of information during set preparation uses a reasonable number of nearest neighbors to predict the class of untagged data.

(4) Adaptive KNN:

For every new piece of information, KNN chooses the same number of nearest neighbors. For each test, adaptable KNN [9] calculates a fit value of K. K is found to have an optimal value early on. The value of K is then set to be comparable to the ideal value of K of its nearest neighbor in the training dataset in order to anticipate the

preparation of the untagged data. The results of the suggested calculation are then tested against a variety of datasets.

(5) KNN with Shared Nearest Neighbors

The use of separated nearest neighbor comparability, which can create similarity among tests with nearest neighbor tests, is presented as an improved K-nearest neighbor computation in [9]. It makes use of the nearest neighbor equality and motivational factors to calculate similarity results for all test-takers. The furthest between these traits is then established.

(6) KNN with K-Means:

One more invented way to deal with the calculation is represented in[6]. These computations divide a set of focuses into K sets or groups, keeping the focuses in each group close to one another. The focuses of these recently made groups are occupied as the new preparation tests. To prepare for the categorization of untagged information, its separation from the newly discovered fixing focus is completed, and the central element that shares the information's base separation is selected for that class. There is the information border K that isn't passed, unlike to standard KNN. One of its earnings is shown by this record.

(7) SVM KNN

An order approach known as Support Vector Machine (SVM) can be used with both direct and indirect information. For visual classification acknowledgement, it is a composite variation of KNN combined with SVM, and is expanded in [4]. K nearest neighbors to the unnamed data of interest are used in this computation to complete the preparation. First off, the K-closest information is not unchangeable. Following that, processing of the pairwise distance between these K information centers occurs. The determined distances are then used to create a distance framework. The got distance framework is then used to plan a kernel network. This bit framework is treated as an SVM classifier contribution. The class of the obscure piece of information is the result attained. SVMs could be used, however one drawback is that they take a lot of time. It also incorporates pairwise distance estimation.

(8) KNN with Mahalanobis Metric

The measuring distance is important when grouping other relevant data. Another distance metric is Mahalanobis, whose methodology is discussed in [8]. The metric ensures that the K-closest neighbors belong to the same class, and it isolates examples that belong to different classes by a great deal of contrast.

(9) Generalized KNN

KNN can also be used to earn consistently high grades in classes. In this structure, the class attribute of the unlabeled information is assigned the typical characteristics decided upon among neighbors. This calculation is carried out by KNN [3] in order to anticipate the constant- esteemed class characteristic.

(10) Informative KNN

It might be challenging to choose the boundary for different applications because the value of K typically varies on the information. [10] presented a different metric that measures how illuminating the things to be grouped are. The value of focuses is estimated by educationalness. There are two information limits in this strategy: K and I. The majority of students in the majority of educational downsizing models will make up the test's class.

(11) Bayesian KNN

A similar likelihood conveyance is used to generate the information values that contain the target and spread out over the appropriate number of neighbors. [11] Recursively calculated the likely of the most recent change-point, moved in the direction of the goal, and recorded the dispersion of the back likelihood over K.

CHAPTER 4
SYSTEM DESIGN AND IMPLEMENTATION

Informal online communities like Facebook and Twitter have been increasingly prevalent in people's day-to-day lives during the past several years. People utilize social media as a virtual community platform to blog about their views and ideas while also keeping in touch with friends and family. These platforms draw in a large number of users as a result of this emerging tendency, making them spammers' favorite targets. The most well-known informal community among teenagers is presently Twitter. For instance, "beauty gurus" or "beauty influencers" who blog about makeup often target young females as their primary audience. Today, 400 million new tweets are created each day by 200 million users. The expansive environments that twitter also give spammers a chance to direct viewers to useless stuff. By tricking users into clicking on links to access harmful websites with malware, phishing, and scams, these irrelevant or uninvited messages assault users. The comments area that appears beneath each user post on Twitter is one of its most prominent features. Users can exchange thoughts and opinions with this tool.

This system makes predictions about the spam comments that will appear in the twitter comments section utilizing the machine learning concept, which is also referred to as a subset of artificial intelligence. The K Nearest Neighbor which suggested classification technique is used to predict and categorize spam comments. The aim of the system is to define the prediction technique and briefly introduce machine learning techniques. Machine learning, which is far more effective than traditional data analysis methods, can create new possibilities for research and improve prediction accuracy.

Using term frequency - inverse document frequency (TF-IDF) and KNN, this system can create a framework for classifying twitter comments and tweet spam. A statistical way to assess a word's significance to the entire corpus is the TF-IDF. The tweets were categorized by using the KNN classifier and placed in the appropriate group.

4.1 Term Weighting Schemes (TF-IDF method)

The term weighting method known as "Term Frequency" (TF) is based on how frequently terms appear in a document. The impact of a word on a document increases as a word's TF value increases. The weighting approach known as Inverse Document Frequency (IDF) bases its calculations on the number of words that appear in each document. One of the most straightforward and effective weighting methods for the data is TF-IDF. Due to its straightforward formulation and effective operation on a variety of different data sets, TF-IDF and its algorithm version are the default choice. This method's formulation is as follows:

$$W(d, t) = tf (t, d) * \log (N / n_t)$$

Equation 4.1

where:

w(t,d) = term weight in document d

tf (t,d) = term frequency in document

N = the total number of document

n_t = number of documents that have term t

4.2 K-nearest Neighbor (KNN)

It is a supervised machine learning approach that employs proximity to classify or predict how a single data point will be grouped. In order to learn by analogy, nearest-neighbor classifiers compare a given test tuple with training tuples that are similar to it. The letter "K" stands for the number of closest neighbors to a new unknown variable that needs to be forecasted or categorized. In order to determine what class, a new unknown data point belongs to, it seeks to locate all of its nearest neighbors. It is also known as a lazy learner algorithm since it saves the training dataset rather than learning from it immediately. Instead, it uses the dataset to perform an action when classifying data. If k=1, then the object is simply assigned to the class of that single nearest neighbor. The advantages of KNN are: Easy to implement, adapts easily ad few hyper-parameter.

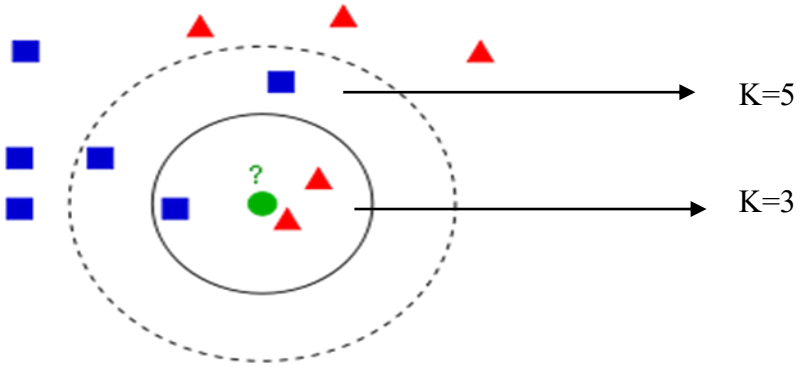

Figure 4.1 Prediction Variations by K Values

KNN works as follows:

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated **Euclidean distance**.

- The training tuples are described by n attributes.
- Each tuple represents a point in an n-dimensional space.
- When given an unknown tuple, a k-nearest-neighbor classifier searches the pattern space for the k training tuples that are closest to the unknown tuple.
- "Closeness" is defined in terms of a distance metric, such as Euclidean distance. The Euclidean distance between two points or tuples, say,
- X1 = (x11, x12,..., x1n) and X2 = (x21, x22,..., x2n), is

$$dist(\boldsymbol{X}_1, \boldsymbol{X}_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}.$$

Equation 4.2

where,

x1, x2 = two points in Euclidean n-space

x1i − $x2i$ = Euclidean vectors, starting from the origin of the space (initial point)

n = n-space

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

4.3 The System Flow

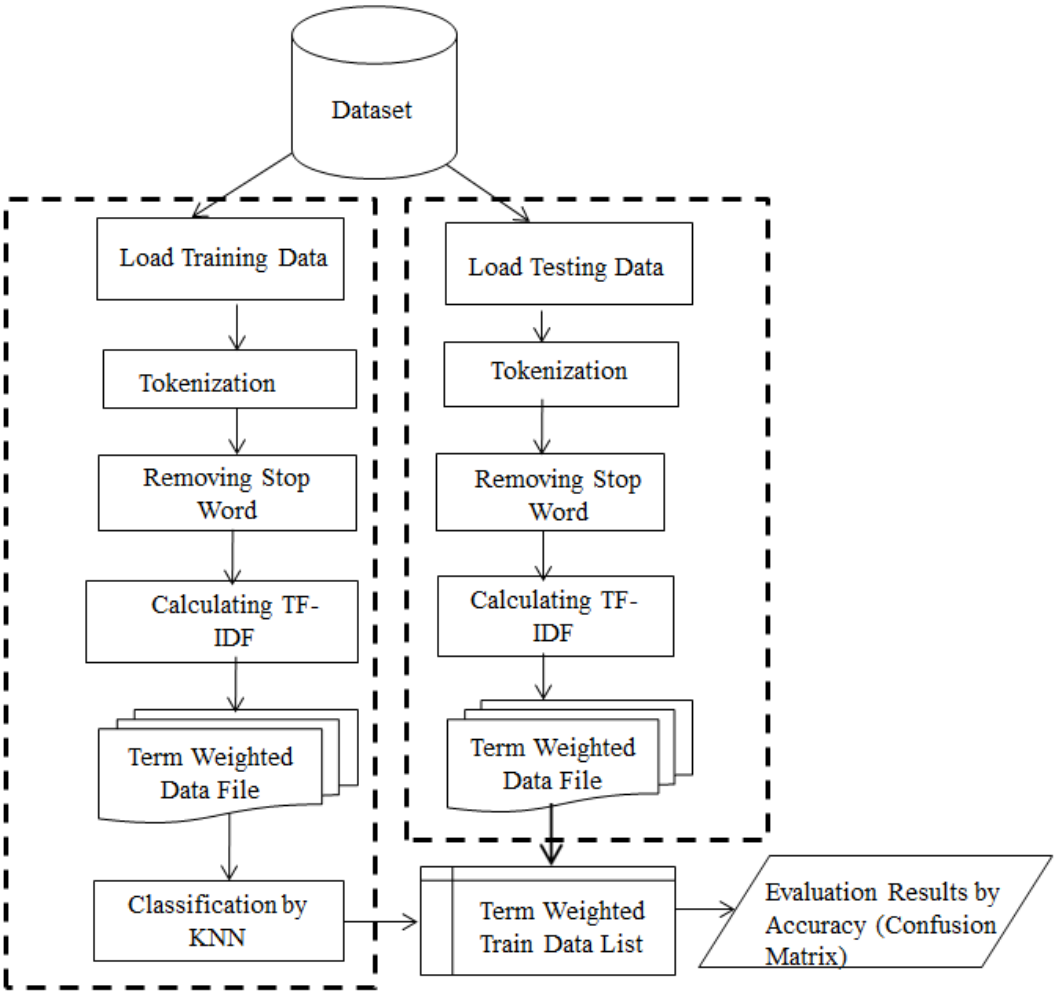

Figure 4.2 Twitter Spam Classification Model

The proposed system's quick process flow is depicted in Figure 4.2 above. The processing of the system phases can be classified into two categories: training phase and testing phase. Before determining weight or classification in any phase, data pre-processing (Tokenization and Removing Stop Words) will be performed. The pre-processing procedures are briefly discussed as follows, and section 4.4 provides a detailed explanation of the subsequent steps.

Tokenization

- Tokenization is the process of breaking down the text corpus into individual elements.

Hello! Dear, I am on leave today.

hello	dear	i	am	on	leave	today

Removing Stop Words

- Stop words are unnecessary word that commonly appeared in the text.
- Example, words such as so, and, or, the, etc., All stop words are removed first. In the figure below the stop words are: you, are, that, have, and the.

You are lucky, that you have won the cash prize

lucky	won	cash	prize

4.3.1 Case Study of K-nearest Neighbor with Sample Dataset

In this section K-nearest Neighbor classification for spam detection is explained with sample twitter data. The sample dataset with values of 2 columns is described in Table 4.1.

Table 4.1 Training Sample Dataset

No.	Tweets	Class
1	Nice songï	0
2	I love song	0
3	Fuck it was the best ever 0687119038 nummber of patrik kluivert his son share !	1
4	The best world cup song ever!	0
5	I like shakira..ï»¿	0
6	SEE SOME MORE SONG OPEN GOOGLE AND TYPE Shakira GuruOfMovieï	1
7	Waka best oneï»¿	0
8	Check out this playlist on YouTube	1
9	¡ remember this song!	0
10	Check out this playlist on YouTube:Central ï»¿	1

Table 4.2 Tokenization of Sample Dataset

No.	Tweets	Class
1	nice / song	0
2	i / love / song	0
3	fuck / it / was / the / best / ever / 0687119038 / nummber / of / patrik / kluivert / his / son / share	1
4	the / best / world / cup / song / ever	0
5	i / like / shakira	0
6	see / some / more / song / open / google / and / type / shakira / guruofmovie	1
7	waka / best / one	0
8	check / out / this / playlist / on / youtube	1
9	i / remember / this / song	0
10	check / out / this / playlist / on / youtube / central	1

Table 4.3 Stop Words Removing of Sample Dataset

No	Tweets	Class
1	nice/ song	0
2	love/ song	0
3	fuck/ best/ nummber/ patrik/ kluivert/ son/ share	1
4	best/ world/ cup/ song	0
5	like/ shakira	0
6	song/ open/ google/ type/ shakira/ guruofmovie	1
7	waka/ best	0
8	check/ playlist/ youtube	1
9	remember/ song	0
10	check/ playlist/ youtube/ central	1

4.3.2 TF-IDF Calculation

Calculate TF-IDF

TF-IDF = tf_n (t , d) . Idf (t)

$$Tf_n(t,d) = \frac{\text{number of word (term t) in document d}}{\text{total number of words in document d}} \qquad \textbf{Equation 4.3}$$

$$IDF(t) = \log\left(\frac{n_d}{n_d\ (t)}\right), \ \frac{\text{total number of documents in dataset}}{\text{total number of documents in which term t appears}}$$

TF.IDF for Comment 1

nice = 1/song = 1/

Total term for Comment 1 ==> 2

TF.IDF (nice) = (1/ 2) * Log(10 / 1) = 0.5

TF.IDF (song) = (1/ 2) * Log(10 / 5) = 0.1505

TF.IDF for Comment 2

love = 1/song = 1/

Total term for Comment 2 ==> 3

TF.IDF (love) = (1/ 3) * Log(10 / 1) = 0.3333

TF.IDF (song) = (1/ 3) * Log(10 / 5) = 0.1003

TF.IDF for Comment 3

fuck = 1/best = 1/nummber = 1/patrik = 1/kluivert = 1/son = 1/share = 1/

Total term for Comment 3 ==> 14

TF.IDF (fuck) = (1/ 14) * Log(10 / 1) = 0.0714

TF.IDF (best) = (1/ 14) * Log(10 / 3) = 0.0373

TF.IDF (nummber) = (1/ 14) * Log(10 / 1) = 0.0714

TF.IDF (patrik) = (1/ 14) * Log(10 / 1) = 0.0714

TF.IDF (kluivert) = (1/ 14) * Log(10 / 1) = 0.0714

TF.IDF (son) = (1/ 14) * Log(10 / 1) = 0.0714

TF.IDF (share) = (1/ 14) * Log(10 / 1) = 0.0714

TF.IDF for Comment 4

best = 1/world = 1/cup = 1/song = 1/

Total term for Comment 4 ==> 6

TF.IDF (best) = (1/ 6) * Log(10 / 3) = 0.0871

TF.IDF (world) = (1/ 6) * Log(10 / 1) = 0.1667

TF.IDF (cup) = (1/ 6) * Log(10 / 1) = 0.1667

TF.IDF (song) = (1/ 6) * Log(10 / 5) = 0.0502

TF.IDF for Comment 5

like = 1/shakira = 1/

Total term for Comment 5 ==> 3

TF.IDF (like) = (1/ 3) * Log(10 / 1) = 0.3333

TF.IDF (shakira) = (1/ 3) * Log(10 / 2) = 0.2330

TF.IDF for Comment 6

song = 1/open = 1/google = 1/type = 1/shakira = 1/guruofmovie = 1/

Total term for Comment 6 ==> 10

TF.IDF (song) = (1/ 10) * Log(10 / 5) = 0.0301

TF.IDF (open) = (1/ 10) * Log(10 / 1) = 0.1

TF.IDF (google) = (1/ 10) * Log(10 / 1) = 0.1

TF.IDF (type) = (1/ 10) * Log(10 / 1) = 0.1

TF.IDF (shakira) = (1/ 10) * Log(10 / 2) = 0.0699

TF.IDF (guruofmovie) = (1/ 10) * Log(10 / 1) = 0.1

TF.IDF for Comment 7

waka = 1/best = 1/

Total term for Comment 7 ==> 3

TF.IDF (waka) = (1/ 3) * Log(10 / 1) = 0.3333

TF.IDF (best) = (1/ 3) * Log(10 / 3) = 0.1743

TF.IDF for Comment 8

check = 1/playlist = 1/youtube = 1/

Total term for Comment 8 ==> 6

TF.IDF (check) = (1/ 6) * Log(10 / 2) = 0.1165

TF.IDF (playlist) = (1/ 6) * Log(10 / 2) = 0.1165

TF.IDF (youtube) = (1/ 6) * Log(10 / 2) = 0.1165

TF.IDF for Comment 9

remember = 1/song = 1/

Total term for Comment 9 ==> 4

TF.IDF (remember) = (1/ 4) * Log(10 / 1) = 0.25

TF.IDF (song) = (1/ 4) * Log(10 / 5) = 0.0753

TF.IDF for Comment 10

check = 1/playlist = 1/youtube = 1/central = 1/

Total term for Comment 10 ==> 7

TF.IDF (check) = (1/ 7) * Log(10 / 2) = 0.0998

TF.IDF (playlist) = (1/ 7) * Log(10 / 2) = 0.0998

TF.IDF (youtube) = (1/ 7) * Log(10 / 2) = 0.0998

TF.IDF (central) = (1/ 7) * Log(10 / 1) = 0.1428

Table 4.4 TF-IDF Calculation Results of Training Sample Dataset

Term (words)	TF-IDF									
	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
nice	0.5									
song	0.150	0.1003		0.0502		0.0301			0.0753	
love		0.3333								
fuck			0.0174							
best			0.0373	0.0871			0.1743			
number			0.0174							
patrik			0.0174							
kluivert			0.0174							

son			0.0174							
share			0.0174							
world				0.1667						
cup				0.1667						
like					0.3333					
shakira					0.2330	0.0699				
open						0.1				
google						0.1				
type						0.1				
guruofmoive						0.1				
waka							0.3333			
check								0.1165		0.0998
playlist								0.1165		0.0998
youtube								0.1165		0.0998
remember									0.25	
central										0.1428
Total	0.6505	0.4336	0.1417	0.4707	0.5663	0.5	0.5076	0.3495	0.3253	0.4422
Class	0	0	1	0	0	1	0	1	0	1

Testing set classified :

Check out our Channel for nice Beats!!ï»¿

Tokenization

check / out / our / channel / for / nice / beats

Stop words remove

check/ channel/ nice/ beats

TF.IDF for Comment

check= 1 /channel = 1/ nice=1 / beats=1

Total term for Comment 1 ==> 7

TF.IDF (check) = (1/ 7) * Log(10 / 2) = 0.0998

TF.IDF (channel) = (1/ 7) * Log(10 / 1) = 0.1428

TF.IDF (nice) = (1/ 7) * Log(10 / 1) = 0.1428

TF.IDF (beats) = (1/ 7) * Log(10 / 1) = 0.14

Table 4.5 TF-IDF Calculation Results of Sample Testing Data

Words	Weight
check	0.0998
channel	0.1428
nice	0.1428
beats	0.1428

Distance Between training set 1 and testing set,

For word “check”,

dist = $(0.5-0.0998)^2 + (0.1505-0.0998)^2$

= 0.1602 + 0.0026

= 0.1628

For word “channel”,

dist = $(0.5-0.1428)^2 + (0.1505-0.1428)^2$

= 0.1276 + 0.0006

= 0.1282

For word “nice”,

dist = $(0.5-0.1428)^2 + (0.1505-0.1428)^2$

= 0.1276 + 0.0006

= 0.1282

For word “beats”,

dist = $(0.5-0.1428)^2 + (0.1505-0.1428)^2$

= 0.1276 + 0.0006

= 0.1282

dist (X_1,X_2) = $\sqrt{0.1628 + 3(0.1282)}$

= $\sqrt{0.5474}$

= 0.7398

Distance Between training set 2 and testing set,

For word “check”,

dist = $(0.1003-0.0998)^2 + (0.3333-0.0998)^2$

= 0.0545

For word “channel”,

dist = $(0.1003-0.1428)^2 + (0.3333-0.1428)^2$

= 0.0381

For word “nice”,

dist = $(0.1003-0.1428)^2 + (0.3333-0.1428)^2$

= 0.0381

For word “beats”,

dist = $(0.1003-0.1428)^2 + (0.3333-0.1428)^2$

= 0.0381

dist (X_1,X_2) = $\sqrt{0.0545 + 3(0.0381)}$

=$\sqrt{0.1688}$

= 0.4109

Distance Between training set 3 and testing set,

For word “check”,

dist = $(0.0373-0.0998)^2 + 6(0.0174-0.0998)^2$

= 0.0446

For word “channel”,

dist = $(0.0373-0.1428)^2 + 6(0.0174-0.1428)^2$

= 0.1055

For word “nice”,

dist = $(0.0373-0.1428)^2 + 6(0.0174-0.1428)^2$

= 0.1055

For word “beats”,

dist = $(0.0373-0.1428)^2 + 6(0.0174-0.1428)^2$

= 0.1055

dist (X_1,X_2) = $\sqrt{0.0446 + 3(0.1055)}$

= 0.6009

Distance Between training set 4 and testing set,

For word “check”,

dist = $(0.0502-0.0998)^2 + (0.0871-0.0998)^2 + 2\,(0.1667-0.0998)^2$

= 0.0025 + 0.0002 + 0.0089

= 0.0116

For word “channel”,

dist = $(0.0502-0.1428)^2 + (0.0871-0.1428)^2 + 2\,(0.1667-0.1428)^2$

= 0.0086+ 0.0031 + 0.0014

= 0.0131

For word “nice”,

dist = $(0.0502-0.1428)^2 + (0.0871-0.1428)^2 + 2\,(0.1667-0.1428)^2$

= 0.0086+ 0.0031 + 0.0014

= 0.0131

For word “beats”,

dist = $(0.0502-0.1428)^2 + (0.0871-0.1428)^2 + 2\,(0.1667-0.1428)^2$

= 0.0086+ 0.0031 + 0.0014

= 0.0131

dist (X_1,X_2) = $\sqrt{0.0116 + 3(0.0131)}$

=$\sqrt{0.0509}$

= 0.2256

Distance Between training set 5 and testing set,

For word “check”,

dist = $(0.3333-0.0998)^2 + (0.2330-0.0998)^2$

= 0.0723

For word “channel”,

dist = $(0.3333-0.1428)^2 + (0.2330-0.1428)^2$

= 0.0444

For word “nice”,

dist = $(0.3333-0.1428)^2 + (0.2330-0.1428)^2$

= 0.0444

For word “beats”,

dist = $(0.3333-0.1428)^2 + (0.2330-0.1428)^2$

= 0.0444

dist (X_1,X_2) = $\sqrt{0.0723 + 3(0.0444)}$

=$\sqrt{0.2055}$

= 0.4533

Distance Between training set 6 and testing set,

For word “check”,

dist = $(0.0301-0.0998)^2 + (0.0699-0.0998)^2 + 4(0.1-0.0998)^2$

= 0.0058

For word “channel”,

dist = $(0.0301-0.1428)^2 + (0.0699-0.1428)^2 + 4(0.1-0.1428)^2$

= 0.0253

For word “nice”,

dist = $(0.0301-0.1428)^2 + (0.0699-0.1428)^2 + 4(0.1-0.1428)^2$

= 0.0253

For word “beats”,

dist = $(0.0301-0.1428)^2 + (0.0699-0.1428)^2 + 4(0.1-0.1428)^2$

= 0.0253

dist (X_1,X_2) = $\sqrt{0.0058 + 3(0.0253)}$

= 0.2858

Distance Between training set 7 and testing set,

For word “check”,

dist = $(0.1743-0.0998)^2 + (0.3333-0.0998)^2$

= 0.0601

For word “channel”,

dist = $(0.1743-0.1428)^2 + (0.3333-0.1428)^2$

= 0.0373

For word “nice”,

dist = $(0.1743-0.1428)^2 + (0.3333-0.1428)^2$

= 0.0373

For word “beats”,

dist = $(0.1743-0.1428)^2 + (0.3333-0.1428)^2$

= 0.0373

dist (X_1,X_2) = $\sqrt{0.0601 + 3(0.0373)}$

=$\sqrt{0.172}$

= 0.4147

Distance Between training set 8 and testing set,

For word “check”,

dist = $3\ (0.1165-0.0998)^2$

= 0.0008

For word “channel”,

dist = $3(0.1165-0.1428)^2$

= 0.0021

For word “nice”,

dist = $3(0.1165-0.1428)^2$

= 0.0021

For word “beats”,

dist = $3(0.1165-0.1428)^2$

= 0.0021

dist (X_1,X_2) = $\sqrt{0.0008 + 3(0.0021)}$

=$\sqrt{0.0071}$

= 0.0843

Distance Between training set 9 and testing set,

For word “check”,

dist = $(0.0753-0.0998)^2 + (0.25-0.0998)^2$

= 0.0232

For word “channel”,

dist = $(0.0753-0.1428)^2 + (0.25-0.1428)^2$

= 0.0160

For word “nice”,

dist = $(0.0753-0.1428)^2 + (0.25-0.1428)^2$

= 0.0160

For word “beats”,

dist = $(0.0753-0.1428)^2 + (0.25-0.1428)^2$

= 0.0160

dist (X_1,X_2) = $\sqrt{0.0232 + 3(0.0160)}$

$=\sqrt{0.0172}$ = 0.1311

Distance Between training set 10 and testing set,

For word “check”,

dist = $3(0.0998-0.0998)^2$

= 0

For word “channel”,

dist = $(0.1428-0.1428)^2$

= 0

For word “nice”,

dist = $(0.1428-0.1428)^2$

= 0

For word “beats”,

dist = $(0.1428-0.1428)^2$

= 0

dist (X_1,X_2) = $\sqrt{0+3(0)}$

= $\sqrt{0}$

= 0

Table 4.6 Distances of Training Test and Testing Test

	Distance	Rank	Class
D1	0.7389	10	0
D2	0.4109	6	0
D3	0.6009	9	1
D4	0.2566	4	0
D5	0.4533	8	0
D6	0.2858	5	1
D7	0.4147	7	0
D8	0.0843	2	1
D9	0.1311	3	0

D10	0	1	1

For k ➔3 : Testing data prediction is 1

Two neighbors are “Ham” class as positive and one neighbor is “Spam” class as negative. So, the predicted class for testing data is “Ham”.

4.4 Implementation of the System

The proposed system uses the ASP.Net programming language on the Microsoft Visual Studio 2015 version IDE and the Microsoft SQL Server 2017 Express edition as the database engine to implement the classification of Twitter spam comments. TF.IDF and KNN Approach will be used to develop the classification analysis. The following illustrates the proposed system's system main page.

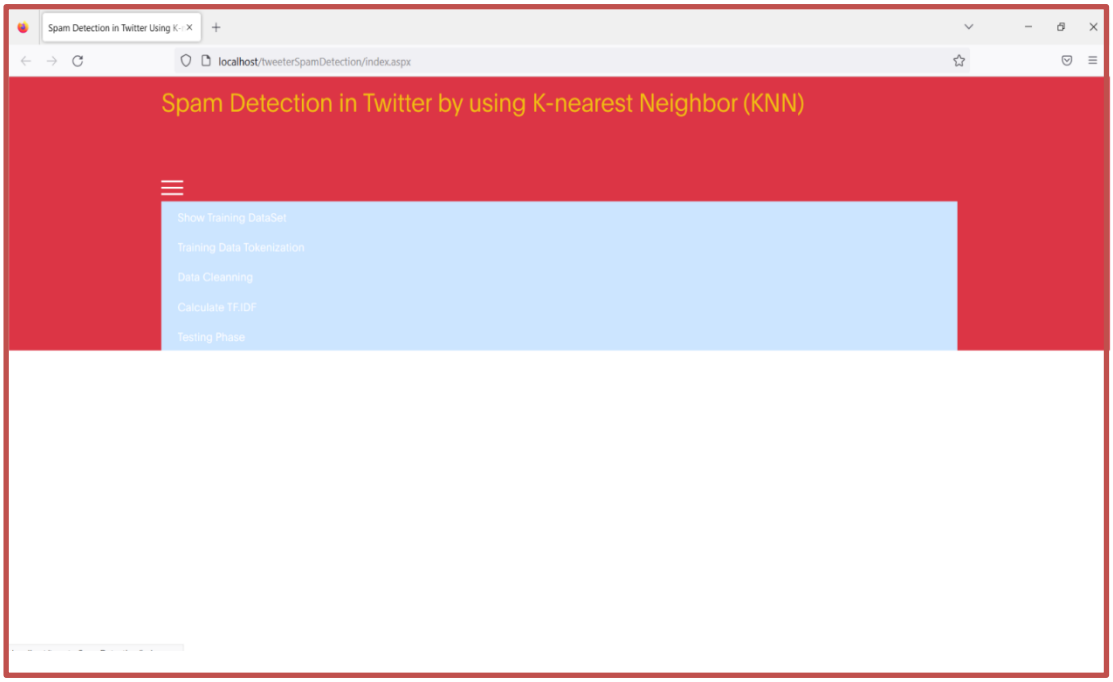

Figure 4.3 System Main Page

The “Show Training Data” button is used for training the load dataset before the tokenization process in shown Figure 4.4.

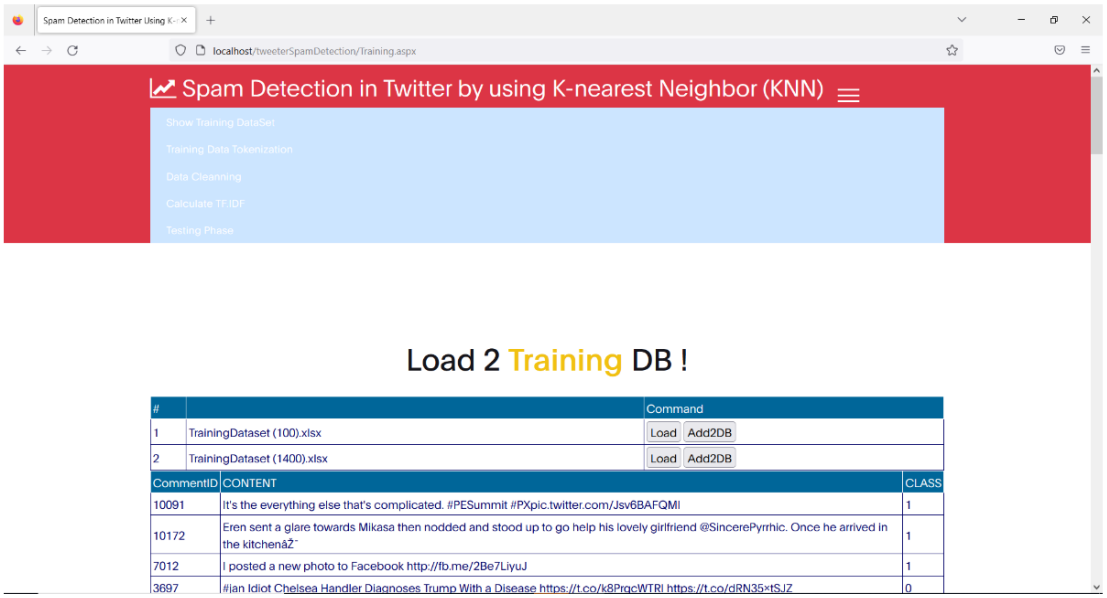

Figure 4.4 Load Training Data

Prior to removing the elements, preprocessing is important. Stop words like the verb to be in the action, pronouns, relational words, and conjunctions don't provide useful information for examining opinions. As a result, the stop words are removed to reduce handling time. The preprocessing steps are shown in Figures 4.5 and 4.6.

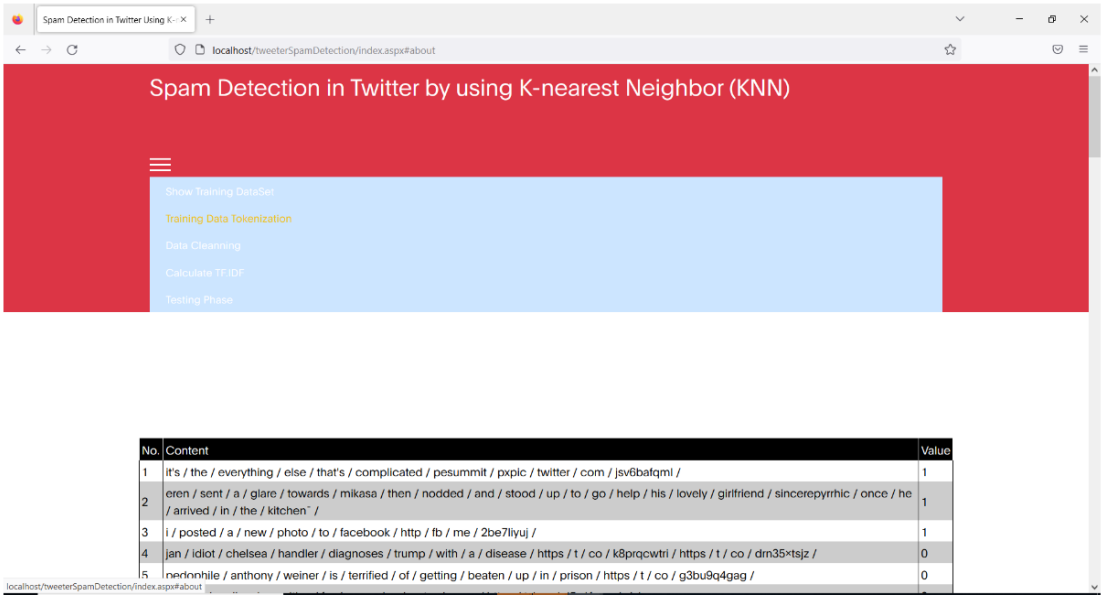

Figure 4.5 Training Data Tokenization

The "Tokenization" phase of the training process is depicted in Figure 4.5. The following are the specific tokenization processes:

- Words are constructed from individual remarks.
- The training comments' words are divided via tokenization.
- String Tokenizer is used to do tokenization.

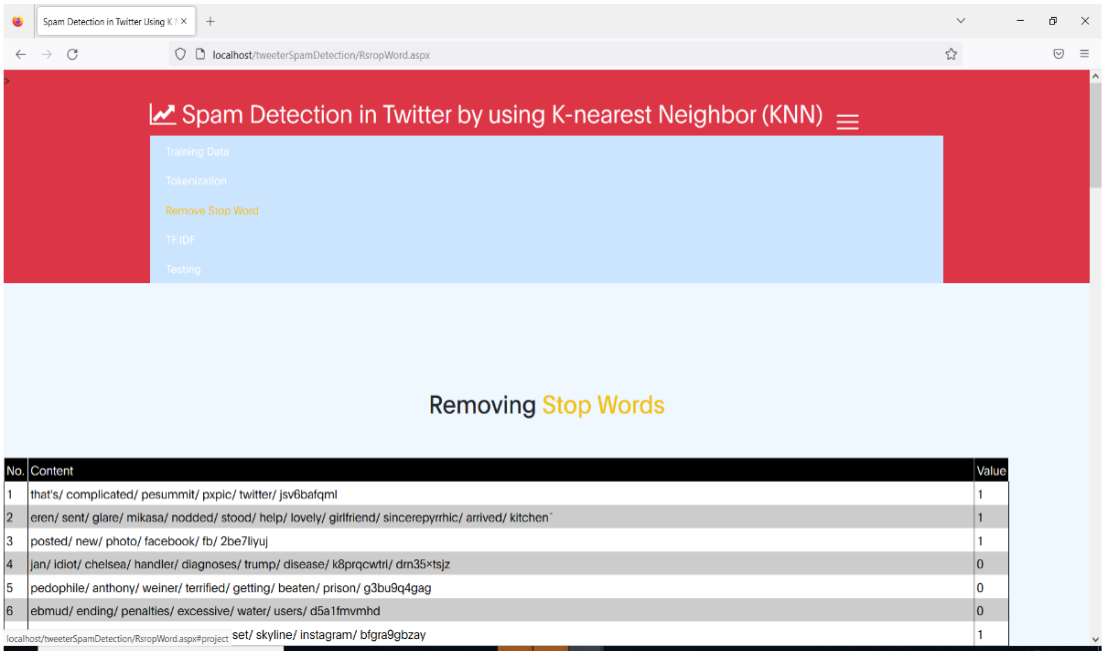

Figure 4.6 Removing Stop-word on Training Data

Figure 4.6 is the step of removing stop-words from training data. Words that do not have particular meaning such as “a”, “and”, “the” and some other common words should be removed. Stop-words lists are stored in the system database.

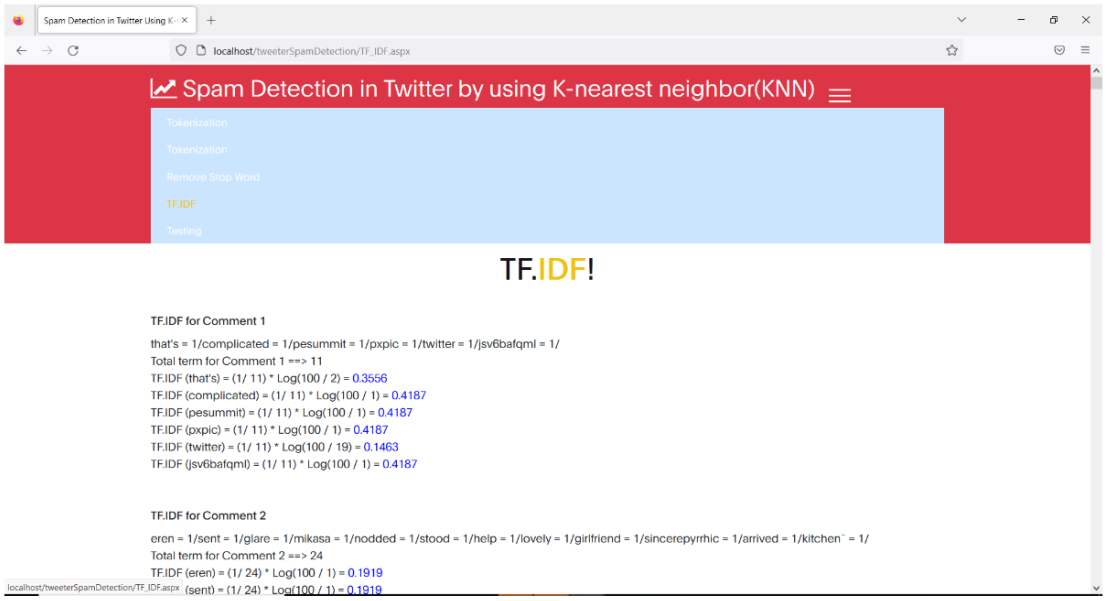

Figure 4.7 Weight Calculations on Training Data (TF.IDF)

The element determination technique, or IF.IDF, is a factual metric that evaluates the importance of a word to a report over a range of reports. This is accomplished by multiplying two metrics: the frequency with which a word appears in a record and the term's backwards report recurrence across a number of records. The preparatory interaction is complete once the component assignment and determination have been completed. The testing phase can then be utilized to continue identifying spam tweets. The accompanying Figure 4.8 shows the testing information stacking pages.

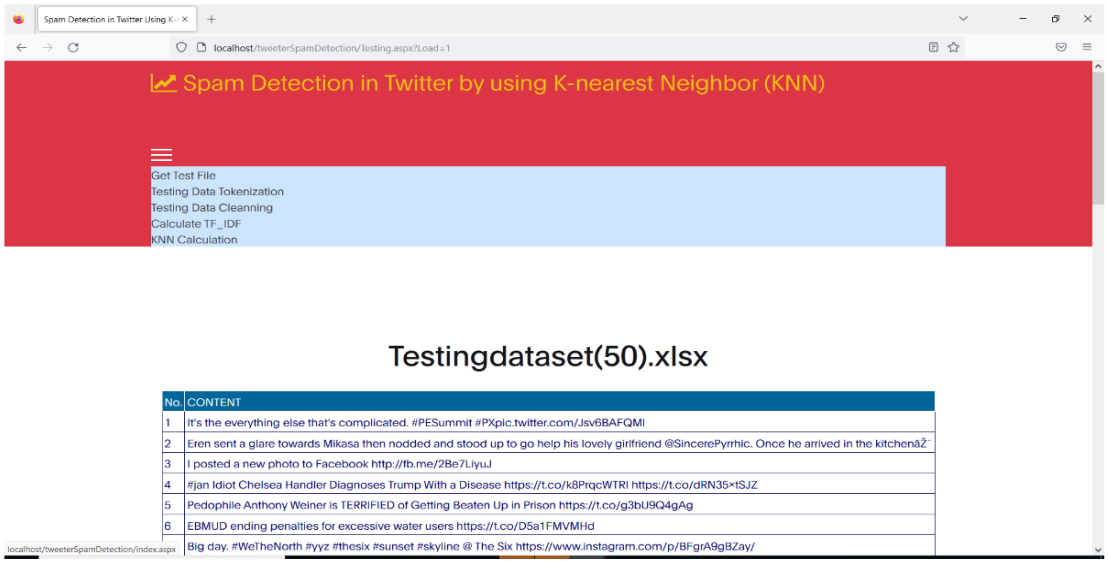

Figure 4.8 Testing Data Loading

In overflow, stop words are available in every human language. By removing these terms, the low level information from the text that was omitted can help the significant information stand out more. The model that has been trained for the task performs poorly when all words that denote a state are removed. The removal of stop words definitely reduces the size of the dataset, and the preparation time is also reduced in this way because there are little tokens to prepare. Following the loading of the testing data, the tokenization and removal of stop words occur in a manner similar to that depicted in Figures 4.5 and 4.6. Then, using pre-processed test data, TF.IDF computation will be performed. The final spam detection result is displayed in Figure 4.9.

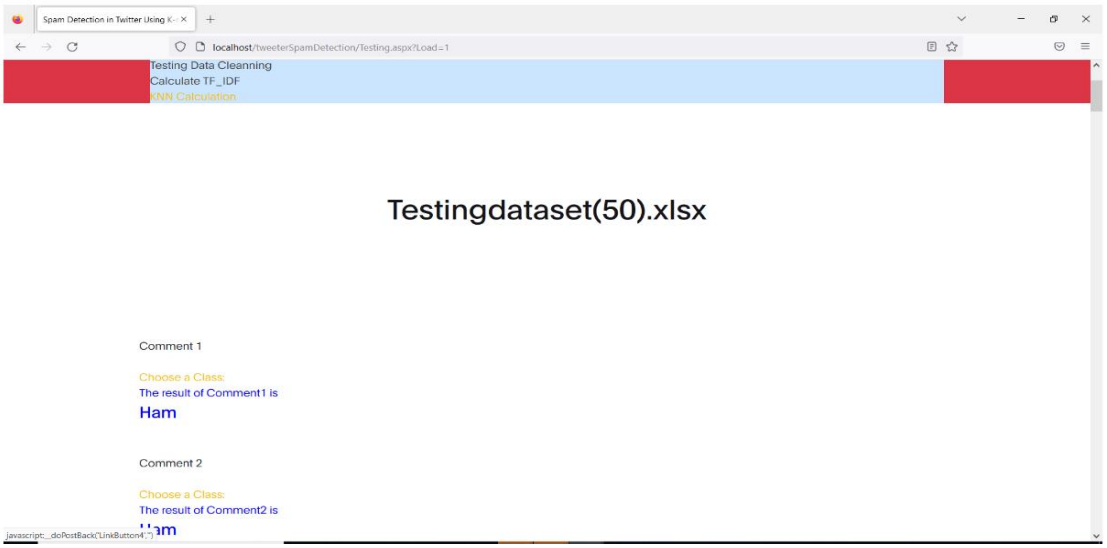

Figure 4.9 Spam Classification Page

4.4.1 Precision, Recall, F-measure

Four evaluation performances which are precision, recall, F-measure and accuracy are used to estimate the efficiency of the system. These are calculated by using Equation (4.4), (4.5), (4.6), (4.7).

$$\textbf{Accuracy} = \frac{TP+TN}{TP+FN+FP+TN} \qquad \textbf{Equation 4. 4}$$

$$\textbf{Recall} = \frac{TP}{TP+FP} \qquad \textbf{Equation 4. 5}$$

$$\textbf{Precision} = \frac{TP}{TP+FN} \qquad \textbf{Equation 4. 6}$$

$$\textbf{F-measure} = \frac{2*Recall*Precision}{Recall+Precision} \qquad \textbf{Equation 4. 7}$$

where:

- TP: It is true positive which means the comment were actually spam and the classifier predicted as ‘spam ‘.
- TN: It is true negative which means the comment were actually real and the classifier predicted as ‘not spam’.
- FP: It is false positive which means the comment were actually real and the classifier predicted as ‘spam’.

- FN: It is false negative which means the comment were actually spam and the classifier predicted the comment as ‘non spam’.

In this tweets’ Spam and Ham three experiments are conducted to test the detecting system. To analysis, the dataset is split into 70% training data and 30% of testing data. This system used Accuracy, Precision, Recall, and F-measure of each analysis to evaluate the performance of the experiment results. As in shown in Figure 4.10, almost performance values are higher than 90%. About 90% accuracy is maintained in this proposed work.

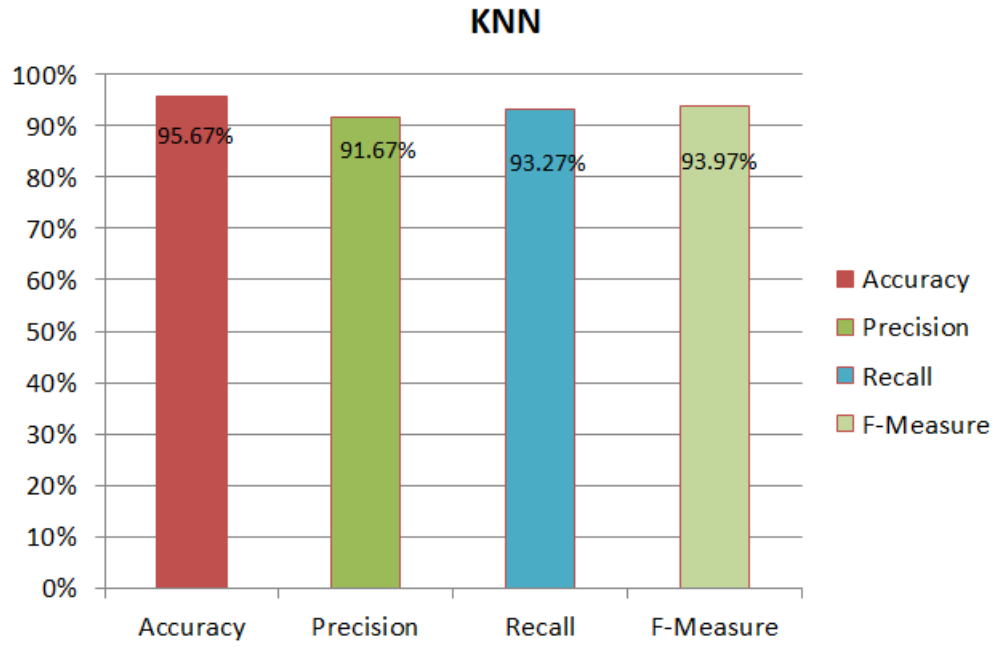

Figure 4.10 Experiment Results of KNN

Based on the analysis, the system can give better classification result if the more trained data can feed to this system.

CHAPTER 5
CONCLUSION, LIMITATION AND FURTHER EXTENSION

Twitter is a great place to start when analyzing social media. The technology preprocesses the gathered spam tweets using machine learning methods. The KNN classifier is used to classify the spam tweets and choose the spam features. With the suggested system, users may easily get a summary of the opinions expressed in the spam tweets.

5.2 Advantages of the System

For Twitter comments, the offered solution provides high-performance and high-accuracy services. The KNN Classifier cannot directly use categorical data. For improved outcomes, TF-IDF is usually used on the text before training the model. Before TF-IDF produced more accurate text classification, stop words were removed. Using the appropriate feature vectors that can be provided by the TF-IDF feature selection methods will improve the performance of the classifier. The system's evaluation shows that after the training phase is through, it can carry out classification tasks with speed and accuracy.

5.3 Limitations and Further Extensions

This audio file can be set up as a separate dataset or a future extension. Utilizing one's own dataset within a popular social media platform may also enable future expansion. Furthermore, such a diversified dataset can be combined with a comparison classifier.

AUTHOR'S PUBLICATION

[1] Shwe Tha Zin, Zin Thu Thu Myint, "Spam Detection in Twitter by Using K-nearest Neighbor (KNN)", National Journal of Parallel and Soft Computing, Yangon, Myanmar, 2022.

REFERENCES

[1] Aryo Pinandito, Rizal Setya Perdana, Mochamad Chandra Saputra, "Spam Nearest Neighbor Classifiers", Information System Department, Computer Science Faculty, Universities Brawijaya, 2017.

[2] B. Biggio, I. Corona, G. Fumera, G. Giacinto, and F. Roli, ―Bagging classifiers for fighting poisoning attacks in adversarial classification tasks," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 6713 LNCS, pp. 350–359, 2011, doi: 10.1007/978-3-642-21557-5_37.

[3] Bratko, A., Filipi˘c, B., Cormack, G., Lynam, T ., Zupan, B.: Spam filtering using statistical data compression models. The Journal of Machine Learning Research 7 (2016) 2673–2698

[4] E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O. E. Ajibuwa, "Machine learning for email spam filtering: review, approaches and open research problems,‖ Heliyon, vol.5,no.6,2019, doi: 10.1016/j.heliyon.2019.e01802.

[5] F. Murtagh, ―Multilayer perceptrons for classification and regression,‖ Neurocomputing, vol. 2, no. 5–6, pp. 183–197, 1991, doi: 10.1016/0925-2312(91)90023-5.

[6] H. Sharma and S. Kumar, ―A Survey on Decision Tree Algorithms of Classification in Data Mining,‖ Int. J. Sci. Res., vol. 5, no. 4, pp. 2094–2097, 2016, doi: 10.21275/v5i4.nov162954.

[7] KamalanathanKandasamy, PreethiKoroth: An Integrated Approach to Spam Classification on Twitter Using URL Analysis, Natural Language Processing, and Machine Learning Techniques , 2018 IEEE Student s' Conference on Electrical, Electronics and Computer Science.

[8] L. Breiman, ―ST4_Method_Random_Forest,‖ Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1017/CBO9781107415324.004.

[9] P. Cunningham and S. J. Delany, ―K -Nearest Neighbour Classifiers, Mult. Classif. Syst., no. May, pp. 1–17, 2007, doi: 10.1016/S0031-3203(00)00099-6.