Detecting Spam in Emails using Advanced Machine Learning Methods

By

Nirali Mistry

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science (MSc) in Computational Sciences

The Office of Graduate Studies
Laurentian University
Sudbury, Ontario, Canada

© Nirali Mistry, 2020

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Office of Graduate Studies/Bureau des études supérieures

Title of Thesis Titre de la thèse	Detecting Spam in Emails using Advanced Machine Learning Methods	
Name of Candidate Nom du candidat	Mistry, Nirali	
Degree Diplôme	Master of Science	
Department/Program Département/Programme	Computational Sciences	Date of Defence Date de la soutenance June 03, 2022

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Kalpdrum Passi
(Supervisor/Directeur(trice) de thèse)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Oumar Gueye
(Committee member/Membre du comité)

Dr. Nirbhay Chaubey
(External Examiner/Examinateur externe)

Approved for the Office of Graduate Studies
Approuvé pour le Bureau des études supérieures
Tammy Eger, PhD
Vice-President Research (Office of Graduate Studies)
Vice-rectrice à la recherche (Bureau des études supérieures)
Laurentian University / Université Laurentienne

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

I, **Nirali Mistry**, hereby grant to Laurentian University and/or its agents the non-exclusive license to archive and make accessible my thesis, dissertation, or project report in whole or in part in all forms of media, now or for the duration of my copyright ownership. I retain all other ownership rights to the copyright of the thesis, dissertation or project report. I also reserve the right to use in future works (such as articles or books) all or part of this thesis, dissertation, or project report. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that this copy is being made available in this form by the authority of the copyright owner solely for the purpose of private study and research and may not be copied or reproduced except as permitted by the copyright laws without written authority from the copyright owner.

Abstract

E-mail is one of the quickest and most professional ways to send messages from one location to another around the world; however, increased use of e-mail has increased to received messages in the mailbox, where the recipient receives a large number of messages, some of which cause significant and varied problems, such as the theft of the recipient's identity, the loss of vital information, and network damage. These communications are so harmful that the user has no way of avoiding them, especially when they come in a variety of forms, such as adverts and other types of messages. Spam is the term for these emails Filtering is used to delete these spam communications and prevent them from being viewed. This research intends to improve e-mail spam filtering by proposing a single objective evaluation algorithm issue that uses Deep Learning, Genetic Algorithms, and Bayes theorem-based classifiers to build the optimal model for accurately categorizing e-mail messages. Text cleaning and feature selection are used as the initial stage in the modeling process to minimize the dimension of sparse text features obtained from spam and ham communications. The feature selection is used to choose the best features, and the third stage is to identify spam using a Genetic algorithm classifier, Support Vector Machine, Bayesian Classifier, Nave Bayes, SVM, Random Forest, and Long-Short Term Memory classifier.

Keywords: Spam Database, Feature Selection, Genetic Algorithm

Acknowledgments

I would sincerely like to express my warmest gratitude and appreciation to my supervisor, Dr. Kalpdrum Passi, who gave me the opportunity and all the possible support to make this research successful. I believe I have got the finest supervisor who has been standing with me throughout the educational journey and he was always available to clear all my doubts and handled all the queries with patience.

I would also like to thank my technical coordinator, Dr. Kalpdrum Passi for providing all the server requirements for the training phase. He was always available and supported all the dependencies required in the environments.

I would like to thank and appreciate my parents, brother, and friends who made my research experience enjoyable. I want to give a shout-out to all my classmates for their help and support during the testing phase.

Finally, and most significantly, I offer special thanks to my parents for their support and patience, in me, and trust in calibre.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Spam is an unwanted and unsolicited electronic communication delivered by a sender who has no established relationship with the recipient [1]. Electronic spam is divided into numerous categories: e-mail, SMS, social media, and online commerce platforms. Spam takes users' time because they have to identify and delete unwanted communications; it also gobbles up inbox capacity and buries valuable personal e-mails [2]. On the other hand, SMS spam is often sent through a mobile network [3]. Due to a large number of spammers and the possible detrimental consequences of social network spam on the convenience and comprehension of all users, social network spam has recently gained greater attention from both researchers and practitioners [4]. As a result, spam filtering is given a lot of thought in the above communication routes.

Spam communications can be manually or automatically screened. Manual spam filtering, which involves recognizing spam messages and eliminating them, is time-consuming. Furthermore, spam communications may contain a security risk, such as links to phishing websites or malware-hosting servers. As a result, researchers and practitioners have worked for decades to improve automatic spam filtering systems. Machine learning algorithms are known for being very good at detecting spam emails. The basic idea behind machine learning algorithms is to create a word list and then give a weight to each word. Spammers, on the other hand, frequently incorporate common valid statements in spam messages to reduce the likelihood of being identified. Neural networks (NNs) [5], support vector machines (SVMs) [6], Naive Bayes (NB) [7] [8], and Random Forest (RF) are some of the known machine learning techniques used in spam filtering [9].

Ensemble learning approaches such as bagging and Random Forest outperform traditional single classifiers, according to a study by Kaur [10]. In comparison to single algorithms, ensemble approaches integrate the predictions of numerous underlying machine learning algorithms to enhance accuracy and precision. In previous studies, traditional classifiers like decision trees were used to effectively filter spam messages using ensemble approaches. Surprisingly little research has been done on Neural Networks (NNs) in ensemble learning. Recent data suggest that NNs using regularization approaches may detect spam in e-mail and SMS messages with excellent accuracy. This is due to improved optimization convergence and overfitting resistance. They also combined regularized Neural Networks with ensemble learning approaches for automatic spam filtering to make use of these properties. Rectified linear units (ReLu) and dropout regularization are utilized in deep feedforward NNs (DFFNNs) to improve the performance of the proposed technique by addressing the optimization convergence to a bad local minimum problem that is typical in classic shallow NN models.

In general, spam filtering is a binary classification issue, in which each message must be classified as spam or ham. In addition to high accuracy, spam filtering algorithms should have a low false-positive rate (when a genuine communication is mistakenly classified as spam) to avoid instances where legitimate messages are not delivered to the intended recipient. Furthermore, a standard classification performance metric based on accuracy ignores the various costs associated with type I and type II mistakes. To determine the outcome of Type I and Type II mistakes, the null hypothesis is used. Type I errors reject the null hypothesis even if it is true, whereas type II errors do not reject the null hypothesis even if the alternative hypothesis is right. These are both called as False Negatives.

Because the minority class (typically the class of spam messages) has less influence on accuracy compared to the majority class of valid communications, using accuracy for sometimes

extremely unbalanced spam datasets might lead to incorrect results. Therefore, multiple performance measures must be considered when evaluating the spam filtering algorithms.

As previously said, the primary principle behind content-based machine learning models is to create a word (list) and give weight to each word or phrase (bag-of-words) or word category (part-of-speech tagging) in the list [11]. Such characteristics, on the other hand, are sparse, making it challenging to capture semantic representations of communications. Ren and Ji (2017) presented a gated recurrent NN (GRU) model to identify review spam to overcome this issue. This method used word embedding derived from the CBOW (continuous bag-of-words) model [12] [13] to map words to vectors depending on context. As a result, global semantic information may be gathered, and the problem of sparse data can be addressed to some extent. This method was said to be more successful than traditional tagging using a bag of words or a chunk of speech [14]. This study was based on these recent results and used word embedding to extract the semantic representation of e-mails, SMS, social network communications, and online reviews.

This research aims to explore machine learning models based on traditional algorithms and ensembles using a high-dimensional feature representation for spam filtering using two major open-source Spam Datasets which are Enron and Spam Assassin.

The term spam is derived from the 1970 "Spam" sketch of the BBC sketch comedy television series Monty Python's Flying Circus. The sketch, set in a cafe, has a waitress reading out a menu where every item but one includes the Spam canned luncheon meat. As the waitress recites the Spam-filled menu, a chorus of Viking patrons drown out all conversations with a song, repeating "Spam, Spam, Spam, Spam… Lovely Spam! Wonderful Spam!

1.2 Importance of Spam Filtering

Spam is simple in concept: send a message to millions of people and benefit from the one person who responds. According to recent surveys by SAINT (Global Security Map), 80 percent of e-mails are spam, with considerable disparities in spam rates among nations. From Figure 1, it can be seen that majority of the cyber security threats can be found in Russia, China, and United States.

Figure 1.1 Global Security Map by SAINT <*https://globalsecuritymap.com*>

As a result, substantial negative repercussions on the global economy have been reported such as poorer productivity, spam delivery costs, and spam and virus/phishing attack delivery costs [15] [16] [17]. As a result, a good spam filter may boost user productivity while also reducing the need for information technology resources like the help desk. Individuals may have more faith in e-mail communication if spam filters are more accurate [18].

According to statistics, spam accounts for a considerable part of email inboxes and all social media messages. For example, according to a survey conducted by Nexgate, a prominent cyber security firm, there was a 355 % increase in spam in the first half of 2013 [19]. Many users have been attracted to social networks because of their expanding capabilities and popularity. The number of social network users is gradually increasing these days, and social networks are used for a significant amount of communication. However, along with genuine and essential information, these networks often distribute improper and harmful content. Spammers do, in fact, target users of social media sites.

1.3 E-mail Spam Filtering

Spammers (those who send spam messages) collect e-mail addresses from a variety of places, including websites and chatrooms, and send unwanted messages in mass. This hurts the receiver, resulting in a waste of time and money. E-mail spam, in particular, has a severe impact on the memory of the email server, CPU performance, and user time. Furthermore, spammers' deceptive techniques may cause victims to suffer significant financial losses.

Even though global spam volume (as a percentage of total e-mail traffic) has decreased to roughly 55% over the last decade [20], the number of e-mail messages carrying hazardous attachments (viruses, ransomware, and other malware) has continually climbed [20]. China is the country with the greatest e-mail spam, accounting for roughly 20% of all e-mail spam. Spam senders are compelled to send spam that avoids spam filters to maximize income. As a result, spam filtering is a difficult operation since spammers employ various strategies to reduce spam detection rate. To get through spam filters, you can use a variety of techniques, such as utilizing irrelevant, odd, or misspelled phrases.

Non-machine learning and machine learning approaches to spam filtering can be distinguished. Legislative initiatives [22], improvements to protocols and operational models [23], rule-, signature-, and hash-based filtering, whitelists (trusted senders), and traffic analysis are among the former [24]. Kaya and Ertugrul developed a useful method based on the likelihood of employing characters in similar order based on their UTF-8 values [25].

Spam filters that use machine learning automatically determine whether or not a communication is a spam based on its content. Automated spam filtering can be characterized as follows, according to Sebastiani (2002) and Zhang (2004) [27] [2].

Let D = {d1, d2, ..., di, ..., dN} be a message set and C = {spam, legitimate} be a class set. A spam filter's purpose is to create a model that can identify each communication as spam or ham. It comes at a price to misclassify a valid communication as spam (a false positive) or to misclassify spam as non-spam (a false negative). This is a difficult endeavour since spammers frequently use acceptable phrases to reduce the likelihood of their communications being recognized as spam [27]. Spam filtering with machine learning methods begins with text pre-processing [28], with tokenization being conducted initially to extract the words (multi-words) in each message. Following that, the original list of words is usually reduced by stemming, lemmatization, and removing stop-words. The vector-space model, often known as the bag-of-words (BoW), is a popular way to express the weights of pre-processed words. The term frequency-inverse document frequency (TF-IDF) is a typical specialized weighting system. Filters or wrappers can then be used to minimize the size of the feature space, which is helpful because not all classification techniques can handle high-dimensional data [7] [8]. Finally, machine learning methods are applied to classify the pre-processed dataset.

Because of their simplicity and computing efficiency, Naive Bayes algorithms were used in the earliest spam classifiers [29] [30]. Another prominent spam-classification method, SVM,

has been proven to be resilient to various datasets and pre-processing strategies [31]. Comparative studies have shown that it outperforms NB, k-nearest-neighbour (k-NN), decision trees, and NN techniques. Another interesting solution for spam filtering is artificial immune systems (AISs) [32]. To increase spam filter efficacy, Zitar and Hamdan (2013) employed a genetic approach to train AISs [30]. Recent assessments of e-mail spam filtering imply that content-based deep learning is the way of the future for e-mail spam filtering [21]. Chapter 2 will explore all research and past studies on spam filtering in detail.

1.4 Objectives

The primary objective of this study is to examine the different methods and machine learning algorithms for detecting Spam in Email messages. This study initially proposed to build baseline models using traditional machine learning algorithms such as Naïve Bayes, Support Vector Machine, Bayesian Classifier and Genetic Classifier with better Accuracy using Feature selection methods and later build ensemble models using Decision Trees, Extreme Boosting, Random Forest and Long Short-Term Memory based Artificial Neural Network. The main objective is to test and validate different methods to get higher accuracy. The performance metrics used to test the models include Accuracy, Precision, recall, and F1-Score.

1.5 Thesis Outline

This thesis is organized as follows:

Chapter 1 reviews the introduction and importance of filtering spam messages and the objectives of this thesis.

Chapter 2 reviews the literature associated with Spam Filtering.

Chapter 3 introduces the proposed research methodology and techniques implemented, and presents the datasets used for the experimental comparison and presents the strategies for data pre-

processing and feature selection outlines the proposed spam filtering model and briefly introduces the state-of-the-art models used for comparisons, experimental settings

Chapter 4 presents the results, as well as a comparative analysis with the state-of-the-art methods used for spam filtering.

Chapter 5 discusses the limitations, suggests possible future directions, and concludes the thesis.

Chapter 2

Literature Survey

Email spam, often known as junk email or unsolicited email, comes under the category of electronic spam. Phishing emails are an example of this type of spam [40] [41]. Spammers have turned their attention to e-commerce sites and their consumers as a result of their popularity. A spammer is someone or something that writes spam, whether it's a human, computer, or software (spam maybe email spam, web spam, review spam, etc.). There are two categories of spammers that we might find when looking for spam. These are Individual Spammer or a single spammer and a group of spammers which divide the group into sub-groups and each of these subdivisions work on spamming. Three types of machine learning techniques are used to detect spam. These are:

1. Supervised Learning Approach: Labeled data sets are required for the supervised learning technique LIWC (Linguistic Inquiry and Word Count), POS tagging, and N-gram TF-IDF scores are the most common characteristics of features used in supervised learning. Following these processes, several classifiers such as SVM, decision tree, logistic regression, Naive Bayes, and others are trained, and their accuracy is assessed. This is one of the most extensively used spam detection methods.
2. Semi-supervised Learning Technique: Semi-supervised learning is similar to supervised learning, with the exception that we don't have to label the entire data set. We can utilize such a learning strategy if we just have a few labeled data sets. There have been very few studies in this field.
3. Unsupervised Learning Technique: We use an unsupervised learning technique to uncover hidden patterns when dealing with unlabeled data. It contains things like k-mean clustering

and mixture models, among other things.

In the past, a lot of effort has been put into spam detection (email spam, web spam, SMS spam). Phishing is a very common attack on the subject of email spam. Phishing is the theft of personal information such as login ids, passwords, credit card numbers, and other sensitive information for malevolent purposes. In their research, Fette [40] has demonstrated that phishing attacks may be easily recognized with high accuracy. They used machine learning to analyse user-generated feature sets such as IP-based URLs, the age of domain names connected to them, non-matching URLs, links to non-modal websites, HTML emails, the number of links, domains, dots, and spam-filter output. The proposed technique proved successful in detecting phishing websites and emails that guide victims to those sites. The accuracy of their system was tested on a set of 860 phishing emails and 6950 non-phishing emails, and it was found to be above 96% [40].

Li's suggested study [41] is likewise based on email spam, in which they looked at ways to combine numerous email filters with multivariate analysis to produce a spam barricade. Multiple email filters for establishing a barrier are stronger than a single filter barrier alone, according to the authors. For calculating the accuracy, they developed the W-Voting method. The training and filtering phases of the algorithm are the most important. The training phase is used to find all of the different filters, while the filtering phase is used to categorize fresh emails. The experiment was carried out on the PU1 (http://www.aueb.gr/users/ion/data/pu1_encoded.tar.gz) and Ling Spam Corpus datasets (https://metatext.io/datasets/ling-spam-dataset). PU1 Corpus included 43.77% spam, while Ling Spam corpus contained 16.63% spam, according to the author.

Abernethy [47] proposed a graph-based method for detecting online spam. They introduced the WITCH algorithm for detecting web spam and compared it to several existing algorithms, concluding that it outperformed all of them. The WITCH algorithm detects spam hosts or similar web pages. WEBSPAM-UK2006 provided the datasets that were used in this study. There were

11,402 hosts in these datasets, of which 7,473 were tagged and all were on the UK domain. In their suggested study, the authors included 236 characteristics, including average word length, the total number of words in the title, PageRank, total number of neighbours, and others proposed in [18] and [19]. Using the SVM classifier, this approach achieves a maximum accuracy of 95.3%.

To detect SMS spam, Karami deployed a combination of content based and LIWC characteristics. Capital words, spam words, SMS segments, unique words, URL, SMS frequency, using the word 'call', the rate of URL to SMS segments, the rate of spam words to unique words, the rate of spam words to SMS segments, the rate of capital words to unique words. The Linguistic Inquiry and Word Count (LIWC) is a text analysis tool that examines 80 various sorts of elements such as text functional aspects, psychological issues such as emotion and perception, and personal interests such as money, religion, and so on [44]. They used 20 LIWC characteristics, such as the ratio of verb scores to total word scores, the difference between the scores of "Money" and "Death", the frequency of punctuations, pronouns, exclamation marks, and so on. The author gathered data from the Grumbletext website[1]. There are 5,574 identified short messages in the sample, 747 of which are SMS spam and 4,827 of which are not. They used various classifiers and discovered that accuracy ranges from 92% – 98%.

The popularity of Bayesian approaches is growing, as is their usage in text and spam categorization applications. It has advantages in a cost-sensitive evaluation since it can give a better level of confidence in the categorization. A Naïve Bayes classifier and a bag of words representation for the email dataset were employed in Sahami's research [48]. In this study, the inclusion of complicated features (such as FREE! f r 3 3), domain name features, and other non-

[1] https://archive.ics.uci.edu/ml/datasets/sms+spam+collection

alphabetic characteristics to the bag of features resulted in increased performance. An ensemble of classifiers approach is also within the scope of the research.

Sakkis [49] has devised a method for constructing a superior ensemble of classifiers that combines many approaches. To get good classification results, this architecture comprises of naïve Bayes and KNN classifiers.

Trivedi and Dey [50] employed an ensemble of classifier approaches with probabilistic classifiers and discovered that this strategy improved the performance of probabilistic classifiers even when just a small sample of relevant features were included. In this field, the Support Vector Machine (SVM) is also widely used. SVM and boosted decision tree are the classifiers that are promising in terms of speed and accuracy, according to Drucker, who conducted a comparison study of SVM with various machine learning classifiers. However, the training speed of SVM was faster than for the Boosted decision tree [51].

Trivedi and Dey investigated the impact of several kernel functions on SVM performance and discovered that Normalized Polynomial Kernel [52] was the most effective technique to improve SVM learning capabilities.

D. Puniskis [53] used a neural network technique to classify spam in his study. Rather than using the context or frequency of terms in the message, his technique uses attributes made of descriptive aspects of the evasive patterns used by spammers. The data utilized was a corpus of 2788 valid and 1812 spam emails that were received over several months. The results demonstrate that ANN is good, but it is not acceptable for use as a spam filtering tool on its own.

Email data were categorized using four distinct classifiers (Neural Network, SVM classifier, Nave Bayesian Classifier, and J48 classifier) in another study. The experiment was carried out with varying data sizes and feature sizes. If it is eventually spam, the final

categorization result should be '1'; otherwise, it should be '0'. This work demonstrates that a basic J48 classifier that creates a binary tree may be effective for datasets that can be categorized as binary trees [54].

For SMS, Goswami employed a machine learning method. Naive Bayes, Support Vector Machine and Random Forest were used to filter and classify spam. There are 5574 measures of two variables in the dataset used in this study. However, because the algorithms' performances are not specified, there are contradictions in the findings [55].

Various supervised machine-learning methods, such as the Naive Bayes algorithm, support vector machine, and maximum entropy algorithm, were used to detect spam and ham communications, according to Pavas. They compared the results of their Ham and Spam message filtering [56].

Atanu and Kumar [57] compared the outcomes of the Logistic Regression, Naïve Bayes, Decision Tree, and Gradient Boosting Machine algorithms for model evaluation. In addition, the authors employed ApacheTM Spark as a platform for evaluating the algorithms' efficiency. Although several research publications use the same dataset, the context and classifiers are completely different. Precision and time efficiency were taken into account while analyzing performance. The findings demonstrated that GBTs took longer to categorize spam messages, but NB performed better in terms of accuracy and runtime.

Aditya compared the performance of eight different classifiers. According to the findings of the classifier evaluation, the Convolutional Neural Network (CNN) classifier obtains maximum precision of 99.19% and Average Recall values of 0.9926 and 0.9994 for the two datasets [58].

Choudhary and Jain [59] suggested a 10-feature SMS spam filtering method based on five machine-learning algorithms, including J48, Decisions Trees, Naive Bayes, Random Forest, and

Logistic Regression. With a true positive rate of 96.1 %, the Random Forest classification algorithm delivers the best results.

El-Alfy and Al-Hasan [60] published a classification model for both email and SMS text messages. They compared many ways before settling on a set of features. For the classification, they employed two algorithms: Nave Bayes and Support Vector Machine (SVM). Emotional visuals, unusual characters, JavaScript code, gappy expressions, recipient address, URLs, plausible spam terms, message metadata, work terms, point zone, and spam space are among the eleven aspects analysed. They put their theory to the test by demonstrating it on five databases for email and SMS.

"Messages Topic Model (MTM)" was the title of Jialin's research [61]. Based on the passive semantic analysis chance premise, they examined image words, context words, and topic terms to speak to spam communications. They used the k-means algorithm to eliminate the scarcity problem by training SMS spam messages into arbitrarily irregular groups. They then used catchword co-occurrence techniques to combine all SMS spam messages into a single record.

Kim [62] proposed an SMS filtering method that is both light and quick and can be implemented separately within mobile phones. The approach includes techniques for removing unnecessary data. These strategies include data filtering, highlight selection, and data grouping, among others. They were able to choose essential characteristics using a relative volume of function values.

Using Bayesian and SVM techniques, Shahi and Yadav [63] proposed spam filtering of Smartphone SMS for Nepali text. The main purpose of the study was to evaluate the performance of spam filters based on Naive Bayes and SVM. The accuracy, precision, and recall of the two spam filters were compared.

Shirani-Mehr [64] used a variety of machine learning methods to solve the challenge of detecting SMS spam. They compared their output to get the knowledge that helped them figure out where the discrepancies were. They developed an application based on one of those algorithms that could channel high precision of SMS spams.

The Naive Bayesian theorem is used to propose an algorithm. Emails were classified as spam or ham by the system. The emails were classified based on the content of the email body [65]. The characteristics and limits of several classification algorithms used in email classification, such as SVM, K-means clustering, vector space model, and others, are examined. The findings revealed that in most situations, unsupervised learning is neglected, and data mining approaches are beneficial [66]. The genetic algorithm and the k-nearest neighbor algorithm are used to provide a method for grouping spam communications. The clustering algorithm was shown to be effective in classifying emails as spam or legal [67].

Various spam detection methods are explored. Techniques for spam detection range from non-machine learning to machine learning. Different methodologies for categorizing emails are investigated based on false positive and false negative rates. The findings revealed that no strategy is 100 percent efficient [68]. A text clustering approach based on a vector space model is presented for spam identification. The vector space model is used to represent the data. A k-means and BIRCH-based algorithm is provided. For smaller datasets, K-means proved to be the most effective [69]. The definition of spam was explored and the numerous issues that spam may create. Trends and strategies for spam filtering are examined. On the receiver side, the approaches outlined are used [70].

A method for detecting spam using characters-words is suggested [71]. Multi-neural networks were utilized as the approach's classifier. The weighting factor for characteristics is the ASCII value of the word characters. It is possible to produce high false-positive and low true

negative rates. The method is used to determine which words are good or terrible. Before being sent to the classifier, messages are pre-processed.

For email categorization, a data reduction strategy has been developed. Before classifying, the Instance Selection Method is used to remove unnecessary data from the dataset. The findings revealed high classification accuracy and a low number of false positives. For data reduction from the training model, cluster classifiers were utilized [72]. A pattern discovery methodology is utilized to classify emails in a novel way. In terms of detecting spam communications, the Naive Bayesian algorithm in conjunction with pattern discovery was found to be the most effective. When compared to other approaches, such as data mining, the proposed strategy produced the best results. Text mining was employed in the suggested method [73]. In spam detection, both classical and learning-based algorithms are investigated, and a summary of current spam filtering approaches is provided. Learning-based algorithms proved to exceed traditional ones because of several qualities [74].

The WEKA tool is used to investigate different classification algorithms in the categorization of emails as spam or ham. The experiment makes use of the Spambase dataset from the UCI repository. Naive Bayes, Logistic Regression, PART, and J48 are the classifiers employed. The findings indicated that the Logistic Regression classification method works best [75]. Various issues with spam and spam filtering technologies are investigated. Traditional and learning-based strategies are being investigated. The findings revealed that no approach can guarantee a 0% false positive or false negative rate [76].

The Enron Email Corpus is a new testbed for spam filtering and text learning research. The applicability of Enron Corpus in terms of email folder prediction was explained. Another email dataset was used to analyse the Enron Corpus. The experiment revealed that the Enron corpus is a huge standard test dataset with a lot of potentials [77]. Spam filtering employs a data categorization

algorithm. Classification and semi-supervised learning are the most often utilized methodologies in content-based detection. Spam filtering is a type of content-based detection system. Content-based strategies outperformed rule-based techniques in the studies [78].

The WEKA tool and the k-mean clustering method are used to suggest an email mining technique. The data were converted from .eml format to .csv format using a text converter. The approach focuses on filtering the email addresses that create the most emails. Users can regain control of their emails using the implemented technique [79]. Traditional spam filtering methods like blacklisting/whitelisting, keyword matching, and content-based detection be ineffective. Spammers have refined their strategies, which has resulted in this. The use of supervised machine learning methods to filter spam is proposed. C4.5 Decision Tree, Naive Bayes Classifier, and multilayer perceptron were employed in this study. The project makes use of a personal mail dataset. Based on training time, properly categorized cases, prediction accuracy, and false positives, the classifiers were compared. The multilayer perceptron has been discovered to outperform other classifiers [80].

An approach based on Natural Language Processing has been developed to improve internet security. It's a step-by-step process that examines both the sender and the content of the email. It employed spam filtering techniques such as blacklisting and keyword matching. The approach provided a threshold counter, which aids in web server congestion reduction and spam filtering effectiveness [81].

In the field of spam filtering, several techniques have been presented. Machine learning methods have been shown to outperform other traditional methodologies in research. However, no approach guarantees a 0% false-positive or false-negative rate. As spammers enhance their spamming strategies, existing spam filtering solutions must improve as well. Several academics have looked at the efficacy of different machine learning algorithms in spam email screening

methods. Different classifiers were assessed in the works [82], [65], [75], and [80] to appropriately categorize spam emails. In publication [77], the use of the Enron corpus in spam filtering research was explored. However, the minimal feature size necessary for efficient filtering and the necessity of pre-processing is not addressed in the study. Machine learning methods were used with

substantial pre-processing techniques and text classification principles in this study. The Enron Corpus, the world's biggest publicly available email collection, was used in the experiment. In this experiment, the TF-IDF value is highlighted. The feature size for various classifiers that will offer the best, classification accuracy is also investigated.

Chapter 3

Methodology & Datasets

This chapter is divided into three sections which are the following.

1. Feature Selection
2. Algorithms
3. Evaluation Metrics

3.1 Preprocessing

After cleaning the text data, features are created using Feature Engineering techniques. In-text mining, common feature selection methods involve Bag of Words or BOW and Term Frequency-Inverse Document Frequency or TF-IDF.

3.1.1 Bag of Words (BoW)

The Bag of Words (BoW) model is the most basic type of numerical text representation. In binary representations, a sentence can be represented as a bag of words vector (a string of integers), similar to the phrase itself.

Text 1: This movie is very scary and long.

Text 2: This movie is not scary and is slow.

Text 3: This movie is spooky and good.

To begin, a vocabulary is created using all the unique terms found in the three texts above. 'This', 'movie', 'is','very','scary', 'and', 'long', 'not','slow','spooky', 'good' are the 11 words in the

vocabulary. Each of these terms is used to form a matrix with n rows of n texts and m columns of several unique words, with 1s and 0s indicating their occurrence as shown in Table 3.1.

Table 3.1. Bag of Words approach

	1 This	2 movie	3 is	4 very	5 scary	6 and	7 long	8 not	9 slow	10 spooky	11 good	Length of the review(in words)
Review 1	1	1	1	1	1	1	1	0	0	0	0	7
Review 2	1	1	2	0	0	1	1	0	1	0	0	8
Review 3	1	1	1	0	0	0	1	0	0	1	1	6

Vector of Review 1: [1 1 1 1 1 1 1 0 0 0 0]

Vector of Review 2: [1 1 2 0 0 1 1 0 1 0 0]

Vector of Review 3: [1 1 1 0 0 0 1 0 0 1 1]

The following are some of the shortcomings of the BoW model.

- If the new sentences contain new words, our vocabulary will expand, and the length of the vectors will expand as well.
- Furthermore, the vectors would contain a large number of 0s, resulting in a sparse matrix which can cause overfitting of the model.
- We don't save any information about the syntax of the sentences or the order in which the words appear in the text.

3.1.2 Term Frequency-Inverse Document Frequency (TF-IDF)

Term frequency-inverse document frequency is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

- **Term Frequency (TF)**

It is a measure of how frequently a term, t, appears in a document, d:

$$tf_{t,d} = \frac{n_{t,d}}{Number\ of\ terms\ in\ the\ document} \qquad (1)$$

Here, in the numerator, n is the number of times the term “t” appears in the document “d”. Thus,

each document and term would have its TF value.

For *text 2*

Number of words in Text 2 = 8

TF for the word ‘this’ = (number of times ‘this’ appears in review 2)/ (number of terms in review 2) = 1/8

Similarly,

- TERM FREQUENCY(‘movie’) = 1/8
- TERM FREQUENCY(‘is’) = 2/8 = 1/4
- TERM FREQUENCY(‘very’) = 0/8 = 0
- TERM FREQUENCY(‘scary’) = 1/8
- TERM FREQUENCY(‘and’) = 1/8
- TERM FREQUENCY(‘long’) = 0/8 = 0
- TERM FREQUENCY(‘not’) = 1/8
- TERM FREQUENCY(‘slow’) = 1/8
- TERM FREQUENCY(‘spooky’) = 0/8 = 0
- TERM FREQUENCY(‘good’) = 0/8 = 0

We can calculate the term frequencies for all the terms and all the reviews in this manner as shown in Table 3.2.

Table 3.2. Term Frequency calculation

Term	Review 1	Review 2	Review 3	TF (Review 1)	TF (Review 2)	TF (Review 3)
This	1	1	1	1/7	1/8	1/6
movie	1	1	1	1/7	1/8	1/6
is	1	2	1	1/7	1/4	1/6
very	1	0	0	1/7	0	0
scary	1	1	0	1/7	1/8	0
and	1	1	1	1/7	1/8	1/6
long	1	0	0	1/7	0	0
not	0	1	0	0	1/8	0
slow	0	1	0	0	1/8	0
spooky	0	0	1	0	0	1/6
good	0	0	1	0	0	1/6

- **Inverse Document Frequency (IDF)**

IDF is a measure of how important a term is. We need the IDF value because computing just the TF alone is not sufficient to understand the importance of words:

$$idf_t = log \frac{number\ of\ documents}{number\ of\ documents\ with\ term\ t} \quad (2)$$

Here we employ logarithm of base 2. The IDF values for all the words in Text 2 are calculated as:

INVERSE DOCUMENT FREQUENCY ('this') = log (number of documents/number of documents containing word 'this') = log (3/3) = log (1) = 0

- INVERSE DOCUMENT FREQUENCY ('movie',) = log (3/3) = 0
- INVERSE DOCUMENT FREQUENCY('is') = log (3/3) = 0
- INVERSE DOCUMENT FREQUENCY('not') = log (3/1) = log (3) = 0.48
- INVERSE DOCUMENT FREQUENCY('scary') = log (3/2) = 0.18
- INVERSE DOCUMENT FREQUENCY('and') = log (3/3) = 0
- INVERSE DOCUMENT FREQUENCY('slow') = log (3/1) = 0.48

We can calculate the IDF values for each word like this. Thus, the IDF values for the entire vocabulary is given in Table 3.3.

Table 3.3. Inverse Document Frequency calculations

Term	Review 1	Review 2	Review 3	IDF
This	1	1	1	0.00
movie	1	1	1	0.00
is	1	2	1	0.00
very	1	0	0	0.48
scary	1	1	0	0.18
and	1	1	1	0.00
long	1	0	0	0.48
not	0	1	0	0.48
slow	0	1	0	0.48
spooky	0	0	1	0.48
good	0	0	1	0.48

Hence, we see that words like "is", "this", "and", etc., are reduced to 0 and have little importance; while words like "scary", "long", "good", etc. are words with more importance and thus have a higher value.

We can now compute the TF-IDF score for each word in the corpus. Words with a higher score are more important, and those with a lower score are less important. The formula for TF-IDF is given in equation (3).

$$(tf\_idf)_{t,d} = tf_{t,d} \times idf_t \qquad (3)$$

We can now calculate the TF-IDF score for every word in Review 2.

TF-IDF ('this', Review 2) = TF ('this', Review 2) * IDF ('this') = 1/8 * 0 = 0

Similarly,

- **TF-IDF ('movie', Review 2) = 1/8 * 0 = 0**
- **TF-IDF ('is', Text 2) = 1/4 * 0 = 0**
- **TF-IDF ('not', Text 2) = 1/8 * 0.48 = 0.06**
- **TF-IDF ('scary', Text 2) = 1/8 * 0.18 = 0.023**
- **TF-IDF ('and', Text 2) = 1/8 * 0 = 0**
- **TF-IDF ('slow', Text 2) = 1/8 * 0.48 = 0.06**

Similarly, we can calculate the TF-IDF scores for all the words with respect to all the Texts as shown in Table 3.4.

Table 3.4. Term Frequency-Inverse Document Frequency calculations

Term	Review 1	Review 2	Review 3	IDF	TF-IDF (Review 1)	TF-IDF (Review 2)	TF-IDF (Review 3)
This	1	1	1	0.00	0.000	0.000	0.000
movie	1	1	1	0.00	0.000	0.000	0.000
is	1	2	1	0.00	0.000	0.000	0.000
very	1	0	0	0.48	0.068	0.000	0.000
scary	1	1	0	0.18	0.025	0.022	0.000
and	1	1	1	0.00	0.000	0.000	0.000
long	1	0	0	0.48	0.068	0.000	0.000
not	0	1	0	0.48	0.000	0.060	0.000
slow	0	1	0	0.48	0.000	0.060	0.000
spooky	0	0	1	0.48	0.000	0.000	0.080
good	0	0	1	0.48	0.000	0.000	0.080

The TF-IDF scores for our vocabulary have now been determined. TF-IDF also offers greater values for less frequent terms and is high when both IDF and TF values are high, indicating that the word is rare in all papers combined yet common in one document. The Bag of Words model simply generates a collection of vectors holding the count of word occurrences in the text (Texts), but the TF-IDF model includes information on both, the more significant and less important words. The Bag of Words vectors is simple to understand. TF-IDF, on the other hand, frequently outperforms BoWs in machine learning models.

We used both methods to create the text features and applied the machine learning models.

3.2 Feature Selection Algorithms

A feature selection, also known as a variable selection, is a method of selecting a subset of features from data. This method is often used in machine learning to address issues with high dimensionality. To simplify and summarize data representation, it selects a subset of significant characteristics and rejects redundant, irrelevant, and all noisy features [83]. Filter models, Wrappers models, and embedding techniques are examples of feature selection approaches. For feature selection, we employed evolutionary algorithms and greedy search approaches. Enron dataset has 21 features for each employee in the datasets and Spam Assassin does not have as many features as Enron dataset. While we are comparing both the datasets, we have applied feature selection. Feature selection improves the machine learning process and increases the predictive power of machine learning algorithms by selecting the most relevant features and eliminating redundant and irrelevant features. Also it reduces overfitting, eliminates noise and improves accuracy.

3.2.1 Evolutionary Algorithms

Evolutionary algorithms use a set of rules that apply evolutionary principles to create adaptable systems. They are probabilistic search methods and optimization heuristics that can find a solution to search and optimization tasks by simulating natural biological evolution; at each generation, the population evolves towards regions in the search space that is better and better by simulating "Darwinian" evolution (selection, recombination, and mutation); the best-known methods in the field of evolutionary algorithms are genetic algorithms. The process of Genetic Algorithms (GAs) is resilient and random [84, 88]. The following phases are included in the genetic algorithm [85, 89].

The initial population is produced randomly to offer the genetic algorithm a speed in terms

of both (the evolutionary and optimal start point processes). Even though the genetic algorithm uses a directionless search, it is utilized to choose the parent.

a. Fitness-based selection: this is an opportunity to pick each chromosome based on its fitness. The initial approach of parent selection included roulette wheel selection and fitness-based selection.

b. Rank-based selection: instead of absolute fitness, the selection is based on probabilistic and relative rank.

c. Tournament-based selection: the best of these parents is chosen at random and returned.

One of the most significant operations in GA is a crossover, which is the process of pluralizing bit strings by replacing segments between pairs of chromosomes. There are several kinds of crossover.

a. 1-Point Crossover: this method selects the bit position that has to be changed at random.

b. 2-point Crossover: selects two places and just changes the bit between them.

Mutation: is selecting any bit position randomly and changing it and has ensured all chromosomes can keep the best gene in the new chromosomes.

3.2.2 Greedy Feature selection

A greedy feature selection, on the other hand, is a basic feature selection technique in which an algorithm selects the best characteristics one by one (ahead selection) or removes the worst features one by one (backward selection).

3.3 Classification Techniques

Features from the above approaches are used to train classifiers i.e., Naive Bayes, Bayesian Network, Support Vector Machine (SVM), Genetic classifier, Random Forest, XGBoost, and Long Short-Term Memory (LSTM).

3.3.1 Naive Bayes

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

It is a Bayesian probabilistic classifier with the strong assumption that all characteristics are independent of one another. Class conditional independence is the name for this type of assumption. Naive Bayes has the benefit of requiring a little quantity of training data to do classification. Because it operates in a single scan, it is one of the fastest classifiers [90].

The Bayes theorem explains how to calculate the posterior probability P (c | x) from P (x | c), P(c), and P(x). The influence of a predictor x (just its value) on a given class c is not reliant on other predictors, according to the Naive Bayes classifier. The formula for determining posterior probability is as follows:

$$P(c|x) = \frac{P(x|c) \times P(c)}{P(x)} \qquad (4)$$

Where:

$P(c|x)$: the posterior probability of the target class on a given attribute.

$P(x|c)$: the probability of predictor on given class (like a hood).

$P(c)$: the prior probability of the class.

$P(x)$: the prior probability of the predictor.

3.3.2 Bayesian Network

The characteristics of the Bayesian network as a classifier are chosen using scoring functions such as the Bayesian scoring function. The scoring functions primarily limit the structure (connections and orientations) as well as the parameters (likelihood) that may be used with the data. The class is only decided by the nodes in the Markov blanket (its parents, children, and children's parents) after the structure has been learned, and all variables provided to the Markov blanket are deleted.

All characteristics are considered attributes and are independent given the class in the Naive Bayesian Network, which is more well-known presently.

Bayesian Network is more complicated than the Naive Bayes, but they almost perform equally well, and the reason is that all the datasets on which the Bayesian network performs worse than the Naive Bayes have more than 15 attributes. That's during the structure learning some crucial attributes are discarded. We can combine the two and add some connections between the features of the Naive Bayes and it becomes the tree augmented Naive Bayes or k-dependence Bayesian classifier.

3.3.3 Support Vector Machine (SVM)

A supervised learning algorithm that offers an alternative view of logistic regression, the simplest classification algorithm, is the support vector machines or SVMs. Support vector machines try to find a model that precisely divides the classes with the same amount of margin on either side, where the support vectors are called samples on the margin. Support Vector Machine (SVM) also known as Support Vector Network in machine learning is a supervised learning technique used for classification and regression. In simple terms, given training examples set, each of them marked belonging to one of any number of categories. SVM training algorithm constructs a model that decides and assigns a new example that falls into one category or the other. Hence SVM classifier is represented by a separating hyperplane. There are three kernels in SVM used widely which are

Linear, Radial basis function (RBF), and Polynomial.

3.3.3.1 Linear Kernel

The feature space corresponding with a linear kernel is the original feature space. In other words, a linear kernel is a direct translation of Euclidian distances, and there is non-dimensional space involved at all. Linear kernel support vector machines have good performance only on very simple problems. The more indecipherable the boundaries between class clusters become, the worse the performance of the SVM algorithm with the linear kernel becomes.

3.3.3.2 Radial Kernel

RBF is the most popular support vector machine kernel choice. RBF is a type of function that is used to approximate other functions which are quite complex, but basically, it translates into the kernel that can create the most complex boundaries. RBFs are also behind kernel smoothing, and through that, kernel density estimation (including Kernel Density Estimation (KDE) plots, which are popular in machine learning and Exploratory Data Analysis (EDA)). Computing an RBF kernel is very expensive, and quickly becomes computationally intractable as the number of samples exceeds the tens of thousands.

3.3.3.3 Polynomial Kernel

The feature space corresponding with a polynomial kernel is the same as it is in polynomial regression. Each feature x is copied over to x^2, x^3, and so on (up to a degree, so only to 3 in this case), and then separating hyperplanes are drawn in that feature space. Drawing them in two dimensions demonstrates their polynomial nature. The degree parameter is unique to the polynomial kernel Support vector machine. The higher the degree the more over-fitted the resulting space. The spaces generated by the separating hyperplane remain continuous and boundless.

3.3.4 Random Forest

When Decision Trees aren't effective, the Random Forest classifier is used. Random Forests are a bagging-based Decision Tree approach that is also regarded as one of the ensemble approaches. In other words, if trees are grown very deep or have an irregular form, i.e., an over-fit training set, random forests act on different parts of the same training set by creating a large number of decision trees throughout training time to average numerous deep decision trees. As a result, Random Forest transforms weak learners into strong learners. The main assumption behind the random forest method is that the majority of the trees can correctly forecast class for the majority of the data.

The Random Forest is a meta-learner which consists of many individual trees. Each tree votes on an overall classification for the given set of data and the random forest algorithm chooses the individual classification with the most votes. Each decision tree is built from a random subset of the training dataset, using what is called replacement, in performing this sampling. That is, some entities will be included more than once in the sample, and others would not appear at all. In building each decision tree, a model based on a different random subset of the training dataset and a random subset of the available variables is used to choose how best to partition the dataset at each node. Each decision tree is built to its maximum size, with no pruning performed. Together, the resulting decision tree models of the Random Forest represent the final ensemble model where each decision tree votes for the result and the majority wins.

3.3.5 Genetic Classifier

The suggested technique for classifying email spam filtering is applied to a training database to produce the best model, and the model's performance is evaluated using a testing dataset.

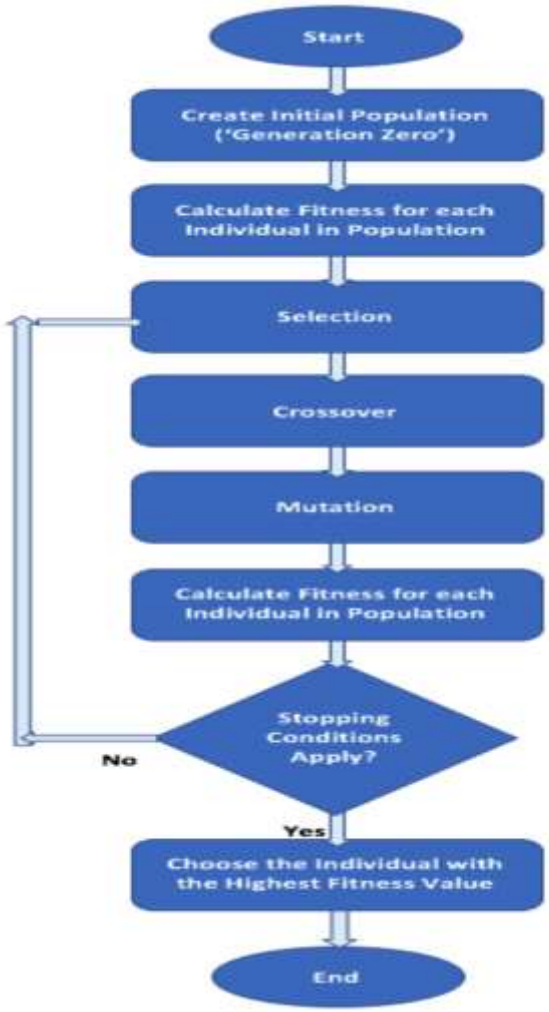

Figure 3.1. Genetic Algorithm Classifier [93]

Each solution gene is encoded using binary encoding, with values greater than 0 as 1 and otherwise 0. Consider a population P of N, which is represented by a length vector of size N. It's possible to put it this way: P = P1, P2…, PN, where N is the population size. The population begins with a random population P_0 and continues until it achieves the stated maximum number of iterations. The genetic Algorithm will apply three key operators at each iteration: selection, crossover, and mutation. After applying the operator, the fitness function suggested to evaluate the quality of genetic algorithm solutions, the single objective evaluation algorithm (SOEA) model is

proposed to solve email spam filter problem difficulties, it expressed by maximizing "accuracy" as in the following equation:

$$Maximizing\ SOEA\ (P) or\ Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \qquad (5)$$

3.3.6 Long Short-Term Memory (LSTM)

Long Short-Term Memory, or LSTM, is a kind of recurrent neural network (RNN) that has been recognized for its ability to process short-term time series data. Furthermore, when compared to other deep neural networks (DNN), LSTM has demonstrated greater temporal series learning capacity. Long Short-Term Memory is a system that uses a cell state and a carry state to maintain information in the form of a gradient that is guaranteed not to be lost when the input sequence is stored deeper in the architecture. The current state, the carry state, and the cell state are all taken into account at each time step in the LSTM. The architecture of the LSTM network is depicted in the Figure 3.2.

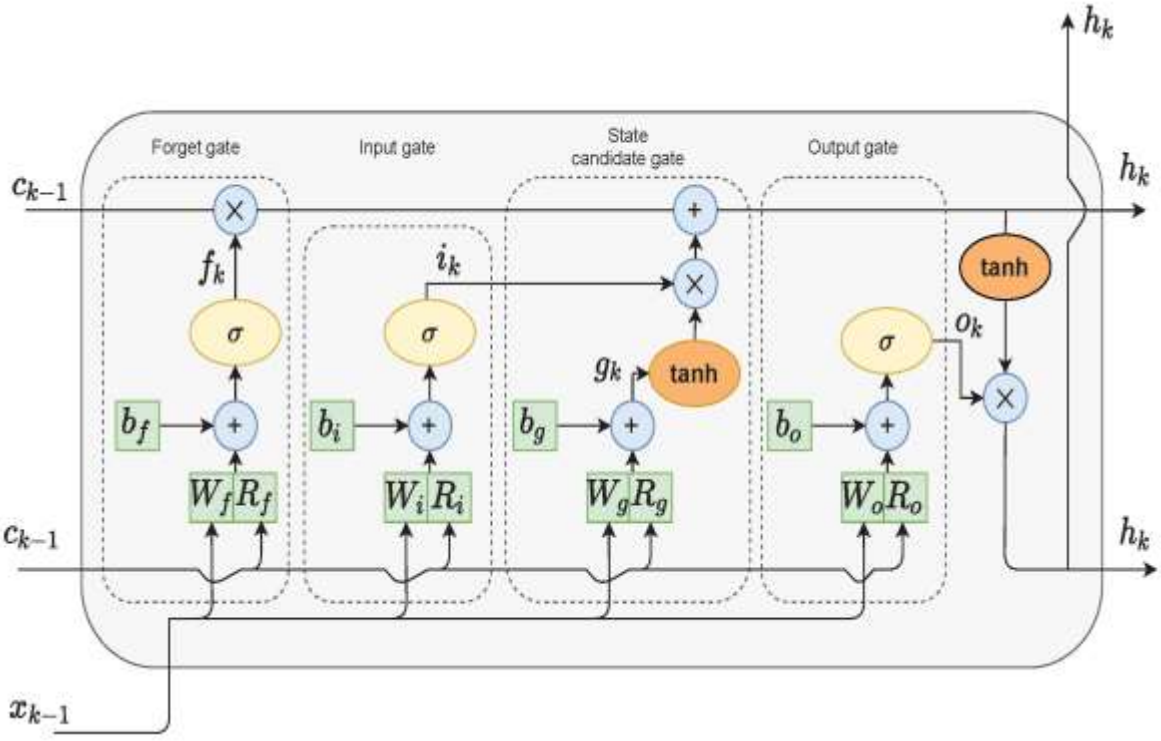

Figure 3.2. LSTM Model Architecture [91]

As previously mentioned, the memory contains three common gates and weight vectors. The forget gate is in charge of forgetting non-essential data, the input gate of working with the current input, and the output gate of providing forecasts for each time point. It's worth mentioning that, as in the Back-Propagation Neural Network training phase, the weights of the network layers, which are started and altered during training, dictate the role of each cell element. Recurrent neural networks RNNs have a long-term dependence problem that LSTM networks were created to solve which is known as vanishing gradient problem. LSTMs vary from standard feedforward neural networks. This property allows LSTMs to process complete data sequences (e.g., time series) without having to handle each point in the sequence separately, instead preserving important knowledge about prior data in the sequence to aid in the processing of incoming data points. As a result, LSTMs specialize in processing data sequences such as text, audio, and time series in general.

3.4 Evaluation Metrics

The evaluation metrics given in Table 3.5 were used in the evaluation and comparison of the models.

Table 3.5 Evaluation Metrics used in the Research

Evaluation Metrics	Description
Confusion matrix	The confusion matrix is one of the most fundamental representations in classification since it describes all performance and can be used to evaluate all metrics. In a matrix, it is the representation of correctly and erroneously classified instances in a test set.
Accuracy score	Percentage of observations that were correctly classified.

Sensitivity Score	Sensitivity is known as the True Positive Rate, also known as **Recall** which is the proportion of actual positive cases which are predicted as positive by our model. **Sensitivity = True Positive / (True Positive + False negative)**
Specificity Score	Specificity is also known as the True Negative Rate which is the proportion of actual negative cases which are predicted as negative by our model. **Specificity = True Negative / (True Negative + False Positive)**
Area Under Curve (AUC) Score	At different threshold conditions, the AUC score is an output measurement for classification problems. The degree or metric of separability is represented by the AUC. It indicates how well the model can differentiate between classes. The higher the AUC, the more accurate the model.

3.5 Datasets

3.5.1 Enron Dataset

After one of the greatest financial scandals in corporate history, Enron, an American energy business, filed for bankruptcy in late 2001. During the Federal Energy Regulatory Commission's inquiry into Enron's demise, the Federal Energy Regulatory Commission collected approximately 600,000 emails generated by 158 Enron workers, now known as the Enron Corpus. FERC made the original Enron dataset public and uploaded it on the web during its investigation into the Western Energy Crisis in 2000-2001 [2]. Leslie Kaelbling of the Massachusetts Institute of Technology later acquired the dataset and discovered various issues with its integrity [7]. Shortly after, a team of researchers led by Melinda Gervasio at SRI International, a non-profit organization

formed by Stanford University, cleaned it up and removed attachments before sending it to Professor William W. Cohen at Carnegie Mellon University, who placed it on his webpage [7].

The data was subsequently posted on the internet, where it has been gratefully prepared, cleaned, and sorted by several people and organizations (a few years later, financial data of top Enron executives were released following their trial). The Enron Corpus is now the world's largest and one of the few publicly available bulk collections of genuine emails that can be studied. The Enron corpus was suited for Spam Classification evaluation of e-mail classification algorithms, according to research evaluating the Enron database presented at a 2004 Conference. The Enron corpus is made up of e-mails from Enron workers, especially top executives, and traders.

Each of the 151 workers' folder information is included in the dataset. Each message in the folders includes the sender's and receiver's email addresses, as well as the date and time, topic, body, text, and other email-specific technical information. We identified 33,715 emails in the dataset after combining all five versions of the Enron Corpus provided into a single data frame; 17,110 spam emails and 16,545 ham emails were found.

3.5.2 Spam Assassin Dataset

The Spam Assassin dataset was obtained from the public corpus website of Spam Assassin. It is made up of two data sets: training and testing. Each dataset comprises a series of emails in plain text format that has been classified as ham or spam at random. The total number of emails in the sample is 3435. The data is ham 78% of the time, and spam 22% of the time. There are 2680 spam emails and 755 ham emails in the database.

The emails are in plain text format in the data. As a result, we need them to turn plain text into characteristics that might represent emails. We may then perform a machine learning algorithm on the emails using these attributes. First, a variety of pre-processing processes are

carried out such lower-casing all the character, removal the numbers, removal of special characters, punctuations etc., stemming, and lemmatizing text.

Chapter 4

Results and Analysis

The outcomes of the analysis will be discussed in this chapter.

4.1 Data Cleaning

We defined a custom function to read and store data from multiple folders for both datasets. To do so, we created an empty list to store the text from files and used a for loop to read and extract all the emails which were separately provided in different files using the filenames of each file in a folder and stored them in an empty list until all folders and files are read. A file consists of an email, which has four parts, tag, lines, heading and body. We extract the body of each file and store them in the list we created above and later create a data frame of both the datasets with one column representing the main body of the email and the second representing the class of the email i.e., spam or ham.

4.2 Exploratory Data Analysis (EDA)

After reading, loading, and preparing the datasets, we conducted some EDA to see some information which is contained in the email corpus for each dataset. Since there are not many options for EDA when it comes to text data, we used word clouds to represent text in each dataset.

4.2.1 Enron Dataset

Word Cloud for Genuine emails

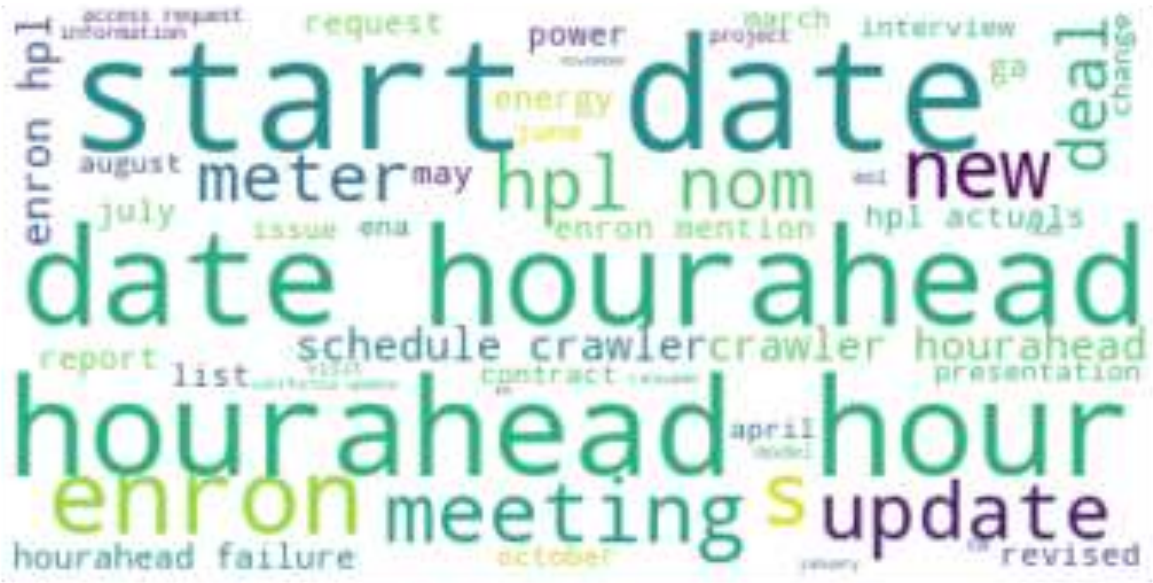

Figure 4.1 Word cloud for genuine emails in Enron Dataset

Word Cloud for Spam emails

Figure 4.2 Word cloud for spam emails in Enron dataset

4.2.2 Spam Assassin

Word Cloud for Genuine emails

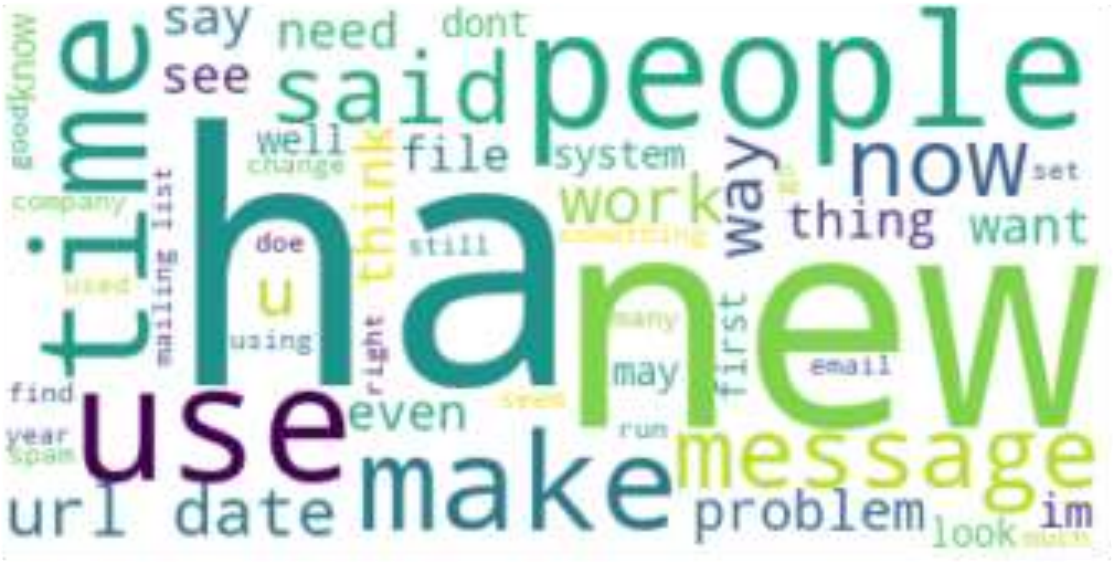

Figure 4.3 Word cloud for genuine emails in Spam Assassin Dataset

Word Cloud for Spam emails

Figure 4.4 Word cloud for spam emails in Spam Assassin Dataset

We can see that emails that are spam are filled with words such as free, receive, money, need, congratulations, want, help, etc. which are typically seen in spam or marketing emails that we receive.

4.3 Machine Learning Algorithms

We tested and evaluated many models using a variety of Machine Learning methods to discover the optimum algorithm for detecting spam in both datasets (Enron and Spam Assassin). The machine learning models were trained and tested using a variety of assumptions. To detect spam in email datasets, we applied seven machine learning methods shown in Table 4.1 below.

Table 4.1. Machine Learning algorithms to detect spam

Algorithm	Full Name
NB	Naïve Bayes
Bayesian	Bayesian Classifier
SVM	Support Vector Machine with Polynomial Kernel
RF	Random Forest
XGB	Extreme Boosting
GA	Genetic Classifier
LSTM	Long-Short Term Memory (ANN)

We also applied two feature selection strategies, Genetic based Feature Selection and Greedy Search Feature Selection, to choose the best features and avoid the model from overfitting with non-significant features while decreasing computing costs. We look through the findings of each of the feature selection techniques individually as well as all of them together to find the optimal feature selection approach and machine learning algorithm for our data to predict spam

with the highest possible accuracy and AUC. All the feature selection approaches utilized in this study are listed in Table 4.2.

Table 4.2. Feature Selection Techniques implemented in the research

Method	Feature Selection Technique Name
Genetic	Genetic Algorithm based feature selection
Greedy	Greedy Search based Feature Selection

We trained the models using training-testing split (80%-20%) to build the model on the training set and evaluate the model using the test set. The datasets were loaded and processed. Figure 4.5 and 4.6 show the statistics and graphs with the distribution of spam/ham in Enron and Spam-Assasin datasets.

Enron Dataset

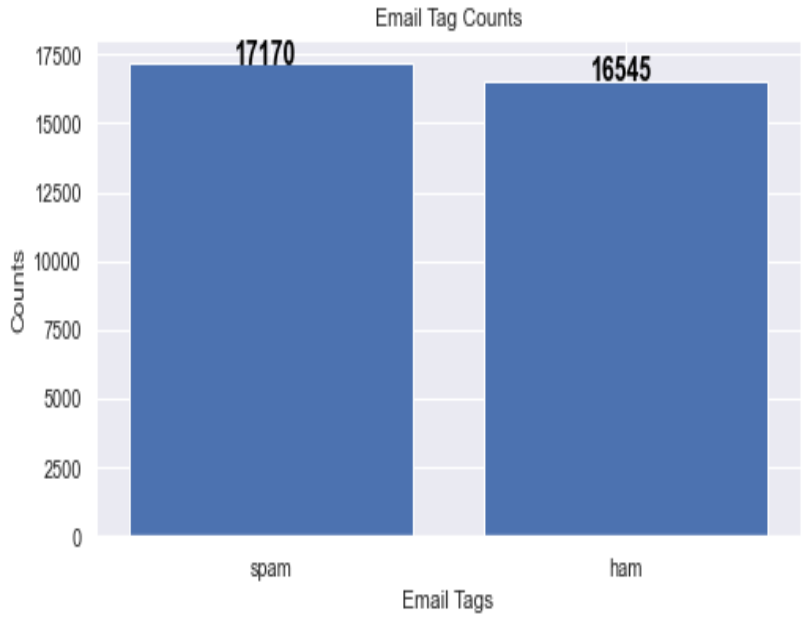

Figure 4.5 Statistics of spam and ham emails in Enron dataset

Spam Assassin Dataset

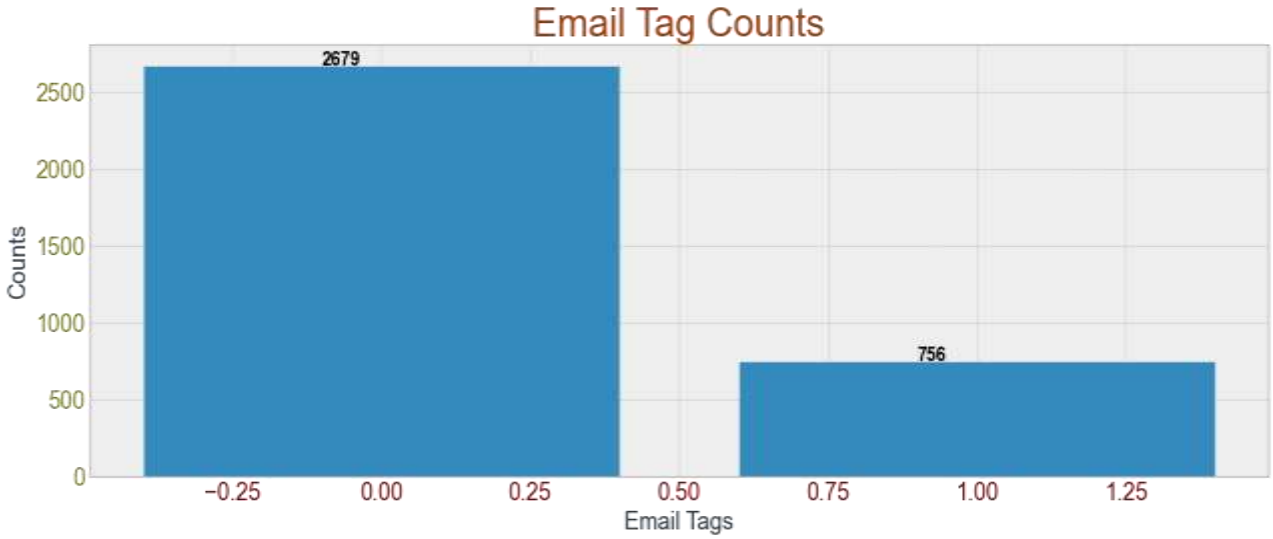

Figure 4.6 Statistics of spam and ham emails in Spam-Assassin dataset

We can see that the Enron dataset is quite balanced where 17170 are spam emails and 16545 are ham emails. On the other hand, Spam Assassin Data is unbalanced to a great extent where 2679 are spam emails and 756 are Ham emails. The distribution of email labels tells us more about the quality and nature of the dataset. We can assume that most of the models will perform better on Enron Dataset as it is balanced as compared to the Spam Assassin dataset.

Confusion Matrix, Sensitivity, Specificity, and Area under Receiver Operating Characteristics (ROC) Curve (AUC) metrics are used to evaluate the performance of the models as shown in Table 3.1. Sensitivity and Specificity, in addition to accuracy, must have high values, indicating a strong model.

Firstly, we compared all the models without the feature selection methods using 80-20 train-test split evaluation. Given below are the evaluations of the model, based on the accuracy score for each of the feature selection methods.

4.4 Comparison of Results

We used three strategies on each dataset and computed evaluation metrics on each of the datasets, which is as follows.

1. No Feature Selection
2. Genetic Algorithm based Feature Selection
3. Greedy Search based Feature Selection

4.4.1 No Feature Selection

4.4.1.1 Enron Dataset

Table 4.3 shows the performance of the various machine learning methods on the Enron dataset without feature selection. It can be seen that SVM and Random Forest have performed the best with the train accuracy of 99.5% and 99.7% and with the test accuracy of 98.2% and 97.5% respectively, which is far better accuracy of SVM and Random Forest, 79.38% and 83.2% given in [94].In LSTM we also converted dataset into

Table 4.3. Performance of classifiers on Enron dataset without using feature selection

Model	Train Accuracy	Test Accuracy	Precision	TPR or Recall	F1 Score	Cohen Kappa Score	FPR or Fall Out
Bayesian	0.970673	0.969301	0.969697	0.969016	0.969261	0.938527	0.030984
Naive Bayes	0.928556	0.927332	0.935927	0.925555	0.926718	0.854058	0.074445
SVM	0.994698	0.982204	0.982513	0.981978	0.982182	0.964367	0.018022

Genetic	0.909425	0.903900	0.920621	0.901319	0.902469	0.806693	0.098681
Random Forest	0.996886	0.974789	0.975053	0.974574	0.974760	0.949522	0.025426
XGboost	0.984651	0.973009	0.974243	0.972454	0.972950	0.945922	0.027546

From Table 4.3, Figure 4.7 and 4.8, we compared different classifiers on Enron Dataset with no Feature Selection. It can be observed that Genetic Classifier and Naïve Bayes Classifier have over fitted because of lower train/test accuracy, precision, recall and F1 score. The Cohen Kappa score of these classifiers is also typically very poor especially for Genetic Classifier as compared to the other models. If we find out the best model with no Feature Selection Method, Bayesian Classifier, XGboost, Random Forest and SVM have performed better based on the evaluation measures. SVM has the lowest Fall Out/False Positive Rate and highest Precision along with other metrics, which makes it very suitable for real-time implementation. We can conclude that with no Feature Selection Method, SVM is the best model, because SVM algorithm is very potent for the identification of patterns and classifying them into a specific class or group.

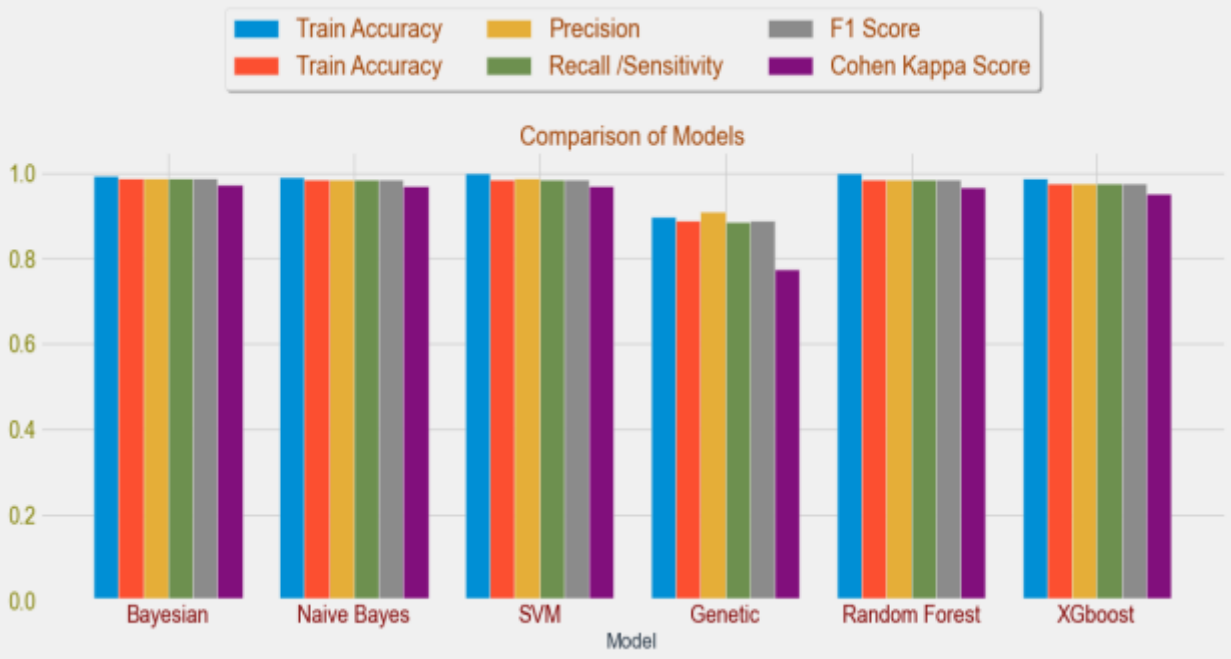

Figure 4.7. Evaluation metrics of each classifier on Enron dataset without feature selection

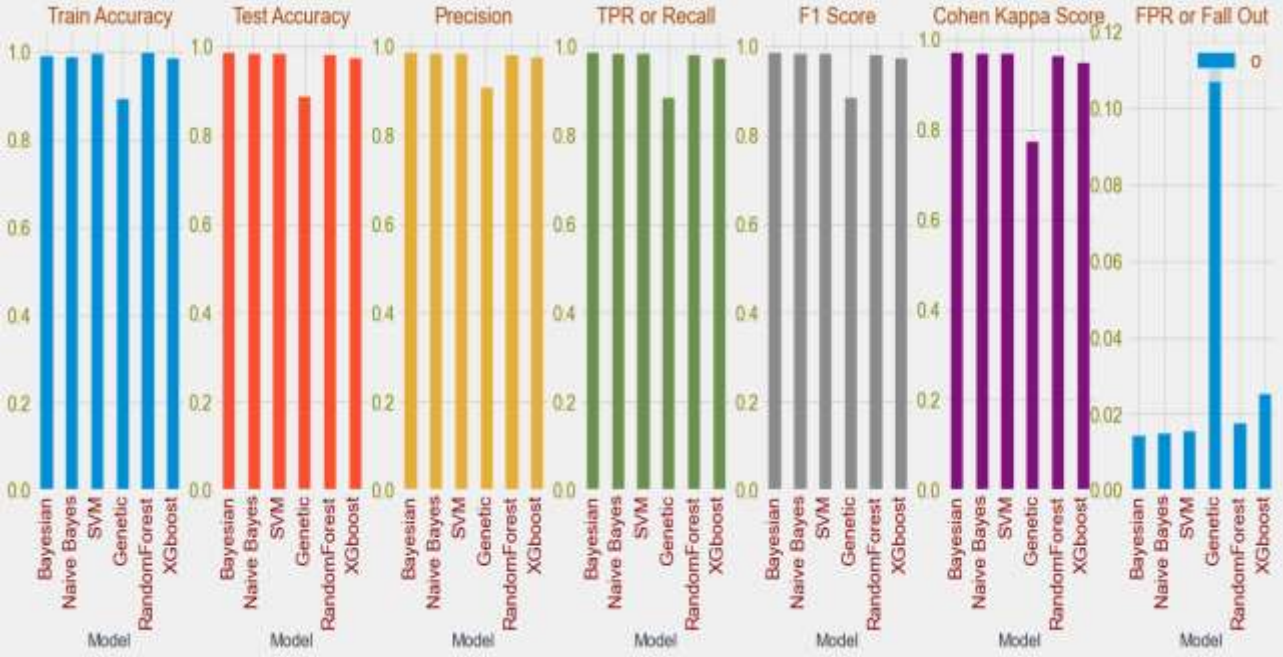

Figure 4.8. Comparison of classifiers on each evaluation metric on the Enron dataset without feature selection

4.4.1.2 Spam Assassin Dataset

Table 4.4 shows the performance of the various machine learning methods on the Spam Assassin dataset without feature selection. It can be seen that SVM, Random Forest and XGboost have performed the best with the highest train/test accuracy, precision, recall, and F1-Score.

Table 4.4. Performance of classifiers on Spam Assassin dataset without using feature selection

Model	Train Accuracy	Test Accuracy	Precision	TPR or Recall	F1 Score	Cohen Kappa Score	FPR or Fall Out
Bayesian	0.893013	0.855895	0.921429	0.682692	0.724970	0.470909	0.317308
Naive Bayes	0.866084	0.822416	0.860908	0.618028	0.640048	0.318153	0.381972
SVM	0.999272	0.965066	0.975473	0.925340	0.947551	0.895262	0.074660
Genetic	0.853348	0.820961	0.905963	0.605769	0.622704	0.293157	0.394231
Random Forest	0.999272	0.962154	0.970837	0.921194	0.943180	0.886534	0.078806
XGboost	0.993086	0.969432	0.967998	0.944010	0.955331	0.910695	0.055990

From Table 4.4, Figure 4.9 and Figure 4.10, we compared different classifiers on Spam Assassin Dataset with no Feature Selection and it can be seen that Genetic Classifier and Naïve Bayes Classifier have over fitted because of lower train/test accuracy, precision, Recall and F1 score. Overfitting occurs when genetic and Naïve Bayes classifier tries to cover all the data points or more than the required data points present in the given dataset. Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the genetic and Naïve Bayes models.

The Cohen Kappa score of these classifiers is also typically very poor, which is 0.293157, especially for Genetic Classifier as compared to the other models, which means classifier works moderately. If we find out the best model with no Feature Selection Method, Bayesian Classifier, Random Forest and XGBoost have performed better based on the evaluation measures. SVM has the lowest Fall Out/False Positive Rate and highest precision along with other metrics, which makes it very suitable for real time implementation. We can conclude that with no Feature Selection Method, XGBoost is the best model with the accuracy and precision of 99.30% and 96.79%, respectively, which is considerably better to be given in [95], which are 95% and 96.47% accuracy and precision correspondingly.

Figure 4.9. Evaluation metrics of each classifier on Spam Assassin dataset without feature selection

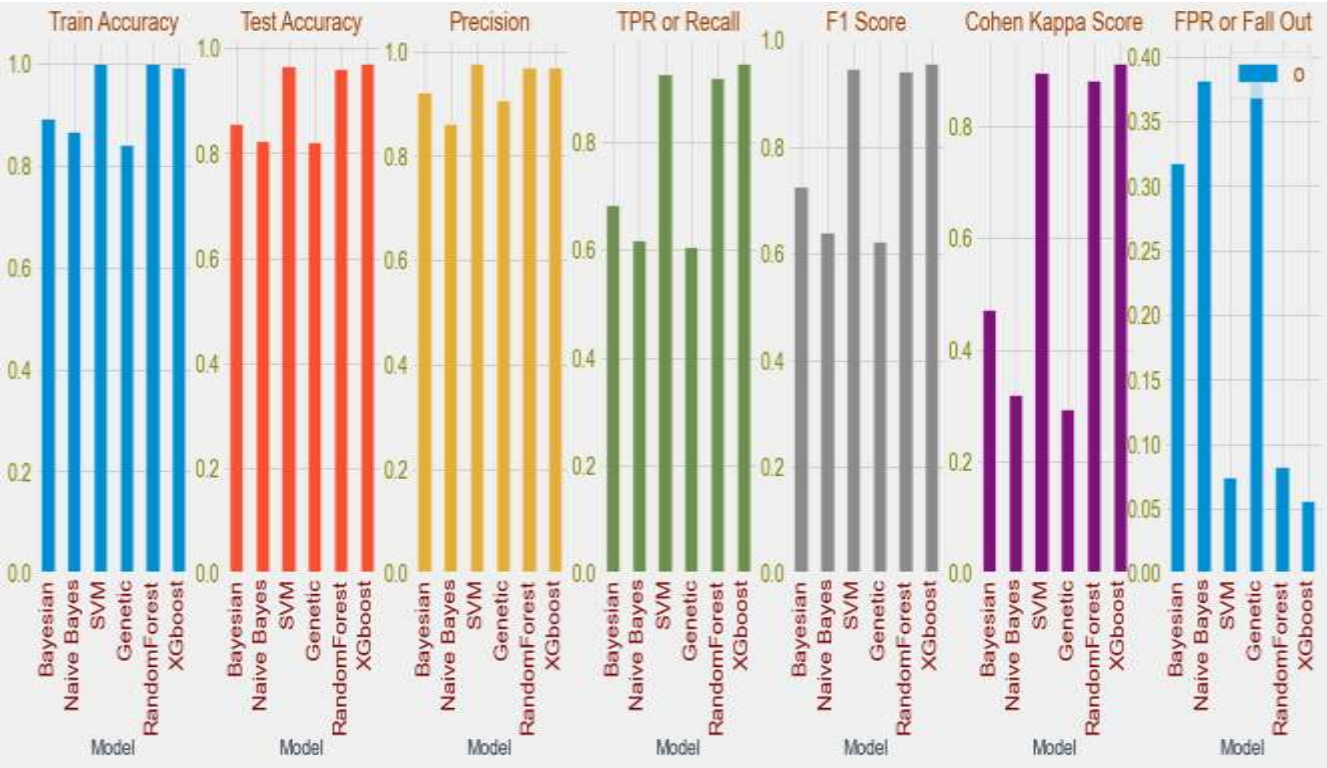

Figure 4.10. Evaluation metrics of each classifier on Spam Assassin dataset without feature selection

4.4.2 Genetic based Feature Selection

4.4.2.1 Enron Dataset

Table 4.5 shows the performance of the machine learning methods for Enron dataset using genetic based feature selection. Based on Genetic Feature Selection method, it can be seen that Bayesian, SVM and Random Forest have performed the best, based on various evaluation metrics.

Table 4.5. Performance of various machine learning models for Enron dataset with genetic based feature selection.

Model	Train Accuracy	Test Accuracy	Precision	TPR or Recall	F1 Score	Cohen Kappa Score	FPR or Fall Out
Bayesian	0.991176	0.985466	0.985507	0.985409	0.985455	0.970910	0.014591
Naive Bayes	0.989285	0.984873	0.984988	0.984760	0.984859	0.969719	0.015240
SVM	0.997887	0.984725	0.985232	0.984419	0.984703	0.969410	0.015581
Genetic	0.887624	0.882248	0.905076	0.879140	0.879879	0.762896	0.120860
Random Forest	0.998443	0.982945	0.983170	0.982764	0.982927	0.965855	0.017236
XGboost	0.985726	0.974937	0.976048	0.974419	0.974885	0.949788	0.025581

From Table 4.5, Figure 4.11 and 4.12, we compared different classifiers on Enron Dataset with Genetic Search based Feature Selection and it can be seen that all the classifiers performed well with greater train/test accuracy, precision, recall and F1-score. Bayesian Classifier performed the best with highest test accuracy, precision, recall and F1-score. Moreover, in genetic based feature selection, the test accuracy and F1-score for SVM is better than as in [96], which is 98.47% and 98.47% respectively.

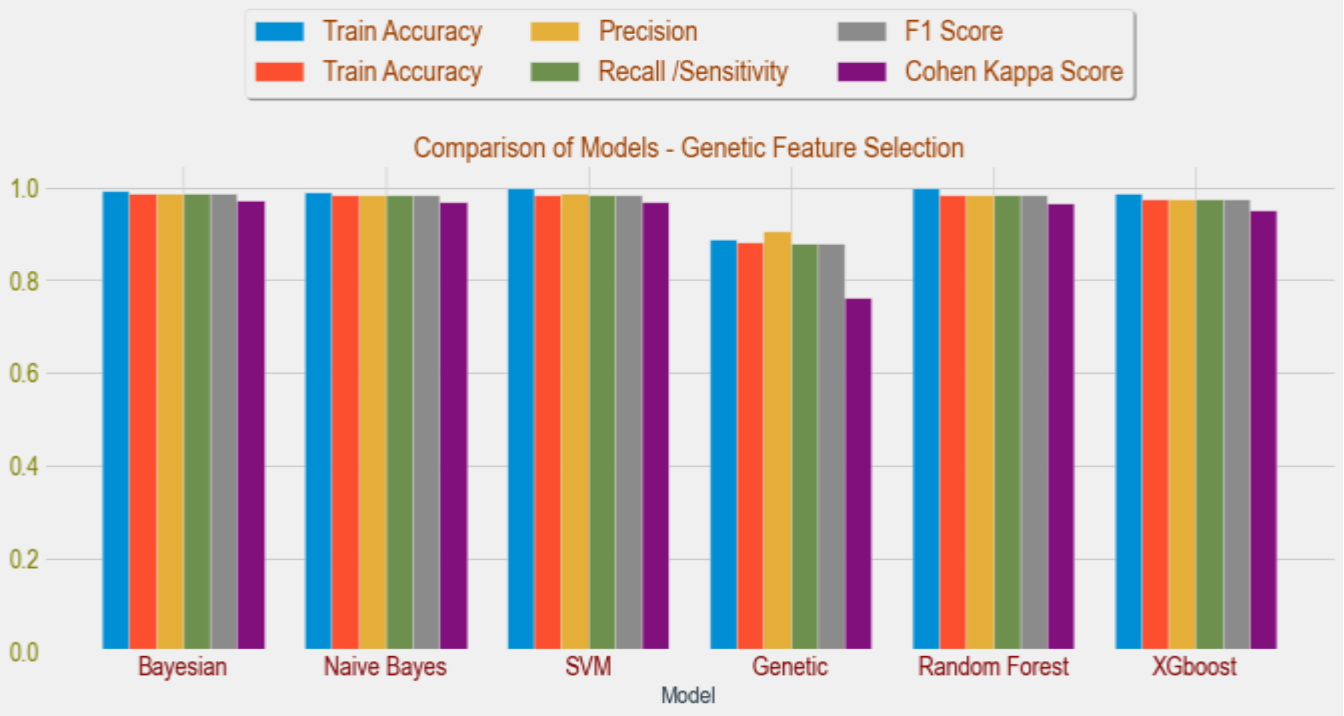

Figure 4.11. Evaluation metrics of each classifier on Enron dataset with genetic based feature selection

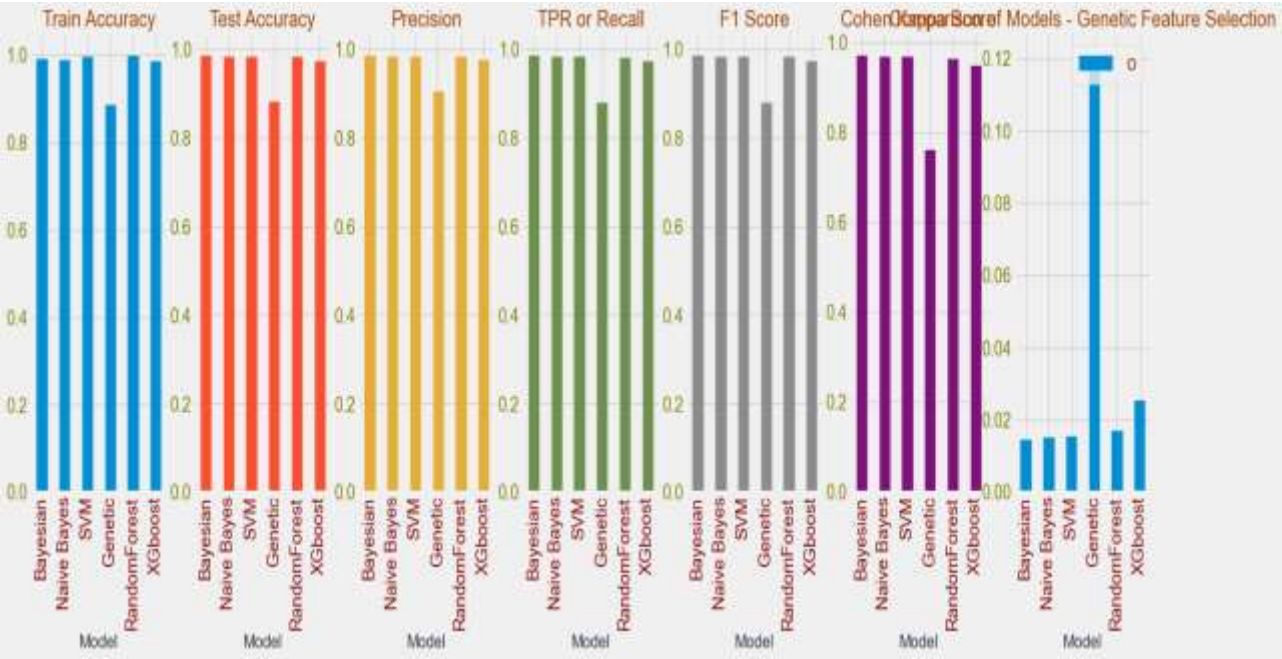

Figure 4.12. Comparison of classifiers on each evaluation metric on Enron dataset with genetic based feature selection

4.4.2.2 Spam Assassin

Table 4.6 shows the performance of the machine learning methods for Spam Assassin dataset using genetic based feature selection. Based on Genetic Feature Selection method, it can be seen that SVM, Random Forest and XGBoost have performed the best, based on various evaluation metrics.

Table 4.6. Performance of various machine learning models for Spam Assassin dataset with genetic based feature selection

Model	Train Accuracy	Test Accuracy	Precision	TPR or Recall	F1 Score	Cohen Kappa Score	FPR or Fall Out
Bayesian	0.893013	0.855895	0.921429	0.682692	0.724970	0.470909	0.317308
Naive Bayes	0.866084	0.822416	0.860908	0.618028	0.640048	0.318153	0.381972
SVM	0.999272	0.965066	0.975473	0.925340	0.947551	0.895262	0.074660
Genetic	0.840611	0.809316	0.901057	0.580128	0.583218	0.227805	0.419872
Random Forest	0.999272	0.960699	0.964492	0.922516	0.941436	0.883006	0.077484
XGboost	0.993086	0.969432	0.967998	0.944010	0.955331	0.910695	0.055990

From Table 4.6, Figure 4.13 and 4.14, we compared different classifiers on Spam Assassin Dataset and it can be seen that Bayesian Classifier, Genetic Classifier and Naïve Bayes Classifier have over fitted because of lower train/test accuracy, precision, recall and F1-score. The Cohen Kappa score of these classifiers is also typically very poor especially for Genetic Classifier as compared to the other models. If we find out the best model based on the Genetic Feature Selection

Method, XGBoost, Random Forest and SVM have performed quite better based on the evaluation measures, which has 96.9%, 96.0% and 96.5% test accuracy, respectively, that is considerably superior as compared to [96]. While, Genetic performs much better in [96]. XGBoost has the lowest Fall Out/False Positive Rate and highest precision along with other metrics, which makes it very suitable for real time implementation. We can conclude that based on Genetic Search based Feature Selection Method, XGBoost is the best model, because each of the trees is grown using information from previously grown trees, unlike bagging, where multiple copies of original training data are created and fit separate decision tree on each, this is how XGBoost work and get better accuracy.

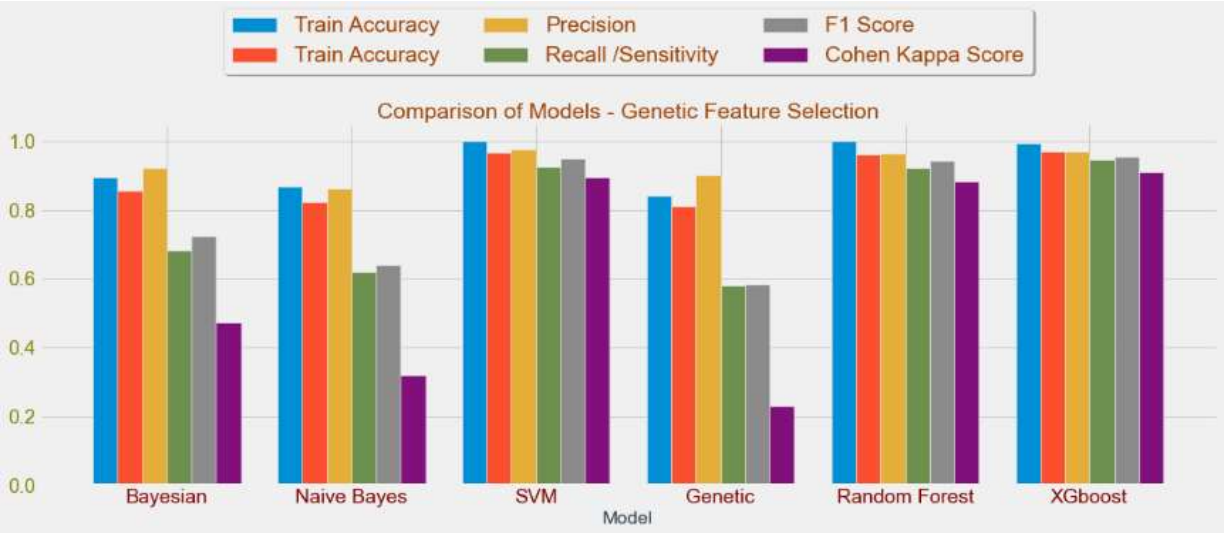

Figure 4.13. Evaluation metrics of each classifier on Spam Assassin dataset with genetic based feature selection

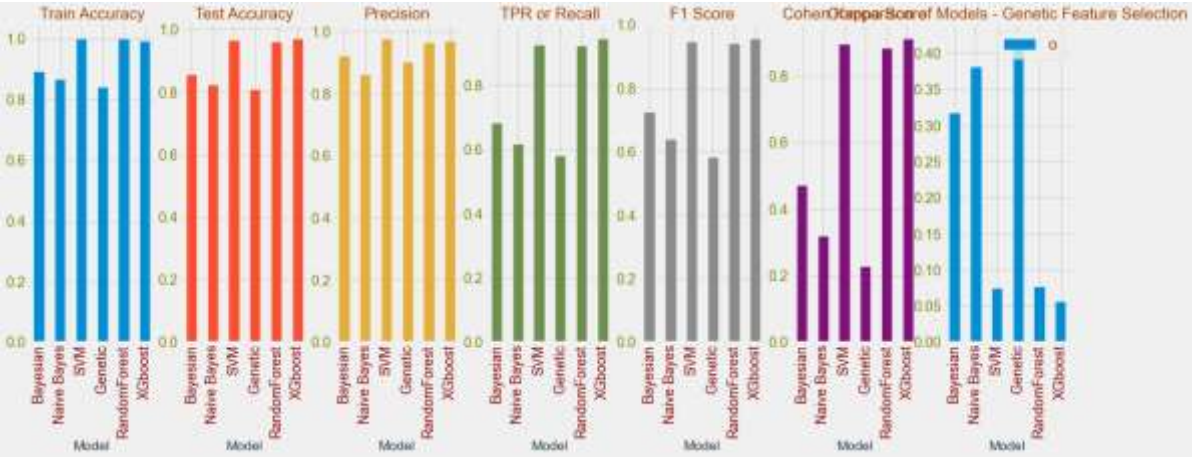

Figure 4.14. Comparison of classifiers on each evaluation metric on Spam Assassin dataset with genetic based feature selection

4.4.3 Greedy Search based feature Selection

4.4.3.1 Enron

Table 4.7 shows the performance of the machine learning methods for Enron dataset using Greedy Search based feature selection. Based on Greedy Search based feature selection method, it can be seen that SVM, Random Forest and XGBoost have performed the best, based on various evaluation metrics.

Table 4.7. Performance of various machine learning models for Enron dataset with greedy search-based feature selection

Model	Train Accuracy	Test Accuracy	Precision	TPR or Recall	F1 Score	Cohen Kappa Score	FPR or Fall Out
Bayesian	0.970673	0.969301	0.969697	0.969016	0.969261	0.938527	0.030984
Naive Bayes	0.928556	0.927332	0.935927	0.925555	0.926718	0.854058	0.074445
SVM	0.994698	0.982204	0.982513	0.981978	0.982182	0.964367	0.018022
Genetic	0.909425	0.903900	0.920621	0.901319	0.902469	0.806693	0.098681
Random Forest	0.996886	0.974789	0.975053	0.974574	0.974760	0.949522	0.025426
XGboost	0.984651	0.973009	0.974243	0.972454	0.972950	0.945922	0.027546

From Table 4.7, Figure 4.15 and Figure 4.16, we compared the different classifiers on Enron Dataset based on the Greedy Search Feature Selection method. It can be seen that Bayesian Classifier, Genetic Classifier and Naïve Bayes Classifier which were overfitting with the Genetic Search based Features selection method, also performed better and comparable with the other classifiers, and didn't over fit as they did with Genetic Search based Feature Selection methods because of improved train/test accuracy, precision, recall and F1-score. The Cohen Kappa score of these classifiers also improved very much. If we find out the best model based on the Greedy Search based Features Selection Method, Random Forest, SVM and XGBoost have performed again better based on the evaluation measures. SVM has the lowest Fall Out/False Positive Rate and highest precision along with other metrics, which makes it very suitable for real time implementation. We can conclude that based on Greedy Search Feature Selection Method, SVM

Classifier is the best model. It shows that performance of evaluation measures of SVM is significantly better than SVM performance as in [96].

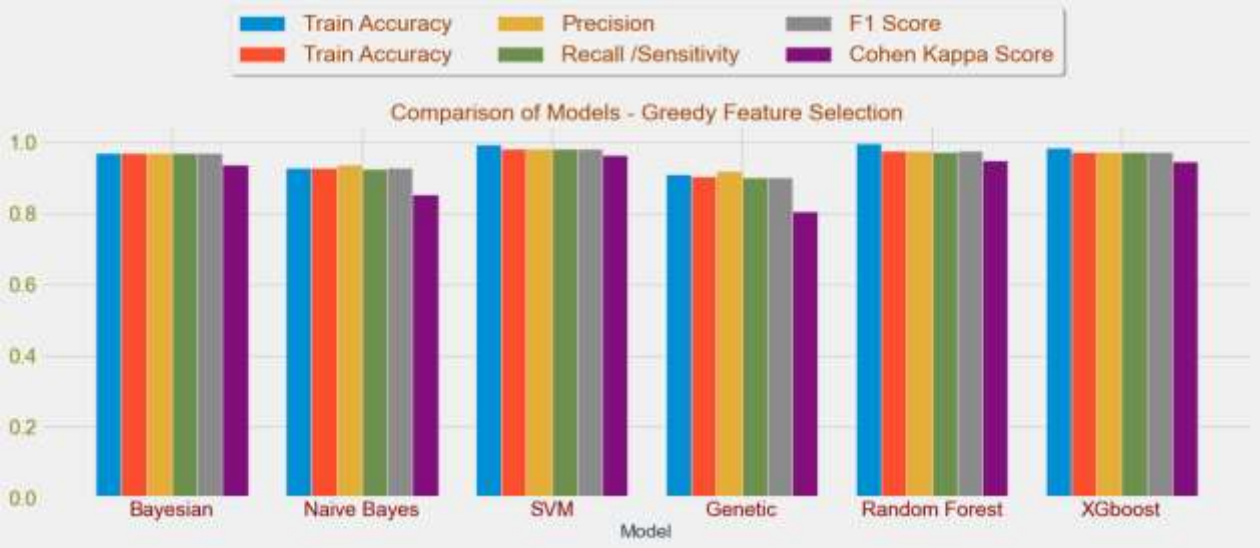

Figure 4.15. Evaluation metrics of each classifier on Enron dataset with greedy search-based feature selection

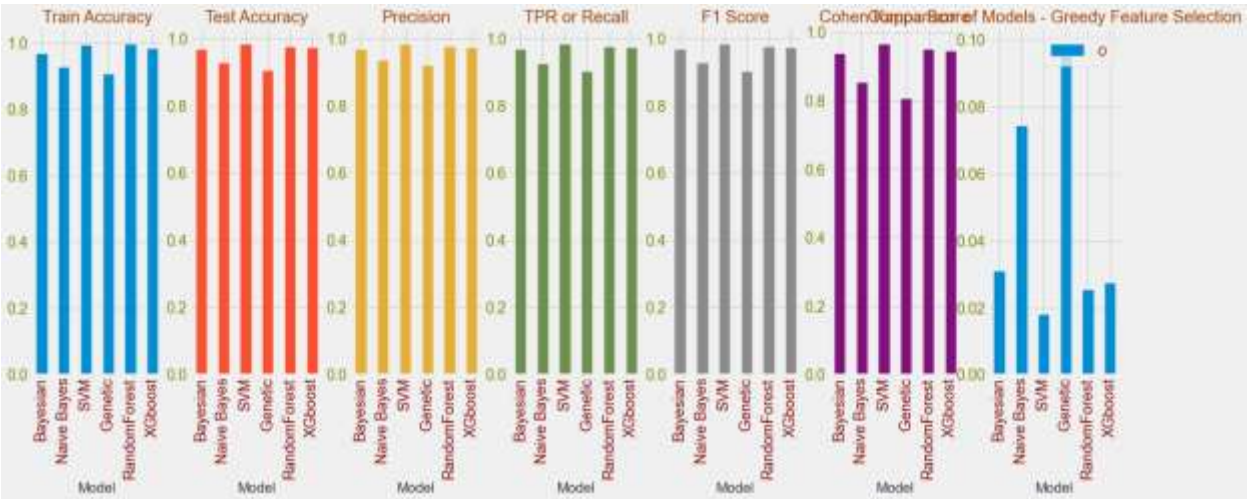

Figure 4.16. Comparison of classifiers on each evaluation metric on Enron dataset with greedy search-based feature selection

4.4.3.2 Spam Assassin

Table 4.8 shows the performance of the machine learning methods for Spam Assassin dataset using Greedy Search based feature selection. Based on Greedy Search based feature selection method, it can be seen that SVM, Random Forest and XGBoost have performed the best, based on various evaluation metrics.

Table 4.8. Performance of various machine learning models for Spam Assassin dataset with greedy search-based feature selection

Model	Train Accuracy	Test Accuracy	Precision	TPR or Recall	F1 Score	Cohen Kappa Score	FPR or Fall Out
Bayesian	0.893013	0.855895	0.921429	0.682692	0.724970	0.470909	0.317308
Naive Bayes	0.866084	0.822416	0.860908	0.618028	0.640048	0.318153	0.381972
SVM	0.999272	0.965066	0.975473	0.925340	0.947551	0.895262	0.074660
Genetic	0.853348	0.820961	0.905963	0.605769	0.622704	0.293157	0.394231
Random Forest	0.999272	0.962154	0.970837	0.921194	0.943180	0.886534	0.078806
XGboost	0.993086	0.969432	0.967998	0.944010	0.955331	0.910695	0.055990

From Table 4.8, Figure 4.17 and Figure 4.18, we compared the different classifiers on Spam Assassin dataset based on the Greedy Search Feature Selection method. It can be seen that again Bayesian Classifier, Genetic Classifier and Naïve Bayes Classifier which were overfitting a lot with the Genetic Search based Features selection method as in Enron Dataset, Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts

the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model, also performed better again with the other classifiers, and didn't over fit as they did with Genetic Search based Feature Selection methods because of improved train/test accuracy, precision, recall, and F1-score. The Cohen Kappa score of these classifiers also improved very much. If we find out the best model based on the Greedy Search based Features Selection Method, XGBoost, Random Forest and SVM have performed again better based on the evaluation measures. XGboost has the lowest Fall Out/False Positive Rate and highest precision along with other metrics, which makes it very suitable for real time implementation. So, we can conclude that based on Greedy Feature Selection Method, XGBoost Classifier is the best model, the train/test accuracy and precision has performed better compared to [94], is 96.9432% and 96.7998% consecutively.

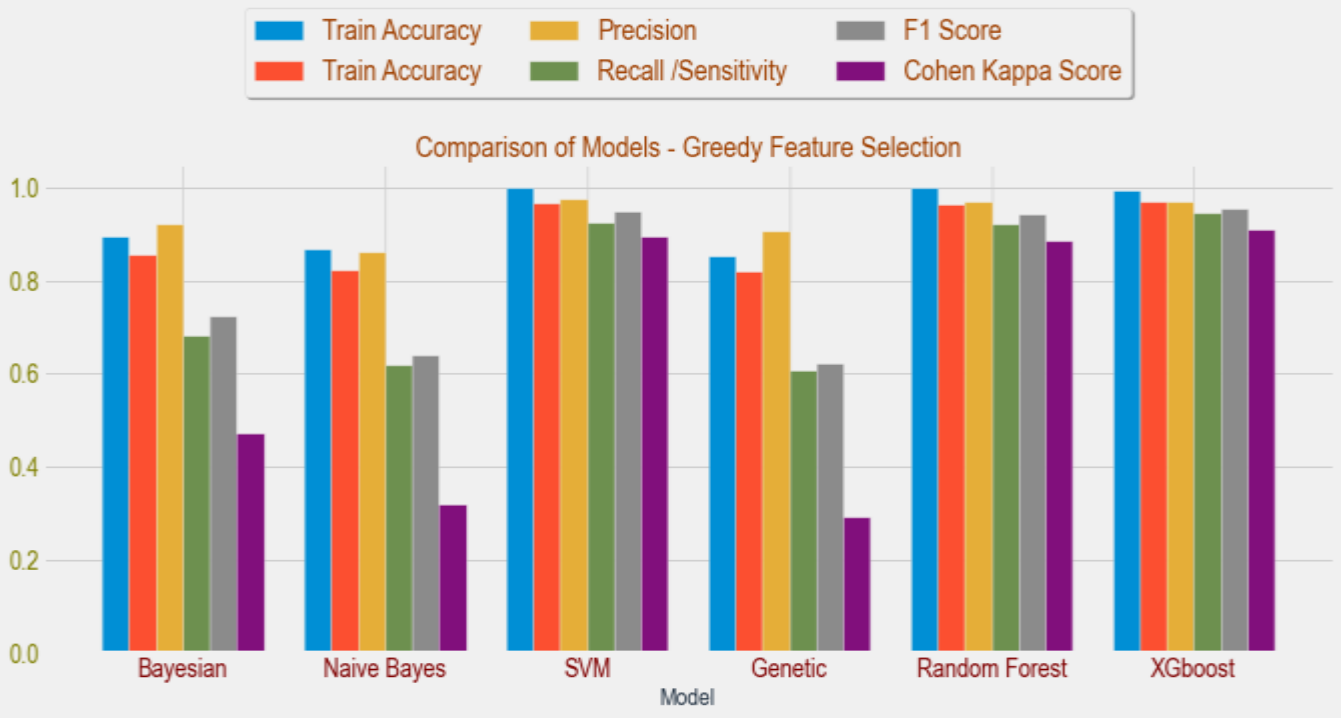

Figure 4.17. Evaluation metrics of each classifier on Spam Assassin dataset with greedy search-based feature selection

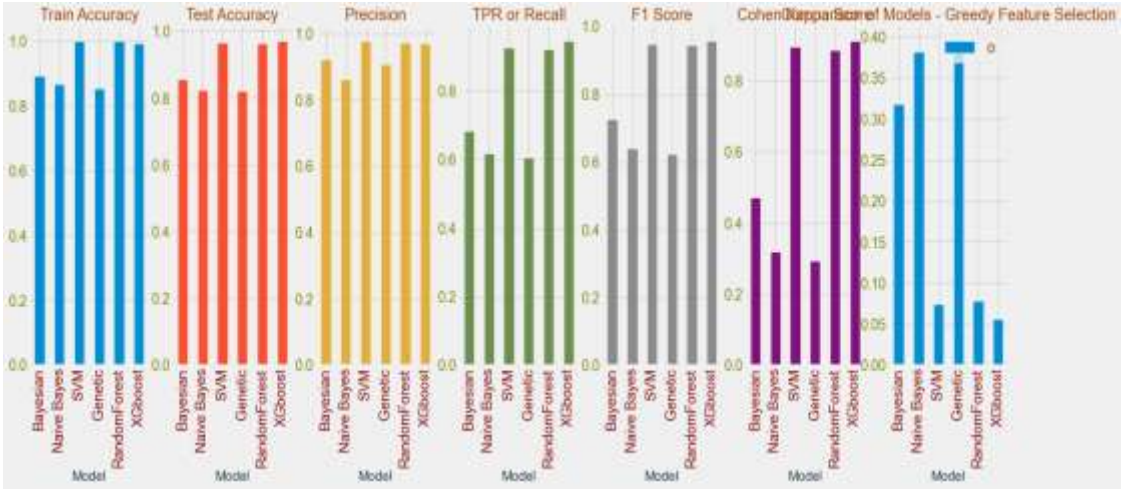

Figure 4.18. Comparison of classifiers on each evaluation metric on Spam Assassin dataset with greedy search-based feature selection

4.4.4 Deep Learning-based Models – LSTM (Long-Short Term Memory)

We used 80% observations in the training set and 20% observations in the testing set and also used Glove pre-train embedding for boosting the performance of the model. We convert both the datasets into numerical form from text form using word2vec method and then compare LSTM with all other machine learning models. The maximum length of the single vector of text was taken as 2000 and the number of features was considered as 2000 as the configuration of the neural network. All the text vectors or inputs were padded to ensure the similar length of each vector, which Neural Network requires before feeding the training data to the network. We used a Bi-Directional LSTM layer as the input layer with the embedding of 128 units followed by Global 1D Max Pooling. The network was flattened after the input layer and dense sequential layers were used of 32 units with dropout to avoid overfitting. Lastly, a single layer was used as the output layer. Bidirectional LSTM layers use tangent hyperbolic activation function, the hidden dense layer was also provided tangent hyperbolic activation function and the output layer was provided

a sigmoid activation function to obtain the probabilities in the range 0-1 at the output layer node. Figure 4.19 shows the architecture of the Bidirectional LSTM.

```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 2000)]            0

embedding (Embedding)        (None, 2000, 300)         5400000

bidirectional (Bidirectiona  (None, 2000, 256)         439296
l)

global_max_pooling1d (Globa  (None, 256)               0
lMaxPooling1D)

dense (Dense)                (None, 32)                8224

dropout (Dropout)            (None, 32)                0

dense_1 (Dense)              (None, 1)                 33

=================================================================
Total params: 5,847,553
Trainable params: 5,847,553
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.19. Architecture of Bi-LSTM Model

4.4.4.1 Enron dataset

The results obtained by LSTM for Enron dataset are summarized as follows:

- Accuracy: 0.976
- Cohen Kappa Score: 0.951
- Precision: 0.98
- Recall: 0.98
- F1-score: 0.98

Model Accuracy versus Number of Epochs for each train and test set

Figure 4.20 shows the learning curve of LSTM model on Enron Dataset. It can be seen that as the number of epochs increases from 0 to 5, the training accuracy gets maximum and stabilized but

the test accuracy achieves its best value after 15 epochs or iterations.

On Spam Assassin dataset LSTM is little overfitting, so there is a drop in the accuracy on test dataset, and so we can observe that there is some difference between accuracy of train and test Datasets.

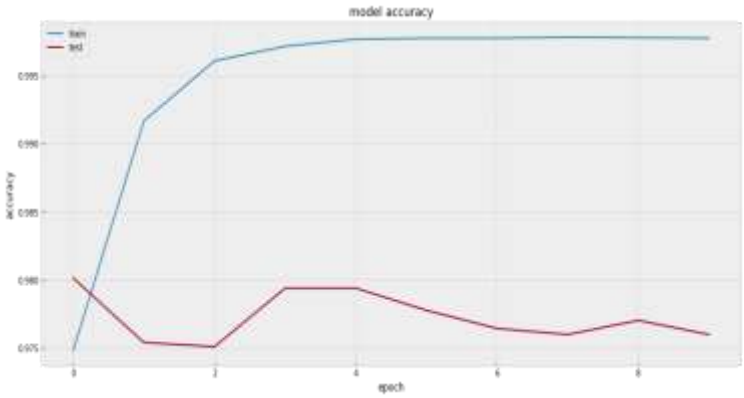

Figure 4.20. LSTM model Accuracy for Enron dataset

The learning curve confirms that LSTM model is not overfitting. The train accuracy and test accuracy are stable after certain number of epochs. If we train the model for more iterations, it would have been seen a flat line for validation data as well, also plotting from y-scale 0 to

Confusion Matrix

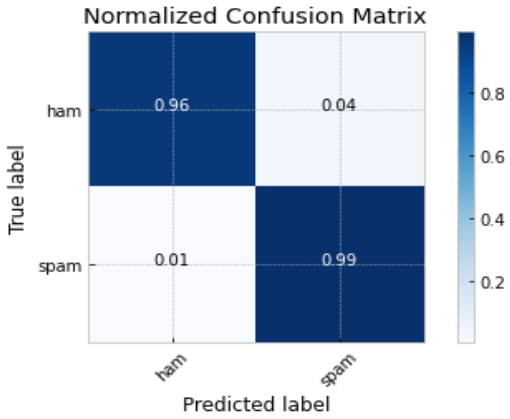

Figure 4.21. LSTM model Confusion Matrix for Enron dataset

From the confusion matrix, the performance metrics are as follows:

- Sensitivity, hit rate, recall, or true positive rate: 0.97568
- Specificity or true negative rate: 0.97568
- Precision or positive predictive value: 0.9766
- Negative predictive value: 0.9766
- Fall out or false positive rate: 0.0243
- False-negative rate: 0.02431
- False discovery rate: 0.0233148

The model is performing really well based on the above metrics.

4.4.4.2 Spam Assassin dataset

The results obtained by LSTM for Spam Assassin dataset are summarized as follows:

- Accuracy Score: 0.935
- Cohen Kappa Score: 0.82003
- Precision: 0.95
- Recall: 0.94
- F1-score: 0.96

Model Accuracy versus Number of Epochs for each train and test set

Figure 4.22 shows the learning curve of LSTM model on Spam Assassin Dataset, it can be seen that as the number of epochs increases from 0 to 5, the training accuracy gets maximum and stabilized but the test accuracy achieves its best value after 15 epochs or iterations.

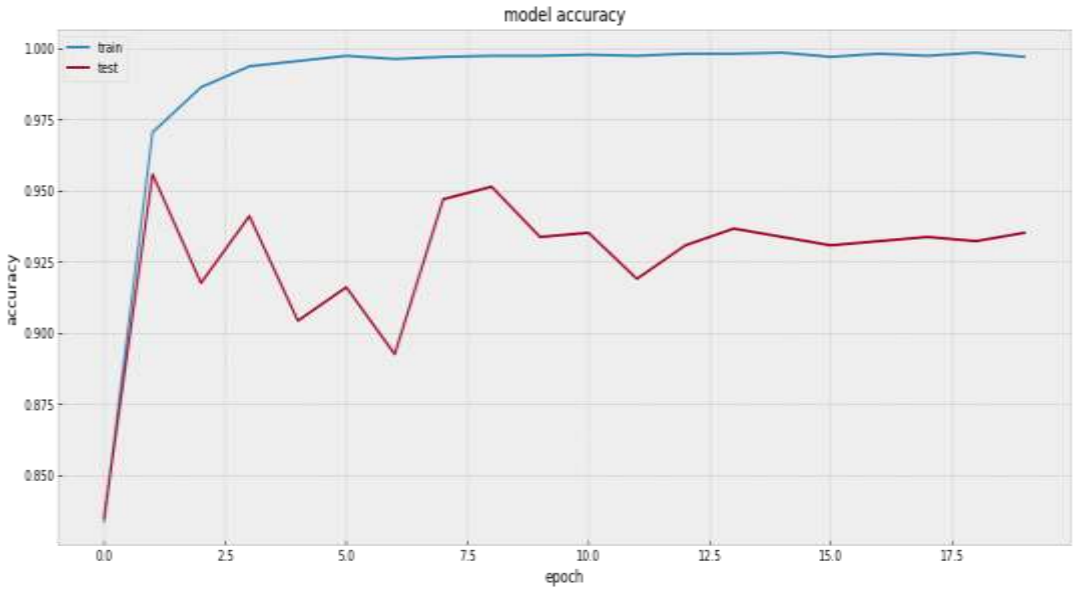

Figure 4.22. LSTM model accuracy for Spam Assassin dataset

The learning curve confirms that LSTM model is not overfitting. The train accuracy and test accuracy are stable after certain number of epochs.

Confusion Matrix

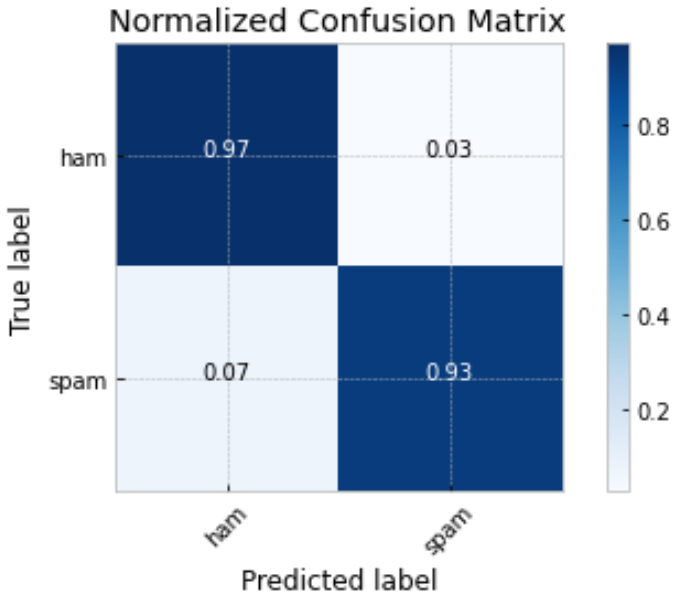

Figure 4.23. LSTM model Confusion Matrix for Spam Assassin dataset

From the confusion matrix, the performance metrics are as follows:

- Sensitivity, hit rate, recall, or true positive rate: 0.94864
- Specificity or true negative rate: 0.94864
- Precision or positive predictive value: 0.8830
- Negative predictive value 0.8830
- Fall out or false positive rate: 0.0513
- False-negative rate: 0.05135
- False discovery rate: 0.11697

4.5 Collective Comparison of all models on Enron Dataset

It can be seen from the results on Enron data in Table 4.9 that SVM, LSTM and Random Forest is performing the best among all the models with highest training accuracy, test accuracy, precision score, Cohen kappa score.

Table 4.9. Complete results of performance for Enron dataset

ENRON DATA								
Model	**Feature Selection**	**Train Accuracy**	**Test Accuracy**	**Precision**	**TPR / Recall**	**F1 Score**	**Cohen Kappa Score**	**FPR / Fall Out**
Bayesian	**None**	**0.971**	**0.969**	**0.970**	**0.969**	**0.969**	**0.939**	**0.031**
Naive Bayes	**None**	**0.929**	**0.927**	**0.936**	**0.926**	**0.927**	**0.854**	**0.074**
SVM	**None**	**0.995**	**0.982**	**0.983**	**0.982**	**0.982**	**0.964**	**0.018**
Genetic	**None**	**0.909**	**0.904**	**0.921**	**0.901**	**0.902**	**0.807**	**0.099**
Random Forest	**None**	**0.997**	**0.975**	**0.975**	**0.975**	**0.975**	**0.950**	**0.025**
XGboost	**None**	**0.985**	**0.973**	**0.974**	**0.972**	**0.973**	**0.946**	**0.028**

Bayesian	**Genetic Algorithm**	**0.991**	**0.985**	**0.986**	**0.985**	**0.985**	**0.971**	**0.015**
Naive Bayes	**Genetic Algorithm**	**0.989**	**0.985**	**0.985**	**0.985**	**0.985**	**0.970**	**0.015**
SVM	**Genetic Algorithm**	**0.998**	**0.985**	**0.985**	**0.984**	**0.985**	**0.969**	**0.016**
Genetic	**Genetic Algorithm**	**0.888**	**0.882**	**0.905**	**0.879**	**0.880**	**0.763**	**0.121**
Random Forest	**Genetic Algorithm**	**0.998**	**0.983**	**0.983**	**0.983**	**0.983**	**0.966**	**0.017**
XGboost	**Genetic Algorithm**	**0.986**	**0.975**	**0.976**	**0.974**	**0.975**	**0.950**	**0.026**
Bayesian	**Greedy Search**	**0.971**	**0.969**	**0.970**	**0.969**	**0.969**	**0.939**	**0.031**
Naive Bayes	**Greedy Search**	**0.929**	**0.927**	**0.936**	**0.926**	**0.927**	**0.854**	**0.074**
SVM	**Greedy Search**	**0.995**	**0.982**	**0.983**	**0.982**	**0.982**	**0.964**	**0.018**
Genetic	**Greedy Search**	**0.909**	**0.904**	**0.921**	**0.901**	**0.902**	**0.807**	**0.099**
Random Forest	**Greedy Search**	**0.997**	**0.975**	**0.975**	**0.975**	**0.975**	**0.950**	**0.025**
XGboost	**Genetic Algorithm**	**0.985**	**0.973**	**0.974**	**0.972**	**0.973**	**0.946**	**0.028**
LSTM	**LSTM Generated Features**	**0.983**	**0.976**	**0.980**	**0.980**	**0.980**	**0.951**	**0.024**

Random Forest and Support Vector Machine are working outstandingly well with any of the approach whether it is Greedy Search feature selection, Genetic Search feature selection or no feature selection. It is also interesting to note that Genetic Algorithm based feature selection is working well with each and every Classifier on Enron Dataset. Genetic Classifier is not working well on Enron Data, or Spam Assassin Dataset, and it was found that Overfitting in genetic algorithms and programming is a major problem that the community is presently researching. The majority of the study focuses on genetic programming and classification/regression model evolution [97].

4.6 Collective Comparison of all models on Spam Assassin Dataset

It can be seen from the results on Spam Assassin data in table 4.10 that SVM, XGboost, LSTM and Random Forest is performing the best among all the models with highest training accuracy, test accuracy, precision score. Cohen kappa score.

Table 4.10. Complete results of performance for Spam Assassin dataset

Spam Assassin								
Model	**Feature Selection**	**Train Accuracy**	**Test Accuracy**	**Precision**	**TPR / Recall**	**F1 Score**	**Cohen Kappa Score**	**FPR / Fall Out**
Bayesian	**None**	**0.893**	**0.856**	**0.921**	**0.683**	**0.725**	**0.471**	**0.317**
Naive Bayes	**None**	**0.866**	**0.822**	**0.861**	**0.618**	**0.640**	**0.318**	**0.382**
SVM	**None**	**0.999**	**0.965**	**0.975**	**0.925**	**0.948**	**0.895**	**0.075**
Genetic	**None**	**0.853**	**0.821**	**0.906**	**0.606**	**0.623**	**0.293**	**0.394**
Random Forest	**None**	**0.999**	**0.962**	**0.971**	**0.921**	**0.943**	**0.887**	**0.079**
XGboost	**None**	**0.993**	**0.969**	**0.968**	**0.944**	**0.955**	**0.911**	**0.056**
Bayesian	**Genetic Algorithm**	**0.897**	**0.864**	**0.921**	**0.683**	**0.735**	**0.471**	**0.315**
Naive Bayes	**Genetic Algorithm**	**0.873**	**0.822**	**0.861**	**0.618**	**0.640**	**0.318**	**0.372**
SVM	**Genetic Algorithm**	**0.999**	**0.965**	**0.975**	**0.925**	**0.948**	**0.895**	**0.077**
Genetic	**Genetic Algorithm**	**0.851**	**0.809**	**0.915**	**0.580**	**0.583**	**0.228**	**0.420**

Random Forest	**Genetic Algorithm**	**0.999**	**0.961**	**0.964**	**0.923**	**0.941**	**0.883**	**0.077**
XGboost	**Genetic Algorithm**	**0.993**	**0.969**	**0.968**	**0.944**	**0.955**	**0.911**	**0.056**
Bayesian	**Greedy Search**	**0.893**	**0.856**	**0.921**	**0.683**	**0.725**	**0.471**	**0.317**
Naive Bayes	**Greedy Search**	**0.866**	**0.822**	**0.861**	**0.618**	**0.640**	**0.318**	**0.382**
SVM	**Greedy Search**	**0.999**	**0.965**	**0.975**	**0.925**	**0.948**	**0.895**	**0.075**
Genetic	**Greedy Search**	**0.853**	**0.821**	**0.906**	**0.606**	**0.623**	**0.293**	**0.394**
Random Forest	**Greedy Search**	**0.999**	**0.962**	**0.971**	**0.921**	**0.943**	**0.887**	**0.079**
XGboost	**Genetic Algorithm**	**0.993**	**0.969**	**0.968**	**0.944**	**0.955**	**0.911**	**0.056**
LSTM	**LSTM Generated Features**	**0.995**	**0.935**	**0.950**	**0.940**	**0.960**	**0.820**	**0.051**

Random Forest and Support Vector Machine are working outstandingly well with any of the approach whether it is Greedy Search feature selection, Genetic Search feature selection or no feature selection. XGBoost is also performing very well, on the other hand Naïve Bayes and Genetic Classifier were not providing good results without or with any feature selection methods. Genetic Algorithm is also overfitting because of high dimensionality of the features, even with feature selection methods. Naïve Bayes is one of the simplest and traditional machine learning algorithms, so it can be understood why Naïve Bayes is not able to perform well comparable to these classifiers.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, it was demonstrated that using high-dimensional document representation obtained using the feature extraction methods such as BOW and TF-IDF followed by feature selection methods such as Genetic Algorithm and Greedy search with SVM and Random Forest algorithms and LSTM are more accurate than traditional machine learning algorithms and state-of-the-art spam filtering methods. Seven popular spam filtering methods (Naïve Bayes, Bayesian Classifier, and Support Vector Machine with Polynomial Kernel, Random Forest, Extreme Boosting, Genetic Classifier, and Long-Short Term Memory (ANN)) were benchmarked using two different datasets from different domains (Enron Dataset and Spam Assassin Dataset).

On the Enron dataset, the results show that in case of no feature selection, SVM with polynomial kernel and Random Forest are the best models as compared to others. In the case of Genetic Algorithm based feature selection, SVM, Random Forest, Bayesian, and Naïve Bayes classifiers outperformed all other classifiers. On the other hand, with Greedy Search-based feature selection, SVM with polynomial kernel and Random Forest are the best models as compared to others. Eventually, it can be seen that the SVM classifier with a polynomial kernel performed the best on Enron Dataset.

On the Spam Assassin dataset, the results show that in case of no feature selection, XGBoost, Random Forest, and SVM with the polynomial kernel are the best models as compared to others. In the case of Genetic Algorithm based feature selection and Greedy Search based feature

selection as well XGBoost, Random Forest, and SVM with polynomial kernel classifier outperformed all other classifiers.

5.2 Limitations

The main limitation of this study is that the Genetic Search feature selection model and SVM with the polynomial kernel are more computationally intensive than the compared algorithms and also require more computational resources and time. The average testing and especially training CPU time is significantly higher compared to the state-of-the-art spam filters. On NVidia GTX 1060 TI GPU with 6 Gigabytes of RAM it usually took 30 minutes to execute and save the model results, while on the other hand with only CPU of 16GB RAM, it took more than 2-3 hours for a single model to execute.

The difficulties of machine learning algorithms in effectively dealing with the spam threat were addressed, and comparative assessments of machine learning approaches accessible in the literature were conducted. We have discovered several outstanding methods with spam filters. Overall, the number and quality of literature we examined indicate that tremendous progress has been done and will continue to be made in this area. Following the discussion of the remaining issues in spam filtering, more research to improve the efficiency of spam filters is required. As a result, academics and business professionals investigating machine learning approaches for efficient spam filtering will continue to be engaged in the development of spam filters. We expect that this study will serve as a platform for qualitative research in spam filtering by employing machine learning, deep learning, and deep adversarial learning algorithms by research students. Moreover, this research also demonstrates the central importance of text pre-processing strategies in detecting spam messages. The results indicate that common patterns can be observed. The

number and length of the extracted word segments have a major effect on the performance of the classifiers.

The models used on both the datasets, Enron and Spam assassin are old datasets. The latest datasets will have to be trained with different classifiers to achieve high accuracy.

5.3 Future work

We can also use more extensive features in further studies such as n-gram models which can boost the performance of the detector. Also, more emphasis can be given to deep learning-based methods such as Convolutional Neural networks or CNN-based Artificial Neural Networks.

Further work can also take into consideration other factors that are a strong indicator of spam other than just the message content, such as email header, URLs, image-based filtering, analysing of SMTP (Simple Mail Transfer Protocol) paths, behaviour-based filtering, and analysing users' social network by analysing 'From', 'To', 'CC' and 'BCC' fields of the email messages.

The model can also be extended to train and learn the features from the emails in a broader range of languages and test how the algorithms are fair for languages other than English.

References

[1] CORMACK, G. V. Email spam filtering: a systematic review. Foundations and Trends® in Information Retrieval, 2006, vol. 1, no. 4, pp. 335–455. https://doi.org/10.1561/150 0000006

[2] ZHANG, L., ZHU, J., YAO, T. An evaluation of statistical spam filtering techniques. ACM Transactions on Asian Language Information Processing, 2004, vol. 3, no. 4, pp. 243–269. DOI: 10.1.1.109.7685

[3] DELANY, S. J., BUCKLEY, M., GREENE, D. SMS spam filtering: methods and data. Expert Systems with Applications, 2012, vol. 39, no. 10, pp. 9899–9908. DOI: 10.1016/j.eswa.2012.02.053

[4] ZHOU, B., YAO, Y., LUO, J. A three-way decision approach to email spam filtering. In: Canadian Conference on Artificial Intelligence, Lecture Notes in Computer Science, vol. 6085. Springer, 2010, pp. 28–39. doi: 10.1007/978-3-642-13059-5_6

[5] BARUSHKA, A., HÁJEK, P. Spam filtering using regularized neural networks with rectified linear units. In: Adorni, G., Cagnoni, S., Gori, M., Maratea, M. (eds.) Conference of the Italian Association for Artificial Intelligence. Lecture Notes in Computer Science,

Springer, Cham, 2016, vol. 10037, pp. 65–75. doi: 10.1007/978-3-319-49130-1_6

[6] BHOWMICK, A., HAZARIKA, S. M. E-mail spam filtering: A review of techniques and trends. In: Kalam A, Das S, Sharma K (eds.) Advances in Electronics, Communication, and

Computing. Lecture Notes in Electrical Engineering, Springer, Singapore, vol. 443, 2018, pp. 583–590. doi: 10.1007/978-981-10-4765-7_61

[7] ALMEIDA, T. A., ALMEIDA, J., YAMAKAMI, A. Spam filtering: how the dimensionality reduction affects the accuracy of Naive Bayes classifiers. Journal of Internet Services and Applications, 2011a, vol. 1, vol. 3, pp. 183–200. DOI: 10.1007/s13174-010-0014-7

[8] ALMEIDA, T. A., HIDALGO, J. M. G., YAMAKAMI, A. Contributions to the study of SMS spam filtering: new collection and results. In: Proceedings of the 11th ACM

[9] CHOUDHARY, N., JAIN, A. K. Towards filtering of SMS spam messages using machine Learning-Based Technique. In: Singh, D., Raman, B., Luhach, A., Lingras, P. (eds.) Advanced Informatics for Computing Research. Communications in Computer and Information Science, vol. 712, Springer, Singapore, 2017, pp. 18–30. doi: 10.1007/978- 981-10-5780-9_2

[10] KAUR, R., SINGH, S., KUMAR, H. Rise of spam and compromised accounts in online social networks: A state-of-the-art review of different combating approaches. Journal of Network and Computer Applications, 2018, vol. 112, pp. 53–88. doi: 10.1016/j.jnca.2018.03.015

[11] CRAWFORD, M., KHOSHGOFTAAR, T.M., PRUSA, J.D., RICHTER, A.N., AND AL NAJADA, H. Survey of review spam detection using machine learning techniques.

Journal of Big Data, 2015, vol. 2, no. 1, pp. 1-23. DOI: 10.1186/s40537-015-0029-9

[12] LE, Q., MIKOLOV., T. Distributed representations of sentences and documents. In:

International Conference on Machine Learning, 2014, vol. 32, pp. 1188–1196.

[13] LECUN, Y., BOTTOU, L., BENGIO, Y., HAFFNER, P. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 1998, vol. 86, no. 11, pp. 2278–2324. https://doi.org/10.1109/5.726791

[14] LILLEBERG, J., ZHU, Y., AND ZHANG, Y. Support vector machines and word2vec for text classification with semantic features. In: 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC), IEEE, 2015, pp. 136–140. doi: 10.1109/ICCI-CC.2015.7259377

[15] HOANCA, B. How good are our weapons in the spam wars? IEEE Technology and Society Magazine, 2006, vol. 25, no. 1, pp. 22–30. DOI: 10.1109/MTAS. 2006.1607720

[16] LAORDEN, C., UGARTE-PEDRERO, X., SANTOS, I., SANZ, B., NIEVES, J., BRINGAS, P. G. Study on the effectiveness of anomaly detection for spam filtering.

Information Sciences, 2014, vol. 277, pp. 421–444. DOI : 10.1016/j.ins. 2014.02.114

[17] OBIED, A., ALHAJJ, R. Fraudulent and malicious sites on the web. Applied Intelligence, 2009, vol. 30, no. 2, pp. 112–120. https://doi.org/10.1007/s10489- 007-0102-y

[18] WEI, C. P., CHEN, H. C., CHENG, T. H. Effective spam filtering: a single class learning and ensemble approach. Decision Support Systems, 2008, vol. 45, no. 3, pp. 491–503. DOI: 10.1016/j.dss.2007.06.010

[19] NEXGATE. 2013 State of Social Media Spam Research Report, 2013. Accessed 10 January 2020, available at: https://go.proofpoint.com/rs/309-RHV-619/images/Nexgate- 2013-State-of-Social-Media-Spam-Research-Report.pdf.

[20] STATISTA. Global spam volume as a percentage of total e-mail traffic from January 2014 to September 2019. 2019a, accessed on 10 January 2020, available at: https://www.statista.com/statistics/420391/spam-email-traffic-share/

[21] DADA, E. G., BASSI, J. S., CHIROMA, H., ADETUNMBI, A. O., AJIBUWA, O. E. Machine learning for email spam filtering: review, approaches, and open research problems. Heliyon, 2019, vol. 5, no. 6, pp. e01802. DOI: 10.1016/j.heliyon. 2019.e01802

[22] CARPINTER, J., HUNT, R. Tightening the net: a review of current and next-generation spam filtering tools. Computers & Security, 2006, vol. 25, no. 8, pp. 566–578. DOI: 10.1016/j.cose.2006.06.001

[23] HENNING, J. L. SPEC CPU2006 benchmark descriptions. ACM SIGARCH Computer Architecture News, 2006, vol. 34, no. 4, pp. 1–17. DOI: 10.1145/1186736.1186737

[24] CARUANA, G., LI, M. A survey of emerging approaches to spam filtering. ACM Computing Surveys, 2008, vol. 44, no. 2, pp. 1–27. DOI: 10.1145/2089125.2089129

[25] KAYA, Y., ERTUGRUL, O. F. A novel approach for spam email detection based on shifted binary patterns. Security and Communication Networks, 2016, vol. 9, no. 10, pp. 1216–1225. doi: 10.1002/sec.1412

[26] SEBASTIANI, F. Machine learning in automated text categorization. ACM Computing Surveys (CSUR), 2002, vol. 34, no. 1, pp. 1–47. DOI: 10.1145/505282.505283

[27] SHEN, H., LI, Z. Leveraging social networks for effective spam filtering. IEEE Transactions on Computers, 2014, vol. 63, no. 11, pp. 2743–2759. DOI: 10.1109/TC.2013.152

[28] HAGENAU, M., LIEBMANN, M., NEUMANN, D. Automated news reading: stock price prediction based on financial news using context-capturing features. Decision Support Systems, 2013, vol. 55, no. 3, pp. 685–697. https://doi.org/10.1016/j.dss.2013.02.006

[29] ANDROUTSOPOULOS, I., KOUTSIAS, J., CHANDRINOS, K. V., SPYROPOULOS,

C. D. An experimental comparison of Naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In: Proceedings of the 23rd Annual Int. ACM SIGIR Conference on Research and Development in Information Retrieval, 2000, pp. 160– 167. https://doi.org/10.1145/345508.345569

[30] METSIS, V., ANDROUTSOPOULOS, I., PALIOURAS, G. Spam filtering with Naive Bayes - which Naive Bayes? In: Third Conference on Email and Antispam (CEAS), 2006, pp. 27–28. doi: 10.1.1.61.5542

[31] DRUCKER, H., WU, D., VAPNIK, V. Support vector machines for spam categorization. IEEE Transactions on Neural Networks, 1999, vol. 10, no. 5, pp. 1048–1054. doi: 10.1109/72.788645

[32] WATKINS, A., TIMMIS, J. Artificial immune recognition system (AIRS): an immune-inspired supervised learning algorithm. Genetic Programming and Evolvable Machines, 2004, vol. 5, no. 3, pp. 291–317. DOI: 10.1023/B: GENP.0000030197.83685.94

[33] ZITAR, R. A., HAMDAN, A. Genetic optimized artificial immune system in spam detection: a review and a model. Artificial Intelligence Review, 2013, vol. 40, no. 3, pp. 305–377. DOI: 10.1007/s10462-011-9285-z

[34] Company Cheated of RM 4.5 Mil Due to Email Spoofing, Jul. 2019, [online] Available: https://www.thestar.com.my/news/nation/2017/06/11/kedah-based-company-cheated-due-to-email-spoofing.

[35] T. L. Shan, G. N. Samy, B. Shanmugam, S. Azam, K. C. Yeo and K. Kannoorpatti, "Heuristic systematic model-based guidelines for phishing victims", Proc. IEEE Annu. India Conf. (INDICON), pp. 1-6, Dec. 2016.

[36] J. V. Chandra, N. Challa and S. K. Pasupuleti, "A practical approach to E-mail spam filters to protect data from advanced persistent threat", Proc. Int. Conf. Circuit Power Comput. Technol. (ICCPCT), pp. 1-5, Mar. 2016.

[37] J. V. Chandra, N. Challa and S. K. Pasupuleti, "A practical approach to E-mail spam filters to protect data from advanced persistent threat", Proc. Int. Conf. Circuit Power Comput. Technol. (ICCPCT), pp. 1-5, Mar. 2016.

[38] A. Han, Leoni AG Loses 40m in an Email Scam, Apr. 2019, [online] Available: https://www.bankvaultonline.com/news/security-news/leoni-ag-loses-e40m-in-an-email-scam/.

[39] M. J. Schwartz, French Cinema Chain Fires Dutch Executives Over CEO Fraud, Apr. 2019, [online] Available: https://www.bankinfosecurity.com/blogs/french-cinema-chain-fires-dutch-executives-over-ceo-fraud-p-2681.

[40] Fette, Ian, Norman Sadeh, and Anthony Tomasic.” Learning to detect phishing emails.” Proceedings of the 16th international conference on World Wide Web. ACM, 2007.

[41] Li, Wenbin, Ning Zhong, and Chunnian Liu.” Combining multiple email filters based on multivariate statistical analysis.” Foundations of Intelligent Systems. Springer Berlin Heidelberg, 2006. 729-738.

[42] Spirin, Nikita, and Jiawei Han.” Survey on web spam detection: principles and algorithms.” ACM SIGKDD Explorations Newsletter 13.2 (2012): 50-64.

[43] Karami, Amir, and Lina Zhou.” Improving static SMS spam detection by using new content-based features.” (2014).

[44] Lim, Ee-Peng.” Detecting product review spammers using rating behaviors.” Proceedings of the 19th ACM international conference on Information and knowledge management. ACM, 2010.

[45] Ott, Myle,” Finding deceptive opinion spam by any stretch of the imagination.” Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

[46] Harris, C.” Detecting deceptive opinion spam using human computation.” Workshops at AAAI on ArtiFIcial Intelligence. 2012.

[47] Abernethy, Jacob, Olivier Chapelle, and Carlos Castillo. “Graph regularization methods for web spam detection.” Machine Learning 81.2 (2010): 207-225.

[48]. Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998, July). A Bayesian approach to filtering junk e-mail. In Learning for Text Categorization: Papers from the 1998 workshop (Vol. 62, pp. 98-105). [49]. Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos,

C. D., & Stamatopoulos, P. (2001). Stacking classifiers for anti-spam filtering of e-mail. arXiv preprint cs/0106040.

[50]. Trivedi, S. K., & Dey, S. (2013). The interplay between Probabilistic Classifiers and Boosting Algorithms for Detecting Complex Unsolicited Emails. Journal of Advances in Computer Networks, 1(2).

[51]. Drucker, H., Wu, D., & Vapnik, V. N. (1999). Support vector machines for spam categorization. Neural Networks, IEEE Transactions on, 10(5), 1048-1054.

[52] Trivedi, S. K., & Dey, S. (2013). Effect of Various Kernels and Feature Selection Methods on SVM Performance for Detecting Email Spams." International Journal of Computer Applications 66.21 (2013)

[53] D. Puniškis, R. Laurutis and R. Dirmeikis, "An Artificial Neural Nets for Spam e-mail Recognition", Electronics and electrical engineering, Vol. 69, No. 5, pp. 73 – 76, 2006.

[54] Youn and Dennis McLeod, "A Comparative Study for Email Classification", Proceedings of International Joint Conferences on Computer, Information, System Sciences and Engineering, 2006.

[55] Goswami, V, Malviya, V, and Sharma, P. (2019). Spam Emails/SMS Detecting Using Various Machine Learning Techniques, Journal of Applied Science and Computations. Volume VI, Issue VI, JUNE/2019

[56] Pavas, N, and Gaurav, D.A.R. (2017) "SMS Spam Filtering using Supervised Machine Learning Algorithms" in IEEE 2018.Ali, W. (2017, January).

[57] Atanu, G, and Ajit, K.P (2018). Identifying spam SMS using Apache Spark MLlib, International Journal of Emerging Technologies and Innovative Research(www.jetir.org), ISSN:2349-5162, Vol.5, Issue 5, page no.893-897, MAY-2018

[58] Aditya, B, Mehul, G, Shubhangi, A, and Pulkit, M. (2018). A Comparative Study of Spam SMS Detection Using Machine Learning Classifiers .2018 Eleventh International Conference on Contemporary Computing (IC3).

[59] Choudhary, N, and Jain, A.K (2017) Towards Filtering of SMS Spam Messages Using Machine Learning-Based Technique, In book: Advanced Informatics for Computing Research, D. Singh (Eds.): ICAICR 2017, CCIS 712, pp. 18–30, 2017. Springer Nature Singapore Pte Ltd. 2017.

[60] El-Alfy, E.S.M, and AlHasan, A.A. (2016) Spam filtering framework for multimodal mobile communication based on dendritic cell algorithm. Future Gen. Comput. Syst. 64, 98–107 (2016)

[61] Jialin, M, Zhang, Y, Liu, J, Yu, K, and Wang, X. (2016) Intelligent SMS spam filtering using the topic model. In: International Conference on Intelligent Networking and Collaborative Systems (INCoS), pp. 380–383. IEEE (2016)

[62] Kim, S.E, Jo, J.T, and Choi, S.H. (2015) ‘‘SMS spam filtering using keyword frequency ratio,’’ Int. J. Secure. Appl., vol. 9, no. 1, pp. 329–336, 2015

[63] Shahi, T.B, and Yadav, A. (2014). Mobile SMS Spam Filtering for Nepali Text Using Naïve Bayesian and Support Vector Machine International Journal of Intelligence Science 04(01):24-28

[64] Shirani-Mehr, H (2013) SMS Spam Detection using Machine Learning Approach, in a Book: CS229 Project 2013, published by Stanford University, pp. 1-4.

[65] C. Science and S. Engineering, “Effective Email Classification for Spam and Non-Spam,” vol. 4, no. 6, pp. 273–278, 2014.

[66] A. R. On and D. Glaucoma, “a Review on Different spam Detection,” vol. 11, no. 6, pp. 2–7, 2015.

[67] R. M. Alguire, R. M. Ali Guliyev, and S. A. Nazirova, “Classification of Textual E-Mail Spam Using Data Mining Techniques,” vol.

[68] M. T. Scholar and C. Science, “A Review on different Spam Detection Techniques.”

[69] M. Basavaraju and R. Prabhakar, “A Novel Method of Spam Mail Detection using Text-Based Clustering Approach,” vol. 5, no. 4, pp.

15–25, 2010.

[70] S. Geerthik and T. P. Anish, “Filtering Spam: Current Trends and Techniques,” vol. 3, no. 8, pp. 208–223, 2013.

[71] A. Nosseir, K. Ngati, and I. Taj-Eddin, “Intelligent Word-Based Spam Filter Detection Using Multi-Neural Networks Intelligent

Word-Based Spam Filter Detection Using Multi-Neural Networks,” no. November 2016, 2013.

[72] “Deakin Research Online,” pp. 1–5, 2010.

[73] “Spam Detection Using Baysian with Pattern Discovery,” no. 3, pp. 139–143, 2013.

[74] S. Nazarova, “Survey on Spam Filtering Techniques,” vol. 2011, no. August, pp. 153–160, 2011.

[75] S. S. Shinde and P. R. Patil, “International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

Improving spam mail filtering using classification algorithms with discretization Filter,” pp. 82–87, 2014.

[76] V. Christina, P. S. G. R. K. College, and P. S. G. R. K. College, “A Study on Email Spam Filtering Techniques,” vol. 12, no. 1, pp. 7–

9, 2010.

[77] B. Klimt and Y. Yang, “The Enron Corpus: A New Dataset for Email Classification Research.”

[78] M. E. Yitagesu and P. M. Tijare, “Email Classification using Classification Method,” vol. 32, no. 3, pp. 142–145, 2016.

[79] A. T. Sharma and M. I. Technology, “Analysis of Email Fraud Detection Using WEKA Tool,” vol. 10, no. 4, pp. 203–207, 2014.

[80] R. Giyanani and M. Desai, “Spam Detection using Natural Language Processing 1 1,” vol. 16, no. 5, pp. 116–119, 2014.

[81] V. Christina, S. Karpagavalli, and G. Suganya, “Email Spam Filtering using Supervised Machine Learning Techniques,” vol. 2, no. 9,

pp. 3126–3129, 2010.

[82] S. Chakraborty and B. Mondal, “Spam Mail Filtering Technique using Different Decision Tree Classifiers through Data Mining

Approach - A Comparative Performance Analysis," Int. J. Comput. Appl., vol. 47, no. 16, pp. 26–31, 2012. 2011, 2011.

[83] De Silva, A. M., & Leong, P. H. (2015), "Grammar Based Feature Generation", In Grammar-Based Feature Generation for Time-Series Prediction (pp. 35-50). Springer, Singapore.

[84] Razi, Z., & Asghari, S. A. (2016), "Providing an Improved Feature Extraction Method for Spam Detection Based on Genetic Algorithm in an Immune System".

[85] Choudhary, M., & Dhaka, V. S. (2015), "Automatic e-mails Classification Using Genetic Algorithm".

[86] Varghese, L., Supriya, M. H., & Jacob, K. P. (2015), "Finding Template Mails from Spam Corpus Using Genetic Algorithm and K- Means Algorithm", Training, 50, 50.

[87] Holland, J. H., & Goldberg, D. (1989), "Genetic algorithms in search, optimization and machine learning", Massachusetts: Addison- Wesley.

[88] Li, W. (2004), "Using genetic algorithm for network intrusion detection", Proceedings of the United States Department of Energy Cyber Security Group, 1, 1-8.

[89] Ferragina, P., & Grossi, R. (1999), "Improved dynamic text indexing", Journal of Algorithms, 31(2), 291-319.

[90] Han, J. and Kamber, M." Data Mining: Concepts and Techniques." 2nd Edition. Morgan Kaufmann Publishers, San Francisco, USA. (ISBN-55860-901-6), 2006.

[91] Federal Energy Research Commission, "FERC: Information Released in Enron Investigation"

[92] William Cohen, “Enron Email Dataset”, Carnegie Mellon School of Computer Science, available at: http://www.cs.cmu.edu/~enron/.

[93] Genetic Algorithms and its use-cases in Machine Learning-available-online-https://www.analyticsvidhya.com/blog/2021/06/genetic-algorithms-and-its-use-cases-in-machine-learning/

[94] Artif Intell Rev (2010), "A study of spam filtering using support vector machines".Ola Amayri · Nizar Bouguila

[95] p.U. Anitha, dr.C.V. Guru Rao, dr. D. Suresh Babu (2021), "Email Spam Filtering Using Machine Learning Based Xgboost Classifier Method”. Vol.12 No.11 (2021), 2182-2190.

[96] Sharwankumar Trivedi, Shubhamoy Dey [2013], "Effect of feature selection methods on machine learning classifiers for detecting email spams".

[97] How to avoid overfitting with genetic algorithm, https://stackoverflow.com/questions/27764825/how-to-avoid-overfitting-with-genetic-algorithm