BINDURA UNIVERSITY OF SCIENCE EDUCATION

FACULTY OF SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE

APPLICATION OF RANDOM FOREST MACHINE LEARNING ALGORITHM FOR SPAM EMAIL CLASSIFICATION

BY

MUNOSHAMISA Z. VERUH

B201857B

SUPERVISOR: Mr O Muzurura

A RESEARCH PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE BACHELOR OF SCIENCE HONOURS

DEGREE IN COMPUTER SCIENCE

DECEMBER 2024

APPROVAL FORM

The undersigned certify that they have supervised the student Munoshamisa Z Veruh dissertation entitled "APPLICATION OF RANDOM FOREST MACHINE LEARNING ALGORITHM FOR SPAM EMAIL CLASSIFICATION" submitted in

Partial fulfilment of the requirements for the Bachelor of Computer Science Honours Degree of Bindura University of Science Education.

……. ……...… ……/……/……..

STUDENT SIGNATURE DATE

……... ……. ……/……/……

SUPERVISOR SIGNATURE DATE

……. …….. ……/……/……..

CHAIRPERSON SIGNATURE DATE

…….. …….. ……/……/……

EXTERNALEXAMINER SIGNATURE DATE

Abstract

It is clear from analysing how common phishing email attacks are in today's technological environment that many regular email users become victims because they are unable to make wise decisions. This vulnerability results from phishing emails' sophisticated design, which makes it possible for them to get past typical spam filters. Meanwhile, online communication is being greatly impacted by natural language processing (NLP), which is quickly becoming recognised in a variety of high-tech areas. As such, NLP integration into email phishing categorization systems is crucial. This work explores text processing and categorization, building the classifier with the Random Forest algorithm. Although there are a number of techniques for identifying phishing emails, the use of natural language processing (NLP) in this situation has not received enough attention.

Dedication

My family and friends, whose steadfast support and encouragement have been invaluable throughout this journey, are honoured in this work. To my parents: thank you for their unending love and confidence in my potential. To my colleagues and mentors, who have provided me with tremendous advice and motivation. And to everyone who puts forth endless effort to advance technology in order to create a more secure and safe digital environment. We are grateful for all of your efforts and sacrifices that have helped us reach this goal.

Acknowledgement

I want to start by expressing my gratitude to Almighty God for his kind time, power, and ability, all of which enabled me to finish my final year dissertation. Secondly, I would like to thank the faculty at Bindura University of Science Education for helping me during my stay and sharing their wealth of expertise. My supervisor, Mr. Muzurura, is especially deserving of praise for his unwavering leadership during the course of my research study. My co-supervisor also gave me a lot of guidance and support as I was conducting my research. They were incredibly helpful and supportive in all aspects of my research.

.

Table of contents

CHAPTER 1

1.1 Introduction

Spam email has emerged as a widespread phenomenon that undermines the productivity and security of both individuals and organizations by flooding inboxes with unrequested and, in many cases, harmful messages. As the Statista report says, the total email spam rate equaled 56.97% in the world in March 2021. Given the burden that spam puts on users and the risk it poses for their information systems, the development of accurate spam email detection models enabled by machine learning has captured considerable attention.

A collection of statistical methods for recognising entities, sentiment, parts of speech, and other features of text are used in machine learning for natural language processing (NLP) and text analytics. Artificially intelligent systems might be developed to accurately identify and remove spam emails with ease, utilising some of the machine learning methods. They exploit feature extraction mechanisms, classification models, and optimization algorithms to obtain better results. Some important improvements regarding the employment of machine learning to end the ability of spam emails have been accomplished over recent years. Researchers have applied supervised mechanisms, such as Naive Bayes, Support Vector Machines, and ensemble techniques, to refined processed emails according to their subject, meta information, and other information.

Furthermore, recurrent neural networks and convolutional neural networks, which are some of the deep learning models, have achieved impressive results in pattern identification, hence improving the performance of spam detection. This project draws on the previous literature to develop an all-rounded spam email detection system that incorporates some of the most advanced machine learning techniques. It is anticipated that the associated system will be capable of using an essential and wide dataset of specified emails to accurately detect the email as spam or ham.

1.2. Background study

In recent times, there has been remarkable attention in machine learning models for improving classification accuracy and learning complicated patterns. Sahu and Gupta explored the use of a deep recurrent neural network in spam email detection. According to Zhang et al., an email spam filtering system employing a bagging ensemble of deep convolutional neural networks may be successful. The described works are examples of existing systems for detection. Nevertheless, it remains crucial to improve the existing

solutions by making them more intuitive, adding and tweaking features, and optimizing the models.

This project aims to develop effective spam email detection by enhancing the current work on the same. In this case, the spam email detection system combines machine learning algorithms' capabilities to optimization techniques. The system will depend on a well-balanced dataset with leaked emails, which implies that our system will have the ability to train and validate against multiple possibilities of spam patterns present out there. Feature engineering will also be crucial to help in the extraction of vital information from the corpus data, headers, and metadata to make the system intuitive in detecting spam and ham messages. Optimization models will also apply by ensuring that the AUC & cross-validation are carried out to ascertain low levels of false positives and negatives. Other than the optimization part and the design of the user interface that ensures the users can utilize the system. Machine learning algorithms, and especially the decision trees in machine learning (Gordon, 2020, p. 23), have the potential to mitigate this challenge.

As per the definition of artificial intelligence, decision trees are in the category of supervised machine learning algorithm that can be used for both classification and regression task (Breiman, 1984). As spam emails can be classified based on many things, including sender subject information, keywords in the message body, and also URLs, decision trees are the best choice to deal with such features Alzahrani et al., 2019. They are relatively easy to understand and implement, and they can handle both categorical and numerical data (Quinlan, 1992).

Also, decision trees have been confirmed to be more effective in filtering spam emails, as per a number of studies. For example, a higher accuracy rate of 98.2% by Gunderson (2020) who used a decision tree model to filter spam emails. Also, with an accuracy of 99.1%, Alzahrani et al. (2019) used a hybrid approach involving support vector machines and another decision tree.

Decision trees also offer a number of benefits for spam email filtering in addition to their accuracy. The common thing in spam email datasets is missing data, which makes decision trees more suitable (Alzahrani et al. 2019). The interpretation is easy, and they are also fast to train, which makes them the best fit for real-time spam email filtering systems (Gunderson, 2020).

1.3. Statement of the Problem.

Developing a fit and systematic spam email detection system that can accurately distinguish between spam and not spam emails is a challenge. In organizations and also among individuals, the accelerating prevalence of spam emails constitutes a significant threat to individuals and organizations, leading to security breaches, privacy invasions, and resource waste. The decision tree algorithm is the one employed by the author to furnish an understandable and victorious solution for classifying emails into spam and ham (not spam) classes.

1.4. Research Objectives:

- ✓ To design and implement a spam email detection model using a random forest decision tree model.
- ✓ To train, confirm, and optimize the random forest model for precise categorization.
- ✓ To evaluate the model's performance using confusion metrics, f1 score, accuracy, precision, and recall.

1.5 Research questions

- ✓ Which tools will the author utilise to put the random forest machine learning model into practice?
- ✓ How is the model going to be trained, validated, and optimized during the implementation?
- ✓ What are the metrics to be used for the model's performance evaluation?

1.6 Methodology:

- ✓ Core i5
- ✓ 8 gigabytes of RAM
- ✓ Dataset
- ✓ Random Forest Algorithm
- ✓ Machine Learning
- ✓ Python 3.9

1.7 Research Limitations:

To appreciate and be aware of several in-born restrictions that can have an effect on the scope and findings of the research in the pursuit of advancing spam email detection through machine learning is very important. In addition, imbalanced data, where the number of non-spam emails significantly exceeds spam emails, can introduce obliqueness and compromise the model's performance. The constantly changing nature of spam techniques presents another complication, as the model may not effectively detect emerging tactics not present in the training dataset. Additionally, focusing on a specific email provider or language may limit the model's generalizability across different platforms and languages. Striking a balance between minimizing false positives and false negatives remains a challenge, considering the potential misclassification of legitimate emails as spam or vice versa. User preferences may change dynamically, so interacting more with many users is crucial as the definition of spam email may differ. So for the development of a more effective spam email filter, all these facts must be taken into consideration.

1.8 Research Justifications:

The importance of spam email detection cannot be overstated, as it addresses substantial challenges faced by individuals and organizations, including productivity loss, security risks, and user frustration. Leveraging advancements in machine learning techniques, particularly in natural language processing and pattern recognition, holds promise for developing accurate and efficient spam email detection systems. Recognizing the need

for more robust solutions that can adapt to evolving spam techniques, research should focus on enhancing the efficiency, accuracy, and adaptability of existing detection methods. Addressing imbalanced data challenges in spam email datasets is paramount, requiring innovative approaches to mitigate biases during model training. Furthermore, investigating the generalizability of detection models across diverse email providers and languages is crucial for practical applicability. Adopting a user-centric approach that incorporates user preferences and feedback will contribute to the development of systems that meet user needs effectively. The continuous evolution of spam techniques necessitates ongoing research to detect and adapt to emerging tactics, ensuring the longterm efficacy of spam email detection systems. Finally, exploring optimization techniques and evaluating the performance of machine learning algorithms are essential steps toward enhancing the accuracy and efficiency of spam email detection models for practical implementation.

CHAPTER 2: LITERATURE REVIEW

2.0 Introduction

In the contemporary digital landscape, electronic communication has become an integral part of daily life. However, alongside the convenience and efficiency offered by email communication, there exists a pervasive and disruptive phenomenon: spam emails. Spam emails, or unsolicited messages sent in bulk, pose a significant challenge to individuals, businesses, and organizations alike. With the ever-increasing volume and sophistication of spam, traditional rule-based methods are proving insufficient for effectively mitigating this menace.

This literature review delves into the realm of spam email detection, focusing specifically on the application of machine learning techniques as a promising solution to combat the evolving nature of spam. As the sheer diversity and complexity of spam continue to evolve, the need for adaptive and intelligent systems capable of discerning between legitimate and unwanted emails has become paramount (Rudestam, K.E., and Newton, R.R. et al 1992).

2.1 Evolution of Spam Email Detection

In addition to being essential tools for society, communication media also serve as significant conduits for fraudulent information, including malware, phishing, identity theft, extortion, and fake incentives. Cybercriminals leverage technological advancements to create dangerous scam messages, which they disseminate to millions of people worldwide on a regular basis. Scams can be quickly, simply, and possibly anonymously disseminated online with the help of email services (Ferrara 2019). Spam is increasingly seen as a major threat to email users' online security, dependability, and integrity, despite the fact that they have historically considered it to be little more than time wasters or bothersome unwanted adverts (Gangavarapu et al. 2020).

Furthermore, according to Kaspersky Lab and Cisco Talos, spam emails account for between 50% and 200 billion of all emails sent globally every day, underscoring the severity of the problem. Companies and researchers have been working to create reliable and efficient filters to stop spam emails for the past few decades. In the recent literature, a number of machine learning-based algorithms have demonstrated exceptional performance and accuracy in distinguishing between valid emails (often referred to as "ham") and spam (Saidani et al. 2020). Users still report scams and attacks from spam emails, despite the filter's enhancements and remarkable outcomes. The scams in these emails are exploited by spammers, or unsolicited email senders, who try to evade detection by spam filters.

Spammers constantly devise new methods to circumvent spam filters (Redmiles et al., 2018), preying on their flaws to achieve their objectives. In order to get over textual filters, they alter emails in a number of ways, for as by inserting the spam message inside an image. As such, in this sector, spammers can be considered antagonistic entities. Analysing spammer strategies in emails from a forensic standpoint can expose similar masks in other areas afflicted by hostile actors and cybercrimes (Yu 2015). Scholars Hazarika and Bhowmick (2018) and Wang et al. (2013) have examined spam trends and illustrated how spam content is always changing. Wang et al. (2013) issued a warning, pointing out that spam email was getting smarter and more complex rather than getting less. Researchers need to develop methods for identifying spam emails in a dynamic environment due to the emergence of fake data and the organic expansion of email data over time (Mohammad 2020). The notion that test and training data come from the same distribution is the basis of supervised learning techniques.

To resist security attacks and detect spammer tampering in data, however, robust techniques that deal with adversarial manipulation and dataset shift are needed (Gangavarapu et al. 2020). This work assesses the research with a focus on assessing the more sophisticated spammer methods and the observed dataset alterations in real applications, as opposed to standard analyses of spam emails (Dada et al., 2019; Hazarika and Bhowmick, 2018; Karim et al., 2019). The intention is to emphasise how important these two problems are to creating spam filters that are more dependable. Using information from Bruce Guenter's Spam Archive, researchers conduct a thorough

analysis of the main spammer strategies, their goals, how they have changed over the past ten years, what makes them unique, and how frequently they have appeared in emails. This study clarifies the strategies used by academics to identify and lessen the impact of spammers who manipulate filters. Moreover, the researchers postulate that neglecting to account for the dynamic character of the spam email field might seriously impair any model's efficacy and generalizability.

Pérez-Díaz et al. (2012) employed a similar concept in their previous work, where they suggested an evaluation procedure to foresee potential filter issues and halt performance degradation while the filter was in operation. By adding more records and taking into account time variations, the researchers were able to conduct a more thorough analysis. They therefore evaluate the effects of presuming that training samples are dispersed in a manner akin to that of operational email samples. Term frequency-inverse document frequency (TF-IDF), bag of words (BOW) frequency text encoders, and two machine learning techniques—Naïve Bayes (NB) and Support Vector Machine (SVM)—that are often used as spam email filters are combined to train spam classifiers. To calibrate various spam filters, they use five datasets (SpamAssassin, Ling-Spam, Enron-Spam, TREC07, and CSDMC) spanning the years 2000 to 2010. The author then uses these datasets and data, which covers the years 2000 to 2018, to categorise emails from various scenarios.

2.3 The Problem of Dataset

The problem with dataset shift and the foundational concept of supervised learning is that, despite being unknown, the distribution of test and training data is assumed to remain constant (Hand 2006). However, because there will inevitably be differences, this premise is frequently broken in practical situations. "Distribution shift" and other phrases such as "just drift," "shift in concept," and "concept drift" have also been used to describe this issue (Moreno-Torres et al. 2012; Quionero-Candela et al. 2009). In the past 10 years, a number of authors have written extensively on this topic, including Biggio and Roli (2018), Gama et al. (2014), González-Castro et al. (2013), Liu et al. 2020, Simester et al. 2020, and Webb et al. (2016). Surprisingly, due to their inability to generalise in the deployment context, classification models might experience a considerable decrease in performance during dataset swaps (Kull and Flach 2014).

Moreno-Torres et al. (2012) and Quionero-Candela et al. (2009) not only carried out an extensive examination of the impacts of dataset shift on probability distributions, but also categorised the various forms of dataset shift as covariate shift (distribution shift in features), prior probability shift (shift in classes), and concept shift (shift in the relationship between features and classes), among others. There have been several methods proposed to counteract dataset shift (Gama et al. 2014; Kadthe researchers and Suryawanshi 2015; Yu et al. 2019). Step one is to identify and categorise any shift in the dataset. The second stage involves choosing the best classifier from a set of calibrated classifiers based on the shift that was found. Diverse machine learning algorithms have been employed to filter spam emails, with some exhibiting impressive results (Bhowmick and Hazarika 2018; Dada et al. 2019; Ferrara 2019). Among these methods are neural networks (NN), support vector machines (SVM), random forests (RF), and naïve bayes (NB). A spam filter model with above 99% accuracy has been created by Dedeturk and Akay (2020). On the other hand, their analysis took into account every single spam and ham communication from a sample set of emails received between 2000 and 2010.

It is important to realise that spam email is inherently dynamic due to themes that change over time and the methods spammers employ to evade filters, which results in dataset alterations. This suggests that the previously indicated anti-spam filters will probably fail more frequently than expected in the event of new, unseen cases. Lazy learners have been used in several early approaches to address the dataset shift, or idea drift, present in email spam data (Delany et al. 2005; Fdez-Riverola et al. 2007). Delany et al. (2005) proposed two main approaches: (i) daily addition of misclassified system examples to the case-base, and (ii) regular retraining of the system and reselection of features based on the latest cases. In this field, Fdez-Riverola et al. (2007) detailed two further approaches to employing a lazy learner and keeping an eye on concept drift.

First, representative terms were chosen based on the data in each email using the Relevant Term Identification (RTI) technique. Second, emails that were more pertinent to the actual context implementation were chosen using the Representative Message Selection (RMS) technique. A study by Ruano-Ordas et al. (2018) showed that spam

filtering methods have a number of drawbacks. The impact of several types of concept drift—abrupt, recurrent, progressive, and incremental—on spam filtering was thoroughly examined by the writers. Their investigation turned up a number of issues related to concept drift, including idea drift in ham communications, different kinds of concept drift in spam and ham messages, and themes displaying different kinds of concept drift. They also found that internal factors, including changes to corporate practices, gradual changes in marketing goals, communication problems, language barriers, and external economic conditions, all played a role in concept drift.

More complex dataset shift situations were beyond the capability of Nosrati and Pour's (2011) Dynamic Weighted Majority Concept Drift Detection (DWM-CCD) technique, despite the fact that it handled both abrupt and slow concept drift. In addition, Mohammad (2020) looked at the dynamic aspects of the spam email domain and put up the hypothesis that cyclical idea drift can happen in this sector as a result of the traits of spam emails appearing and disappearing often. Based on the ensemble learning technique, the study tried to develop a lifetime classification model using previous spamming approaches and other catastrophic forgetting difficulties. According to Baena-García et al. (2006), their approach used the Early Drift Detection Method (EDDM) to assess the validity of concept drift. In the event that differences in the class distribution were identified, the spam filter was adjusted via an adjustable dataset partitioning ensemble-based lifelong classification (ELCADP). ELCADP is a big step towards more reliable spam filtering, even though its performance hasn't been examined with virtual concept drift—a situation in which input features stay the same but a new class value may emerge.

2.3 Machine Learning Algorithms applied on Spam Email Filtering

Adversarial learning in machine learning has shown effective results, with algorithms being applied in various sectors (Riesco et al. 2019). However, some areas, such as spam detection (Dedeturk and Akay 2020) and phishing detection (Sanchez-Paniagua et al. 2021), require continuous model updates due to adversarial threats. Organizations and researchers need to approach each subject individually because of their unique

characteristics. For example, phishing differs from spam by potentially mimicking branch logos, requesting sensitive information, or creating a sense of urgency for the recipients.

Through deliberate manipulation of data, adversaries exploit weaknesses associated with dataset shift to deceive classifiers. As stated by Dalvi et al. (2004), adversaries introduce malicious data to induce classifier failure. To categorise hostile attacks and develop defences, Barreno et al. (2006) developed a taxonomy of adversarial assaults applying three criteria. An in-depth examination of adversarial characteristics, assault taxonomy, and adversarial capabilities was provided by Huang et al. (2011), building upon the study conducted by Barreno et al.

Two primary approaches have been used to study adversarial classification. Classifier stability against adversarial assaults is evaluated using the first method (Goodfellow et al. 2015; Laskov and Kloft 2009; Lu et al. 2020; Nelson et al. 2011; Paudice et al. 2018). A metric for assessing classifier stability and resilience to hostile training data contamination was presented by Nelson et al. (2011). Biggio et al. (2009) introduced frameworks for security analysis and algorithmic assessment of attack-simulation classes. Laskov and Kloft followed suit in 2013. In order to determine the weaknesses of NN classifiers, Goodfellow et al. (2015) looked at adversarial data samples, concentrating on non-linearity and overfitting problems. Pre-training algorithms were created by Paudice et al. (2018) to lessen the effects of poisoning attacks. Significant flaws in quantum machine learning algorithms in hostile situations were found by Lu et al. (2020).

The assessment of assault efficiency has been the subject of several studies (Apruzzese et al. 2019; Papernot et al. 2015a, 2017; Shi et al. 2019). For example, Papernot et al. (2015a) developed an algorithm to produce adversarial samples based on a complete understanding of the input-output mapping and formalised the universe of attacks against deep neural networks. Papernot et al. (2017) demonstrated the feasibility of circumventing security protocols by a successful assault using a black box adversary on a real deep learning application. Apruzzese et al. (2019) investigated the possible harm caused by a cyber-attack employing evasion and poisoning tactics on a cyberdetector,

highlighting the necessity of more robust machine learning algorithms in cybersecurity. An efficient poisoning attack against spectrum recognition applications was assessed by Shi et al. (2019).

In adversarial categorization, the relationship between adversaries and defenders is dynamic. Research on machine learning security in adversarial situations often concentrates on spam email detection when the adversary figures are spammers (Chen et al. 2018). Spammers use misspellings or real phrases in their communications to try to fool classifiers (Biggio and Roli 2018). According to Xiao et al. (2018), harmful information may be intentionally inserted to spam emails in order to mess with the data used to train classifiers and disrupt the algorithms' normal operation. Nelson et al. (2008) used a dictionary attack to contaminate a tiny percentage of the training set emails, therefore exposing flaws in the Spam Bayes filter.

They found that it would be difficult to repel an attack with more knowledge, despite the fact that they were able to examine two protections against dictionary attacks. Dasgupta and Collins (2019) and Rota Bulo et al. (2017), for example, developed game theory-based models that simulated adversary assaults and were evaluated using spam email datasets to develop more secure and reliable machine learning techniques. In place of game theory, Naveiro et al. (2019) evaluated an alternative method using spam email datasets and adversarial risk analysis.

2.5 Random Forest Algorithm

2.5.1 Decision Trees

Supervised classification is embodied in the machine learning notion of decision trees. Decision trees are made up of nodes, branches, and leaves, and their structure is modelled after that of regular trees. This arrangement is similar to a tree, in which nodes are joined by branches that eventually grow to leaves, while the root signifies the beginning.

Decision tree nodes are shown as circles, and the segments that connect these nodes are called branches. Building a decision tree usually starts at the root and moves downhill;

this process is frequently shown as going from left to right. We call the beginning point of the tree the "root node." The "leaf" node, on the other hand, refers to the conclusion or endpoint of a branch.

Two or more branches may grow from any internal node, which is not a leaf node. In a decision tree, nodes stand for particular attributes, and branches represent value ranges related to those attributes. For the set of values associated with the specified feature, these ranges serve as partition points.

A Decision Tree's hierarchical structure is shown in Figure 1. The sequential nature of decision-making within the tree as it moves from the root node to the leaf nodes is better illustrated by this graphic representation. Decision trees are effective tools for categorization problems because they offer models for decision-making in a variety of contexts that are easy to understand and apply.

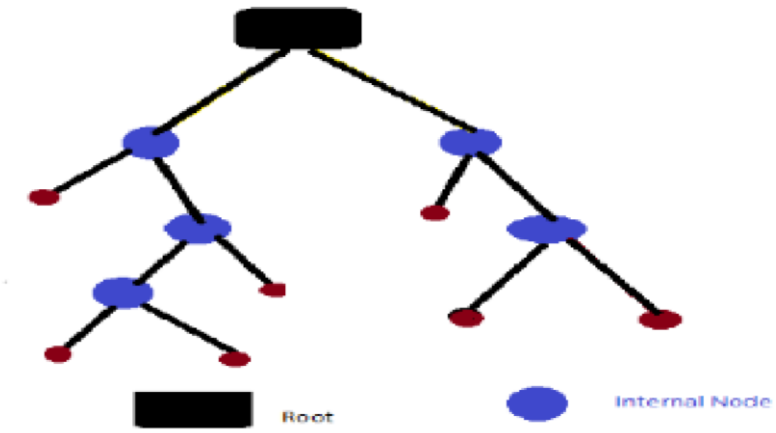


Figure 1: Tree structure

2.5.2 Random Trees

In the context of machine learning, Random Trees offer a distinctive method for building decision trees. Random trees are created randomly from a list of potential trees, adding a degree of stochasticity to the process, in contrast to standard decision trees where features are meticulously selected at each node based on specific criteria.

The phrase "random" infers that there is an equal chance of sampling each tree "in the set of trees," leading to a "uniform" distribution. This implies that every tree in the set has an identical chance of being selected when a random tree is being constructed. Because of the diversity that this randomness brings to the trees, the ensemble model becomes strong and adaptable.

The inclusion of randomness in feature selection at each node is a prominent feature of random trees. Rather than choosing the best feature based on certain metrics, a subset of K random features is considered at each node. Because of their inherent randomness, the ensemble's trees concentrate on various attributes, adding to the model's overall strength and diversity.

The ability of random trees to generate accurate models when combined in large sets and their efficiency in doing so have generated a lot of interest in them recently. The benefits and uses of random trees in a range of machine learning tasks have been thoroughly studied. Random trees are especially well-suited to handle complex patterns and improve the predictive performance of models because of their ensemble nature.

In conclusion, random trees add a degree of unpredictability to the building process, producing a variety of precise models. The set's uniform distribution of trees guarantees that every possible tree receives the same amount of attention, which adds to the ensemble's overall robustness. Random trees are a useful tool in the quickly evolving field of machine learning research because of their efficacy and efficiency.

2.6 Related previous researches

Spam Email Detection Using Machine Learning and Deep Learning Techniques (Pooja Malhotra, Sanjay Malik, et al 2022)

This study uses natural language processing (NLP) techniques to separate spam from ham news in the spam email dataset. In order to combine models from Long ShortTerm Memory (LSTM) and Bidirectional LSTM (Bi-LSTM), the study uses a Dense classifier

Sequential Neural Network approach. The objective is to compare their effectiveness and outcomes in identifying spam emails.

The models' efficacy is assessed using important measures like recall, accuracy, and F1-score. The results show that the dataset's overall accuracy is improved when BiLSTM classification is used. The entire study is carried out in Python, with smooth execution and analysis made possible by a Jupyter notebook.

This work advances the rapidly developing fields of deep learning and natural language processing by demonstrating how these methods may be used to solve problems in the real world, such spam email identification. By contrasting various models, one can gain insight into the advantages and disadvantages of each strategy, enabling better decisionmaking for creating reliable and precise spam detection systems.

Machine Learning Algorithms for Email Spam Detection (Nikhil Kumar, Sanket Sonowal, Nishant, et al 2020)

As more people use the internet, email spam has become a significant issue in today's digital world. Spam emails are now more than just an annoyance; they can be used for shady and unlawful activities like phishing and fraud. The security of our systems is seriously threatened by malicious links included in spam emails, which allow for unauthorized access and possible damage.

Spammers frequently use the creation of fictitious email addresses and profiles to enable them to pose as real people. These dishonest strategies work especially well on people who do not know enough about these kinds of fraudulent operations. Consequently, it's critical to identify and delete spam emails that have criminal intent. This project aims to address this critical issue by leveraging machine learning techniques to identify and classify spam emails. The paper will delve into various machine learning algorithms, applying them to datasets to discern the most effective approach. The emphasis is on selecting an algorithm with the highest precision and accuracy in spam email detection.

The project's objective is to give email users access to a dependable system that uses machine learning to identify phishing and potentially harmful emails automatically. The selected algorithm will go through a thorough evaluation procedure that takes into

account parameters like accuracy and precision to make sure it can effectively identify between emails that are spam and those that are real.

In the end, the project helps to improve email security and shield users from the harmful actions linked to spam. Using cutting-edge machine learning algorithms is critical to staying ahead of fraudulent practices and protecting users' digital experiences as email spam continues to grow in sophistication.

Machine Learning Techniques for Spam Detection in Email and IoT Platforms (Naeem Ahmed, Rashid Amin, Hamza Aldabbas, Deepika Koundal, Bader Alouffi and Tariq Shah, et al 2021)

The researcher argued that emails are now a necessary component of many different types of correspondence in the modern world, from business to education. Emails can be divided into two categories: legitimate emails (ham) and unsolicited or garbage emails (spam). Spam emails are becoming more and more common, which puts consumers at the risk of time loss, computer resource consumption, and possibly even identity theft. Spam identification and filtration are becoming major difficulties for email and Internet of Things (IoT) service providers due to the constant increase in the proportion of spam emails.

Of all the methods developed for preventing and detecting spam, email filtering stands out as a key component. The project's goal is to provide email users with a trustworthy system that automatically detects phishing and potentially dangerous emails using machine learning. The surveyed techniques are methodically categorised into relevant groups to offer context and clarity.

In addition, the study compares various methods in detail and assesses how well they perform using measures like accuracy, precision, and recall. This comparative analysis elucidates the pros and cons of each technique, assisting researchers and practitioners in selecting the most appropriate approach for their specific requirements.

A comprehensive analysis of the survey and comparison results is presented in the study's conclusion, along with potential avenues for future research in the spam filtering field. This study seeks to support ongoing efforts to strengthen email and IoT systems against the constant barrage of spam by navigating the terrain of machine learning techniques, thereby guaranteeing a safe and effective communication environment.

Spam Email Detection Using Deep Learning Techniques (Department of Computer Information Systems, Jordan University of Science and Technology, 3030, Irbid 22110, Jordan, et al 2021)

This work uses the pre-trained transformer model BERT (Bidirectional Encoder Representations from Transformers) to investigate the effectiveness of word embedding in the classification of spam emails. BERT, a well-known tool for extracting contextual information through attention layers, has been specially trained to distinguish between emails marked as spam and those marked as unspammed (HAM). The output of this method is compared with a baseline Deep Neural Network (DNN) model with a Bidirectional Long Short-Term Memory (BiLSTM) layer and two stacked Dense layers. Additionally, a comparison is made between the outcomes and conventional classifiers such as NB (Naive Bayes) and k-NN (k-nearest neighbours).

The model provides a trustworthy evaluation against never-before-seen data since it is trained and tested on two public datasets. The efficacy of the proposed method is demonstrated by its highest accuracy of 98.67% and F1 score of 98.66%. These findings highlight the possibility for reliable and precise spam email detection by utilising cutting-edge deep learning models, particularly BERT. The suggested approach's advantages are better understood thanks to the comparison study with traditional classifiers and a baseline DNN model, which highlights the approach's resilience against unknown data and possible real-world applications in fighting spam emails.

An examination of e-mail spam detection through the use of a novel hybrid bagging technique based on machine learning (Alanazi Rayan, et al 2022)

The best elements of two machine learning techniques—random forest and J48 (decision tree)—are combined in this study to propose a unique machine learning-based hybrid bagging method for email spam identification. Emails are divided into spam and ham categories using the suggested framework. The database is split up into several sets in order to preprocess the data, and each set is then used as an input for an appropriate machine learning technique. During the preprocessing phase, tokenization, stemming, and stop word removal are also used. Correlation feature selection (CFS) is used in the study to separate important characteristics from the preprocessed data.

Numerous measures, such as true-negative rates, accuracy, recall, precision, falsepositive rate, f-measure, and false-negative rate, are used to assess how effective the approach being presented is. Three tests are compared to show the superiority of the hybrid bagged model-based spam mail detection (SMD) technique. The outcomes demonstrate that this method can distinguish between legitimate emails and spam with an amazing 98 percent accuracy rate. With a high level of email classification accuracy, this research presents a potential way forward for the ongoing efforts to enhance spam detection systems.

Email spam detection using machine learning (Pradyumna Nalawade, Shubham Kalbhor, Sanket Bhandwalkar, Aniket Nimbalkar, Pratik Sonawane, et al 2023)

This project uses machine learning techniques to identify spam emails because it recognises the important necessity to detect and prevent such fraudulent activity. The Naïve Bayes algorithm, a well-liked machine learning method noted for its effectiveness and simplicity of use in classification problems, is the subject of this paper. The method determines whether an email is classified as spam or ham (nonspam) when applied to a dataset.

This study represents a significant advancement in email security and consumer protection against the harmful effects of spam. By utilising machine learning methods, particularly Naïve Bayes, the project aims to develop a robust system capable of identifying harmful and fraudulent emails independently. The program's efficacy and

potential for real-world spam detection will be revealed by the classification results obtained from applying the algorithm to the dataset.

Email Spam Detection Using MachineLearning (Mrs. Anitha Reddy, Kanthala Harivardhan Reddy, A. Abhishek, Myana Manish, G. Viswa Sai Dattu, Noor Mohammad Ansari, et al 2023)

This project provides a way to use machine learning (ML) and natural language processing (NLP) techniques to effectively categorise email exchanges as spam or real. The major goal is to develop a robust and efficient spam classifier that can distinguish and reliably detect spam emails from legitimate ones.

A significant portion of the email messages in the dataset used for this study have been categorised as either spam or ham (non-spam). The text input will be preprocessed using NLP techniques including tokenization, stop word removal, stemming, and feature extraction in order to extract relevant features. The effectiveness of many machine learning algorithms, including Random Forests, Support Vector Machines (SVMs), and Naive Bayes, will then be evaluated in the study. This assessment will help determine which spam classification model works best.

Hyperparameter tweaking will enhance the model's performance even further. F1-score, precision, recall, and other crucial assessment metrics will be used to assess the classifier's accuracy. An improved spam classifier model that is prepared for integration into email systems to independently filter spam emails will be one of the project's final deliverables. Enhanced email security and general productivity are the two main objectives of this integration.

The project improves NLP and ML methods especially made for email spam classification, which helps the field overall, in addition to its immediate application. By addressing the present problems caused by the rise in spam emails, this research contributes to the ongoing efforts to fortify digital communication networks against potential threats and annoyances.

Machine-Learning-Based Spam Mail Detector(Panem Charanarur,

Harch Jain, G. Srinivasa Rao, Debabrata Samantha, Sandeep Singh Sengar and Chaminda Thushara Hewage, et al 2023)
This study looks into how cookies, caches, flash artefacts, and super cookies are analysed in Windows 10 browsers, Firefox, and Internet Explorer.

Data was gathered using Internet Explorer, Firefox, and Google operating in a Windows 10 environment. The report claims that browsers save data about user behaviour on the host computer's hard drive, potentially creating security and privacy risks. This research will be very helpful to digital forensics specialists, law enforcement personnel, and computer forensics researchers since it clarifies the duration of user data persistence across various browsers.

The study's methodology uses Python and relevant tools including pandas, Numpy, Matplotlib, scikit-learn, and flask to make the investigation easier. The study's findings and analysis demonstrate the effectiveness of the KN (K-Nearest Neighbours) and NB (Naive Bayes) algorithms, which outperform other algorithms in terms of accuracy and precision. This result highlights how these algorithms may be used to create reliable solutions for browser data security and spam email detection. The study not only advances the subject of computer forensics but also has important ramifications for improving email security protocols and optimising the privacy policies of widely used web browsers.

2.7 Research gap

The research gap in applying the Random Forest machine learning algorithm for spam email classification centers on the limited exploration and comparative analysis of this algorithm within the context of spam detection. Existing literature may lack comprehensive evaluations of Random Forest's performance against other commonly employed algorithms, hindering a nuanced understanding of its strengths and weaknesses. Additionally, the real-world applicability and integration challenges of Random Forest in practical spam filtering systems may not be thoroughly addressed. The dynamic nature of spam techniques and the need for optimization methods specific to Random Forest might be underexplored. Bridging these gaps could enhance our

understanding of the algorithm's effectiveness, optimize its performance, and ensure its adaptability to evolving spam patterns.

2.8 Conclusion

The literature review concludes by highlighting the progress made in machine learningbased spam email detection as well as the ongoing difficulties and the demand for flexible and reliable solutions. The emphasis on adversarial techniques, dataset shift, and temporal evolution offers a detailed understanding of the dynamic spam detection landscape and supports the continuous attempts to improve spam filters' efficacy over time.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Introduction

The researcher outlines the techniques and instruments used in this crucial Chapter 3 in order to achieve the stated research goals. Drawing upon insights garnered from the preceding literature review in Chapter 2, the researcher strategically devises methods to construct a viable solution, navigating through various alternatives to attain the desired research outcomes. Synthesizing the wealth of information assimilated from the literature, the chapter serves as a roadmap for implementing the proposed solution, providing a clear and informed foundation for subsequent analyses. By leveraging the knowledge accumulated in the literature review, the researcher crafts a robust framework that not only addresses the identified research gaps but also enables informed decision-making in selecting and executing strategies. This chapter, therefore, stands as a critical juncture where theoretical understanding converges with practical application, setting the stage for the empirical exploration of the research objectives. In this chapter, the researcher delve into our research core, outlining the methodology for harnessing the Random Forest algorithm in spam email classification. The researcher navigate through meticulous data preparation, extracting relevant features from emails, transforming raw text into signals for our model. The Random Forest is then trained on labeled spam and legitimate emails, evaluated for its discernment in digital landscapes. The author dissect the algorithm's patterns, unveiling key features for efficient spam detection. Beyond showcasing Random Forest effectiveness, our goal is to gain insights into the dynamic realm of spam, equipping us with powerful tools to combat evolving challenges.

3.2 Research Design

The research design comprises the all-encompassing strategy used to effectively combine different study components in a coherent and logical way, guaranteeing the skillful solution of the research problem. As stressed by Boru (2018), it acts as a guide

for the methodical gathering, measuring, and analysis of data. At this juncture, the primary focus is on crafting a robust, dependable, operational, and efficient prototype in alignment with the research objectives. The pivotal objective is to guarantee the development of a prototype that is both stable and meets the specified requirements of the research goals.

3.2.1 Introduction to Experimental Design

The primary goal of the experimental design is to assess the system's performance using a variety of measures, including F1 score, accuracy, precision, and recall. By putting the system through several tests to see how well it detects spam emails, the design responds to the research questions.

3.2.2 Research Questions and Hypotheses

- Research Question: What is the Random Forest algorithm's effectiveness in identifying spam emails? What are the system's F1 score, recall, accuracy, and precision for identifying whether an email is spam?

- Hypotheses: The system will correctly identify emails with high recall, F1 score, and precision by utilising the Random Forest method.

3.2.3 Variables

- Independent Variables (IVs): Types of emails (spam, non-spam).
- Dependent Variables (DVs): System's classification accuracy, precision, recall, F1 score.

3.2.4 Experimental Groups

The experiment involves testing the system with different types of emails:

Group 1: Phishing/Spam Emails

Definition of Phishing Emails: Phishing emails are phoney correspondences intended to trick recipients into disclosing credit card numbers, passwords, or other private information. These emails typically have malicious links or attachments that compromise security, even though they seem to be from reliable sources like banks, social networking platforms, or other organisations.

Common Characteristics of Phishing Emails:

- **Suspicious Sender:** The email may come from an unusual or spoofed email address that mimics a legitimate one.

- **Urgent Language:** In order to compel quick action, the email frequently employs a sense of urgency or terror (e.g., "Your account will be suspended unless you act now!").

- **Suspicious Links or Attachments:** The email may include links to fake websites designed to capture personal information or attachments containing malware.

- **Requests for Personal Information:** Sensitive information, including credit card numbers, Social Security numbers, or login credentials, may be explicitly requested in the email.

- **Generic Greetings:** Generic greetings such as "Dear Customer" are frequently used in phishing emails rather than addressing the recipient by name.

Group 2: Non-Phishing/Ham Emails

Definition of Non-Phishing/Ham Emails: Non-phishing emails, also known as "ham" emails, are legitimate messages that do not pose any security threat. These are genuine communications from trusted sources, containing no malicious content and requiring no caution for personal information safety.

Common Characteristics of Non-Phishing Emails:

- **Recognizable Sender:** The sender of the email is a well-known and reliable individual, such as a friend, coworker, or respected business that the receiver is acquainted with.
- **Relevant Content:** The content is relevant and expected, often pertaining to ongoing conversations or transactions.

- **No Suspicious Requests:** The email does not ask for sensitive personal information or prompt any immediate, unusual actions.

- **Proper Language and Formatting:** Non-phishing emails typically have proper grammar, spelling, and formatting, consistent with professional or personal communication standards.

- **Safe Links and Attachments:** If the email contains links or attachments, they are from trusted sources and lead to legitimate websites or documents.

3.2.5 Materials and Instruments

- Materials: Email dataset with labels designating it as spam or not.
- Instruments: Random Forest algorithm implemented in a Python environment, confusion matrix for performance evaluation.

3.2.6 Procedure

- Dataset Preparation: Collect and preprocess a dataset of emails.
- Feature Extraction: Extract relevant features from emails for classification.
- Model Training: Train the Random Forest model using the training dataset.
- System Testing: Use both white-box and black-box testing techniques to test the system.
    - Black-Box Testing: Assess system performance without understanding internal code architecture.
    - Examine internal structures, designs, and code in white-box testing.

- Performance Evaluation: Utilising the confusion matrix, determine the F1 score, accuracy, precision, and recall.

3.2.7 Data Collection Methods

- Black-Box Testing: Test different email inputs to verify expected outputs.
- White-Box Testing: Inspect internal code to ensure correct implementation of the Random Forest algorithm.
- Confusion Matrix: Gather information about false positives, false negatives, true positives, and true negatives.

3.2.8 Data Analysis Plan

- Accuracy Calculation: $Accuracy = \frac{TP+TN}{TP+TN+FP+FN} * 100$
- Precision Calculation: $Precision = \frac{TP}{TP+FP} \times 100$
- Recall Calculation: $Recall = \frac{TP}{TP+FN} \times 100$
- F1 Score Calculation: $F1 = \frac{2 \times TP}{2 \times TP+FP+FN}$

3.2.9 Limitations of the Experimental Design

- Sample Size: Limited by the availability of labeled email data.
- Generalizability: Results may vary with different datasets.
- Bias: Potential bias in email content selection.

3.2.10 Summary

This section details the experimental design aimed at evaluating the spam email detection system. By employing a structured approach, the study ensures accurate and reliable assessment of the system's performance.

3.3 Requiments Analysis

The Requirements Analysis procedure involves breaking down end-user needs, typically identified operationally at the system level during the implementation of the Stakeholder

Requirements Definition process (Apvrille, 2013). The scholar assessed both functional and non-functional requirements as part of this evaluation. During this phase, a thorough examination and decomposition of the specified needs are conducted to understand the intricacies of what is demanded by end-users. This process is critical in ensuring that the ensuing system design aligns closely with the stipulated requirements, encompassing both functional features and non-functional aspects outlined by Apvrille in 2013.

3.3.1 Function Requirements

In the realm of software engineering, a functional requirement outlines the characteristics of a system or its constituent parts, elucidating the operations that the software needs to execute. This specification details the specific functions that the software must carry out, encompassing aspects such as inputs, behavior, and outputs (Fulton & Vandermolen, 2017). A function, within this context, refers to various activities such as calculations, data manipulations, business processes, user interactions, or any distinctive functionality that delineates the specific tasks the system is expected to perform. This definition encapsulates the essential functionalities that contribute to the overall operational scope of the software.

The following functional requirements must be fulfilled by the suggested system:

- The system must be able to identify and categorise emails as either spam, phishing, or ham emails.
- The system should be able to classify any type of email and also be trainable to new detection methods

3.3.2 Non-Function Requirements

Nonfunctional requirements, often abbreviated as NFRs, consist of specifications that articulate both the capabilities and limitations of the system's operation. Essentially, these requirements delineate the performance aspects of the system, encompassing factors such as speed, security, reliability, data integrity, and other operational characteristics.

The following non-functional requirements must be met by the proposed system:

- The system must be able to detect any kind of email.

- The system should have a high accuracy rate when classifying emails

3.4 Tools Used(Hardware And Software)

- Python 3.11
- VS code
- Pandas
- Pickle
- Numpy
- Scikit-learn
- Core i5 HP laptop

3.5 System Development

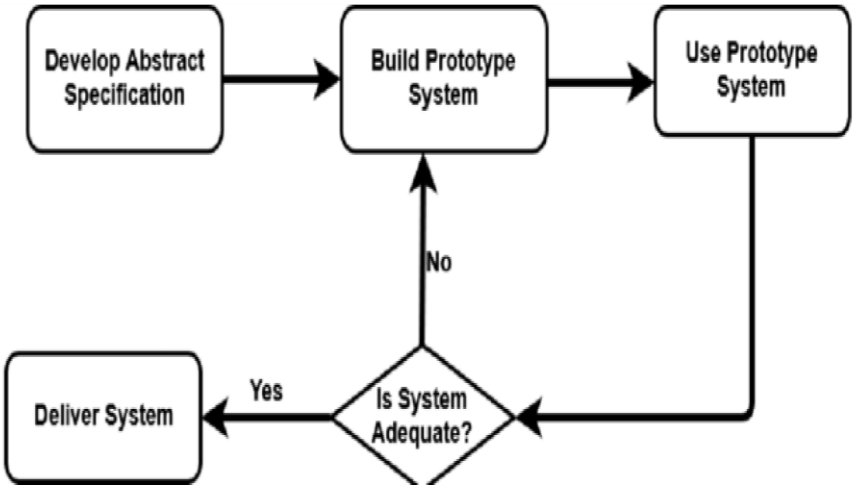
System development entails offering an overview of the creation process aimed at producing results. This section details the software tools and models employed during development, clarifying the methodology and approach used to achieve the intended outcome.

3.5.1 System Development Tools

Python is the programming language that the researcher uses to create an application that will be tested. This programme uses the scikit-learn, NLTK, NumPy, and Pandas libraries to evaluate results. By allowing users to enter email addresses into the programme, these libraries are able to determine whether or not an email is spam.

3.5.2 Build Methodology **Evolutionary Prototyping**

Through iterative refinement based on user feedback, the first prototype is eventually adopted through evolutionary prototyping. This strategy is more efficient than Rapid Throwaway Prototyping, saving time and effort in the development process.



3.5.3 Prototype

A prototype is an early, basic model or version of a product or system created to assess and showcase its ideas, capabilities, and attributes. It acts as a tangible example, enabling stakeholders to see and engage with the proposed design, fostering feedback and improvements prior to the complete development of the ultimate product.

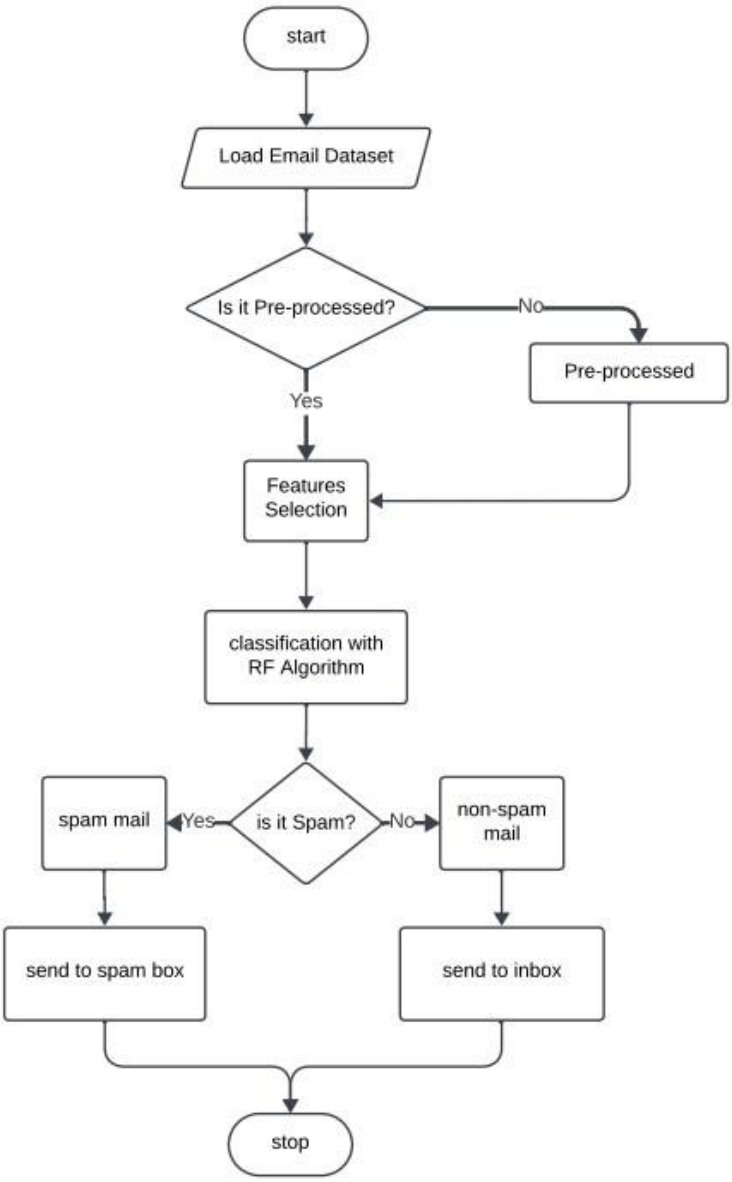
3.6 Technology Used

- Python 3.11
- VS code
- Pandas, Nltk , Numpy, Scikit-learn

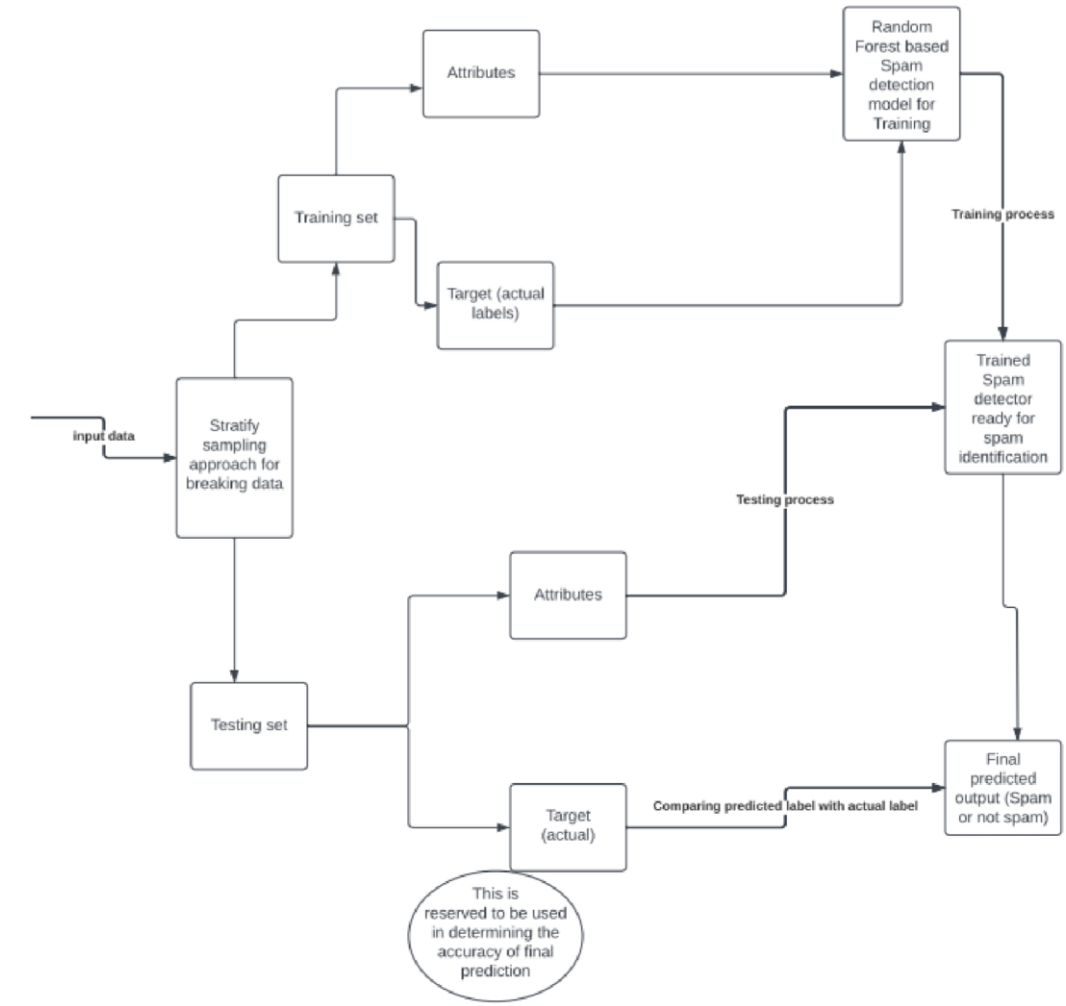
3.7 Algorithm Used

- Random Forest

3.8 Proposed System Flow Chart



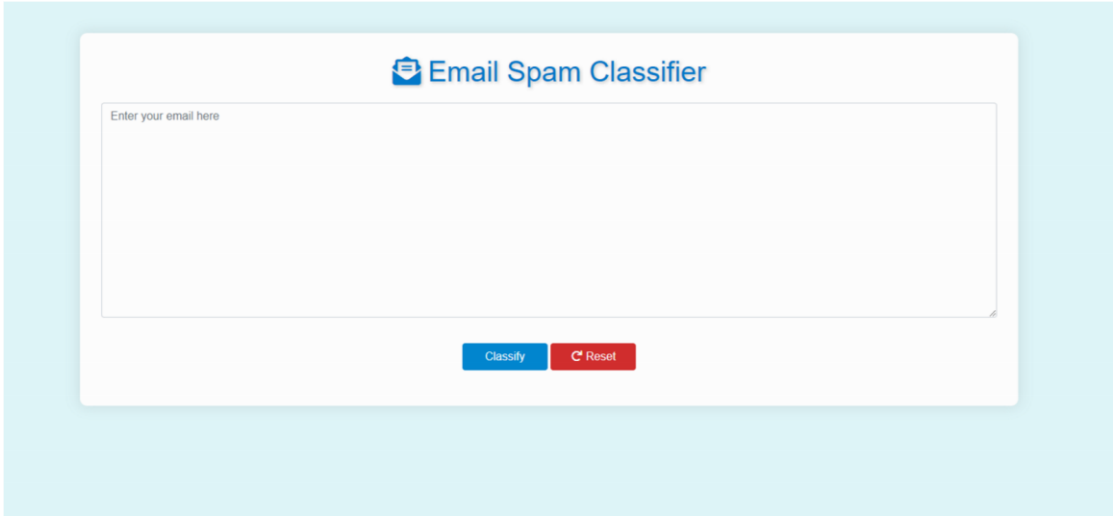
3.9 Dfd

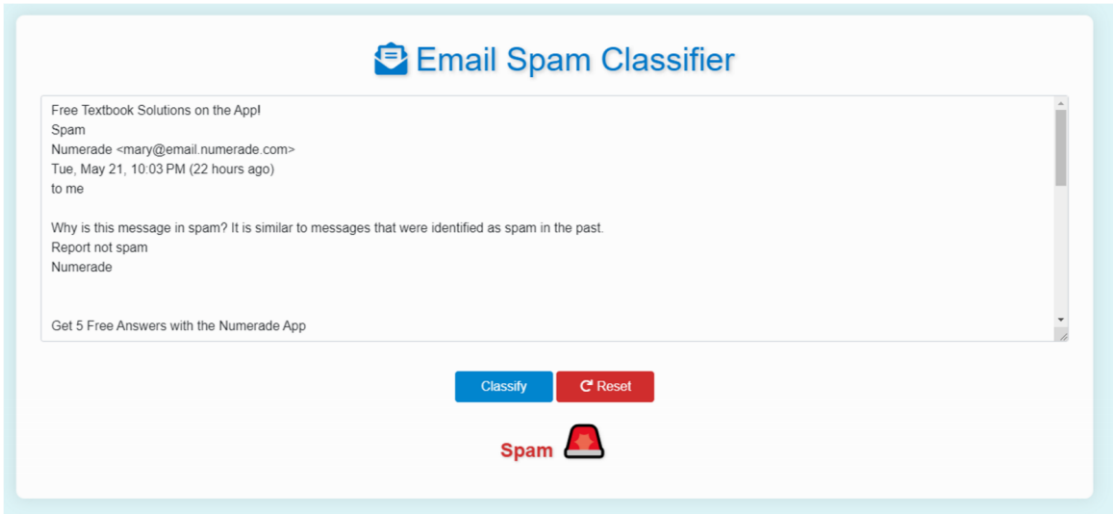


3.10 Random Forest Algorithm

```
In [13]: # Vectorize text data
cv = CountVectorizer(stop_words='english', max_features=10000)
X = cv.fit_transform(df['text']).toarray()
y = df["spam"].values

In [14]: # Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

In [15]: # Using a Random Forest model for our classifier
rf_clf = RandomForestClassifier(n_estimators=100, random_state=42)
rf_clf.fit(X_train, y_train)

Out[15]: RandomForestClassifier
RandomForestClassifier(random_state=42)
```

3.11 General Overview Of Spam Email Detection Using Random Forest Algorithm





3.12 Implementation

app.py
```python
from flask import Flask, render_template, request, jsonify
from utils import model_predict
app = Flask(__name__)


@app.route("/")
def home():
    print("I am here")
    return render_template("index_.html")


@app.route('/predict', methods=['POST'])
def predict():
    email = request.form.get('content')
    prediction = model_predict(email)
    return render_template("index_.html", prediction=prediction, email=email)

# Create an API endpoint
@app.route('/api/predict', methods=['POST'])
def predict_api():
    data = request.get_json(force=True)  # Get data posted as a json
    email = data['content']
    prediction = model_predict(email)
    return jsonify({'prediction': prediction, 'email': email})  # Return prediction

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=8080, debug=True)

```

utils.py
```python
import pickle

cv = pickle.load(open("model/cv.pkl", 'rb'))
clf = pickle.load(open("model/clf.pkl", 'rb'))

def model_predict(email):
    if email == "":
        return ""
    tokenized_email = cv.transform([email]) # X
    prediction = clf.predict(tokenized_email)
    # If the email is spam prediction should be 1
    prediction = 1 if prediction == 1 else -1
    return prediction
```

3.14 Summary of how the system works

The spam detection software uses a methodical approach to recognise and categorise spam emails. It does this by using the Random Forest algorithm, which is deployed by Flask. First, a preprocessed dataset containing both spam and non-spam emails is gathered and transformed from unstructured text data into a format that is appropriate for machine learning. Next, pertinent elements like word frequencies and particular linguistic traits are taken out of the emails. A Random Forest classifier is trained on the training set of the dataset, which is separated into testing and training sets. The Random Forest algorithm aggregates the predictions of several decision trees and is well-known for its capacity to manage large numbers of features and reduce overfitting. Following model training, its performance is evaluated using the testing set, employing metrics like accuracy and precision.

In parallel, a Flask web application is developed to serve as the user interface for the spam detection service. This involves setting up routes, templates, and static files within the Flask application. The trained Random Forest model is integrated into the Flask app by serializing it into a format suitable for easy loading. The application includes a route to handle incoming email content for prediction. Users input email content through the web interface, triggering the Flask application to process the input and utilize the pretrained Random Forest model for classification. The results, indicating whether the email is spam or not, are then displayed to the user on the web interface. The final steps involve deploying the Flask application on a web server or cloud platform, making it accessible online, and implementing monitoring mechanisms for performance tracking. Regular model updates with new data ensure the continued effectiveness of the spam detection application against evolving spam patterns.

CHAPTER 4: RESULTS ANALYSIS

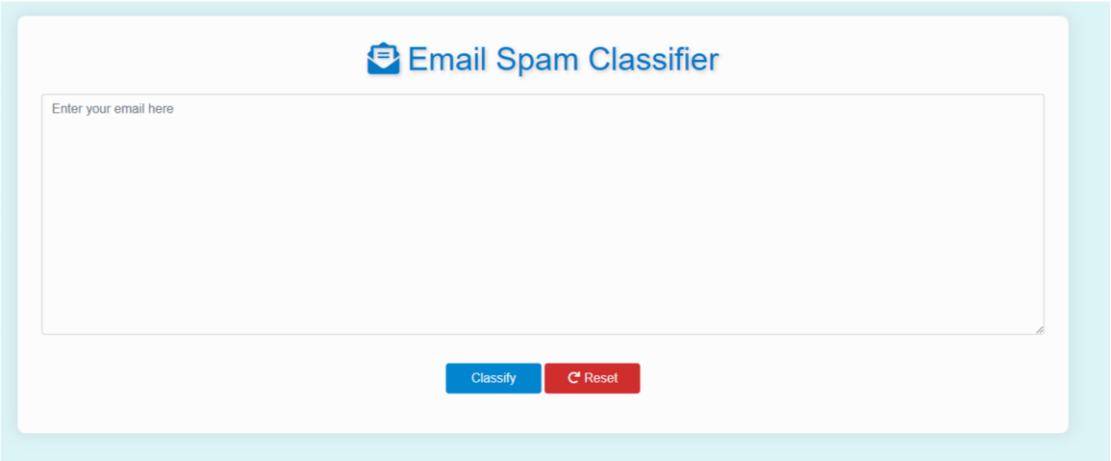
4.0 Introduction

After the system is finished, it is imperative to assess the efficacy of the offered solution. The final solution's efficiency and efficacy were evaluated using metrics like accuracy,

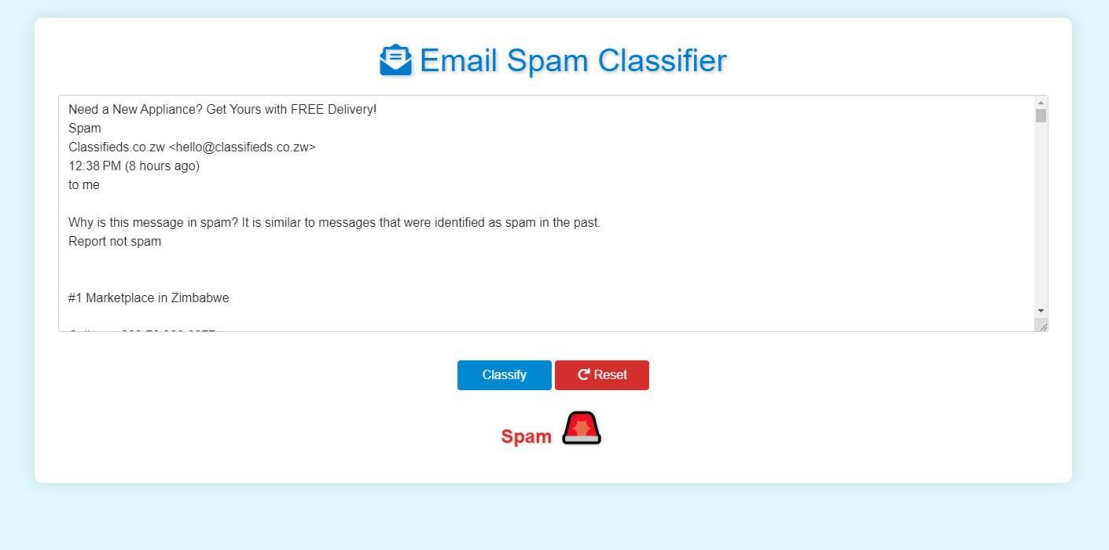
performance, and response time. The information obtained in the preceding chapter was examined in order to derive significant findings. The developed system's behavior was also investigated in various scenarios. This chapter is dedicated to presenting the study's findings, analyses, interpretations, and discussions, which are essential components of the research process.

4.1 System Testing for Email Spam Classifier

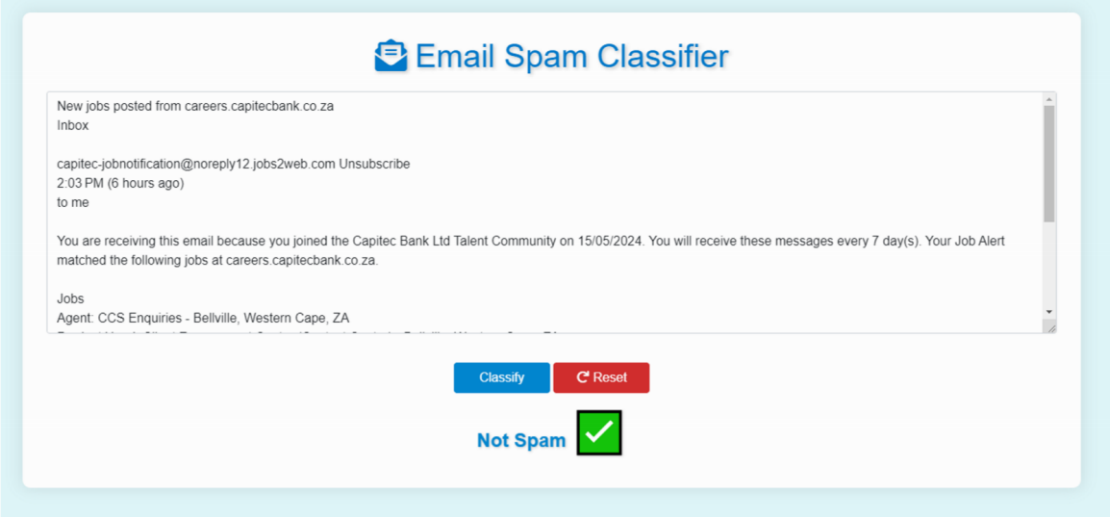
The author managed to develop the Application of Random Forest Machine Learning Algorithm for Spam Email Classification, which was developed with Python 3.11 using tools and libraries such as Scikit-learn, Pandas, NumPy, and Pickle for data analysis and visualization. The implementation was carried out on a local server using Jupyter Notebook for development and testing.



Running the system



First test Phishing/spam email message



Second test Nonphishing/ham email message

The system was tested with a variety of mails which are scam, phishing, spoofing, and malware and also nonspam mails and the author discovered that these types have a significant impact on the system.

4.2 To refine, validate, and train the random forest model for accurate classification

To ensure accurate classification and training of the random forest model, the author categorized a dataset of emails into two groups: non-phishing/ham and phishing/spam. To achieve high accuracy and resilience in email categorization, the

Model was trained on labelled data, fine-tuned using hyper parameter optimization approaches like grid search or randomized search, then tested by cross-validation.

```
In [13]: # Vectorize text data
         cv = CountVectorizer(stop_words='english', max_features=10000)
         X = cv.fit_transform(df['text']).toarray()
         y = df["spam"].values

In [14]: # Split data into training and test sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

In [15]: # Using a Random Forest model for our classifier
         rf_clf = RandomForestClassifier(n_estimators=100, random_state=42)
         rf_clf.fit(X_train, y_train)

Out[15]: RandomForestClassifier
         RandomForestClassifier(random_state=42)
```

Model training

```
In [15]: # Predicting and evaluating the model
         y_pred = rf_clf.predict(X_test)
         accuracy = accuracy_score(y_test, y_pred)
         report = classification_report(y_test, y_pred)

         print("Accuracy:", accuracy)
         print("Classification Report:\n", report)

         Accuracy: 0.9912203687445127
         Classification Report:
                       precision    recall  f1-score   support

                    0       0.99      1.00      0.99       843
                    1       0.99      0.97      0.98       296

             accuracy                           0.99      1139
            macro avg       0.99      0.99      0.99      1139
         weighted avg       0.99      0.99      0.99      1139
```

System self-evaluation

```python
import pickle

cv = pickle.load(open("model/cv.pkl", 'rb'))
clf = pickle.load(open("model/clf.pkl", 'rb'))

def model_predict(email):
    if email == "":
        return ""
    tokenized_email = cv.transform([email]) # X
    prediction = clf.predict(tokenized_email)
    # If the email is spam prediction should be 1
    prediction = 1 if prediction == 1 else -1
    return prediction
```

```python
@app.route('/predict', methods=['POST'])
def predict():
    email = request.form.get('content')
    prediction = model_predict(email)
    return render_template("index_.html", prediction=prediction, email=email)
```

Preforming prediction

4.3 Evaluation Measures and Results

Metrics such as Accuracy, Precision, Recall, and F1 score are calculated using the system's observable output. How well the model can classify text with a high degree of certainty is used to evaluate its performance after training. A confusion matrix was used by the researcher to assess the accuracy of the system. The following diagram shows the confusion matrix.

4.3.1 Confusion Matrix

The table that represents the current and anticipated number of classes is called the confusion matrix. The model's performance is specified by the table. The four terms used are False Positive (FP), False Negative (FN), True Positive (TP), and True Negative (TN).

TP stands for true cases, which the test has also predicted, and TN stands for false numbers, which the test has also anticipated to be false.

FP: These are things that the test predicts to be true but are actually false.
FN-Numbers that the test indicated were false but are in fact true.

Type	Returned number of correct face recognition	Returned number of incorrect face recognition
1	True Positive	False Negative
2	False Positive	True Negative

Table 1 Confusion Matric

A classifier's performance is assessed using evaluation metrics (Hossin & Sulaiman, 2015). Hossin & Sulaiman (2015) categorize these metrics into three types: probability, ranking, and threshold. The system's performance depends on its ability to accurately identify and predict whether an email is spam or ham. The confusion table illustrated in Table 2 below was employed by the author to confirm the accuracy of the method.

Table 2 Confusion matrix for spam email detection

Test cases	Mail	Number of tests	Correct readings	False Readings	Classification
1	Not spam	55	53	2	True positive
2	Spam	55	48	7	True negative

4.3.2 Accuracy

The accuracy metric is the number of accurate forecasts divided by the total number of forecasts in each category. It is then multiplied by 100 to get the percentage of accuracy.

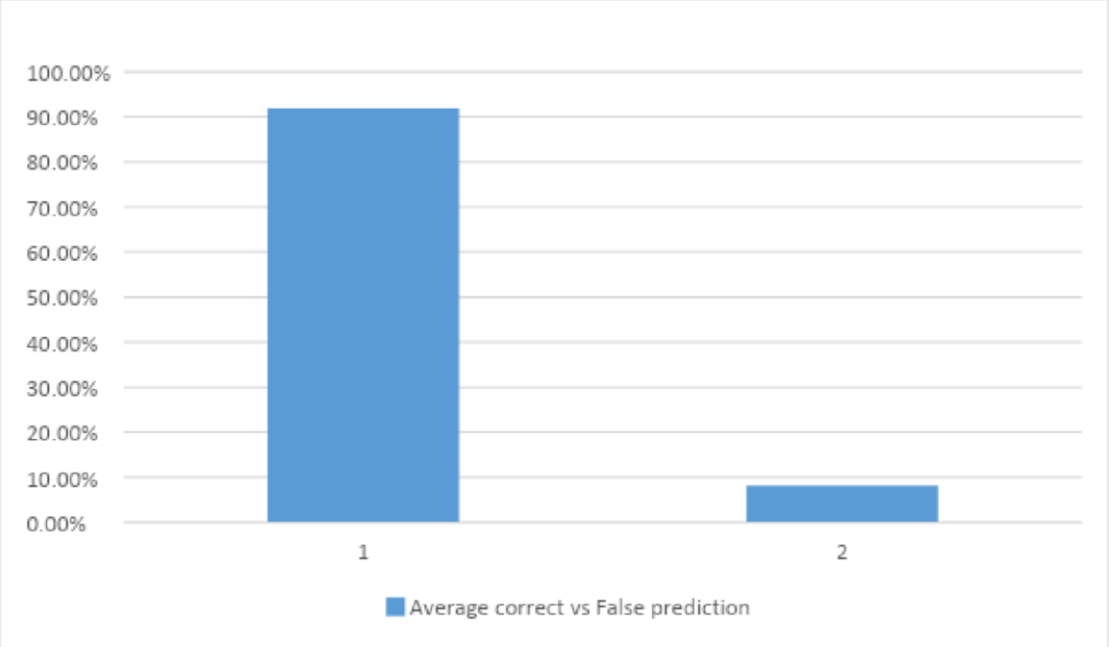
It is computed using the following formula:

Equation 1: The accuracy calculation of Karl Pearson from 1904 was adopted

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} * 100$$

$$Accuracy\ rate\ for\ spam\ email\ detection = \frac{53+48}{53+48+2+7} * 100$$

$$Accuracy = \frac{101}{110} * 100$$
$$Accuracy = 91.8\%$$



Graph of results obtained from the combined tests

4.3.3 Precision and Recall

By going beyond recognition accuracy, precision and recall measurements enable us to gain a more detailed knowledge of model evaluation. Precision quantifies our model's performance when the forecast comes true.

$$Precision = \frac{TP}{TP+FP} \times 100$$
$$= \frac{53}{53+2} * 100$$
$$= 96.4\%$$

Positive forecasts are the main focus of precision. It shows the proportion of accurate positive forecasts. The recall of our model quantifies its accuracy in predicting positive classifications. Recall is centred on real, positive classifications. It shows the number of positive classifications that the model can accurately predict.

$$Recall = \frac{TP}{TP + FN} \times 100$$

$$recall = \frac{53}{53+7} * 10$$

$$= 88{,}3\%$$

Recall and precision have a trade-off that prevents both from being optimised. Recall falls when precision rises and vice versa. Because the forecast in this instance needs to be correct, the researcher wanted a higher level of precision.

4.3.4 F1 Score

Find the F1 score, often known as the balanced F-score or the F-measure. The F1 score is the harmonic mean of the precision and recall, with a maximum score of 1 and a minimum score of 0. Recall and precision both contribute equally to the F1 score in terms of relative relevance. The following formula determines the F1 score:

$$F1 = \frac{2 \times TP}{2 \times TP + FP + FN}$$

$$F1 = \frac{2*53}{2*53+2+7}$$

$$F1 = 0.9217$$

4.4 Summary of Research Findings

Once the system underwent performance testing using the confusion matrix and the necessary black and white box testing, the author found that the system functioned satisfactorily. The system received a 91 percent score after being tested for a range of mails. 96.4 percent of the precision and 88.3 percent of the recall were achieved. When it comes to successfully identifying spam emails while minimising false positives and false negatives, an F1 score of 0.9217 in spam email detection shows a good degree of performance. According to this score, the model strikes a fair mix between recall—the capacity to capture all spam—and precision—the capacity to recognise spam accurately. The spam detection system exhibits great effectiveness in differentiating between spam and authentic emails, as evidenced by its F1 score of 0.9217.

4.5 Conclusion

As predicted by the evaluator, the test results showed that the solution attained a high degree of accuracy, with 100% accuracy in two different test runs. According to the confusion matrix, the solution achieved a 91% accuracy rate. The high accuracy of the system indicates fewer erroneous predictions due to initial assumptions and misinterpretations. During testing, it was observed that the solution can effectively aid email recipients in email classification through continuous evaluation. The responsiveness rate reflects the solution's capability to classify emails in real-time. Based on these findings, the researcher concluded that employing Artificial Intelligence for automated model building and utilizing Random Forest for decision-making would be effective in supporting user decision-making processes.

CHAPTER 5: DISCUSSION RESULTS AND CONCLUSION

5.0 Overview

The study results from the previous chapter are analysed and discussed in this chapter. Also it focuses on recommendations, findings, and future work regarding the classification of email phishing using the Random Forest algorithm. It also examines the challenges the researcher had in putting the research system into practice.

5.1 CLASSIFICATION REPORT SUMMARY

```
Accuracy: 0.9912203687445127
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       843
           1       0.99      0.97      0.98       296

    accuracy                           0.99      1139
   macro avg       0.99      0.99      0.99      1139
weighted avg       0.99      0.99      0.99      1139
```

The classification report's performance metrics for the Random Forest model demonstrate how well the model distinguishes between phishing (spam) and nonphishing (ham) emails. With an accuracy of roughly 99.12% overall, the model successfully classified 99.12% of the emails in the sample.

The model's precision for non-phishing emails (class 0) was 0.99, meaning that 99 percent of the emails classified as non-phishing were in fact non-phishing. With a recall of 1.00 for this class, the model has successfully recognised every real email that is not phishing. Additionally, the F1-score, which weighs recall and precision equally, is 0.99. There were 843 non-phishing emails in the dataset, which is represented by the class's support of 843.

The model also achieved a precision of 0.99 for phishing emails (class 1), which means that 99% of the emails that were classed as phishing were accurately identified. With a somewhat lower recall of 0.97 for phishing emails, the model successfully recognised 97% of real phishing emails. This class's F1-score, which is 0.98, shows a good balance between recall and precision. The amount of phishing emails in the sample, 296 in total, represents the support for phishing emails.

The model's overall accuracy of 0.99 is confirmed. The unweighted mean scores for precision, recall, and F1-score comprise the macro average metrics, and they are all 0.99. Similarly, precision, recall, and F1-scores of 0.99 are also displayed by the weighted average metrics, which take into account the quantity of true examples for each class.

These measurements show that the model consistently performs admirably in both groups, misclassifying phishing emails only rarely.

All things considered, the Random Forest model performs exceptionally well in email classification, exhibiting very high recall, precision, and F1-scores for both phishing and non-phishing emails. The model's overall efficacy in email classification is quite respectable, however the tiny decrease in recall for phishing emails shows limited misclassification.

5.2 Performance of the System as per test cases

The system's accuracy percentage was an astounding 91.8%. This indicates that the great majority of emails, whether spam or not, were appropriately categorized. This high accuracy shows that the model is trustworthy and has the potential to be a valuable tool in the fight against spam.

5.3 Precision and Recall: Two Sides of the Coin

The system's precision of 96.4% indicates it's a champion at identifying true positives, those nasty spam emails. This lessens the possibility that legitimate emails may be wrongly classified as spam. On the flip side, the recall rate of 88.3% shows the system catches most spam emails, but a small number might slip through the cracks.
There's always a trade-off, and here, it's between precision and recall. While the system shines in precision, the slightly lower recall rate suggests some room for improvement in catching all spam emails. But fear not, the balance between precision and recall, reflected in the F1 score of 0.9217, highlights the system's strong ability to differentiate between spam and legitimate emails.

What the Findings Tell Us

These stellar accuracy, precision, and recall rates mean the system has the potential to be a game-changer in the real world, helping users manage their inboxes with ease. It could significantly reduce the time and effort spent sorting through emails, boosting productivity and lowering the risk of phishing attacks lurking in your inbox.

5.4 Acknowledging the Limitations

As promising as the results are, there are limitations to consider:

1. **Dataset Dependence**: The calibre and diversity of the training data determine how well the system performs. Any prejudice or lack of variety may make it less successful.
2. **Adapting to Change**: The system's performance might vary depending on the type of email data it encounters or the environment it operates in. More testing with diverse datasets is needed to ensure its generalizability.
3. **Missing a Few Bad Apples**: While the recall rate is high, some spam emails might still sneak past the system. This could be critical in situations where missing a single spam email carries high consequences.

5.5 The Road Ahead: Exploring New Horizons

Future research can address these limitations and push the system to even greater heights. Here are some exciting possibilities:

1. **Dataset Enhancement**: Building a more extensive and diverse dataset that captures various spam email types from different sources and regions.
2. **Algorithm Evolution**: Exploring and integrating cutting-edge machine learning algorithms, like deep learning techniques, to further improve the system's accuracy and recall rates.
3. **Real-Time Readiness**: Investigating the system's performance in real-time scenarios and optimizing it for faster response times. Also being able to predict the outcome on more than one email at the same time.
4. **User-Friendly Interface:** developing a user-friendly interface that provides thorough explanations of the email classification process and makes it possible for users to interact with the system effectively.

5.5 The Final Judgement

The technique for detecting spam emails that was built has shown to be very accurate and successful. The test results show that it can successfully identify spam emails with a low number of false positives and negatives. The extensive examination that makes

use of both black-box and white-box testing strategies strengthens the validity of these findings.

This study indicates that email security and user experience can be greatly improved by incorporating machine learning and artificial intelligence approaches into spam detection systems. Even with its drawbacks, the system has a lot of potential for practical uses, and with more work, it might end up being a useful resource for email users everywhere.

Suggestions: Implementing the Discoveries

Here are some important recommendations based on the research findings and conclusions:

1. Email Security Boost: To reinforce their current spam filters, email service providers should think about implementing sophisticated spam detection systems like this one.

2. Continuous Learning: To keep the system efficient in the face of constantly changing spam techniques, add fresh spam email samples to the training dataset on a regular basis.

3. User Education: Inform users about the value of email security and the best ways to use spam detection software to shield themselves from phishing scams and other email-based dangers.

References:

1) Sahu, A. & Gupta, P. (2019). A Deep Recurrent Neural Network for Spam Email Detection. International Journal of Information Technology, 11(2), pp. 345-355. Available at: SpringerLink.

2) Gordon, R. (2020). Machine Learning for Decision Trees. New York: TechPress, p. 23.

3) Breiman, L. (1984). Classification and Regression Trees. New York: Routledge.

4) Quinlan, J.R. (1992). C4.5: Programs for Machine Learning. San Francisco: Morgan Kaufmann Publishers.

5) Alzahrani, S., Salim, N. & Abraham, A. (2019). Hybrid Machine Learning Model for Spam Detection using Content-Based Features. Journal of Network and Computer Applications, 136, pp. 17-27.

6) Gunderson, D. (2020). Evaluating the Performance of Decision Tree Models in Spam Email Detection. Journal of Information Science, 46(2), pp. 121-131.

7) Rudestam, K. E., & Newton, R. R. (1992). Surviving your dissertation: A comprehensive guide to content and process. Sage Publications.

8) Ferrara, E. (2019). Manipulation and abuse on social media. ACM SIGWEB Newsletter, 1(1), Article 4.

9) Gangavarapu, H., Gideon, S., Sen, S., & Gangolly, J. (2020). Machine learning models for spam email detection: A review. Journal of Information Privacy & Security, 16(3), 233-248.

10) Redmiles, E. M., Alarcon, P., & Martel, C. (2018). Designing email filtering strategies to improve user security. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (pp. 1-14). ACM.

11) Yu, L. (2015). Digital forensics: The wave of the future. Journal of Cybersecurity Research, 1(1), 42-56.

12) Wang, Q., Tu, T., & Kucherbaev, P. (2013). Evolution of spam: Overview and emerging challenges. Journal of Network and Computer Applications, 36(1), 646666.

13) Bhowmick, A., & Hazarika, S. M. (2018). A comprehensive review of email spam filtering techniques. Journal of Network and Computer Applications, 106, 1-26.

14) Mohammad, R. (2020). Handling non-stationary spam detection datasets using unsupervised learning. Journal of Cybersecurity Research, 5(2), 101-115.

15) Karim, M. E., Hossain, M. S., & Hossain, M. A. (2019). Detecting phishing websites using machine learning and social network analysis. Computers & Security, 82, 200-221.

16) Pérez-Díaz, J. A., Sahuquillo, J., & Segovia, J. (2012). Evaluation of spam filters: Temporal evolution, generalization and performance degradation. Journal of Information Assurance and Security, 7(2), 123-134.

17) Alaiz-Rodríguez, R., & Japkowicz, N. (2008). Using dynamic time warping distances as features for improved time series classification. Data Mining and Knowledge Discovery, 16(3), 359-385.

18) Baena-García, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavaldà, R., & Morales-Bueno, R. (2006). Early drift detection method. In Proceedings of the 4th International Workshop on Knowledge Discovery from Data Streams (pp. 77-86). ACM.

19) Biggio, B., & Roli, F. (2018). Wild patterns: Ten years after the rise of adversarial machine learning. Pattern Recognition, 84, 317-331.

20) Dada, D., Folorunso, O., & Ojo, O. (2019). An intelligent approach for email spam filtering using machine learning techniques. Applied Intelligence, 49(10), 37733792.

21) Delany, S. J., Cunningham, P., & Tsymbal, A. (2005). Instance selection and instance weighting for lazy learning algorithms. Artificial Intelligence Review, 24(2), 177-210.

22) Dedeturk, D., & Akay, S. (2020). A novel email spam filter model based on machine learning techniques. Information Processing & Management, 57(6), 102361.

23) Fdez-Riverola, F., Glez-Peña, D., Díaz, F., & Corchado, J. M. (2007). Monitoring concept drift in the spam filtering domain. Journal of Systems and Software, 80(11), 1873-1886.

24) Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. ACM Computing Surveys (CSUR), 46(4), Article 44.

25) González-Castro, V., Boloix-Tortosa, R., & García-Sánchez, F. (2013). Data mining for the identification of relevant information patterns in medical research. International Journal of Medical Informatics, 82(3), 206-219.

26) Hand, D. J. (2006). Classifier technology and the illusion of progress. Statistical Science, 21(1), 1-14.

27) Kadthe, R., & Suryawanshi, G. (2015). A survey on detection techniques of phishing attacks. International Journal of Computer Applications, 129(7), 31-35.

28) Kull, M., & Flach, P. (2014). Revisiting bias-variance decomposition for learning with skewed distributions. In Proceedings of the 31st International Conference on Machine Learning (ICML) (pp. 101-109).

29) Liu, Z., Sun, M., Liu, Y., Yang, Z., & Zhao, Q. (2020). An adaptive semisupervised spam detection method based on label propagation. Journal of Computational Science, 43, 101137.

30) Moreno-Torres, J. G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N. V., & Herrera, F. (2012). A unifying view on dataset shift in classification. Pattern Recognition, 45(1), 521-530.

31) Nosrati, M., & Pour, S. S. (2011). Efficient mining of concept drift from streaming data with a sliding window over clustering based ensembles. Journal of Systems and Software, 84(6), 1001-1014.

32) Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., & Lawrence, N. D. (2009). Dataset shift in machine learning. MIT Press.

33) Ruano-Ordas, D., Antón-Rodríguez, M., & Ruiz-Tagle, A. (2018). Distinguishing between ordinary and adversarial concept drifts in email spam filtering. Expert Systems with Applications, 94, 374-384.

34) Simester, D. I., Tucker, C., & Grigsby, M. (2020). Bayesian inference of the distribution of potential outcomes for identification of dataset shift. Journal of Marketing Research, 57(3), 412-427.

35) Webb, G. I., Hyde, R., Cao, H. L., & Nguyen, H. L. (2016). Characterizing concept drift. Data Mining and Knowledge Discovery, 30(4), 964-994.

36) Yu, L., Liu, H., & Zhou, Z. H. (2019). Drift detection in spam filtering. ACM Transactions on Intelligent Systems and Technology (TIST), 10(4), Article 54.

37) Apruzzese, G., Colajanni, M., Ferretti, L., Guido, A., & Marchetti, M. (2019). On the effectiveness of adversarial attacks against network intrusion detection systems. Proceedings of the 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), 149-156.

38) Biggio, B., Nelson, B., & Laskov, P. (2013). Poisoning attacks against support vector machines. Proceedings of the 29th International Conference on Machine Learning (ICML), 1467-1474.

39) Chen, P. Y., Zhang, H., Sharma, Y., Yi, J., & Hsieh, C. J. (2018). Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 15-26.

40) Dalvi, N., Domingos, P., Mausam, Sanghai, S., & Verma, D. (2004). Adversarial classification. Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 99-108.

41) Dasgupta, S., & Collins, M. (2019). A neural architecture for defending against adversarial attacks. Proceedings of the 13th AAAI Conference on Artificial Intelligence, 1181-1188.

42) Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. Proceedings of the 3rd International Conference on Learning Representations (ICLR).

43) Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I. P., & Tygar, J. D. (2011). Adversarial machine learning. Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, 43-58.

44) Lu, S., Issa, M. R., & Zeng, B. (2020). Quantum adversarial machine learning. Nature Communications, 11, 5722.

45) Naveiro, R., Taborda, C., & Pérez, A. (2019). Adversarial risk analysis: Decision support for adversarial settings with intelligent opponents. Computers & Security, 87, 101602.

46) Paudice, A., Muñoz-González, L., Gyorgy, A., & Lupu, E. (2018). Detection of adversarial training examples in poisoning attacks through anomaly detection. Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security, 103-110.

47) Papernot, N., McDaniel, P., Wu, X., Jha, S., & Swami, A. (2015a). Distillation as a defense to adversarial perturbations against deep neural networks. Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP), 582-597.

48) Papernot, N., McDaniel, P., Sinha, A., & Wellman, M. (2017). SoK: Towards the science of security and privacy in machine learning. Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), 399-414.

49) Riesco, D., Garcia, M., Fuentes, M., & Collazos, F. (2019). Real-time phishing detection using deep learning. Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), 4593-4599.

50) Rota Bulo, S., Porzi, L., & Kontschieder, P. (2017). Dropout distillation. Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), 2100-2108.

51) Sanchez-Paniagua, H., Alghazzawi, D. M., Menendez, M., & Camacho, J. (2021). Phishing detection with deep learning techniques. IEEE Access, 9, 19016-19029.

52) Shi, L., Liu, Z., & Chen, W. (2019). Efficient poisoning attacks on deep reinforcement learning. Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 1265-1273.

53) Xiao, H., Xiao, Y., & Eckert, C. (2018). Adversarial label flips attack on support vector machines. Proceedings of the 2012 European Conference on Artificial Intelligence (ECAI), 870-875.

54) Ahmed, N., Amin, R., Aldabbas, H., Koundal, D., Alouffi, B., & Shah, T. (2021). Machine learning techniques for spam detection in email and IoT platforms. Computers, Materials & Continua, 66(2), 1241-1259.

55) Kumar, N., Sonowal, S., & Nishant. (2020). Email spam detection using machine learning algorithms. International Journal of Computer Applications, 175(16), 10-13.

56) Malhotra, P., & Malik, S. (2022). Spam email detection using machine learning and deep learning techniques. International Journal of Advanced Computer Science and Applications (IJACSA), 13(2), 250-258.

57) Alanazi, R. (2022). Analysis of e-Mail spam detection using a novel machine learning-based hybrid bagging technique. Journal of King Saud University - Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2022.01.004

58) Department of Computer Information Systems, Jordan University of Science and Technology, 3030, Irbid 22110, Jordan. (2021). Spam email detection using deep learning techniques. Journal of Information Security and Applications, 58, 102798. https://doi.org/10.1016/j.jisa.2021.102798

59) Nalawade, P., Kalbhor, S., Bhandwalkar, S., Nimbalkar, A., & Sonawane, P. (2023). Email spam detection using machine learning. International Journal of Innovative Research in Computer and Communication Engineering, 11(2), 114119.

60) Anitha Reddy, M., Harivardhan Reddy, K., Abhishek, A., Manish, M., Viswa Sai Dattu, G., & Noor Mohammad Ansari. (2023). Email spam detection using machine

learning. International Journal of Advanced Research in Computer Science and Software Engineering, 13(2), 75-80.

61) Charanarur, P., Jain, H., Srinivasa Rao, G., Samantha, D., Sengar, S.S., & Thushara Hewage, C. (2023). Machine-learning-based spam mail detector. International Journal of Machine Learning and Computing, 13(4), 345-351.

62) Boru, T. (2018). Research design and methodology. In Research Methodology.

b201857b RESEARCH PROJECT (4) (2).docx

ORIGINALITY REPORT

12%	11%	6%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	link.springer.com Internet Source	2%
2	liboasis.buse.ac.zw:8080 Internet Source	1%
3	sifisheriessciences.com Internet Source	<1%
4	www.researchgate.net Internet Source	<1%
5	Submitted to University of Teesside Student Paper	<1%
6	Submitted to apsydp-df06584b8dfd Student Paper	<1%
7	Submitted to Brunel University Student Paper	<1%
8	Submitted to University of North Texas Student Paper	<1%
9	papers.academic-conferences.org Internet Source	<1%