

**CLIENT SIDE MULTI-LINGUAL MODEL FOR ENHANCING
PERFORMANCE IN SHORT MESSAGE SERVICE SPAM DETECTION**

ANDREW KIPKEBUT

**A Thesis Submitted to the Institute of Postgraduate Studies of Kabarak University in
Partial Fulfillment of the Requirements for the Award of Doctor of Philosophy in
Information Technology (Security and Audit)**

KABARAK UNIVERSITY

NOVEMBER 2022

DECLARATION

1. I do declare that:

(i) This thesis is my own work and to the best of my knowledge it has not been presented for the award of a degree in any university or college.

(ii) That the work has not in-cooperated material from other works or a paraphrase of such material without due and appropriate acknowledgement

(iii) That the work has been subjected to processes of anti-plagiarism and has met Kabarak University 15% similarity index threshold

2. I do understand that issues of academic integrity are paramount and therefore I may be suspended or expelled from the University or my degree may be recalled for academic dishonesty or any other related academic malpractices

Signed:  _____

Date: 28/03/2022

Name of Student: Andrew Kipkebut

Admission Number: **GDS/M/0206/01/16**

RECOMMENDATION

To the Institute of Postgraduate Studies

The research thesis entitled “**A Client side multi-lingual model for enhancing performance in short message service spam detection**” written by **Andrew Kipkebut** to the Institute of Postgraduate studies of Kabarak University. We have reviewed the thesis and recommend it be accepted in partial fulfillment of the requirement for award of degree of Doctor of philosophy in I.T security Audit.

Signed  Date 28/03/2022

Dr Moses Thiga
School of Science, Engineering and Technology
Kabarak University

Signed  Date 28/03/2022

Dr Elizabeth M Okumu
School of Science, Engineering and Technology
Kabarak University

COPYRIGHT

©2022

ANDREW KIPKEBUT

All rights reserved, No part of this thesis may be reproduced or transmitted in any form by means of either mechanical, including photocopy, recording or any other information storage or retrieval system without permission in writing from author or Kabarak University

ACKNOWLEDGEMENT

This work could not have been done if not for the opportunity given to me by Kabarak university .They gave me the prerequisites needed to test these technologies .A very special gratitude goes out to Kabarak University for the serene learning environment and fantastic research facilities.

I would also like to thank my supervisors, Dr Moses Thiga and Dr Elizabeth Okumu from the school of science engineering and technology of Kabarak University, for the guidance and advice they has provided throughout my time as a PhD student. I have been extremely lucky to have supervisors who cared so much about my work, and who responded to my questions and queries so promptly. I would also like to thank all the members of staff at Kabarak University who helped me in my supervisor's absence. I must express my gratitude to my wife, for her continued support and encouragement. I was continually amazed by her willingness to proof read countless pages of meaningless machine learning algorithms, and by the patience of my kids, who experienced all of the ups and downs of my research. Completing this work would have been more difficult were it not for the support and friendship provided by the other members of school of science engineering and technology including the institute of postgraduate studies of Kabarak university. I am indebted to them for their help. Thank you and God bless you all.

DEDICATION

I dedicate this work to my late Father Wilson Kipkebut and mother Leah Kipkebut, my wife Betty Bob, my children Ivy Chebet, Avianna Kayane and Myles Kebut for the great support.

ABSTRACT

Millions of money are lost by mobile phone users every year due to short message service spam, a social engineering skill attempting to obtain sensitive information such as passwords, personal identification numbers and other private data by posing as a trustworthy entity through short message service. Most spammers are constantly developing new sophisticated methods, rendering previous techniques obsolete. A thoughtful deficiency in most sms spam detection methods is lack of satisfying accuracy, reliability, low performance and comprehensibility especially when individual classifiers are used, these remains important aspects to be considered for an optimal model development. Sms spam detection using machine learning techniques is a new approach especially in ubiquitous computing devices such as mobile phones, moreover the design of short message spam detection techniques in a mobile platform is challenging task due to the non-stationary distribution of the data and the multi-lingual nature of text messages from users. It is in this background that the research proposes a multi-stage ensemble hybrid client side multilingual sms spam detection model for a mobile environment using machine learning techniques. It involves enhanced use of pre-processing techniques, content based feature engineering techniques, multilingual natural language processing, data training and testing. A hybrid ensemble machine learning method is used to combine the classifiers based on a combination algorithm. The contributors of multi-lingual messages data include a combination of secondary data from University of California Irvine public repository and primary data from local users and sampled local repositories in Kenya. Machine learning and data mining experiments are conducted using Java based Waikato environment for knowledge analysis. The results and discussions are analyzed and presented in form of descriptive statistics. The effectiveness of the proposed model is empirically validated using ensemble classification methods that gave an overall classification accuracy of 98.2606%. The results from this study demonstrates that the proposed ensemble model improves the overall performance by increasing the accuracy and reducing false positives.

Keywords: Algorithm, Classifiers, Detection, Machine learning, Mobile, Ensemble Model, WEKA.

TABLE OF CONTENTS

DECLARATION	ii
RECOMMENDATION	iii
COPYRIGHT	iv
ACKNOWLEDGEMENT	v
DEDICATION	vi
ABSTRACT	vii
TABLE OF CONTENTS	viii
LIST OF FIGURES	xiii
LIST OF TABLES	xv
LIST OF ALGORITHMS	xv
LIST OF ABBREVIATIONS	xvi
OPERATIONAL DEFINITIONS OF TERMS	xvii
CHAPTER ONE	1
INTRODUCTION	1
1.1 Introduction.....	1
1.2 Background of the study	1
1.3 Statement of the problem.....	4
1.4 Main objective	6
1.4.1 Specific objectives.....	6
1.5 Research questions.....	6
1.6 Justification of the study	7
1.7 Scope and limitations of the study	8
1.8 Chapter summary	8
CHAPTER TWO	9
LITERATURE REVIEW	9
2.1 Introduction.....	9
2.1.2 Sms spam (smishing) –A social engineering challenge.....	10
2.2 SMS spam preprocessing techniques	18
2.3 Machine learning classification algorithms for feature engineering.....	20
2.3.1 Naïve bayes classification	21
2.3.2 Support vector machine	27

2.3.3 Artificial neural network	28
2.3.4 J48 Decision tree based algorithm.....	32
2.3.5 Apriori algorithm.....	36
2.3.6 K^{th} nearest neighbor	36
2.4 Feature engineering techniques.....	37
2.4.1 Feature selection.....	38
2.4.2 Feature selection process for sms spam detection.....	40
2.5 Correlation-based feature selection for machine learning	46
2.5.1 Symmetrical uncertainty feature selection method	49
2.5.2 Information gain	50
2.5.3 Mutual information.....	50
2.5.5 Dimensionality reduction	52
2.6 Multilingual natural language processing.....	53
2.6.1 Components of words.....	54
2.6.2 Word tokens	54
2.6.2.1 Lexemes	54
2.6.2.2 Typology	55
2.6.2.3 Morphological models	55
2.6.2.4 Dictionary lookup	55
2.6.2.5 Finite state morphology	56
2.6.2.6 Sentence boundary detection	57
2.6.3 Word features	59
2.6.3.1 Punctuations, numbers, and symbol features.....	59
2.6.3.2 Natural language parsing	61
2.6.3.3 Top-down parsing	61
2.6.3.4 Predictive parsing.....	63
2.6.4 Machine translation	64
2.6.4.1 Decoding a language model.....	65
2.6.4.2 Cube pruning.....	65
2.6.4.3 Visualization	65
2.6.4.4 Visualization trees.....	65
2.6.4.5 Receiver operating characteristics.....	67

2.7 Ensemble classification using machine learning	68
2.7.1 Ensemble feature selection techniques	70
2.7.1.1 Bagging	71
2.7.1.2 Boosting	72
2.7.1.3 Stacking.....	74
2.8 Ensemble model prototyping	76
2.8.1 Testing android mobile applications	77
2.9 Related work on SMS spam detection	78
2.9.1 Conceptual framework	89
2.9.2 Chapter summary.....	91
CHAPTER THREE	92
RESEARCH METHODOLOGY	92
3.1 Introduction.....	92
3.2 Research design	92
3.2.1 The WEKA workbench.....	93
3.2.2 Hyper parameter tuning.....	95
3.3 The ensemble modeling steps	96
3.4 Ensemble dataset pre-processing techniques	97
3.4.1 Word tokenization	98
3.4.2 Stemming and lemmatization of words.....	99
3.4.3 Stop words removal.....	99
3.4.4 Word N- gramming modeling	100
3.4.5 Bag of words modeling	101
3.5 Sampling procedure and sample size.....	104
3.5.2 Data collection.....	105
3.6 Ensemble based feature engineering methods	105
3.6.1 Ensemble feature ranking techniques.....	106
3.6.2 The ensemble feature selection algorithm.....	107
3.6.3 KNN modeling using euclidean and manhattan distancing	110
3.7 Ensemble training, testing and validation.....	111
3.7.1 Clustering in machine learning	112
3.7.1.1 K- means clustering algorithm.....	112

3.7.1.2 Hierarchical clustering	114
3.7.1.3 Cobweb clustering	115
3.7.2 Methods of measurement	115
3.7.3 Confusion matrix.....	118
3.7.4 Bagging and boosting with six individual classifiers.....	119
3.7.5 Stacking as a meta- classifier with six algorithms	121
3.7.6 The ensemble hybrid machine learning model	121
3.8 Model prototype development and implementation.	122
3.8.1 Software Testing.....	122
3.9 Ethical considerations	123
3.9.1 Chapter summary.....	123
CHAPTER FOUR.....	125
DATA ANALYSIS, PRESENTATION AND DISCUSSION	125
4.1 Introduction.....	125
4.2 Model pre- processing techniques	126
4.3 Enhanced ensemble multi- filter feature engineering methods.....	130
4.3.1 Ranking the retained feature sets by feature selection method	134
4.4 Improving learning using clustering techniques	150
4.5 Model comparison with existing studies.....	154
4.6 Prototype design and implementation.....	156
4.6.1 Software development tools	160
4.6.2 Coding modules.....	161
4.6.2.1 Main module	161
4.6.2.2 Stopping module	167
4.6.2.3 Stemming module	167
4.6.2.4 Vector space model module.....	168
4.6.3 Prototype module testing.....	168
4.7 Chapter summary	173
CHAPTER FIVE	174
SUMMARY, CONCLUSIONS AND RECOMMENDATIONS	174
5.1 Summary	174
5.1.1 To evaluate the optimal text preprocessing techniques	174

5.1.2 To enhance the feature engineering methods for ensemble learning.....	174
5.1.3 To enhance class correlation using ensemble hybrid of classifiers	174
5.1.4 To cluster short message short message spam service spam detection	177
5.1.5 To design and test the prototype on android platform	177
5.2 Conclusion	177
5.3 Recommendations.....	179
5.4 Areas of Further research.....	179
5.4.1 Data set.....	179
5.4.2 Data set imbalance	179
5.4.3 Toleration for misclassification	179
5.4.4 Dimension reduction.....	180
REFERENCES.....	181
APPENDICES	202
Apendex I JAVA programming code extract	202
Apendex II Data set in Arff format extract.....	210
Apendex III Publications	217
Apendex IV Letter of Introduction	219
Apendex V Research Permit.....	220

LIST OF FIGURES

Figure 1: A sequence diagram for a smishing attack based on URL click	13
Figure 2: Client side smishing example	14
Figure 3: A Naïve bayes algorithm implementation equation	22
Figure 4: SVM boundary decision scheme	28
Figure 5: Artificial neural network.	30
Figure 6: Four key steps of the feature selection process.	41
Figure 7 Grammar rules.	62
Figure 8: Top down parsing.	62
Figure 9: Top down backtracking.	62
Figure 10: Final parsed tree for a given sentence.	63
Figure 11: Sample decision tree.	67
Figure 12: Stacking of algorithm based on meta learner.	75
Figure 13: Architecture of the SMS spam transmission line	79
Figure 14: Conceptual framework	90
Figure 15: Weka GUI chooser.	94
Figure 16: Term-document matrix.	103
Figure 17: Percentages of the improvement recorded for each classifier	128
Figure 18: Receiver operating curve.	149
Figure 19: word clustering	154
Figure 20: A comparison graph for related studies.	155
Figure 21: Android based prototype architecture.	156
Figure 22: The translation example.	158
Figure 23: Training module	159
Figure 24: Testing module for a spam SMS	159
Figure 25: Testing module for non-spam SMS.	160
Figure 26: Test cases vs. failed test cases	170
Figure 27: Bar chart representing defect severity.	171
Figure 28: Defects distributions	172

LIST OF TABLES

Table 1: Smishing attack characteristics	2
Table 2: Summary of the major research studies on SMS spam	87
Table 3: Hyper parameter tuning for individual classifiers	96
Table 4: Data set list	98
Table 5: Tokenization extract	99
Table 6: Bag of words histogram.....	103
Table 7: Evaluation measures for SMS spam filters.....	116
Table 8: Confusion matrix	119
Table 9: Ensemble modelling parameters.....	120
Table 10: Ten fold cross validation scores for evaluating pre-processing techniques	127
Table 12: Reduction of features with attribute selection filter application on the data set....	130
Table 13: Feature engineering techniques with feature search algorithms.....	132
Table 14: Attribute selection with 10 fold cross-validation.....	133
Table 15: Rankings based on retained reduced feature sets	134
Table 16: Accuracy performance of feature selection methods with individual classifiers. .	135
Table 17: Classifiers execution time as per the feature selection methods.....	136
Table 18: KAPPA statistic metric per feature model on classifiers.....	137
Table 19: MAE statistic metric per feature model on a classifier.....	138
Table 20: RMSE statistic metric per feature model on a classifier.....	139
Table 21: Accuracy performance of feature selection techniques on different classifiers	141
Table 22 Bagging and boosting performance evaluation	142
Table 23: Performance of naïve bayes as a base learner and stacking with six classifier	144
Table 24: Stratified 10 fold cross validation summary for the stacked model.	147
Table 25: Detailed accuracy of the ensemble hybrid stacked model.....	148
Table 26: Confusion matrix	149
Table 27: Clustering of 785 spam messages using three algorithms.	151
Table 28: Cluster centroids for selected words.....	153
Table 29: Comparison of various model with the stacked model.....	155
Table 30: Test cases plan vs executed report.....	169
Table 31: No of defects identified by status and severity	170
Table 32: Module defects distribution	171

LIST OF ALGORITHMS

Algorithm 1: Naïve bayes algorithm.....	24
Algorithm 2: Decision tree algorithm	33
Algorithm 3: Decision tree pruning algorithm technique	35
Algorithm 4: Ensemble KNN algorithm.....	37
Algorithm 5: Generalized filter algorithm	42
Algorithm 6: Generalized wrapper algorithm	43
Algorithm 7: Hybrid feature selection algorithm.....	45
Algorithm 8: Tokenization algorithm	53
Algorithm 9: Entropy algorithm	66
Algorithm 10: Bagging algorithm.....	72
Algorithm 11: AdaBoost algorithm technique.....	74
Algorithm 12: Stacking algorithm	75
Algorithm 13: N-gram modeling algorithm.....	101
Algorithm 14: Ensemble feature selection algorithm	109

LIST OF ABBREVIATIONS

ANN	Artificial neural network
ARFF	Attribute- relation file format
BN	Bayesian network
APWG	Anti-phishing working group
CSV	Comma separated values
CFS	Correlation feature selection
CS	Chi square
EHMLA	Ensemble hybrid machine learning algorithms
GR	Gain ratio
IDF	Inverse document frequency
IG	Information gain
KDD	Knowledge discovery database
ML	Machine learning
ROC	Receiver operating Curve
SMS	Short message service
SMO	Sequential minimal optimization
SU	Symmetric uncertainty
TF	Term frequency
SVM	Support vector machine

OPERATIONAL DEFINITIONS OF TERMS

Classification: - In this study classification is the process of identification to which of a set of categories belongs to class for example we can classify a message as either a spam or not a spam (Alpaydin and Ethem, 2010).

Model training set: A process of feeding a machine learning algorithm with data to help it identify and learn good values for all attributes involved in the study (Mahesh, 2020).

Algorithm: A step by step computational procedure for solving a machine language problem for example a feature selection algorithm may be used to select the best feature for the ensemble model.

Ensemble learning: is the process by which multiple classifiers are strategically generated and combined to solve the spam detection problem, the aim is to obtain a better predictive performance for the model.

Clustering: A number of objects of the same kind e.g. a cluster of SPAM, a cluster of respondents, cluster of algorithms.

Naïve bayes method : In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features (Zhang et al.,2021)

Pre-processing: Data preprocessing is a data mining technique which is used to transform the raw text messages in a useful and efficient format.

Cross validation: is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set (Berrar, 2019).

CHAPTER ONE

INTRODUCTION

1.1 Introduction

This chapter gives a brief introduction and background information that informed the study on SMS spam detection. It further proceeds to research problem, the main objective, specific objectives, research questions and the significance of the study.

1.2 Background of the study

Consider this scenario¹, imagine receiving a text message from an online retailer store you always visit, informing you of a flash sale in the next five minutes. Not wanting to miss out on this opportunity, you click the link on the text message. Within the next few seconds, your Internet browser brings you to a familiar retail website, prompting for your credentials. Almost like a reflex, you key in your authentication credentials. The next pop-up that greets you, is a set of familiar light blue fonts saying “The password was recently changed”. For the next thirty seconds, you proceed to type on your phone keyboard all known usernames and passwords combinations that you can remember, mumbling to yourself “One of these must be right...” Before you know it, the flash sale is over and you have just missed one of the greatest deals of your life. On the other side of the world, a hacker is happy as he has just received the keys to your online empire, along with thousands of other similar victims. You have just been smished!

Scenario 2 , a mobile phone user A also receives an sms alleging that he or she has received money from an unknown mobile user B, the unknown user B calls and “pleads” that he be refunded the money since he sent the money to a “wrong” user A, unfortunately most users are like mobile phone user A and are likely to respond to the sms and even transfer the excess money to mobile phone user B, user A may be a victim of Sms spam fraud if he/she actually did not receive the money sent, now the question is how can we know whether a received message is Spam or Not? From both scenarios? Most mobile phone users rely on manual crafted rules/self-judgment when it comes to SMS spam detection also known as Smishing in

this study. Sonowal and Kuppusamy (2018) defined SMS phishing (Smishing) as an incarnation of phishing attack, which utilizes Short Messaging Service (SMS) or simple text message on mobile phones to lure the victim's in an attempt to harvest user credentials, by posing as a trusted party in digital communication using Short Message Service . Many client-side solutions have been proposed to detect these Smishing attacks, but attackers find weaknesses in these systems (Goel and Jain 2017). .Emigh (2006) in their study found out that some web pages are designed to mimic genuine websites and Some attackers tricks client-side users by using SMS social engineering tact's such a threat to suspend an account if they don't make current updates. Table 1 presents features, information netted and potential consequences of a smishing attack.

Table 1: Smishing attack characteristics (Jain and Gupta, 2018)

Delivery message	Information netted	Potential consequences
Good news or bad news	Account information	Financial loss
Sense of urgency	Usernames and passwords	Identity theft
Delicate or confident matter	Personal identification numbers	Loss of data
Impersonation	Other private information	Political gain
Clicking of a link	Transactions details	Malware implantation
General greetings	Credentials	Credential harvesting
Account update/validation/suspension	Credentials	

Frauds are old as the human race, fraudsters chase the money such as credit cards, E- wallets, and E- cash such as M-Pesa, are well-known for being targeted by fraudulent activities especially in developing countries such as Kenya (Gajjala and Tetteh, 2014). As the popularity of this platform increases, there has been a surge in the number of unsolicited commercial advertisements sent to mobile phones using text messaging therefore there is a need for effective systems that are able to detect and fight this social engineering threat. A number of major differences exist between spam-filtering in text messages and emails. Unlike emails, which have a variety of large datasets available, real databases for SMS spams are very limited. Additionally, due to the small length of text messages, the number of features that can be used for their classification is far smaller than the corresponding number in emails. For example, in text messages there is no header. Additionally, text messages are full of abbreviations and have much less formal language than what one would expect from emails. Given the amount of sms data and their popularity in different countries, users text topical messages in different languages (Cambria, 2017). Most expressions are written in different languages, this multi-lingual nature makes classification even complicated. All of these factors combine has led to a serious degradation in detection accuracy. One approach of solving this NLP social engineering problem is the use of machine learning algorithms, however ensemble machine learning has been a promising path, since it uses multiple classifiers by training and combining multiple learners using various methods (Zhou, 2021), this approach reduces bias, improve robustness and generalization compared to single model, however this method can cost more to create, train, test and deploy.

In this thesis, the goal is to apply heterogeneous machine learning classifiers and algorithms to the SMS spam classification problem, compare their performance to gain insight and further explore the problem, and design an application based on one of the most promising

combination of algorithms, that can filter SMS spams with the high level of accuracy i.e. high true positives and low false positives rates, through enhancement of pre-processing and feature engineering techniques.

1.3 Statement of the problem

The existing work in sms spam detection are unable to detect sms spam effectively due to dynamic and complex language nature of text messages especially when the message is multi-lingual. Moreover spammers are developing more sophisticated tactics, rendering previous features and methods ineffective (Karami and Zhou 2014). Machine learning can optimize the entire performance of a sms spam detection system by reducing the feature scope, classification of unlabeled large data set to their appropriate atomic class. Moreover most existing detection techniques for SMS spam have been adapted from other context such as emails, therefore ignore some of the unique characteristics of SMS spam detection. To add on this text messages are highly influenced by the presence of informal languages like regional words, idioms, phrases and abbreviations (Gupta et al., 2018, Silva et al ., 2017).

Recent research done on machine learning for SMS spam detection, focuses more on boosting efficiency rate of machine learning classifiers especially in increasing true positives and reducing false positives and excluding other evaluation metrics that are equally important. There exist many automated approaches to sms spam detection from individual classifiers to multiple classifiers (Gupta et al., 2019) both for client –side and server side. Ensemble learning is a reliable machine learning style which has shown great advantages in a lot of applications, and by using several learners, the general ability of an ensemble methodology is way better than that of a single learner by minimizing bias and improving performance Ahmed et al., (2015). A thoughtful deficiency of most current ensemble methods is lack of satisfying

accuracy and reliability. Improving performance and comprehensibility of ensemble method is an important aspect, yet largely understudied (Gupta et al., 2019).

Many research findings shows that despite the high detection accuracies recorded , there is still room for improvement on areas like text preprocessing ,content based feature engineering, training, testing and hyper parameter tuning and optimization (Onan, Korukoğlu and Bulut ,2016).

The Short Messaging Service (SMS) mobile communication system is attractive for criminal gangs for a number of reasons i.e. it is easy to use, fast, reliable and affordable technology (Delany, Buckley and Greene, 2012). Financial losses due to SMS fraud have affected not only individual mobile users and merchants but also business organizations. If the bank loses money, customers eventually pay as well through higher interest rates, higher membership fees and high penalties therefore the problem cuts across the entire business system environment. SMS spam is a growing problem largely due to the availability of very cheap bulk pre-pay and post pay SMS packages and the fact that SMS normally stimulates higher response rates as it is a trusted and a personal service, sms spam detection is notably a challenging problem because fraud strategies change in time and habits keeps changing. Few examples of these frauds are available, therefore very hard to identify fraudulent behavior, and not all frauds are reported and if reported there are large time lapse. There is also too much trust on text messages that are delivered to our portable devices such as mobile phones, perhaps because these devices are personal. Fraudsters are clever and they make the spam messages appear to be coming from a trusted source like a friend, a co-worker, a favorite retail store you do business with, your school, your financial institution among others.

1.4 Main objective

The main objective of this research is to design a client side multilingual model for enhancing performance of SMS SPAM detection using ensemble of hybrid machine learning algorithms (EHMLA).

1.4.1 Specific objectives

1. To evaluate the optimum text pre-processing techniques for modeling sms spam detection system.
2. To determine the best feature engineering techniques for multilingual text message processing for ensemble hybrid machine learning.
3. To enhance the performance of class correlation model using ensemble of machine learning algorithms.
4. To classify text messages using ensemble hybrid of machine learning techniques as either SPAM class or NOT.
5. To design and test the ensemble hybrid model using a prototype on android mobile environment.

1.5 Research questions

The research questions for the study are listed below:-

1. How to determine the best text processing techniques for sms spam detection?
2. What are the best features set for multilingual languages using ensemble hybrid feature engineering techniques?
3. How can a sms spam class –content correlation hybrid model be enhanced to detect sms spam for a mobile environment?
4. How can machine learning techniques be used in classifying text messages as either Spam or not, considering the multilingual, multicultural nature of users?

5. How can a model prototype be designed and tested on a mobile phone such as the android operating system?.

1.6 Justification of the study

Detecting sms spam is a difficult task because of massive amount of very low quality message data set from users (Delany et al., 2012). The ensemble feature selection and feature extraction techniques from this study will help gather a feature set that is comprehensive enough to classify spam data from non-spam data much faster by applying the proposed techniques (Bach and Gunnarsson,2010).

- i. The heterogeneous linguistic style of sms language makes classification even complex, enhanced sms analysis through textual sentimental analysis and composition will significantly improve the quality of sms spam detection and hence improve precision by utilizing important features of a language.
- ii. By determining features that have high correlation with the output class (spam, not spam), through this we can improve spam detection performance.
- iii. Through the ensemble method we can significantly reduce the feature set of the data while maintaining and improving accuracy of classification.
- iv. The enhanced ensemble model based on the machine learning algorithms (SVM, J48, Naïve Bayes, KNN, ANN) will increase the accuracy speed in detecting multilingual SPAM messages, therefore reducing false positives rates and therefore optimizing device performance.
- v. Through the prototype, users are able to automatically detect fraudulent SMS without the use of manual crafted rules.

1.7 Scope and limitations of the study

The scope of this study is restricted to the following limitations: The study is limited to a combination of locally data collected by the researcher (60 text messages count) and publicly available SMS Spam data at university of California Irvine (UCI) Machine learning repository which is available in raw format in publicly repositories. The research focuses on analyzing short messages service (SMS) as a guide in designing an efficient ensemble hybrid client side SMS spam detection model. According to Milian (2009) one of the limitation in sms analysis is the maximum length of an SMS message which is only 160 characters, which means there is limited messages for content-based filtering because of the short message length available. On the other hand SMS subscribers use a personal language subset with abbreviations, phonetic contractions, bad punctuation, emoticons, slang language, single letters to replace normal words (Pannu and Tomar, 2010, p. 241). The primary goal of this study is increase in true positive, reduction of false positives (misclassification) and redundant alerts. In this research only Sms spam detection as a type of social engineering is studied. Any other type of social engineering attack is beyond the scope of this study.

1.8 Chapter summary

SMS generate huge volume of unstructured text data of low quality evidenced from many data sources (Almeida, 2012). To solve this problem , this work proposes a novel approach that is based on hybrid ensemble machine learning techniques that can be used to improve the SMS spam detection efficiency and precision .The effectiveness of the proposed model is empirically validated using multiple classification techniques, and the results demonstrates that the model can improve performance of SMS spam detection. This model automatically detect sms spam messages using ensemble of hybrid machine learning techniques.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter is organized as follows , it starts with an overview of the SMS technology, Sms spam as a social engineering challenge, Existing sms spam detections techniques, their challenges , and limitations of existing techniques and models including ,the gap of knowledge based on these existing model ,challenges and limitations , from these the ensemble hybrid model conceptual framework is developed.

2.1.1 The concept of Short Message Service (SMS)

Pannu and Tomar (2010) stated that the data that can be held by an SMS message is very limited. A single SMS message can have at most 0.14 kilobytes of data, with up to 160 characters are allowed, on the other hand SMS text messaging supports many languages such as English, Arabic, Malay, Chinese, Punjabi ,Japanese, Korean, Swahili among others. In addition text messages can also carry binary data, therefore it is possible to send ringtones, pictures, organization logos, wallpapers, animations, business cards to a mobile phone with this technology (Zheng et al., 2018).

Most payment plans provided by mobile carriers consist of cheap SMS messaging service, this reason makes most fraudsters find it appropriate platform to con users both in developed and developing countries. An SMS center (SMSC) normally is responsible for organizing the SMS operations of a wireless network. When an SMS message is sent from a mobile phone, it will get to an SMS center first. The SMS center then releases the SMS message to the recipient. An SMS message may need to pass through more than one network unit before reaching the destination. The main function of an SMSC is to route SMS messages and regulate the process.

If the recipient is unreachable the SMSC will store the SMS message, and then forward the SMS later when the user is available.

In practical use, SMS messages are not encrypted by default during transmission. A CRC is provided for SMS information passing across the signaling channel to ensure short messages do not get corrupted, this however is a pitfall because fraudsters can remain anonymous. Since encryption is not applied to short message transmission by default, messages could be altered using techniques such as man in the middle attack.

2.1.2 SMS spam (smishing) –A social engineering challenge

Social engineering is a form of fraud which is conducted by the perpetrators by making a deeper approach to the target to obtain the desired information. The social engineering actors make users as their target, to gain access to a given system. Often system users are not aware when social engineering hackers are targeting them.

According Anti phishing work Group (2016) recently fraudsters are using sophisticated traditional Smishing and social media to defraud internet users. They use modern up to date tricks to make it harder for responders to stop these scams. SMS spam attacks affect a lot of internet users and this is a big cost load for businesses and spam victims (Almomani, Gupta, Wan and Altaher, 2013). Gartner research (2014) found out that data given to spoofed sites yielded no profit for American banks and credit loan issuers to the amount of one billion two hundred million US dollars. Litan (2014) resonated by saying that phishing scam are becoming a significant threat to users and businesses worldwide. These studies have shown negative economic impact of these threats.

Statistica research conducted in 2018 reported that there were about 4.7 billion subscribers using mobile phones worldwide by 2017, and 3.7 billion users of the internet in 2018, clearly there is an enormous wide spread of mobile phones technologies and the significant number of

mobile devices, which is increasing rapidly, this has provided more and more opportunities for mobile transactions leading to increase in market share hence more security threats.

According Landesman (2016) SMS spam campaigns in the United States rose by 400% in the first half of 2016, and about one-third of all SMS spam includes sms spam attempts. This is not a surprise that this social engineering attacks attempts is growing so rapidly, because criminals follow money. Currently there are more than six billion cell phone subscribers in the world today and nearly 60% of all adults with cell phones use text messaging as a means of communication. Another interesting fact worth noting is the high open rate for text messages which is 90% and this happens in less than fifteen minutes (Landesman, 2012). This metric on open rate is a key factor to why many sms spam are successful, fraudsters take advantage of this immediate responsiveness to commit the fraud. These messages usually have a sense of urgency that makes recipient act without much thought. There messages may be an offer for a product for free or at a great price, or an urge to respond immediately to keep something terrible from happening (Table 1). For example, you might get a message that appears to come from your branch bank, telling you that your payment instrument is going to be canceled, unless you validate your account right away, or you can get a free offer from a retailer if you are one of the first people to visit a web page which happens to be fake, or you have won with a lottery company/betting companies or a promotion that you never took part.

Sms spamming is bad for individuals and corporates, as more and more people use their personal devices at work, corporate data and networks can be affected too. Like phishing, Sms spam can be used to install a malware such as a key logger or botnet code, these techniques can be used to harvest personal credentials.

According to the report released by Kenya communications authority (2016-2017) confirmed that number of mobile subscriptions stood at 38.5 million as at 2017. The total number of mobile money subscriptions and mobile money agents were recorded at 31.0 million and 169,698 respectively during the quarter under review. A total of 400.6 million transactions were made valued at over Ksh.1 trillion. The period also recorded 247.9 million mobile commerce transactions amounting to Ksh 447.3 billion. The total value of person to person transfers during the quarter was valued at Ksh. 474.5 billion. During this period under review, the number of messages sent through the Short Messaging Service (SMS) rose to reach 12.2 billion up from 11.6 billion sent during the preceding quarter. The volume of international incoming mobile SMS was recorded at 9.6 million messages down from 9.8 million messages recorded in the preceding quarter representing a drop of 2.1 per cent during the quarter. On the contrary, international outgoing messages rose by 62.7 per cent to record 10.7 million messages. From this report it can be seen that Kenyans are seriously indulged with mobile communication.

According to Joo and Singh (2017) Sms spam (Smishing) has continued to grow and evolve in popularity as a social engineering tool for cybercriminals. Smishing can trick the user into clicking on a link in a text message which can lead the user to entering personal data. The figure 1 shows a ten step sequence diagram that shows a Smishing attack based on universal resource locator (URL).

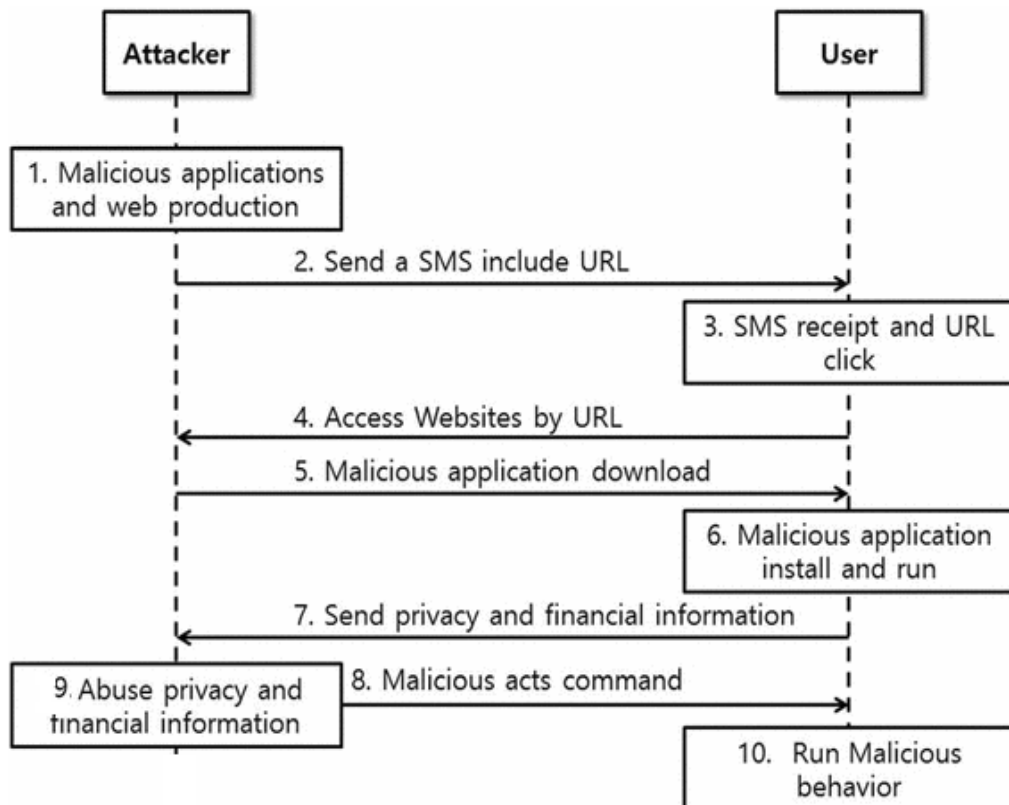


Figure 1: A sequence diagram for a smishing attack based on URL click (Joo et al., 2017).

The objective of this is to access sensitive person information such as usernames, PINs and passwords among others. Additionally, many Sms spam messages will include malicious link with malware/spyware waiting for unknowing users to click on it. If one clicks on the infected site link, it may download malware, which compromises the mobile device or the web site will ask you to enter personal information such as, social security number, credit card type, credit card number and PIN etc. If you call the phone number given, it will sound very official and will ask you to enter personal information such as, social security number and Identifications, credit card type, credit card number and your digital wallet pin. The fraudster will use this information to duplicate a debit/credit/ATM card and begin to use it. The downloaded malware software may allow the fraudster to remotely control your phone and use your phone to access

your banking information. Fraudsters can use the information collected to perform identity theft. A Smishing text message is determined on the basis of the basic attribute of the text message. Whether the text message includes a Universal Resource Locator (URL) or a telephone number or just plain text messages (Kang and Kim, 2013) as shown in figure 2

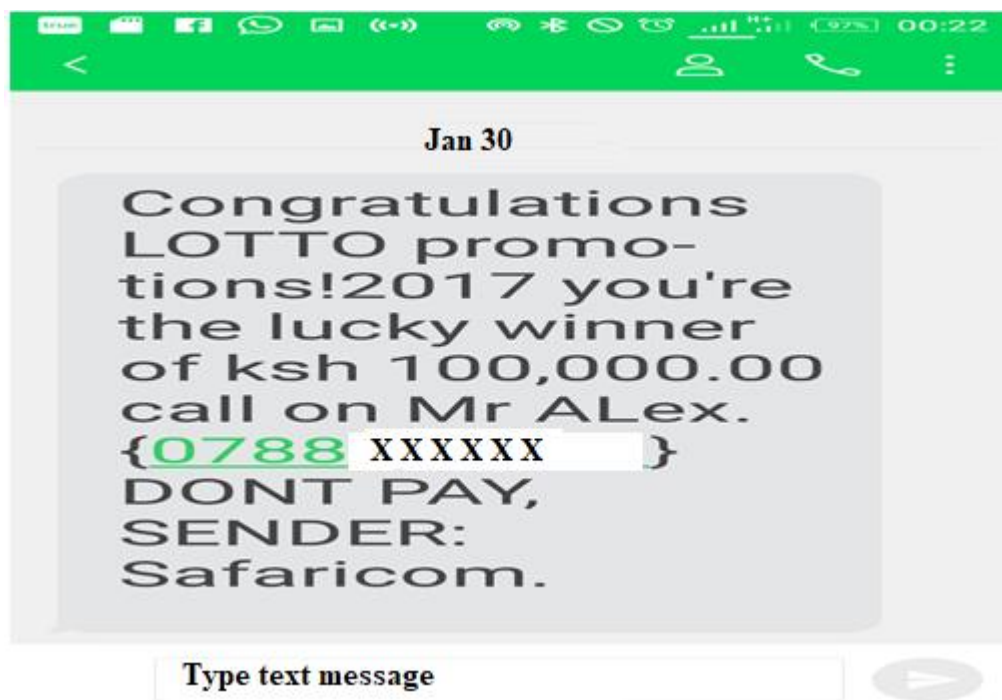


Figure 2: Client side smishing screen shot example (source, Safaricom).

Most of the time the Sms spam is easy to spot, with claims that you have won a contest you never entered or an “unclaimed refund” waiting for you. Many of the same techniques used in phishing email have made their way over to the sms world. Most SMiShing attempts will show up in unfamiliar number, making them appear suspicious to attentive users but many users may not spot the difference.

According to Mohaddes et al., (2016) recent campaign in China sent out messages saying the customer’s bank account will not be accessible, followed by instructions to log in and account

validation information. Obviously clicking on the link would take the user to a website that mimic the true bank website.

2.1.3 Challenges in smishing attack detection

Smishing detection has been a challenging task due to a number of reasons stated below:-

i) Huge volume of low quality sms data

There is a huge number of sms repositories, of which most of them are poor quality from language style, because this text messages are not formatted for grammar. This is attributed to the nature of sms which most of the time is a personal language.

ii) The SMS Language

SMS text messages are very short and can be written in Lingo language, using lingo, broadens a given language by adding more words. Language isn't static, and a language such as English is a collection and reinvention of the words of many other languages such as Latin and Greek, as well as the romance languages of Europe (Jain and Gupta, 2018). Slang language is also used in texting, these words are non-dictionary words that keeps on changing over time. Swahili language also presents another aspect of language , for example a simple word could have different meaning e.g. “Nyanya yangu amekufa” , the sentence could mean two things , the first interpretation may mean that that “my grandmother is dead” and the second meaning of the sentence is “My tomato is dead”?.

iii) The attacker behavior

One of the challenge of detecting SMiShing is that SMiShing strategies change in time, as well as attacker’s behavior is also dynamic. There are limited examples of SMiShing frauds available, which makes it difficult to model the intent of a fraudulent behavior. Many frauds through SMiShing are also not reported, and if they are reported there is a large time lapse and delay, which means few of these attacks attempts can be timely investigated. In fact mobile

banking users in China recently have been receiving sms spam texts that appear to come from their legit bank's official phone number (Deylami et al., 2016) , this makes it difficult to differentiate between spam and non-spam messages based on the SMS originator. Mobile phones are developed to move from base station to another in order to maintain a stable connection .fraudsters may use such design to set up a fake base station with sms sending tool that send out text messages that appears to be completely legitimate, anyone traveling through the area with fake station will receive the text message number (Deylami et al., 2016) This scenario raises concern on the lifespan of any learning model with time.

iv) Feature extraction

Feature extraction is important since it allows appropriate feature selection. Unlike emails it is very hard to extract relevant features in sms messages because they are generally short and many contain a lot of abbreviations, further sms spam can be multilingual, that is , it can be done with many languages in one text message(English, Swahili, Arabic), the language is dictated by the two communicating parties. , some messages may contain inverted words e. g instead of hello, an attacker would write “hello” Viagra, v1agra, use of acronyms, slang language among others. With these issues, it becomes even challenging when it comes to training a given model to detect these nature of attacks (Lota and Hossain, 2017).

v) Data mining challenge

The most basic ingredient of data mining is data, “data is the new oil”. A data mining algorithm takes as input a set of data, an individual datum in the data set as its own structure, it consists of values for several attributes, which may be of different types or take values from different ranges, conversely knowledge discovery in databases and data mining have enjoyed great recognition and success in recent years, however there is a diverse lack of a generally accepted model for data mining. The present lack of such a model is perceived as an impediment to the

further development of the field. The current state of the art for data mining research is too “ad-hoc”, and that the techniques designed are for individual problems and there is no unifying model (Yang and Wu, 2006) that cuts across this diverse field of text mining and classification.

According to Tan et al., (2016) much of the existing research in data mining is about mining complex data, e.g. text mining, web link mining, mining social network data, multi-media data mining among others which means a technique will require a lot data labeling ,and therefore many grand challenges for data mining are centered on these areas . Additionally the challenge is to treat the mining of different types of structured data in a uniform fashion, which is becoming increasingly more difficult. Another challenge include high dimensional data, integrity and privacy concerns from the data sources.

Sumathi and Sivanandam (2006) resonates that data mining algorithms work with vector valued data, and it is an important challenge to extend data mining algorithms to work with unstructured distributed data such as text messages, audio, images and videos. Data mining is a semi- automated process and should be easy to use, relevant techniques include improving user interface, supporting casual browsing and visualization of massive and distributed data sets, including developing techniques and systems to manage the meta-data required for data mining, which may involve developing appropriate languages and protocols for providing casual access to data. In addition, the development of data mining and knowledge discovery environments which address the process of collecting, processing, mining, and visualization of data, as well as the collaborative and reporting aspects is important (Sumathi et al., 2006). All these challenges need to be addressed by developing proper SMS spam solutions .This is an open and comparatively new research that needs to be exploited (Lota and Hossain, 2017).

2.2 SMS spam preprocessing techniques

Pre-processing the data is the process of cleaning and preparing the text for classification. It plays an essential role in disambiguating the meaning of short-texts, not only in applications that classify short-texts but also for clustering and anomaly detection. Pre-processing have a considerable impact on overall model performance; this techniques it is less explored in most literature in comparison to feature engineering (Naseem et al., 2020).

According to Nayak et al., (2016), pre-processing is a three step mechanism composed of Tokenization, Stop Word Removal and Stemming. Stemmers are used to consolidate terms to optimize retrieval performance and/or to reduce the size of indexing files. Stemming will, in general, increase memory at the cost of decreased precision. Studies of the effects of stemming on retrieval effectiveness are unclear, but generally stemming has either no effect, or a positive effect, on retrieval performance using various measures. Several Stemming algorithms have been developed over the years to optimize the data. Porter's Algorithm is one of the efficient techniques for the English language as per several studies, however this can be applied to multi-lingual environments (Singh and Gupta, 2020).

Vijayarani, Ilamathi and Nithya, (2016) discussed the purpose of preprocessing of text data, highlighting the applications of text mining and its various contingencies. Text mining is the process of seeking or extracting the useful information from the textual data. It tries to find interesting patterns from large corpus. On the other hand Kannan et al., (2017) analyzed the importance of pre-processing in text mining, natural language processing and information retrieval. They evaluated the issues in pre-processing methods for text archives. The study examined the need for text pre-processing in NLP systems. Their proposed pre-processing methods use tokenization, stop word removal, and stemming. Tokenization probes the sentences and makes a list of tokens which can be used as input for further algorithms. Further

their experiment shows that tokenization is one of the crucial steps in preprocessing text data because it splits the “bag of words” into identifiable words known as tokens. Tokenization also gives information indicating the frequencies of each token, which can be used in further steps of information retrieval. Stop word removal and Stemming algorithms were then applied and the output contained only those tokens that are deemed valuable by the pre-processing algorithms.

Another research by Moral et al., (2016) from Universidad Politécnica de Madrid, Spain conducted an assessment of stemming algorithms used in information retrieval applications. Their approach on stemming had three main purposes; Clustering words according to topic , improves the effectiveness of information retrieval algorithms and words sharing the same stem leads to reduction of the number of words that needs to be processed. Saif et al., (2012) found that pre-processing led to a significant reduction of the original feature space. After pre-processing, the vocabulary size was reduced by 62%. However, they did not discuss the effect on the performance of the classifiers used. Most of the studies conducted use a single or limited pre- processing technique and therefore they still lack in accuracy and precision.

According to Jianqiang and Xiaolin (2017) before feature selection, a series of pre-processing (e.g., removing stop words, removing URLs, replacing negations) are applied to reduce the amount of noise in the text. Pre-processing is performed extensively on existing approaches, especially in machine learning-based approaches (Yoon et al., 2016). However, few studies focus on the effect of pre-processing method on the performance in text analysis.

The study by ElKah and Zeroual (2021) , in their experiments , three pre-processing techniques and all their possible combinations were considered: stop words removal, stemming, lemmatization, stop words removal and stemming, stop word removal and

lemmatization, stemming and lemmatization, and finally, all the three techniques combined on naïve Bayes , support vector machine and decision trees algorithms , these algorithms achieved the best results when all the three pre-processing techniques were involved (an average of +28.73%). This experiment further confirms that a well-selected ensemble of pre-processing techniques have a great impact on text classification. In summary, ensemble pre-processing allows the selection of several techniques and their combinations that, in a complementary way, leads to an improved model (Mishra et al., 2020).

2.3 Machine learning classification algorithms for feature engineering

Cormen et al., (2009) defined an algorithm as a well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output, thus a sequence of computational steps that transform the input into the output. For example machine learning algorithms are organized into taxonomy, based on the desired outcome of the algorithm: - Supervised, Unsupervised and reinforcement are the most common categories of machine learning algorithms. Supervised Learning consists of a target / outcome variable (or dependent variable) which is to be predicted from a given set of predictors (independent variables). Using these set of variables generated, a function maps inputs to desired outputs. The iterative training process continues until the model achieves a desired level of accuracy on the data set, Examples of Supervised Learning include Decision Tree, Random Forest, Kth nearest Neighbor (KNN), and Logistic Regression among others.

Unsupervised Learning does not have any target or outcome variable to predict or estimate, It is used for clustering population in different groups, which is widely used for segmenting customers and/or users in different groups for specific intervention, Examples of Unsupervised Learning includes hierarchical clustering k-Means algorithm, Gaussian mixture models, Self-organizing maps, Hidden Markov models among others (Hastie et al., 2009). In Reinforcement

learning, the machine is trained to make specific decisions by being exposed to an environment where it trains itself continually using trial and error. The machine learns from past experience and tries to capture the best possible knowledge to make accurate business decisions; an example of reinforcement learning is Markov decision process. The following is a list of algorithms and techniques that can be paired with feature engineering techniques in order to achieve better classification.

2.3.1 Naïve bayes classification

Lee and Kim (2013) stated that Naïve Bayesian classifier that is based on Bayes Theorem, a probabilistic classifier is simple and universal enough to be applied to document classification. Even though its assumptions are very simple, Naïve Bayesian classifier works better than expected in complicated actual conditions (Dietterich et al., 2016). In terms of text classification, it shows similar performance as neural network or decision tree learning. The larger the data, the higher accuracy it has (Lee et al, 2013). The naïve bayes equation can be written as follows:-

$$P(C_k/x) = P(x/C_k) P(C_k) P(x) \text{ (Kim and Lee, 2013).}$$

Where the data x of attribute value is given, it determines whether the data belongs to the specified class C_k , $p(C_k)$ is prior probability, meaning prediction before the relevant work occurs. $P(C_k|x)$ is the probability of the data x in the condition of the specified class C_k . Here, based on the given data of $p(C_k/x)$, $p(C_k)$, $p(x)$. The probability is calculated and can be used to come to a conclusion as to what the class data x belongs to as shown in Figure 3.

Likelihood

Class prior probability

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

Posterior probability

Predictor prior probability

$$P(c | x) = P(x_1 | c) \times P(x_2 | c) \cdots P(x_n | c) \times P(c)$$

Figure 3: A Naïve bayes algorithm implementation equation

Russell and Norvig (2003) also defined Naive Bayes classifiers are a family of simple probabilistic classifiers based on Bayes' theorem with strong independence assumptions between the features. It has been studied extensively since the 1950s., and was introduced under a different name into the text retrieval community in the early 1960s, it remains a baseline method for text categorization, the problem of judging messages as belonging to one category or the other (spam or ham) with word frequencies is used as features. With appropriate pre-processing, it is competitive in the classification domain with more advanced methods including support vector machines (Rennie, Shih, Teevan, and Karger, 2003).Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of features/predictors in a learning problem, and is very suitable for text classification because of the nature of text data. Maximum-likelihood training can be done by evaluating a closed-form expression (Russell et al., 2003) which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

Naive Bayes (NB) is a classifying algorithm uses data about prior events to estimate the probability of future events (Raschka, 2014).While many algorithms typically ignore features with weak effects, Naïve bayes uses all available features information to subtly change

predictions (Bali, Sarkar and Lantz, 2013). It works by evaluating the probability of different text data belonging to different classes. Most of the current sms Spam detection systems use keywords to detect spam. These keywords can be written as intentional misspellings for example: baank or bannk instead of bank. Misspellings are changed from time to time and hence spam detection system needs to constantly update the blacklist of keywords to detect spam containing these misspellings (Hamsapriya and Renuka,2010) . A Naive Bayes classifier will converge quicker than most algorithm models, it requires less training, it is easy to build and particularly useful for very large data sets (Goyal,2012). Naive Bayes is known to outperform even highly sophisticated classification methods (Romero, 2010) .Naive Bayes algorithm uses the following steps:-

1. Given the training data set D that contains messages belonging to two different classes for example A for spam and B for Ham

2. i) First Calculate the prior probability of class A=number of objects of class A / Total number of objects

ii) Calculate the prior probability of class B=number of objects of class B / Total number of objects

3. Find n_i , i.e. the number of word frequency of each class

n_a = The total number of word frequency of class A

n_b = The total number of word frequency of class B

4. Find conditional probability of keyword occurrence given a class by:-

$P(\text{word1}/\text{Class A} = \text{WordCount} / n_i(A).$

$P(\text{word1}/\text{Class B} = \text{WordCount} / n_i(B).$

$P(\text{word2}/\text{Class A} = \text{WordCount} / n_i(A).$

$P(\text{word2}/\text{Class B} = \text{WordCount} / n_i(B).$

.

$P(\text{word}_n/\text{Class B} = \text{WordCount} / n_i(B).$

Avoid Zero Frequency problem by applying uniform distribution.

Classify a new message C on the probability

$P(C/W)$

By Finding $P(A/W)=P(A)* P(\text{word1}/\text{Class A})*P(\text{word2}/ \text{Class A}).....* P(\text{Word}_n/\text{class A})$

By finding $P(B/W)=P(B)* P(\text{word1}/\text{Class B})*P(\text{word2}/ \text{Class B}).....* P(\text{Word}_n/\text{class B})$

Assign text message to a class that has higher probability as spam or Ham.

Algorithm 1: Naïve bayes algorithm

Another flavor of naive bayes is Multinomial Naïve Bayes algorithm that is based on naïve bayes classifier theorem; it uses multinomial distribution on conditional probabilities function. In this research thesis the researcher used the term frequency (tf (t, d)), as the number of times a given term t such as word appears in a given message d, the term frequency is normalized by splitting the raw term frequency by the d given as length as per the normalization term frequency equation (Xu et al., 2017) in eq (1).

$$\text{Normalized term frequency} = \frac{tf(t, d)}{n_d} \quad \text{Eq (1)}$$

Thus -

- tf (t,d) raw term frequency
- n_d : The accumulate number of terms in the message.

In eq (2) the term frequencies is used to calculate the maximum likelihood estimate based on the training data for spam/ham. Class-conditional probabilities for multinomial model as given by the formula:-

$$\hat{P}(x_i | \omega_j) = \frac{\sum tf(x_i, d \in \omega_j) + \alpha}{\sum N_{d \in \omega_j} + \alpha \cdot V} \quad \text{Eq (2)}$$

Where

- x_i : a word from the feature space vector x.

- $\sum_{d \in \omega_j} \text{tf}(x_i, d)$: The sum of raw term frequencies of word x_i from all messages in the training belonging to a given class ω_j .
- $\sum_{d \in \omega_j} N_d$: The total of all term frequencies in the training dataset for class ω_j .
- α : smoothing based on Laplace smoothing in this case $\alpha=1$
- V : Total number of dissimilar words in the training set.

The class based conditional probability of encountering the text say x is calculated as the product from the likelihoods of the individual words as per naïve bayes assumption property of conditional independence.

On the other hand multi-variant Bernoulli model is based on binary data. Every token in the feature vector of a document is associated with the value 1 or 0. The feature vector has \mathbf{m} dimensions where \mathbf{m} is the number of words in the whole vocabulary (See Section on the Bag of Words Modeling; the value 1 means that the word occurs in the particular document, and 0 means that the word does not occur in this document). The Bernoulli model is written as (Ortega et al., 2020).

$$P(\mathbf{x} | \omega_j) = \prod_{i=1}^m P(x_i | \omega_j)^b \cdot (1 - P(x_i | \omega_j))^{(1-b)} \quad (b \in 0, 1). \quad \text{Eq (3)}$$

Let $\hat{P}(x_i | \omega_j)$ be the maximum-likelihood estimate that a particular word such as a token x_i occurs in class ω_j

$$\hat{P}(x_i | \omega_j) = \frac{df_{x_i, y} + 1}{df_y + 2} \quad \text{Eq (4)}$$

Where

- $d_{f_{x_i,y}}$ is the number of messages in the training dataset that contain the feature x_i and belong to a given class ω_j (spam, not spam)
- d_{f_y} is the number of messages in the training dataset that belong to a given class ω_j (spam, not spam)
- $+1$, $+2$ are the parameters for Laplace smoothing

2.3.2 Support vector machine

Support vector machines (SVM), also support vector networks are supervised learning models with associated learning algorithms that analyzes data used for classification and regression analysis (Cortes et al, 2020). Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a framework that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM framework is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible (figure 4). New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. In summary Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. The model extracts a best possible hyper-plane / line that segregate the two classes as shown in the diagram below. One way to avoid high dimensional input spaces using SVM is to assume that most of the features are irrelevant. Feature selection tries to determine these irrelevant features however in text categorization

there are only few irrelevant features, this can be done using information gain and then applied to a classifier of the feature ranks i.e. it is trained using only those features.

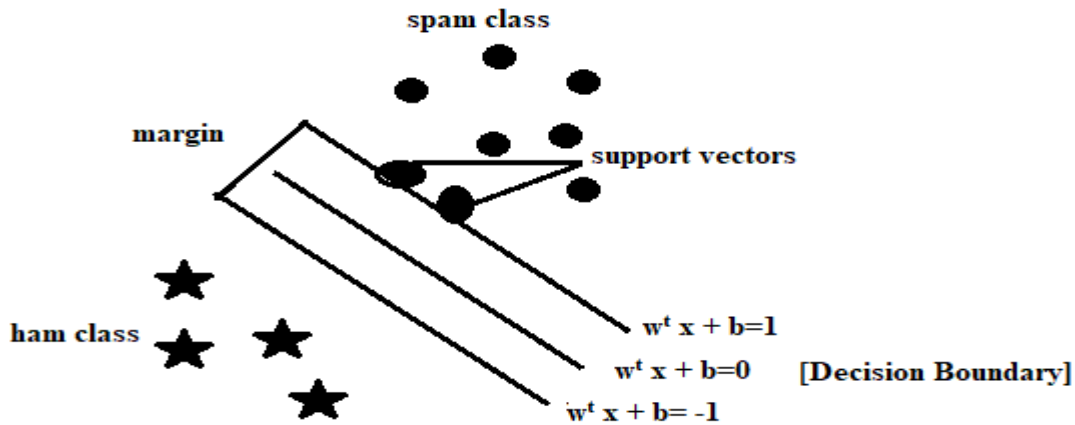


Figure 4: SVM boundary decision scheme

Sequential minimal optimization is a technique for solving the quadratic programming (QP) problem arising during the training of support-vector machines (SVM). SMO is widely used for training support vector machines and is implemented by the popular LIBSVM tool in WEKA. At each step SMO chooses two elements α_i and α_j to jointly optimize, It find the optimal values for those two parameters given that all the others are fixed, and updates the α vector accordingly (Tahir , 2020).

2.3.3 Artificial neural network

Artificial neural network contains a collection of computing elements that are highly networked and can transform a set of inputs to a set of desired outputs through a threshold function. The result of the transformation is primarily determined by the behavior of the single elements and the weights associated with the network involved. A neural network performs an analysis of these information and then provides a probability estimate value that it matches with the data it has been trained to know. The neural network gains the experience by training the model or

a system with both the input and output for a given problem. The network configuration continuously refined until satisfactory optimized results are achieved. The experience is gain over a period of time, as it is being trained on the data related to the problem. A multi-layer feed forward for example is capable of making multi-class classifications.

Training a neural network involves arranging all the weight by considering two major steps, forward and backward propagations, in forward we give a collection of weights to the input and then figures out the output, for the initial feed forward propagation the inputs are chosen randomly, while in backward propagation we ration the margin of error of the output and then modified the weights accordingly to minimized error. Neural network iterate both forward and backward propagation till the weights are balanced inorder to predict the output precisely. The processing layer is used in the hidden layers of the neural network that sums the input with weights and maps the accurate output, some of the function used to achieve this includes linear, sigmoid, hyperbolic tangents, ReLu among others. Typically in Neural networks the first stage is the feed forward where each input (x_i) receives an input signal then send it to the hidden units (z_1, z_2, z_3, \dots), then computes its activation signal and then sends that signal to the output unit, where each unit uses the activation signal to compute the output or response for the given input signal or pattern. Once the output value is generated then each output unit checks its activation (y_k) With the target value (t_k) that is the value needed to obtained after the input pattern is processed as seen in figure 5.

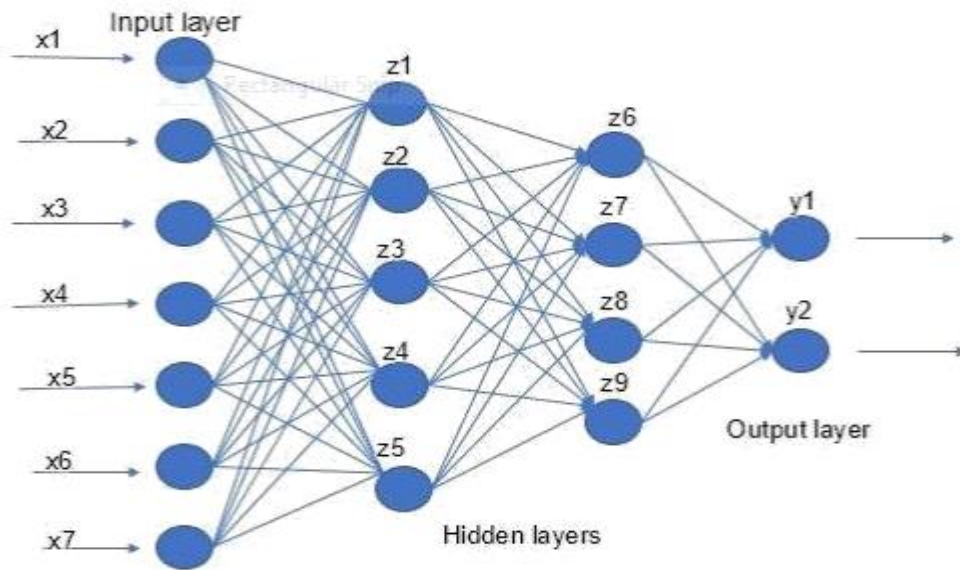


Figure 5: Artificial neural network.

In natural language processing, nowadays the core task in text is how to present features. Most techniques used include bag of words, unigram, bigram and mostly N-gram. In order to extract more useful and distinct features, many methods have been developed such as Latent Dirichlet allocation (LDA) (Hingmire, Chougule, Palshikar, and Chakraborti, 2013) that compares precision, recall and F-measure, PLSA (Cai and Hofmann, 2003), frequency and MI (Cover and Thomas, 2012). In spite of the fact that many researchers have developed sophisticated features such as tree kernel (Post and Bergsma, 2013) to extract more contextual information and accurate word order, there still exist issues with data sparseness as it has great impact on the classification accuracy. In deep machine learning, most successful learning methods involve deep neural networks that involves pre-trained word embedding. Word embedding is a distributed feature learning over sequences of words that tackles the sparseness issue. Pre-trained word embedding has the ability to extract useful syntactic and semantic regularities. To construct sentence representation (Socher, Huang, Pennington and Manning, 2011a) suggested the idea of Recursive Neural Network (Recursive NN) that has ended up being effective.

Recursive NN has the ability to extract the semantic of a sentence by using tree structure technique. Its performance heavily relies on upon the execution of the textual tree development. Nevertheless, the time complexity of contracting such textual tree is at least exponential $O(n^2)$ where n is the length of the text. If a sentence or document is so long, this approach would be time consuming. Additionally, it can be very difficult to develop a relationship between two sentences by a tree structure. Consequently, Recursive NN is unsatisfactory for modeling long sentences or document. Recurrent Neural Network (Recurrent NN) is another type of model that shows a time complexity of linear time $O(n)$. This model investigates a text word to word and saves the semantics of all the past text in a rigid-sized hidden layer (Elman, 2014). There is no doubt that it has the ability to capture the semantics of a big text but it is a biased model, because it focuses on later words than earlier words. This minimizes the efficiency of capturing the semantics of a whole document as all the words have the same probability to appear in the sequences of words. Word Vectors have a long, rich history in NLP, but all methods depend in some way or another on distributed hypothesis that states that words that appear in the same context share same semantic meaning. Multi-Layer Perceptron algorithm. A perceptron algorithm is a type of neural network, consisting of only one neuron, and is typically used for pattern recognition. A perceptron attempts to separate input into a positive and a negative class with the aid of a linear function (Freund, Schapire and Abe, 1999). The inputs are each multiplied by weights, random weights at first, and then summed. Based on the sign of the sum a decision is made.

In order for the perceptron to make the right decision, it needs to train with input for which the correct outcome is known, so that the weights can slowly be adjusted until they start producing the desired result. It is a classification algorithm that makes its predictions based on

a linear predictor function combining a set of weights with the feature vector. The variables are first defined as:-

$y = f(z)$. Denotes the output from the perceptron for an input vector z

$D = \{(x_1, d_1), \dots, (X, d_s)\}$ is the training set of s samples where:

X_j is the n -dimensional input vector

D_j is the desired output value of the perceptron for that input

The value of the features can be shown as follows

X_{ji} is the value of the i^{th} feature of the j^{th} training input vector

$X_{ji} \in \{0, 1\}$

To represent the weights we have:-

W_i is the i^{th} value in the weight vector, to be multiplied by the value of the i^{th} input feature

Unlike other linear classifications there is no need for a learning rate in the perceptron algorithm (Montavon, Orr and Müller, 2012).

2.3.4 J48 decision tree based algorithm

According to Quinlan (1993), J48 classifier is a simple C4.5 decision tree for classification, which is an extension of Quinlan's earlier ID3 (Iterative Dichotomiser 3) algorithm. It is based on binary tree. The decision tree approach is most useful in classification of a problem. With this technique, a tree is constructed to model the classification process, once the tree is built; it

is applied to each tuple in the database and results in classification for that tuple (Dunham, 2006). Algorithm 2 shows the decision Tree based algorithm.

```
Input: currentNode, trainingData
buildNode(currentNode, trainingData) Data: splitInformation
splitInformation = getSplitInformation(trainingData);
if splitInformation.action == LEAF then
currentNode.type = LEAF;
currentNode.probability = calculateProbability(trainingData);
return;
end
Data: rightBranchTrainingData;
forall the message in trainingData do
if message.featureValue(splitInformation.feature) larger than
splitInformation.featureThreshold then
rightBranchTrainingData = message;
trainingData.remove(rightBranchTrainingData);
end
end
buildNode(currentNode.leftBranchNode, trainingData);
buildNode(currentNode.rightBranchNode, rightBranchTrainingData);
```

Algorithm 2: Decision Tree algorithm

Decision tree learning is a method commonly used in data mining (Rokach and Maimon, 2008). The goal is to create a model that predicts the value of a target variable based on several input variables. In the tree build up, J48 ignores the missing values i.e. the value for that item can be predicted based on what is known about the attribute values for the other records. The idea is to divide the data into range based on the attribute values for that item that are found in the training sample. J48 for example allows classification via either decision trees or rules

generated from them. Algorithm 2 depicts how the nodes in the tree is being built recursively node by node from left branch then right branch inorder. The decision to split a node or not to, is decided by calling the get-Split-Information function. This method analyses whether a split is possible by checking if training threshold data is available. Inorder to allow a split, a minimum amount of training messages has to be exceeded. If this is the case, it computes the information gain feature and information gain ratio for every feature by being fed the training data in the current node. If a split is possible, it returns the feature values the node should split messages on. In this case the feature with the maximum information gain ratio is then selected. In case of several splits with the same gain ratio the first one is selected. Each possible split must also have an information gain higher than the average information gain of all the possible evaluated splits for it to be chosen. If doing a split is not possible, then it will tell the buildNode algorithm to make a leaf out of the node. When the current node becomes a decision node the training data are split into two groups, the messages with the current feature having a value over the split threshold is moved to the right branch, while the others are moved into the left. This is done recursively as seen in chapter 4 of this study.

```

Input: currentNode
Data: errorLargestBranch = 0
Data: errorNodeAsLeaf = 0
Data: errorSubTree = 0
if currentNode.type != LEAF then
prune(currentNode.leftBranchNode);
prune(currentNode.rightBranchNode);
if usingSubtreeRaising then errorLargestBranch = errorLargestBranch(currentNode);end
errorNodeAsLeaf = errorForNode(currentNode);
errorSubTree = errorForTree(currentNode);
if errorNodeAsLeaf less than or equal to errorSubTree AND (not
UsingSubTreeRaising OR errorNodeAsLeaf less than or equal to errorLargestBranch) then
currentNode.type = LEAF;end
Else if subTreeRaising AND errorLargestBranch less than or equal to errorSubTree then
Data: minorBranchTrainingData
MinorBranchTrainingData = currentNode.getMinorBranchData ();
redistributeTrainingDataToSubtree(currentNode.getLargestBranch,
minorBranchTrainingData);
currentNode = currentNode.getMajorBranchNode();
End return;

```

Algorithm 3: Decision Tree Pruning algorithm technique

After the tree has been constructed using the appropriate methods, the optional pruning process is then executed. This process has two major steps to achieve, one is to find out the errors if any for the current node, the left and right sub tree. Two is to make a decision whether pruning is required on the node or not. In case pruning is required, this is done using a recursive function from top to bottom of the tree and from left to right. At the leaf level there will be no pruning possible therefore backtracking is done. In algorithm 3 if the current node is a leaf the method will instantly backtrack, else the method prune is recursively called on for the branches of the node, in order to climb up the tree. The pruning then starts by calculating the necessary errors for deciding whether the pruning should continue or stopped. If there should be a sub tree replacement or if sub tree rising should be done.

Sub tree rising is only done in pruning if specifically it is enabled. The sub tree pruning chooses the branch with the least amount of training data leading to it, and then redistributes the data to the larger branch. Thus there will be new probabilities in the leaves because of the new data reaching this point.

The bigger sub tree root replaces the current node and the pruning step is completed successfully. When this process is initialized, the outcome is a built pruned decision that can now be used for classification as seen in chapter 4 of this research.

2.3.5 Apriori algorithm

This algorithm is used to generate association rules which can applied to new SMSs to classify them into spam or legitimate. Association rule is one of the most research areas in data mining and machine learning. Association rule mining is a valuable tool that has been used widely in various areas (Nofal et al., 2011). Association rule uses two criteria, support and confidence. Ishtiaq et al., (2015) proposed a SMS spam classification algorithm using the combination of Naive Bayes classifier and Apriori algorithm. They integrated association rule mining using Apriori algorithm with Bayesian algorithm. Apriori is used to retrieves the most frequent words occurred together then Bayesian calculates the probability of occurring a word independently and together with other words, in spam or ham messages. The main disadvantage of Apriori algorithm is of Apriori algorithm is time and space. It generates numerous uninteresting item sets which lead to generate various rules which are of completely of no use.

2.3.6 Kth nearest neighbor

The kth nearest neighbor algorithm support both classification and regression , it works by storing the entire training data set and querying it to locate the K most similar training patterns when making a prediction as seen in algorithm 4, it is a simple algorithm , but one that does not assume very much about the problem other than that the distance between data instances

, it is meaningful in making prediction, it is for this reason that its performance is good. KNN normally take the mean of the K most similar instance in the training data set especially for regression problems. The size of the neighborhood is controlled by the k parameter known as IBk in WEKA, if k is set to 1, then the predictions are made using the single most similar training instance to given new pattern. Other parameters used by KNN include the Euclidean distance and Manhattan distance (Brownie 2019)

for each corpus **do**

 Split into corpus.training and corpus. Test

for each corpus-comp \rightarrow corpus and corpus-comp.class \rightarrow corpus.class **do**

knn= knn(corpus.data, corpus-comp.data, corpus.target, corpus-comp.target)

Results= knnfoward(corpus.target)

end for

 Compute average accuracy

 Compute voting accuracy

end for

Algorithm 4: Ensemble KNN algorithm

2.4 Feature engineering techniques

In text feature engineering involves identifying important features and removing irrelevant and less important ones. According to Chorghé et al., (2016) there are several relevant features to an email phishing which can also be found in SMS, and these features have found their way in mobile applications. Phishing Sms have lots of occurring keywords like verify, cancel/suspend, winning. Generally a setup of keywords could include congratulations, promotions, Lucky winner, don't pay sender among others. Zhang and Wang (2012) in their study revealed some new aspects of the common features that appear in phishing URLs, and introduce a statistical machine learning classifier to detect the phishing sites, which relies on selected features, their

model, however did not utilize an ordinary feature extraction method. Basnet et al., (2012) used sum of keyword divided by cumulative total of words in an email. Some small amount of keywords if available in emails are counted and organized while cluster of words with same meaning use same feature (Basnet et al., 2012).

2.4.1 Feature selection

The main reason behind the feature selection is that classifiers trained on reduced feature space are more robust and accurate than classifiers constructed on the original large feature space. In feature selection, the researcher particularly searched for features or correlated features. The features which do not provide useful information are called irrelevant features and the features which do not provide more information than the currently selected features are called redundant features (Vipin and Sonajharia, 2014). The features which are not related or uncorrelated to class (spam , not spam) variables are called noise which actually introduces bias in prediction and reduce classification performance. Hence, noise should be handled for improving the performance of prediction and it can be made possible with dimensionality reduction. It can be achieved by either Feature extraction or by Feature selection (Kumar et al., 2014).

In feature extraction, new features are derived from the original input by choosing a new basis for the data. Feature selection helps in reducing the effect of high dimensionality on the dataset by finding the subset of features which will effectively define the data. Directly evaluating the subset of features becomes the NP-Hard problem (Chandrashekar and Sahin, 2014). Feature selection can be used in the reduction of large data in text classification .It enhances the classification process by removing irrelevant and noisy unwanted data and then picks a representative subset of all data to minimize the complexity of the classification process. The existing class content correlation models deals with huge amount of data that may contain no values, incomplete information and non-irrelevant features, this may be time consuming and

error prone especially for a learning model (Chandrashekar and Sahin, 2014). Features set is selected from correlational content based feature selection. Correlation based feature selection normally select features that are highly correlated with a given class (spam or not spam) and are not correlated with each other. Most redundant occurring features are deleted because of high correlation with the residual feature.

Recent studies have showed that machine learning techniques can be affected by irrelevant ,noisy and redundant text information(Noureldien and Yousif , 2016). Most machine learning algorithm are very sensitive to irrelevant features and this leads to slow training and exponential growth in time and space complexity(Aziz, Verma and Srivastava, 2016). The naïve bayes for example is affected by redundant features due to its premise that the features do not dependent on the overall class (Juanchaiyaphum , Arch-int and Saiyod ,2015).Decision trees such as J48 sometime over fit training data, this may result to dense trees (Manandhar and Aung 2014).

Feature selection as a dimensional reduction technique has been the focus in many text classification (Onan, Korukoğlu and Bulut, 2016). It identifies some of the important attributes in a data set that are appropriate in identifying a class by reducing the number of features and deleting irrelevant, redundant and noisy features. Apart from reducing time to train it facilitates visualization of data, boosts modeling, prediction and speeds up the classification process.

Dimensionality reduction techniques like feature extraction and feature selection can be used in machine learning to optimize results. Feature extraction techniques attempt to transfer the input features into a new feature set, while Feature selection looks for the most important

information features from the original input (text message) therefore enhancing the performance of the detection model.

2.4.2 Feature selection process for SMS spam detection

Feature selection process contains four basic steps (Kumar and Minz, 2014) illustrated in figure 6 subset generation, subset evaluation, stopping criterion and results validation.

i) Subset generation Subset generation is a process in an informed search strategy (heuristic search), where with each state in the space we define a candidate subset for evaluation. This process is determined by deciding the start point that will influence the search direction. The search may start with an empty set and successively add features, or start with a full set and successively remove features or done using a bidirectional technique (Doak, 1992).

ii) Evaluation of subset

In this step every new generated subset is evaluated by an evaluation function. The quality of a subset is always determined by a certain criterion, and this criterion selected using one criterion may not be optimal according to another criterion. The evaluation criteria can be broadly categorized into two groups: independent and dependent criteria (kumar and Minz, 2014).

Independent criteria considers the important characteristics of the training data without involving any mining techniques to evaluate the quality of a feature set. The dependent criteria involve known mining techniques for feature selection to select features based on the performance of the mining algorithm.

iii) Stopping criterion

A stopping criterion is used to determine when the feature selection process should be halted, some the criterion includes.

1. Completing a search
2. Some given upper bound or lower bound is reached (minimum number of features or maximum number of iterations).
3. Subsequent addition or removal of any feature that does not produce a better subset results
4. A satisfying results of a subset is selected that is based on some error threshold.

iv) Validation of results

Feature selection method must be validated by carrying out different tests and comparing the results with previously established results or comparison with the results with competing methods using artificial datasets, real world datasets, or both.

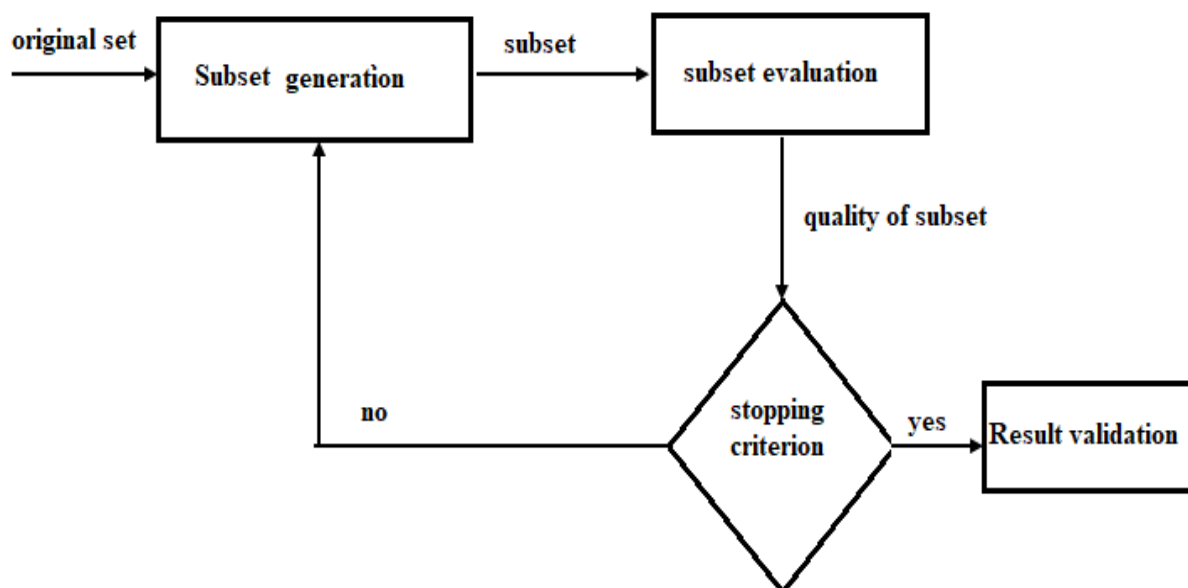


Figure 6: Four key steps of the feature selection process.

There are three general approaches of feature selection: Filter, wrapper and hybrid

i) The filter method

The filter approach incorporates an independent measure for evaluating features subsets without involving a learning algorithm M . This approach is computational efficient. However, filter methods can miss features that are not useful by themselves but can be very useful when combined with others.

Filter Algorithm

```
input:    $D(F_0, F_1, \dots, F_{n-1})$  // a training data set with  $N$  features
           $S_0$  // a subset from which to start the search
           $\delta$  // a stopping criterion
output:  $S_{best}$  // an optimal subset

begin
  initialize:  $S_{best} = S_0$ ;
   $\gamma_{best} = eval(S_0, D, M)$ ; // evaluate  $S_0$  by an independent measure  $M$ 
  do begin
     $S = generate(D)$ ; // generate a subset for evaluation
     $\gamma = eval(S, D, M)$ ; // evaluate the current subset  $S$  by  $M$ 
    if ( $\gamma$  is better than  $\gamma_{best}$ )
       $\gamma_{best} = \gamma$ ;
       $S_{best} = S$ ;
  end until ( $\delta$  is reached);
  return  $S_{best}$ ;
end;
```

Algorithm 5: Generalized filter algorithm (Yu, 2005).

ii) The wrapper method

The wrapper algorithm is very similar to filter algorithm only that it utilizes a predefined data mining algorithm A instead of an independent measure M for subset evaluation. For each generated subset S , it evaluates its fitness by applying the mining algorithm A to the data with feature subset S and evaluates the goodness of the mined results. The variable D represent the function generated. In this case different mining algorithms will produce different feature selection results. Varying the search strategies via the function generated (D) and mining

algorithms (**A**) can give different results based on the wrapper algorithms (Wijaya, Saptono and Douwes, 2016).

Mining algorithms are primarily used to control the feature selections, the wrapper model tends to give greater performance as feature subsets found are better suited to the predetermined mining algorithm. Consequently, the wrapper method is more computationally expensive as compared to filter method (Yu, 2005).

```

Wrapper Algorithm
input:    $D(F_0, F_1, \dots, F_{n-1})$  // a training data set with  $N$  features
            $S_0$  // a subset from which to start the search
            $\delta$  // a stopping criterion
output:  $S_{best}$  // an optimal subset
begin
  initialize:  $S_{best} = S_0$ ;
   $\gamma_{best} = eval(S_0, D, A)$ ; // evaluate  $S_0$  by a mining algorithm  $A$ 
  do begin
     $S = generate(D)$ ; // generate a subset for evaluation
     $\gamma = eval(S, D, A)$ ; // evaluate the current subset  $S$  by  $A$ 
    if ( $\gamma$  is better than  $\gamma_{best}$ )
       $\gamma_{best} = \gamma$ ;
       $S_{best} = S$ ;
  end until ( $\delta$  is reached);
  return  $S_{best}$ ;
end;

```

Algorithm 6: Generalized wrapper algorithm (Yu, 2005)

iii) Hybrid algorithm

The hybrid model is meant to handle huge data sets (Das, 2001). A typical hybrid algorithm (algorithm 7) makes use of independent measure and a mining algorithm to evaluate feature subsets.

The hybrid model normally uses independent measure to decide the best subsets for a given cardinality and uses the mining algorithm to select the final best subset among the best subsets

across different cardinalities. It starts the search from a given subset it iterates to find the best subsets at each increasing cardinality. In each round for a best subset with cardinality c , it searches through all possible subsets of cardinality $c + 1$ by adding one feature from the remaining features.

Each newly generated subset S with cardinality $c + 1$ is evaluated by an independent measure M and compared with the previous best one. If S is better, it becomes the current best subset S'_{best} at level $c + 1$.

At the end of each iteration, a mining algorithm A is applied on S'_{best} at level $c + 1$ and the quality of the mined result Θ is compared with that from the best subset at level c . If S'_{best} is better, the algorithm continues to find the best subset at the next level; otherwise, it stops and outputs the current best subset as the final optimal subset. Like mention earlier the goodness of the results from a mining algorithm provides the stopping criterion in the hybrid model (Yu, 2005).

Hybrid Algorithm

input: $D(F_0, F_1, \dots, F_{n-1})$ // a training data set with N features
 S_0 // a subset from which to start the search

output: S_{best} // an optimal subset

begin

initialize: $S_{best} = S_0$;
 $c_0 = \text{card}(S_0)$; // calculate the cardinality of S_0
 $\gamma_{best} = \text{eval}(S_0, D, M)$; // evaluate S_0 by an independent measure M
 $\theta_{best} = \text{eval}(S_0, D, A)$; // evaluate S_0 by a mining algorithm A

for $c = c_0 + 1$ **to** N **begin**

for $i = 0$ **to** $N - c$ **begin**

$S = S_{best} \cup \{F_j\}$; // generate a subset with cardinality c for evaluation
 $\gamma = \text{eval}(S, D, M)$; // evaluate the current subset S by M
 if (γ is better than γ_{best})

$\gamma_{best} = \gamma$;
 $S'_{best} = S$;

end;

$\theta = \text{eval}(S'_{best}, D, A)$; // evaluate S'_{best} by A
 if (θ is better than θ_{best});

$S_{best} = S'_{best}$;
 $\theta_{best} = \theta$;

else;

break and return S_{best} ;

end;

return S_{best} ;

end;

Algorithm 7: Hybrid feature selection algorithm (Xing, Jordan and Karp, 2001).

In text categorization there are massive volume of online text data on the Internet such as emails, sms, social sites, e commerce site among others. Therefore, automatic text categorization and clustering is an important task. A major problem with text classification or clustering is (Kumar et al, 2014) is the high dimensionality of the document features. A moderate size text document may have hundreds of thousands of features. Therefore, feature selection (dimension reduction) is highly enviable for the efficient use of mining algorithms (Saghapour, Kermani and Sehhati, 2017). In many literature, many applications of feature selection techniques are effectively used in the area of text mining. In summary Filter method

is more accurate and particularly effective in computation time and robust to over fitting (Hamon, 2013). It is great while doing exploratory data analysis (EDA), it can also be used for checking multi co-linearity in data. Wrapper methods main disadvantages are: increased over fitting risk when the number of observations is insufficient and the significant computation time when the number of variables is large. Embedded method combine the advantages of filter and wrapper ,it takes advantage of its own variable selection process and performs feature selection and classification simultaneously, It gives more accurate results but computationally expensive(Saghapour, Kermani and Sehhati, 2017).

2.5 Correlation-based feature selection for machine learning

A central problem in machine learning is identifying a representative set of features from which to construct a classification model for a particular task. This study addresses the problem of feature selection for machine learning through a correlation based technique. The central hypothesis is that good feature sets contain features that are highly correlated with the class, yet uncorrelated with each other Cai et al., (2018). Correlation feature selection is a simple filter algorithm that ranks feature subsets according to a correlation based heuristic evaluation function. The bias of the evaluation function is toward subsets that contain features that are highly correlated with the class (spam or not spam)and uncorrelated with each other. Irrelevant features should be ignored because they will have low correlation with the class. Redundant features should be screened out as they will be highly correlated with one or more of the remaining features (wahba et al., 2015). The acceptance of a feature will depend on the extent to which it predicts classes in areas of the instance space not already predicted by other features. In many literature cases CFS gave comparable results to the wrapper, and in general, outperformed the wrapper on small datasets (CFS executes many times faster than the

wrapper, which allows it to scale to larger data sets (Hall, 2009). The following equation 5 gives the merit of a feature subset S consisting of k features:

$$\text{Merit}_{S_k} = \frac{k\overline{rcf}}{\sqrt{k + k(k-1)\overline{rff}}}. \quad \text{Eq (5)}$$

Here, \overline{rcf} is the average value of all feature-classification correlations, and \overline{rff} is the average value of all feature-feature correlations. The CFS criterion is defined as follows:

$$\text{CFS} = \max_{S_k} \left[\frac{r_{cf_1} + r_{cf_2} + \dots + r_{cf_k}}{\sqrt{k + 2(r_{f_1f_2} + \dots + r_{f_i f_j} + \dots + r_{f_k f_{k-1}})}} \right]. \quad \text{Eq (6)}$$

The $r_{cf_1} + r_{cf_2} \dots r_{cf_k}$ variables are referred to as correlations, but are not necessarily Pearson's correlation coefficient or Spearman's ρ . Hall (1999) dissertation used neither of these, but used three different measures of relatedness, minimum description length (MDL), symmetrical uncertainty, and relief.

In eq(7) x_i is the set membership indicator function for feature f_i ; then the above can be rewritten as an optimization problem that can be solved using branch and bound algorithm((Hai, Katrin and Slobodan ,2009).

$$\text{CFS} = \max_{x \in \{0,1\}^n} \left[\frac{(\sum_{i=1}^n a_i x_i)^2}{\sum_{i=1}^n x_i + \sum_{i \neq j} 2b_{ij} x_i x_j} \right].$$

Eq (7)

Several researchers have explored the possibility of using a particular learning algorithm as a pre-processor to discover useful feature subsets for a primary learning algorithm. Hall (1999) describes the application of decision tree algorithms to the task of selecting feature subsets for use by instance based learners. C4.5 was applied to three natural language data sets; only the features that appeared in the final decision trees were used with a k nearest neighbor classifier. The use of this hybrid system resulted in significantly better performance than either C4.5 or the k nearest neighbor algorithm when used alone. In a similar approach, Cheng & Greiner (2013) used a greedy oblivious decision tree algorithm to select features from which to construct a Bayesian network. Oblivious decision trees differ from those constructed by algorithms such as C4.5 in that all nodes at the same level of an oblivious decision tree test the same attribute. Feature subsets selected by three oblivious decision tree algorithms where each employing a different information theoretic splitting criterion—were evaluated with a Bayesian network classifier on several machine learning datasets. Results showed that Bayesian networks using features selected by the oblivious decision tree algorithms outperformed Bayesian networks without feature selection and Bayesian networks with features selected by a wrapper.

Arif et al., (2018) described an algorithm called RELIEF that uses instance based learning to assign a relevance weight to each feature. Each feature's weight reflects its ability to distinguish among the class values. Features are ranked by weight and those that exceed a user-specified

threshold are selected to form the final subset. The algorithm works by randomly sampling instances from the training data. For each instance sampled, the nearest instance of the same class (nearest hit) and opposite class (nearest miss) is found. An attribute's weight is updated according to how well its values distinguish the sampled instance from its nearest hit and nearest miss. An attribute will receive a high weight if it differentiates between instances from different classes (spam and not spam) and has the same value for instances of the same class. Practical machine learning algorithms often make assumptions or apply heuristics that trade some accuracy of the resulting model for speed of execution, and comprehensibility of the result. While these assumptions and heuristics are reasonable and often yield good results, the presence of irrelevant and redundant information can often fool them, resulting in reduced accuracy and less understandable results. Feature subset selection can help focus the learning algorithm on the important features for a particular problem. It can also reduce the dimensionality of the data, allowing learning model to operate faster and more effectively.

2.5.1 Symmetrical uncertainty feature selection method

Symmetrical Uncertainty (SU) feature evaluation filter method assess the usefulness of an attribute by evaluating the symmetrical uncertainty in relationship to the class. This attribute evaluation filter requites for the bias in IG(Weka, 2014). Symmetrical uncertainty of a can be expressed as shown in equation in the equation 8.

$$SU(X, Y) = \frac{2 * Gain(X|Y)}{H(X) + H(Y)} \quad \text{Eq (8)}$$

where, $H(X)$ is the entropy of the discrete random variable, and $H(X|Y)$ is the condition impurity/entropy which computes the other uncertainty of a stochastic variable given that the value of other random variable is well known.

2.5.2 Information gain

IG measures the number of information bits obtained for class c_i prediction by knowing the presence or absence of a term t_k in a text message (Mesleh, 2011; Ghareb et al., 2016). Information gain (IG) is used as a measure the worth of an attribute based on entropy, (measure of uncertainty) .Gain ratio is also an important aspect of information gain proposed by ross Quinland (1993) meant to reduce a bias towards multi-valued attribute by taking the number and size of branches in a decision tree to choose an attribute. IG (information gain) can reduce the dimension of vector space model by setting the threshold, but the problem is that it is too hard to set the appropriate threshold(Kaufman,1997).

$$IG(t_k, c_i) = \sum_{c \in \{c_i, \bar{c}_i\}} \sum_{t \in \{t_k, \bar{t}_k\}} P(t, c) * \log\left(\frac{P(t, c)}{P(t) * p(c)}\right)$$

Eq (9)

IG feature evaluation is a Waikato environment for knowledge analysis (WEKA). Filter method that checks the value of feature by measuring the Information gain (IG) in relation to the class (spam or not spam). Information Gain can be simplified as:

$$\text{Information gain (Class, Attribute)} = P(\text{Class}) - P(\text{Class}|\text{Attribute})$$

2.5.3 Mutual information

Mutual information also known as gain ratio (GR) is a measure of dependence relationship between variable (t_k and a class c_i), that is, it determines the worthiness of a feature by calculating the gain ratio with respect to the spam and non spam class (Giveki et al., 2012). If the mutual information for a term the $t_k = 0$ then a term t_k and a class c_i are independent. The deficiency of mutual information is that the score is extremely impacted by marginal

probabilities of words. The mutual information can be defined as per equation 10 (Mesleh, 2011; Ghareb et al., 2016):

$$MI(t_k, c_i) = \frac{\log(P(t_k, c_i))}{P(t_k) * P(c_i)} \quad \text{Eq (10)}$$

Gain Ratio (GR) feature check in WEKA assess the usefulness of a feature by measuring the GR in relation to the class. It is not a uniform filter that satisfy present bias issues found with information gain .This can be simplified by the equation 10.

$$GR(Class, Attribute) = \left(H(Class) - \frac{H(Class | Attribute)}{H(Attribute)} \right) \quad \text{Eq (11)}$$

2.5.4 Chi- square feature selection method

This method measures the absence of independency between text in a message language features such as verbs, nouns, punctuations and dictionary words plays an important role in sms spam detection. In addition topics dealing with money, leisure, death, and winning ,are good indicators for spam .Chi -square feature selection method (yang et al., 2016) is very effective in this context when proper training and testing methods are used. In the Chi-square formula eq (11) the best terms t_k for the given class c_i (spam, not spam) are the ones distributed most differently in the sets of positive and negative examples of the class c_i .

$$\text{Chi - square}(t_k, c_i) = \frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)} \quad \text{Eq (12)}$$

In the equation 11 the variables are described as follows.

N = Total number of messages i ,

A = Number of messages in class c_i that containing the term t_k ;

B = Number of messages that contain the term t_k in other classes;

C = Number of messages in class c_i that do not contain the term t_k ;

D = Number of messages that do not contain the term t_k in other classes.

Each feature is assigned a score in each class, all these scores are combined with a single final score $\max(\text{Chi-square}(t_k, c_i))$ (Bahassine, S., et al. F, 2018).

2.5.5 Dimensionality reduction

In machine learning and statistics, dimensionality reduction or dimension reduction is the process of reducing the number of random variables under consideration (Roweis and Saul, 2000) via obtaining a set of principal variables. It can be divided into feature selection and feature extraction. It refers to the process of converting a set of data having vast dimensions into data with lesser dimensions ensuring that it conveys similar information concisely. These techniques are typically used while solving machine learning problems to obtain better features for a classification or task. Dimension reduction reduces the time and storage space required. It becomes easier to visualize the data when reduced to very low dimensions. In this research dimensional reduction was applied to the corpus in order to convey same information, this helped to save time, algorithm such as potter stemming , tokenization and stop words came in handy in achieving dimensional reduction .In addition slang words dictionary was used to handle the informal words which is very common in Kenya. The tokenization algorithm 8 steps are illustrated below:-

```
Data: begin, end=0
  Data: Raw_messages
  Result: tokens
  While begin is not end of message do
  Begin = nonDelimiterCharacterPosition (message, end);
  End = delimiterCharacterPosition (message, start);
  Tokens += startToEnd (start, end, message);
  End
```

Algorithm 8:Tokenization algorithm

2.6 Multilingual natural language processing

Solving a natural language processing problem may be demanding and a complicated phenomenon. It is used to express human thoughts, and through language, we receive information and concluded a meaning. Understanding linguistics is important in natural language processing, though the expressions are not unorganized. Further they show structure of different kinds and complexity and consist of more elementary components whose co-occurrence in context refine the notions they refer to in isolation and implies further meaningful relations between them. Language experts for decades have been looking at different aspects of languages in detail for example morphology is the study of forms and functions of words , the syntax involves the arrangement of words into phrases, the sentences and Structure of words problem due to pronunciation are described by an area known as phonology. In linguistic, words are perhaps the most intuitive unit of any language that is sometimes difficult to define. Knowing how to handle them, in particular the development of syntax and semantic is the key in multilingual natural language processing.

2.6.1 Components of words

Words are the basic stemmed unit of a phrase that can stand on its own. The parts of words that deliver meaning to them are known as morphemes. Depending on communication method, morphemes are constructed using graphemes which are symbols for writing letters or characters. In linguistics, it is sometimes difficult to decide and agree on the precise sentence boundaries in separating words from morphemes and phrases (Martin, 2017).

2.6.2 Word tokens

Most words in linguistics are separated by whitespace and punctuation, for example in English the statement "will you take the balance? , Will you take it? I won't take it." To handle the syntax and etymology of these statements, two words are worth considering: "balance" and "won't" this being a compound word, "balance" has an interesting derivational structure. Distinguishing words in linguistics can be very challenging. Linguists prefer to analyze words independently and then revert to its normalized form. In English, this kind of tokenization and normalization may apply to just a limited set of cases, but in other languages these phenomena have to be treated in a less trivial manner (Cohen and Smith, 2007).

2.6.2.1 Lexemes

A lexeme can be defined as a meaningful linguistic unit that is, an item in the vocabulary of a language. Lexemes can be categorized by their behavior into the lexical levels. In English for example this includes verbs, nouns, adjectives, conjunctions, particles, or other parts of any speech. Another important concept in lexemes is a lemma, this is a word or phrase defined in a dictionary. When a word is converted into other forms, such as conversion may be thought of as a lemma. When we convert a lexeme into another lexeme that is morphologically associated, regardless of its lexical category, we say we obtain the lexeme: for instance, the nouns "mshidi" and "mshindani" are derived from the verb "shinda" in Swahili.

2.6.2.2 Typology

Morphological typology divides languages into groups by characterizing the prevalent morphological phenomena in those languages. It considers various criteria, and during the history of linguistics, different classifications have been proposed (B. Bickel and J. Nichols, 2008). The following is a typology based association between words, their morphemes, and attribute selected (Bikel and Zitouni ,2013).

Isolating typology: most languages involves no or relatively few words that would comprise more than one morpheme.

Synthetic languages typology: they have more morphemes in one word and can be further sub divided into fusional languages.

Agglutinative languages typology: they have morphemes associated with only a single function.

Fusional languages typology: are defined by their feature as per morpheme ratio which is must be higher than one.

Concatenate languages typology: relating morphs and morphemes sequentially.

2.6.2.3 Morphological models

There exist a lot of approaches in developing and implementing morphology models. For a longer period, linguistics have recorded an improvement on a number of formalisms and frameworks, in particular grammars of different types and expressive power, with which to address whole classes of problems in natural processing as well-known formal languages.

2.6.2.4 Dictionary lookup

Dictionary can be thought of as data structure that directly involves obtaining some pre-computed results, which are word analysis. Hence dealing with dictionaries it is important to optimize it inorder to allow quick search and lookup. Lookup operations are relatively simple

and usually easy to implement. As a data structure the dictionary can be implemented as lists, binary search trees, graphs, hash tables etc. Since the set of relationship between word forms and their desired descriptions is declared by plain listing, the coverage of the model is finite and the generative potential of the language is unknown (Bikel & Zitouni ,2013). Developing as well as verifying the association list is unexciting, lots of errors, inefficient and not accurate unless the data are retrieved automatically from large and reliable linguistic sources .Despite all that, an enumerative model is often sufficient for the given purpose, deals easily with exceptions, and can implement even complex morphology. For instance, dictionary based approaches to Swahili depend on a large dictionary of all possible combinations of all morphs and morphological alternations. Most of these approaches do not allow development of reusable morphological rules.

2.6.2.5 Finite state morphology

Finite-state models are computational devices extending the power of finite-state automata. This models consist of a finite set of vertices connected by edges labeled with pairs of input and output symbols. In such a graph, vertices are also known as states, the edges on the other hand called links or arcs. Visiting the network from the set of initial states to the set of goal states along the links is equivalent to reading the order of encountered input symbols and writing the sequences of corresponding output symbols. In languages such as Swahili for example, a finite-state model could analyze the surface string *watoto* into the lexical string *motto* [plural]. This associations on languages can also be viewed as functions.

For example, suppose a relation R denoted by $[\Sigma]$, the set of all order over some set of symbols Σ , the range of R is a subsets of $[\Sigma]$. Therefore R is a function mapping an input string into a set of output strings.

$$R:: [\Sigma] \rightarrow \{[\Sigma]\} R:: \text{String} \rightarrow \{\text{String}\} \quad \text{Eq (13)}$$

This model has been the subject of study for long especially on algebraic properties and have proven to be suitable for man language problems. Their applications encoding the surface rather than lexical string associations as rewrite rules of phonology and morphology have been around since the two-level morphology model, further presented in Computational Approaches to Morphology and Syntax and Morphology and Computation(Roark and Sproat,2007). Morphological operations and processes in human languages can, in the overwhelming manner of cases and to a sufficient degree, can be expressed in finite state terms. Beesley and Karttunen (2003) stress concatenation of transducers as a method for factoring surface and lexical languages into simpler models and propose a somewhat unsystematic compile replace transducer operation for handling non-concatenative phenomena in morphology. On the other hand Roark and Sproat (2007) argued that constructing a morphological models in general using transducer composition, is a pure universal approach. A theoretical limitation of finite-state models of morphology is the problem of capturing reduplication of words or their elements (e.g., to express plurality) found in most formal languages. A formal language that contains only words of the form λ^{1+k} , where λ is some arbitrary sequence of symbols from an alphabet and $k \in \{1, 2, \dots\}$ is an arbitrary natural number indicating how many times λ is repeated after itself, is not a regular language, not even a context-free language. General reduplication of strings of unbounded length is thus not a regular language operation (Roark and Sproat, 2007).

2.6.2.6 Sentence boundary detection

The purpose of sentence boundary construction is to link the message elements within a text. SBD is the first processing step in nearly all the natural language processing applications. The

features normally used for developing the sentence splitter - μ are obtained from the tri-gram contexts of training corpus, that includes the words before and following the potential boundary punctuation marks such as period(.), exclamation mark (!), colon (:), semicolon (;), question mark(?), quotation marks (“), brackets (), and dash (-). Normally, the punctuation marks of the period and exclamation and question marks are only considered as the potential sentence boundaries in related SBD studies (Stamatatos Fakotakis, and Kokkinakis, 2014). However, there are some cases where the punctuation marks may also denote a sentence boundary. On the other hand to maximize the system adaptability for more languages and text genres, the features associated are converted into a binary vector similar to one-hot encoding style, every module corresponds to a possible feature value of a feature in the feature set extracted. The converted feature- type for constructing the SBD system is therefore non-dependent of specific corpus and alphabet language since the system does not directly rely on the orthographic information.

In SBD the initial upper case character of a word, including the immediately preceding word $f_1(w_{i-1})$ and the after word $f_2(w_{i+1})$ is characterized by the feature function as illustrated in eq(14).

$$f_{1,2}(w_i) = \begin{cases} 1 & c_1 \in \mathcal{C}, w_i = \{c_1, c_2, \dots, c_n\} \\ 0 & \text{otherwise,} \end{cases} \quad \text{Eq (14)}$$

where \mathcal{C} is the collection of capital letters; $c_i \in \mathcal{C}$ is a character, and n is the word length $|w|$. The upper casing of a word gives a necessary clue to signal the named entities and objects such as names of places, people, entities among others (Lee and Rim,2004).

2.6.3 Word features

In natural language processing word are feature and are very important in determining a message meaning. The word before and after a specific word contribute a lot in determining the context of a particular word. The following feature method is important for word neighboring to determine the meaning of a sentence.

$f_{3(w_{i-1})}$ and $f_{4(w_{i+1})}$ is defined as.

$$f_{3,4}(w_i) = \begin{cases} 1 & \forall c_i \in \mathcal{C}, w_i = \{c_1, c_2, \dots, c_n\} \\ 0 & \text{otherwise.} \end{cases} \quad \text{Eq(15)}$$

Features 3 and 4 is considered as the previous words and next words, that is, $f_{3(w_{i-1})}$ and $f_{4(w_{i+1})}$. Since in in natural language processing the order of words matter especially in a multilingual corpuses. Features 5 and 6 can be considered as the previous words and next words, that is, $f_{5(w_{i-1})}$ and $f_{6(w_{i+1})}$, respectively and this can be given as

$$f_{5,6}(w_i) = |w_i|. \quad \text{Eq (16)}$$

Abbreviations is most widely used in shortening a word or a phrase, typically it is constructed using the first letter or letters of the word or a phrase. In word feature space, abbreviations introduces a major source of ambiguities in sentence boundary determination.

2.6.3.1 Punctuations, numbers and symbol features

Word features are used to build the surrounding context in general. However, in order to maximize the capability of detecting the SBD in text with different genres and domains, the

researcher must include the non-lexical text information as features, that concerns the pattern of numbers, punctuations and symbols. The pattern collocation of punctuation marks and symbols leads to the ambiguities and makes the boundaries detecting a challenging task (Agarwal, Ford and Schneider, 2005). Therefore the features used to extract this information includes $f(w_i, w_{i+1})$, for detecting the collocation of colon with hyphen , fullstop/period and semicolon.

$$f_7(w_i, w_{i+1}) = \begin{cases} 1 & w_i = \text{colon}, w_{i+1} \in \mathcal{P}_{\text{dps}} \\ 0 & \text{otherwise,} \end{cases} \quad \text{Eq (17)}$$

Where $\mathcal{P}_{\text{dps}} = \{\text{dash, period, semi colon}\}$. The checking of kshs sign $f_8(w_i)$ and number $f_9(w_i)$ is given by

$$f_8(w_i) = \begin{cases} 1 & w_i = \text{"kshs"} \\ 0 & \text{otherwise,} \end{cases}$$

$$f_9(w_i) = \begin{cases} 1 & w_i \in \mathcal{N} \\ 0 & \text{otherwise,} \end{cases} \quad \text{Eq (18)}$$

Where \mathcal{N} is numeric literals. $f_{10}(w_i, w_{i+1})$ describes the expression of potential punctuations followed by either dash or left quotation mark:

$$f_{10}(w_i, w_{i+1}) = \begin{cases} 1 & w_i \in \mathcal{P}^*, w_{i+1} \in \mathcal{P}_{\text{dq}} \\ 0 & \text{otherwise.} \end{cases} \quad \text{Eq (19)}$$

\mathcal{P}^* is the potential punctuations that signal the boundaries of sentences, and $p_{dq} = \{\text{dash, left_Quotes}\}$. $f_{11}(w_i, w_{i+1})$ on the other hand, denotes the expression that excludes left quote which immediately follows the boundary terminal p^* and is defined as

$$f_{11}(w_i, w_{i+1}) = \begin{cases} 1 & w_i \in \mathcal{P}^*, w_{i+1} \notin \mathcal{P}_q \\ 0 & \text{otherwise,} \end{cases} \quad \text{Eq (20)}$$

Where $p_q = \text{left_Quote}$.

2.6.3.2 Natural language parsing

Breaking down a sentence to its component words in order to determine the grammatical type of each word or to break a sentence into more easily processed form is known as parsing. In other words this is the breaking up a sentence into atomic levels. The reason for this is to analyze a sentence and bring out its meaning truly.

2.6.3.3 Top-down parsing

Natural Language processing has two commonly used parsing techniques; Top-Down and Bottom Up parsing. In Top-Down parsing technique a parser determines the syntactic structure of a message by looking at its constituent using a parse tree starting from the root node S to the children of the root node and down to the leaves. The top down parsing algorithm begins by assuming the input can be derived by the designated start symbol S. The next step is to find the tops of all the trees which can start with S, by looking on the grammar rules with S on left hand side, all the possible trees are generated. Top down parsing is a goal directed search (Jurafsky and Manning, 2000). The search starts from the root node labeled (level 0). I.e. starting symbol, construct the child nodes by applying the rules with left hand side equals to S, further expands the internal nodes (siblings). Using next productions with left hand side equals

to internal node, if non terminal, and continues until leaves are Part-of-speech (terminals). If the leaf nodes i.e. Part-of-speech do not match the input string, we need to backtrack to the latest node processed and apply another production. Top-Down parsing is similar to preorder tree traversal.

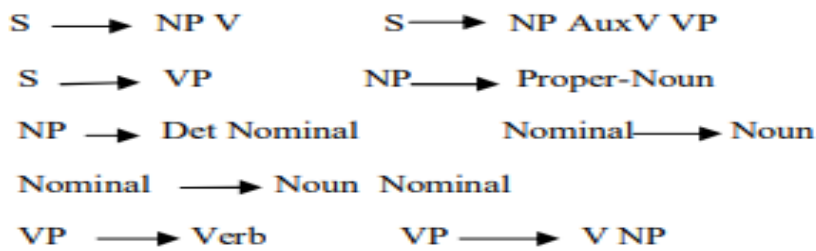


Figure 7 Grammar rules.

Taking the sentence: “John umeshinda pesa”, and applying Top-down parsing we get

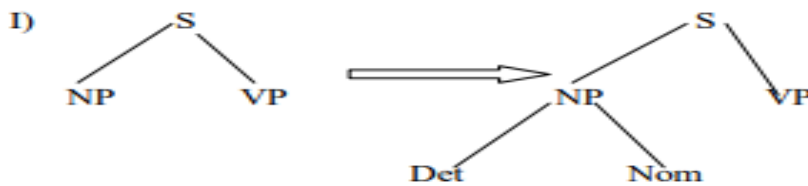


Figure 8: Top down parsing.

Since Part-of-speech does not match the input string, backtrack to the node NP

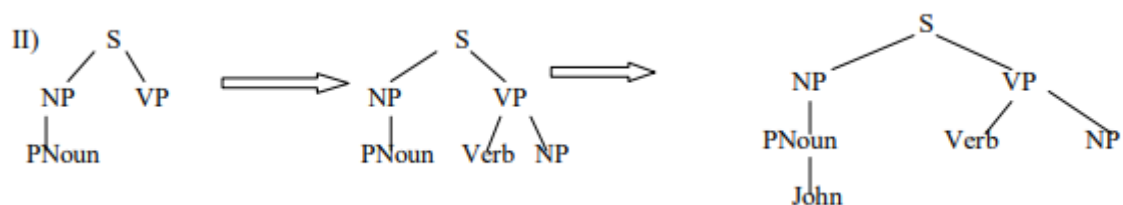


Figure 9: Top down backtracking.

Part-of-speech verb does not match the input string; backtrack to the node S, since PNoun is matched

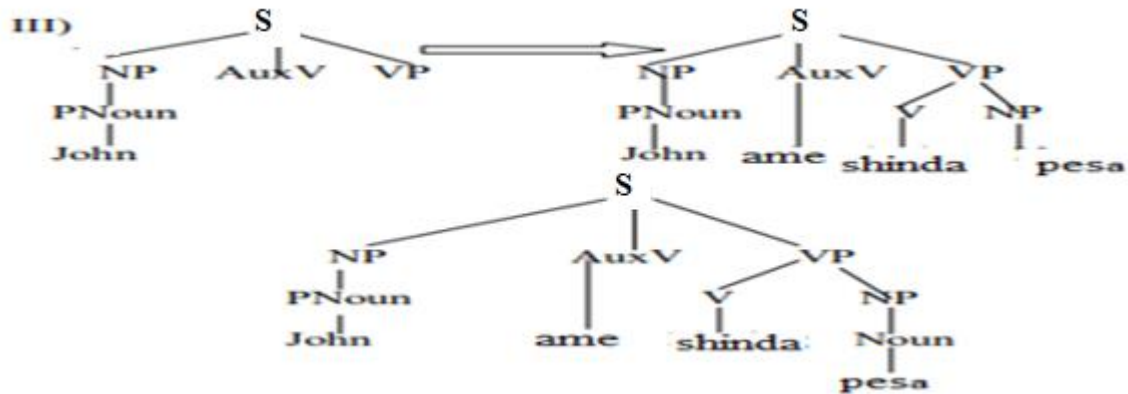


Figure 10: Final parsed tree for a given sentence.

Top-Down method advantage is that it does not waste time traversing the trees that aren't providing viable solution that is it never explores sub-trees that cannot find a place in some root tree. One of the disadvantage of top down method is that it leads to backtracking (Jurafsky and Martin 2000). It also spends considerable effort and time on trees that are not consistent with the input. These weakness in Top-Down parsing arises from the fact that they can generate trees before examining the input. While traversing and expanding the non terminals it becomes difficult to decide which right hand side production method should be selected.

2.6.3.4 Predictive parsing

Predictive parsing provides the solution to the backtracking problem in top-Down Strategy. Predictive Parsing is characterized by the use of the most next (k) tokens inorder to select which production to apply way ahead .Basic idea is given A & A b, the parser should be able to choose between "a" and "b". To make the correct choice it needs First(a) sets and Follow(A) sets. First(a) sets describes the set of tokens that appears as the first symbol in some string that

derives from “a”. Follow(A) is the set of tokens that appears immediately to the right of A in some sentential form. Predictive parsing imposes restriction on the grammar to be used i.e., the grammar must possess LL (Chang ,Quang and Roth ,2006) property, in which the first L states that can scan the input from left- right, second L says we create leftmost derivation first and ‘1’ means an input symbol for look-ahead-Grammar method should not be left -recursive.

2.6.4 Machine translation

Machine translation is the process converting a language it to another language for example from English to Swahili and vice versa.it involves the following.

1. Token separation also known as lexical analysis -. These tokens may be of any type in vector set tokens which are separated in this phase.
2. Token identification - After tokens are separated, they are identified in this step. This separation is of type noun, verb, adverb etc.
3. Sentence type determination after token identification, the type of the sentence is determined in this step.
4. Determination of phrases – Depending on step 3 above certain decisions are need to be taken. For example if a sentence is a phrase, exclamation and question, then different type of rules on the subtasks are needed to be followed as per the translation.
- 5 The language is generation after completing all the steps above steps.

2.6.4.1 Decoding a language model

A statistical based language model is a probability distribution over sequences of words in a sentence/paragraph. If such a sequence exist of length n we can assigns a probability

(P_{w_1, \dots, w_n}) to the entire sequence. The language model provides a way to distinguish between words and phrases that looks alike.

2.6.4.2 Cube pruning

Cube Pruning uses a binarised Synchronous Context Free Grammar -SCFG , therefore every node of the decoding hyper graph, any combination of a rule with the two possible children must be considered, this will result in a three dimensional search space.

Cube pruning as a K-best parsing algorithms (Zoo et al., 2013) is applied to machine translation (Chiang, 2007). K – best parsing algorithm is a heuristic algorithm used accelerate machine translation. Cube Pruning is applied to a monotonic search space, this method can be optimized with K-best algorithm to lower the time and space complexity in a search space.

2.6.4.3 Visualization

The primary goal of data visualization is to communicate information clearly and efficiently via statistical graphics, plots and information graphics. Numerical data may be encoded using dots, lines, or bars, to visually communicate a quantitative message (Friendly, 2008). It is concerned with methods that foster the perceptual, cognitive and creative capabilities of users in order to support data exploration, analysis, and presentation. Visualization trees, Threshold curve and ROC (receiver operating characteristic) curves are some of the visualization tools used in this thesis as shown in chapter 4.

2.6.4.4 Visualization trees

The Visualization tree algorithm goal is to build a tree like structure according to a set of rules from the training data set, for unlabeled data. In this study, the researcher considered a well-

known J48 a variant of iterative dichotomiser 3 (ID3) algorithm .This algorithm builds the decision tree based on entropy and the information gain. Entropy measures the impurity of an arbitrary collection of samples while the information gain calculates the reduction in entropy by partitioning the sample according to a certain feature (Lee and Lee 2006). The algorithm can be implemented as follows.

-
- 1) Entropy of every feature (F) at every value (V) in the training dataset was calculated.
 - 2) The feature F and value V with the minimum entropy was chosen. If more than one pair has the minimum entropy, arbitrarily one is chosen
 - 3) The dataset was split into left and right sub trees at $\{F, V\}$ pair.
 - 4) Steps 1-3 is repeated on the sub trees until all the resulting dataset was pure, i.e., only contains one category.
-

Algorithm 9: Entropy Algorithm

The pure data are contained in the leaf node as shown in Figure 11, by splitting dataset according to minimum entropy, the resulting dataset has the maximum information gain and thus impurity of the dataset is minimized. After the tree is built from the training dataset, testing data point can be fed into the tree. The testing data went through the tree according to the predefined rules until reaching a leaf node state as seen figure 11. The label in the leaf node is then assigned to the testing data.

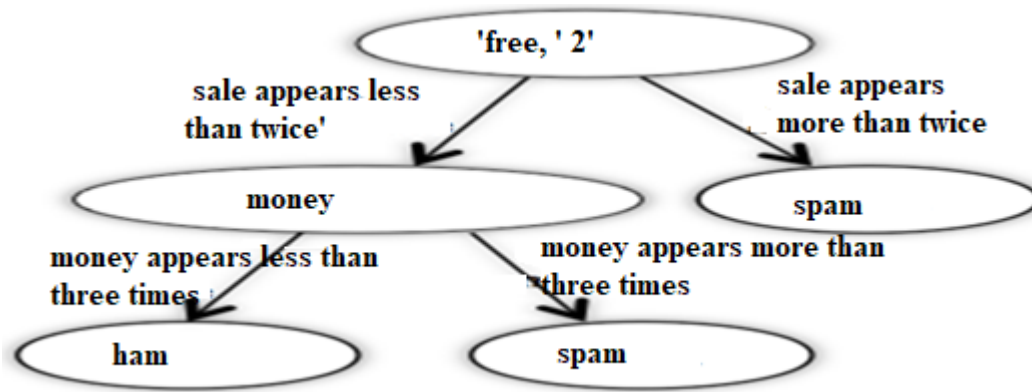


Figure 11: Sample decision tree.

2.6.4.5 Receiver operating characteristics

One of the earliest use of ROC graphs in machine learning was Meistrell and Spackman (1989), who demonstrated the value of ROC curves in evaluating machine learning algorithms. Recent years have seen an increase in the use of ROC graphs in the machine learning community, due in part to the realization that simple classification accuracy is often a poor metric for measuring performance (Provost and Fawcett, 1997; Provost et al., 1998). In addition to being a generally useful performance graphing method, they have properties that make them especially useful for domains with skewed class distribution and unequal classification error costs. These characteristics have become increasingly important as research continues into the areas of cost-sensitive learning and learning in the presence of unbalanced classes. ROC graphs are conceptually simple, but there are some non-obvious complexities that arise when they are used in research. There are also common misconceptions and pitfalls when using them in practice. ROC graphs are two-dimensional graphs in which tp (true positives) rate is plotted on the Y axis and FP (false positive) rate is plotted on the X axis. A ROC graph depicts relative tradeoffs between benefits (true positives) and costs (false positives). To compare classifiers researchers have reduced receiver operating curve (ROC), efficiency to one scalar value representing

expected performance outcome. Commonly used method is to compute the area under the curve, abbreviated AUC (Hanley and McNeil, 1982; Bradley, 1997). Because the area under the curve part of the area of the unit square, and the value is between 0 and 1.0. However, since random guess output a straight diagonal line between the points (0, 0) and (1, 1), that has an area approximately 0.5, through experiments, no classifier should have an area under the curve (AUC), less than 0.5.

The AUC has a beneficial statistical property i.e. the AUC of a classifier is similar to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. These curves are very useful for visualizing and evaluating classifiers in a much clearer way. They also provide extensive measure of classification performance than scalar measures such as accuracy, error cost, error rate, since they simplify classifier performance from class skew and error costs, they have advantages over other evaluation measures such as precision and recall. In this these methods have been used for visualizing and evaluating the classifiers cost.

2.7 Ensemble classification using machine learning

Ensemble learning is a machine learning paradigm where several learners are trained to solve the same problems. In contrast to single machine learning approaches which try to learn one hypothesis from training data, ensemble techniques tends to construct a set of hypotheses and combine for a given problem. According to Brown and Kunicheva (2010) the underlying principle of ensemble learning is an acknowledgment that in real-world situations, that each single model has limitations and will always make errors. Given that each single model will have these limitations the aim of ensemble learning is to manage their strengths and weaknesses, leading to the best possible decision being taken overall. Several theoretical and empirical results have shown that the accuracy of an ensemble can meaningfully exceed that

of a single model (Brown et al., 2010). Ensemble methods operate best when the classifiers are as independent of one another. One way to get diverse classifiers is to train them using very different algorithm approaches, as this increases the chance that they will make very different types of errors, therefore improving the ensemble's accuracy (Dietterich et al.2016), moreover the performance of the ensemble can only improve upon the use of best base classifier. If the ensemble has a sufficient pool of accurate and diverse classifiers, then the model will be more accurate and precise. The success of ensemble depends on a balance between accuracy and diversity.

An ensemble model contains several classifiers normally know as base learners. Abstraction form of an ensemble is normally more stable than the base learners. Correctly, an ensemble learning is effective in that, it is able to enhance weak learners which are slightly better than random guess to strong learners which can make very accurate predictions. Therefore, “base learners” can also be referred to as weak learners. It is significance, that nevertheless majority of theoretical study work on weak learners, base learners experiment in practice may not be necessarily weak since using non weak base learners often results in better accuracy.

Base learners are normally generated from training data set by a base learning technique or algorithm such as naïve bayes (NB), artificial neural network (ANN), a pruned decision tree (PDT), or any other kind of machine learning technique. Majority of ensemble methods/models use one base learning algorithm style to give a homogeneous base learners. Other methods also exists that use multiple learning algorithms to produce heterogeneous learners but this depends on the nature of the problem being solved. In the heterogeneous case there is no single base learning algorithm and thus, some research work names the learners as singly learners.

A study conducted by Wang, Fan, Yu, and Han (2013), determine that predictions made by combining a set of classifiers are accurate than predictions made by one best single classifier.

2.7.1 Ensemble feature selection techniques

There exists several methods for model combination. The base of ensemble classification systems is to create a set of accurate and diverse classifiers, then combine their outputs such that it outperforms all single classifiers (Basnet, Sung and Liu, 2012). Classifiers are diverse if they accurately predict and make errors at different instances, these ensemble can be built in two stages: generation and combination. In the generation step the individual components of the ensemble model also known as the base classifiers are generated, this can be done by iterating through the training set (Donghai et al., 2014).

In the second step, the decision is made by the ensemble members which are combined are combined to obtain one global decision, this is done using classifier fusion and selection process. In classifier fusion, the decision from all the members of the ensemble are combined in a manner to make a final ensemble decision, some of the strategies used include average, voting, weighting and meta algorithms. Meta-algorithms combine several machine learning techniques into one predictive model in order to decrease variance (bagging), bias (boosting) and improve predictions (stacking). The best criteria is to have a combination that will improve the performance in any application area (Miskovic, 2014).

According to Joshi and Srivastava(2014) ensemble learning is a two stage decision making process, the first step is related to the decision of individual classifier and the second is related to the decision to the combined model. In classification theory classifiers requires other Meta classifier that computes the results of the single classifiers, the two major methods to evaluate the results of an ensemble of classifiers are:

All cumulative count ensemble- A class is chosen by the majority of the classifiers as opposed to minority. If a sms is classified by five classifiers and three classifiers gives a spam

classification out of five, then spam is the output of the ensemble classification by all cumulative count.

Winners takes it all method – This method uses the results output from the algorithm which is confident for its choice. For K^{th} nearest neighbor algorithm for example, this means that the algorithm that finds more matching neighbors is the winner and can make the sole decision about the class the message belongs. An ensemble of 3-nearest neighbor for example used for the experiments often has shown several winners majority count. In an example, if three classifiers get two spam neighbors and four classifiers find zero spam neighbors, then a message will be classified as not spam because of the win from the majority four.

Improved classification can be obtained using diverse classifiers techniques such as bagging, boosting and stacking. Recent studies have focused on how to improve classification rates in many application areas, most of this studies have found that most classifier perform better differently, and by combining several techniques it improves performance significantly, for example in sms spam detection, many approaches, however seems none of them is able to completely detect sms Spam precisely and efficiently. Combining diverse methods as an hybrid model is required to achieve high true positives and low misclassifications.

2.7.1.1 Bagging

Bagging also known as bootstrap aggregating is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid over fitting (Zhou Zhi-Hua, 2012)). It considers homogeneous weak learners, learns them independently from each other in parallel and combines them following some kind of deterministic averaging process. One way to reduce the variance of an estimate is to average together multiple estimates (Smolyakov, 2017). For example, we can train different trees on different subsets of the data

(chosen randomly with replacement) and compute the ensemble. Bagging uses bootstrap sampling to obtain the data subsets for training the base learners. For aggregating the outputs of base learners, bagging uses voting for classification and averaging for regression. Although bootstrapping is asymptotically consistent, it does not provide general finite-sample guarantees. The result may depend on the representative sample. The apparent simplicity may conceal the fact that important assumptions are being made when undertaking the bootstrap analysis (e.g. independence of samples), another limitations is that bootstrapping can be time-consuming. On getting the base learners, Bagging puts them together by majority voting and the most voted class is determined, for example the random forest. The Bagging algorithm is illustrated as algorithm 10.

Input: Data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$;
 Base learning algorithm \mathcal{L} ;
 Number of learning rounds T .

Process:
 for $t = 1, \dots, T$:
 $\mathcal{D}_t = \text{Bootstrap}(\mathcal{D})$; % Generate a bootstrap sample from \mathcal{D}
 $h_t = \mathcal{L}(\mathcal{D}_t)$ % Train a base learner h_t from the bootstrap sample
 end.

Output: $H(\mathbf{x}) = \operatorname{argmax}_{y \in \mathcal{Y}} \sum_{t=1}^T 1(y = h_t(\mathbf{x}))$ % the value of $1(a)$ is 1 if a is true and 0 otherwise

Algorithm 10: Bagging algorithm

2.7.1.2 Boosting

In general Boosting is a family of algorithms normally used to enhance the weak learners. The most popular boosting algorithm is the AdaBoost algorithm shown as algorithm 11, Freund and Schapire (2009) formulate AdaBoost, short for "Adaptive Boosting", it is a machine learning meta-algorithm formulated used in conjunction with many other types of learning algorithms to improve performance. The output of the other weaker learners is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive

in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the over fitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing first the algorithm assigns symmetrical weight values to all the training samples. Let's label the distribution of the weighted values at the t -th learning iteration as D_t from the training data and from D_t the algorithm say generates a base learner $h_t : X \rightarrow Y$ by recursively calling the given base learning algorithm. It then uses the training examples to test the base learner h_t , here the weights of not so correctly classified sample will be increased. Therefore an up-to-date weight distribution D_{t+1} will be generated from the training data set and D_t .

AdaBoost generates another base learner by recursively calling the base learning algorithm again and again. This process however is normally repeated say T times, each of which is called an iteration or a round, the final learner is derived by weighted majority voting of the T base of learners, and the weights of the learners are computed during the training process. In real world scenario, this base learning algorithm may be a learning algorithm that can use weighted training examples proportionally, or these weights can be computed by exploiting the training samples according to the weight distribution D_t .

Input: Data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$;
Base learning algorithm \mathcal{L} ;
Number of learning rounds T .

Process:

$D_1(i) = 1/m.$ % Initialize the weight distribution
for $t = 1, \dots, T$:
 $h_t = \mathcal{L}(\mathcal{D}, D_t);$ % Train a base learner h_t from \mathcal{D} using distribution D_t
 $\epsilon_t = \Pr_{i \sim D_t}[h_t(\mathbf{x}_i) \neq y_i];$ % Measure the error of h_t
 $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t};$ % Determine the weight of h_t
 $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \exp(-\alpha_t) & \text{if } h_t(\mathbf{x}_i) = y_i \\ \exp(\alpha_t) & \text{if } h_t(\mathbf{x}_i) \neq y_i \end{cases}$
 $= \frac{D_t(i) \exp(-\alpha_t y_i h_t(\mathbf{x}_i))}{Z_t}$ % Update the distribution, where Z_t is a normalization
factor which enables D_{t+1} to be a distribution
end.

Output: $H(\mathbf{x}) = \text{sign}(f(\mathbf{x})) = \text{sign} \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$

Algorithm 11: AdaBoost algorithm technique.

2.7.1.3 Stacking

Stacking, that often considers heterogeneous weak learners, learns them in parallel and combines them by training a meta-model to output a prediction based on the different weak models predictions. Individual learners are generated from the training data by using dissimilar algorithms techniques. These individual learners are then combined by a second set of learner known as Meta learner as shown on figure 12, the algorithm is shown as algorithm 12.

Input: Data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$;
 First-level learning algorithms $\mathcal{L}_1, \dots, \mathcal{L}_T$;
 Second-level learning algorithm \mathcal{L} .

Process:

```

for  $t = 1, \dots, T$ :
     $h_t = \mathcal{L}_t(\mathcal{D})$     % Train a first-level individual learner  $h_t$  by applying the first-level
                        % learning algorithm  $\mathcal{L}_t$  to the original data set  $\mathcal{D}$ 
end;
 $\mathcal{D}' = \emptyset$ ;    % Generate a new data set
for  $i = 1, \dots, m$ :
    for  $t = 1, \dots, T$ :
         $z_{it} = h_t(\mathbf{x}_i)$     % Use  $h_t$  to classify the training example  $\mathbf{x}_i$ 
    end;
     $\mathcal{D}' = \mathcal{D}' \cup \{((z_{i1}, z_{i2}, \dots, z_{iT}), y_i)\}$ 
end;
 $h' = \mathcal{L}(\mathcal{D}')$ .    % Train the second-level learner  $h'$  by applying the second-level
                        % learning algorithm  $\mathcal{L}$  to the new data set  $\mathcal{D}'$ 

```

Output: $H(\mathbf{x}) = h'(h_1(\mathbf{x}), \dots, h_T(\mathbf{x}))$

Algorithm 12: Stacking algorithm

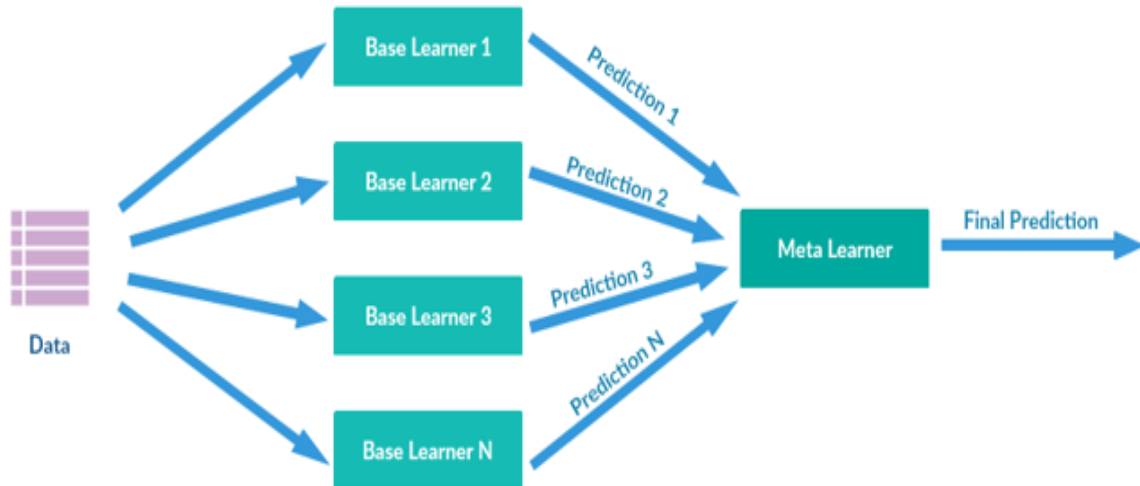


Figure 12: Stacking of algorithm based on meta learner.

In summary no existing ensemble technique that outdoes other ensemble methods consistently (Zhou et al., 2012). Further empirical studies on most ensemble methods studied by Bauer and Kohavi, (1999), Ting and Witten (1999), Opitz and Maclin (1999) showed that using more base learners led to a better performance, on the other hand Zhou et al., (2012) proved the “many could be better than all” but selecting some base learners instead of using all of them to compose an ensemble is a better selection.

2.8 Ensemble model prototyping

Mobile communication devices have been the most adopted means of communication both in the developed and developing countries with its penetration more than all other electronic devices put together (Okediran et al ., 2016). Every mobile communication device needs some type of mobile operating system to run its services: voice calls, short message service, camera functionality, and so on. Google Android, Apple iOS and Microsoft Windows Phone are most common types of mobile operating systems (Novac et al., 2017). According Jain and Sharma (2018) Android’s percentage share in the market is increasing at an alarming rate, Google android is rapidly taking its place in the eyes of today’s youth and every person today wants affordable and the best operating system which Android guarantees to provide to its users. Dollah et al., (2017) research on mobile device ownership, the research indicated that 159 out of 225 respondents (70.4%) had Android based device for their mobile phones followed by others (19%), Apple iPhone (11.1%), and Windows Phone (2.2%). The least was Blackberry mobile phone with a percentage of (1.3%) only. The possible factors that led to the high ownership rate of Android based mobile phones may be attributed to the competitive price of these devices. However, in lieu of this finding, the simplicity, reliability and functionality may be best attributed to others, such as, Apple iPhone and windows Phone.

A research conducted by Richerzhagen et al., (2015) on the increasing market penetration of mobile devices, such as smartphones and tablets, poses additional challenges on the design of distributed systems. Due to the heterogeneous environment consisting of both, mobile and fixed devices, a multitude of effects on different scales need to be considered. Microscopic effects, such as an individual user's interaction with the device, as well as macroscopic effects, such as scalability with the number of users have an impact on the system's performance. The combined evaluation of micro- and macroscopic effects requires both, simulations and prototypical deployments. It is also common practice for developers of user-facing software to transform a mock-up of a graphical user interface (GUI) into code. This process takes place both at an application's inception and in an evolutionary context as GUI changes keep pace with evolving features (Moran et al., 2018). Using stable IDE software such as android studio developers can develop applications for phone that accelerate development and help you build the highest-quality apps (Hagos, 2018).

2.8.1 Testing android mobile applications

There is a growing need for automated testing techniques aimed at Android apps. A critical challenge is the systematic generation of test cases. One method of systematically generating test cases for Java programs is symbolic execution. But applying symbolic execution tools, such as Symbolic Pathfinder (SPF), to generate test cases for Android apps is challenged by the fact that Android apps run on the Dalvik Virtual Machine (DVM) instead of JVM. In addition, Android apps are event driven and susceptible to path-divergence due to their reliance on an application development framework (Mirzaei, et al., 2012). Recent introduction of a dynamic permission system in Android, allowing the users to grant and revoke permissions after the installation of an app, has made it harder to properly test apps. Since an app's behavior may change depending on the granted permissions, it needs to be tested under a wide range of

permission combinations. At the state-of-the-art, in the absence of any automated tool support, a developer needs to either manually determine the interaction of tests and app permissions.

The study by Zein, Salleh and Grundy (2016) focuses on mapping the testing techniques for mobile applications. Additionally, the study emphasizes the need for testing metrics to be included and adhere to address mobile application testing lifecycle conformance. The major lags in the mentioned techniques for a smartphone application testing lie in the automation of testing. According to the authors, this is an emerging and future of mobile and other testing, but very few studies have implemented this technique over complex applications. Automated testing techniques perform well over small to medium and simple mobile applications, but very little work is done over the implementation and analysis of this technique over complex mobile applications. On android ecosystem there is a large selection of different testing tools, libraries and frameworks available for Android. It is hard to understand which tool to use for what type of tests whether unit testing, mocking, user interface testing or integration testing (Morgado & Paiva, 2019).

2.9 Related work on SMS spam detection

Several spam classification techniques with different degrees of efficiency have been proposed, designed and developed. Recent research work on mobile SMS spam detection show that many of these techniques are used primarily to detect, filter or classify spam text messages (Shafil et al., 2017). These solutions are developed to function either on client side or server side. The client side famously is utilized on the user portable device itself. The server side approach is designed to be deployed at the mobile Service provider. Figure 13 shows the basic architecture of Sms spam communication from the spammer to random mobile user.

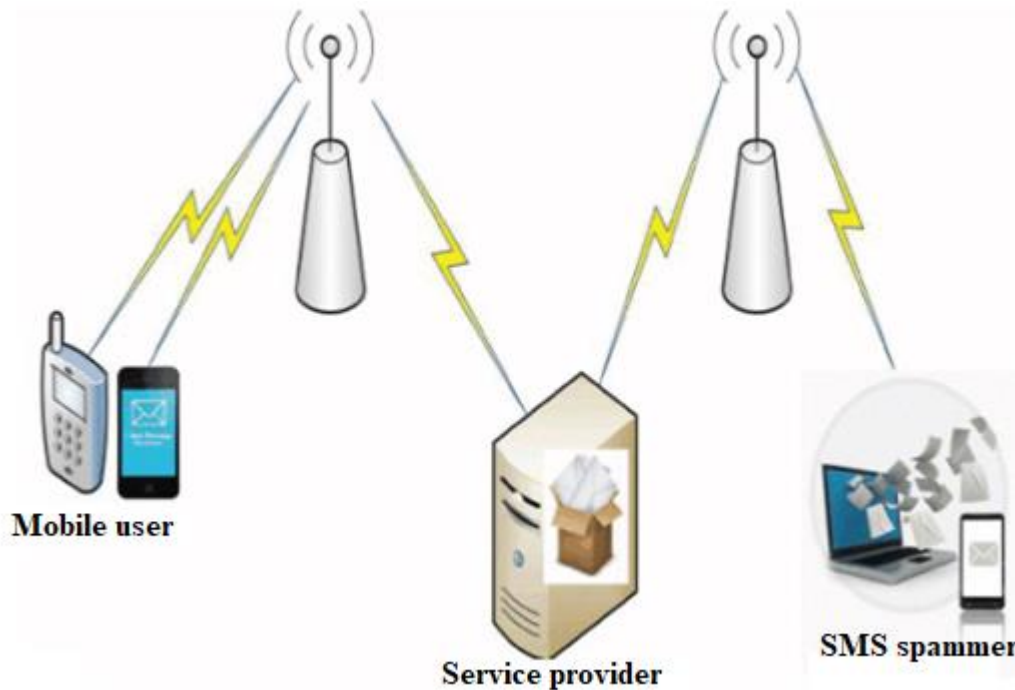


Figure 13: Architecture of the SMS spam transmission line (Abdulhamid, 2017).

One of the earliest study in SMS spam detection (Xiang and Chowdhury, 2004) used Support Vector Machines (SVMs). The same classification method was used by (Gomez et al., 2006), which selected tokens using Information Gain (IG). Longzhen et al., (2009) applied a K-nearest neighbor algorithm (K-NN) along with other spam detection methods on a dataset which contained 750 spam and ham SMS. Liu et al., (2012) used the frequency of some text units as input attributes by using a number of classification algorithms. Jie et al., (2010) added a weight to words to increase the cost of higher false positive to this study. Uysal et al., (2012) suggested a client side solution based on KNN and SVM classification of real time mobile system for Android mobile phones. Several combinations of the Bag of Words and structural feature attribute are fed into widely used classification algorithms in order to classify the SMS messages. In the experiment, a collection of Bag of words features were based on Chi-square (CHI) and information gain (IG) techniques, the number of features span from one to hundred percentage of the entire Bag of words feature space. Experimental outcome and findings on

the important test sets showed that the combination of Bag of words and Structured features instead of Bag of words behavior only allowed for a more effective and accurate performance classification on the test sets, this filtering framework is evaluated on both Turkish and English SMS message datasets and the main limitation of this study is that the efficiency of using the characteristics selection processes varied from one language to another.

Nuruzzaman et al., (2011) showcased a text classification method that used Naïve Bayes method with word occurrences table. The technique was applied to a Google Nexus android phones Operating System, Qual-comm 1 GHz Processor and a Micro SD memory card among others. Several experiments conducted using this method shows a scenario where the applicability is very low since the user is required to have a humongous amount of text data during the start of the training data. The study results shows that the proposed spam filtering system on an independent mobile phone achieves great accuracy with low storage consumption and an average case execution time but lacks computer system support by increasing hardware cost and communication cost between mobile phone and the computer system.

Junaid and Farooq (2011) implemented a progressive learning classifier algorithm to design an sms detection system that classifies spam SMS at the client side of a mobile phone. It evaluates an SMS message in the hexa-decimal number system (base 16, it mines out two features from this format, octet 1 bigrams and frequency distribution of bytes). They evaluated the applicability of a number of evolutionary and non-evolutionary classifiers for the system. The results of the experiments shows that the classifier hexadecimal system achieves an overall detection accuracy rate above 89% and a 0% false alarm rate which is unrealistic and show some element overfitting of the model. This detection rate is still low in precision as compared to other models, it is also unreasonable true to get a 0 % false alarm rate considering dynamic sms spam attack behavior.

Kim et al., (2015) developed a keyword frequency ratio and WEKA 3.7 machine learning tool simulation scheme for the light and fast system. The mobile message filter is executed within the phones autonomously through the use of keyword frequency ratio (FR). Each message is broken down into a set of keyword components by utilizing the function 'string to word vector' (STWV) in a WEKA interface, then pre-processing is completed on 5,574 SMS messages from UCI data set. The WEKA 3.7 smart tool was used to conduct the experiment using the following machine learning algorithms J-48 Decision Trees, Naive Bayes and Logistic regression. The experiment was conducted both with a full training set and 10 fold cross validation. The 10-folds are based on aimlessly selected data which are distributed into 10 separate subsets of almost all identical dimension. Each subset is used as the justification set while the remainder are used to form the classifier. This justification set is subsequently used to enhance the accuracy. The accuracy estimate is the mean of the estimates for each of the classifiers. The results of the experiments shows that the Naive Bayes returns 0.01 seconds of central processing unit (CPU) time and 94.70% of accuracy, the J48 algorithm 0.02 and 94.82%, and Logistic regression algorithm 0.1 and 94.71% respectively. The limitation of this study is that data should be added constantly for a precise analysis, the individual classifier still remain unstable especially when the data set is scaled up .Further the WEKA version used is also limited in library as compared to latest versions of WEKA. Another study proposed by Zainal et al., (2015) used a Bayesian method on Rapid Miner and Weka experiments. To conduct the experiment, the two free software tools were used, Rapid Miner and WEKA based on the dataset from the University of Irvine California (UCI) Machine Learning Repository. The dataset contains 5,572 messages, 4,825 messages which are labeled as ham and 747 as spam. Generally their output shows that both free software tools performed equally accurate using the same data set for classification and clustering.

Cai et al., (2008) earlier had designed a system for spam detection using winnow algorithm technique. His experiments were particularly conducted using Chinese language SMS messages. Although the experimental results illustrated that this system works well, it is possible that the system could be further enhanced by developing the feature selection method and the decision making procedure.

Joe and Shim (2010) used Support Vector Machine for spam filtering for mobile system by applying experience based learning to identify spam SMS. The keyword and terms inside a SMS text are extracted while passing through a pre-processor and a language dictionary. If the homogenized term is contained in the feature list, the word catalogue is set to one or zero. The output vector values are utilized as learning data to change the support vector machine hyper plane. After each attribute vector is marked zero or one, a learning process is concluded from the side to side support vector machine classifier. Further a Gaussian radial basis function is used as the kernel function to enhance efficiency, where the static values are set as 10 20 and 40, gamma values are set as 0.01 0.05 and 0.1. This technique display its performance with a feature vector rate of 150, a static constant rate of 20 with a gamma rate of 0.01. A general weakness of this method is that the detection rate is largely impacted especially when a pre-processing does not separate words in a sentence correctly.

Yadav et al., (2011) provides an approach is similar to Deng and Peng (2006) in that they propose a client side Naive Bayes filter which uses the occurrence of keywords that appear in spam messages to determine a spam score. Messages that score above a certain threshold are labeled as spam. Their solution also requires user feedback to confirm and correct errors made by the classifier and therefore their filter can learn new spam keywords from client reports to a central server, which are in turn pushed out to other clients, however it relies on third party messages which may not be correct.

Almeida et al., (2011) tested 13 classification algorithms on a dataset that contained more than 5500 SMS (747 spam and 4827 ham). Their results show that SVM combined with the alphanumeric tokenization performed the best. The tokenization includes separating non-alphanumeric and alphanumeric characters. They finally extracted 81,000 tokens from short messages. Delany et al. (2012) modified the method and dataset used in (Almeida et al. 2011). This study focused on spam class evaluation factors but did not report F-Measure and Receiver operating curve (ROC) area. The overall accuracy showed that their method did not perform so well for non-spam SMS. Almeida et al., (2013) improved the evaluation factor of their previous work slightly by introducing new features though the features were not enough to generate high quality filtering. Another study by Mathew and Isaac (2011) compared a class of Bayesian classifiers with other classifier algorithm techniques for mobile spam filtering, they used WEKA tool for their experiments. Since WEKA does not read text strings ,all strings were converted into feature vectors, that is very similar to on- hot encoding technique, and from their experiment , this was achieved using ‘StringToWordVector’ WEKA function. The variety of the Bayesian methods they used proved to be very effective with an overall success rate of about 98%, however one limitation of the techniques used especially the Bayesian classifier is independent assumptions of data. Earlier study conducted by Sohn et al., (2009) used a stylistic-feature based shallow linguistic technique as the new feature that indicates the writing style of SMSs for content-based mobile spam filtering. The simulations were performed using K-fold cross validation support (k-10), and the results from the experiment indicate that the techniques outperform the comparative Bayesian scheme earlier proposed by Gómez et al., (2006). Though Shallow linguistic yielded less expressive and structural representations, which directly captures short-distance dependencies and in addition it is only efficient with Korean mobile spam compared to others.

Serrano et al.,(2014) designed a writing style-based spam filter using extrinsic information, sequential labeling extraction and term clustering to store the writing style of spam and non-spam SMS. This is achieved by preserving the order of the content in the feature space. All the classifiers use the K-fold cross validation (K=10) in WEKA. The experimental results show that the technique produces a low dimensional feature space. Yadav et al. (2011) designed ‘SMSAssassin’ a mobile spam filtering application based on Bayesian learning and blacklisting the sender’s mechanism. To achieve higher accuracy, Support Vector Machine (SVM) is used along with Bayesian classifier. Since the patterns and keywords in the spam messages are changed frequently, so crowd sourcing is used to keep it updated. The application requires a feedback from the user to correct the error made by the classifier so that filter can learn new spam keywords. Mobile phones having SMSAssassin application can share their reported spam list and can update their Spam Keyword Frequency (SpamKeywordsFreq) list for better filtering. The Bayesian SVM set up assume a linear classification suffers from decision boundaries problem and is limited in heterogeneous data such text messages. A research by Mahmoud and Mahfouz (2012) that was conducted using Artificial Immune System, an SMS spam classification scheme for filtering SMS spam using artificial intelligence. The system uses a set attributes as an input to the spam classifier. It classifies text messages by using a trained dataset which consists of phone numbers and spam key words detectors. The experimental results are obtained using the IOS operating system .The result of the experiment depicts that the method proposed has the ability to classify text messages either as spam or non-spam accurately and converges faster than Naive Bayesian algorithm. The study is skewed to Naïve Bayes algorithm leaving out other potential algorithm that could perform better. Foozy et al., (2014) experiment their research using a combination of Naive Bayes and J48 machine learning algorithm techniques to classify the spam and ham messages. This research used a

collection of message in Malay language as alternative method to overcome SMS spam and phishing. The Malay SMS collection is tested using the conditional Naïve Bayes algorithm coupled with J-48 decision tree algorithm (Othman and Din, 2019). It further compares these algorithms with other unsupervised machine learning techniques. Their experiments show that the performance of both methods used (Naïve bayes and J48) were relatively similar however the study was only limited to Malay language. Further their experiment did not used public available data such as the UCI, Grumble text, Kaggle, Google among others. Onashoga et al .(2015) presented a server side collaborative and adaptive filtering system using artificial immune system, the study deployed the learning abilities of the human immune system to learn and unlearn sms keywords, a data set of 5240 sms messages was used in this study. The study showed that the server side collaborative gave a better accuracy than Client side naïve bayes and artificial immune system (AIS). A deep perception in this review shows that most studies used Support Vector machines and Bayesian network for developing classifiers for SMS spam detection. However, most of these applications have significant limitations (Sharma and Dey, 2013). This study used a boosted SVM based sentiment analysis approach for online opinionated text. In Proceedings of the 2013 research in adaptive and convergent systems (pp. 28-34).). SVM as a sole classifier is not good for the prediction of categorized class labels as compared to naïve bayes. To add on, it is important for the kernel function in support vector machine to fulfil the Mercer's condition which must have a continuous symmetric kernel (Sharma and Dey, 2013). In (Abdulhamid et al.,2017) study , they suggested that single classifier needs to be ensemble or made hybrid with the other evolutionary algorithm for SMS spam classification and detection, since this will improve effectiveness and efficiency(Chiroma, 2015) of the overall system and reduce bias. According to Bottazzi et al., (2015) the MP-Shield framework consist of Blacklist Application programming Interface(

API) which is used to generate spam list ,machine learning classification engine, and watchdog. It searches for Google Safe Browsing and provides the phishing blacklist service of Google. In this way, it detects malicious URLs (Yan et al., 2009) by profiling social behaviors of sms users for anomaly detection. Choudhary et al., (2017) present a novel approach that can detect and filter the spam messages using machine learning classification algorithms. They study the characteristics of spam messages in depth and then found ten features, which can efficiently filter SMS spam messages from ham messages. This adopted approach achieved 96.5% true positive rate and 1.02% false positive rate for Random Forest classification algorithm. True caller a general spam detector app is created by user community who voluntary choose to report unsolicited calls and sms by enabling the creation of a spam directory, however users have freedom to suggest any sms as spam including targeting business rivalry .True caller has reported on several occasions of having delisted some users report as their spam reports falls below the required threshold, trucaller also uses phone numbers as the key field to label an incoming sms as Spam, the researcher believes it's only the content of the message that can determine the spamicity of a message. On many occasions users have reported on privacy invasions when using trucaller app (Vijayumar, 2016), It also blocks an sms once it has been identified as Spam, this is not a good approach because the user has no chance to look at the contents of the message .Further trucaller provides an email support.eu@trucaller.com for reporting mistakenly (false positives) labeling spam messages and phone numbers. It is important to evaluate the most used algorithm techniques of filtering SMS spam messages based on performance metrics mention in chapter four in this study. Table 2 shows a summary of some of the research studies that influence this thesis direction.

Table 2: Summary of the major research studies on SMS spam

Study	Proposed Techniques	Data Set	Compared techniques	Major Findings	Gap of knowledge
Onashoga et al.,(2015)	Collaborative content based	5420 sms source undisclosed	Naïve Bayes and AIS	Collaborative and content based performed better than NB and AIS	Computational intensive. Low performance.
Almeida et al., (2013)	SVM	Grumble text SMS corpus	Naïve Bayes, Linear SVM,KNN	SVM performed better than other techniques	The study did not include evaluation metrics. Lacks heterogeneous text.
Zhang, H.-yan, and Wang, W. (2009).	Naïve bayes	Jon Stevenson corpus	SVM C45	SVM performed better than other techniques	Naive Bayes classifier makes a very strong assumption on the shape of data distribution-biased.
Yadav et al.,(2011)	SMSAssasin and Bayes	2195 non spam 2103 spam	Bayesian learning and SVM	This technique produces an average of 84.75% classification accuracy	The Bayesian SVM set up assume a linear classification suffers from decision boundaries , low accuracy

Hidalgo et al., (2012)	Near duplicate detection approach	Grumble text, Tag's PhD Thesis NUS SMS corpus	No evaluative	The outcome suggest that the method proposed does not lead to near duplicates	The study has limited bag of words and limited to one language
Bottazzi et al.,(2015)	Proxy service on TCP/IP	Not disclosed	J48, SMO Bayes Net, SGD, IBk	Provides good level of protection	Model once built does not gather new knowledge from current classifications(server side)
Choudhary N et al., (2017)	Random forest	2608 messages	Naïve bayes, logistic regression, J48, SVM	achieved 96.5% true positive rate and 1.02% false positive rate	Features used limited and the data set used is not enough. Higher false positive. Poor quality data

However despite previous efforts on sms spam filtering , there is still need for a better filtering taxonomy (sodiya,2015).In addition many studies mention on the summary table did not compare their models with other existing models and techniques ,this makes some models difficult to establish their capability of detecting spam. Other important points noted from this review is that some studies showed very high false-positive rate and others very low efficiency due to large number of input features. In addition, some studies did not use evaluation metrics at all such as F-measure, precision, MCC, ROC area among others, this makes some classifiers difficult to evaluate. This study intends to address these limitations and gap of knowledge by considerably using an ensemble method, decreasing the number of input features, and

providing a comprehensive assessment by adopting the use of evaluation metrics .Further for a method to be effective in filtering SMS, the researchers believes in combining several algorithms and techniques together than using a single biased classifier for the entire model on heterogeneous data. Presently, these weaknesses of sms spam filtering techniques have remained unresolved and continuous to be researched (Abdulhamid et al., 2017). This study is informed by this gap of knowledge. After having evaluated many techniques, the research is informed by the conceptual framework in figure 14.

2.9.1 Conceptual framework

In this context, an ensemble hybridized model is used in that several classifiers are trained on the same problem (sms spam), but on different subsets of spam and ham messages. An instance is tagged as the class that is chosen by combination of classifiers. If a message is classified by majority classifiers as spam, then spam is the result of the ensemble classification as per the classifiers and vice versa. In this study a newly incoming text message is likely to belong to one category only, depending on its content. For that scenario, Many classifiers would be trained, one for each combination of ham and spam. The text data is received, then preprocessing techniques are applied on the classifiers after each classifier is parameter tuned , to allow easy training and testing as this parameters will define the ultimate architecture of the model, these parameters include setting the maximum depth of the tree ,no of trees, no of neurons ,grid search, no of iterations, estimators , sigma, kernel among others the next step is feature engineering which is done using the techniques aforementioned, the combination of both pre-processing and feature engineering techniques is use for the ensemble meta model, data is trained and tested on this meta model using 10 fold cross validation, a method that perform the fitting procedure 10 times, with each fit being performed on a training set consisting of 90 percent of the total training set selected at random , the remaining 10 percent

is used as a hold out set for validation purposes, this approach reduces overfitting bias and computational time significantly. The Meta model then determines the class of data as either spam or not based on data set provided. Finally an android based prototype is implemented on the client phone to detect the class for the messages.

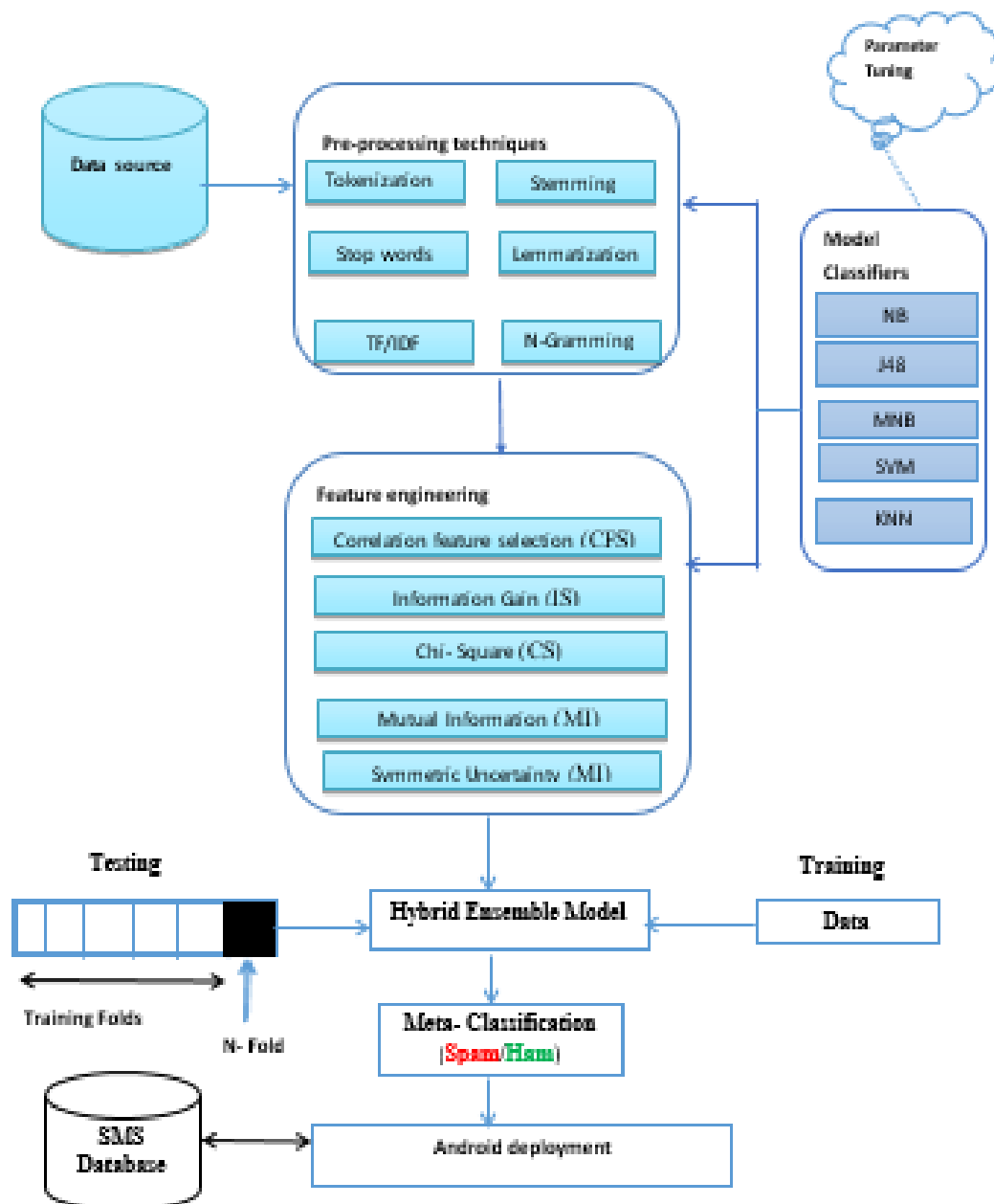


Figure 14: Conceptual framework

2.9.2 Chapter summary

The aim of this chapter was to provide an in-depth analysis on existing literature on sms spam detection as per the five objectives, the challenges and limitations of existing methods, gap of knowledge and the conceptual framework that provides the direction for the research under study. Most notably from this review is that most studies exhibited limitations such as ; very high false-positive rates ,very low efficiency due to large number of input features, high computations and generally low accuracy and precision. In addition, some recent studies did not use evaluation metrics to evaluate their study. This study intends to address these limitations and gap of knowledge by considerably using an ensemble method as per the conceptual framework in this chapter by using ensemble of pre-processing and feature engineering techniques.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 Introduction

This chapter provides a description of the methods and approaches adopted in carrying out the hybridized ensemble machine learning model .It also provides detail description on how to achieve the objectives aforementioned ,the research framework ,the data set used, the experimental tools and programming adopted .

3.2 Research design

This study uses quantitative approach as the preferred method because of the evaluation metrics adopted such as F- measure , recall among others and also the nature of the problem being addressed that require binary classification (Spam or Not Spam).This study proposes a client side ensemble sms spam detection model that uses machine learning techniques, it addresses issues of improving performance of sms spam detection, this approach is updatable and does not need very large amount of SMS data in advance. Thus, only the text Classification algorithms with low computational complexity for both training and filtering were considered. The design of the model as per the research problem is based on the following five steps:

1. Message preprocessing : most text message are raw in the sense that they are different styles format size syntax and punctuations, to take care of this , the model represents the messages in an appropriate format to enhance training.
2. Feature selection method: Message content have huge features, it is important only select those features that are relevant in classification, since redundancy in feature has high impact on model performance.

3. Message classification: many sms spam detection model produces high false positives alerts, therefore is important to cluster message in order to differentiate false positives, redundancies and true positives through clustering algorithms.

4. Performance evaluation: for true reflection of model performance it is important to evaluate the model .Correlation performance needs to be enhanced especially among attributes because message attributes may have dependency relationships. Several performance evaluation methods were used for comparing results of classification; some of these methods include precision, recall, accuracy, true positive, false negative, true negative and false negative-rates

3.2.1 The WEKA workbench

In this study machine learning experiments were conducted using WEKA. WEKA (Waikato Environment for knowledge analysis), an Open source data mining tool written in Java programming that can be used for data pre-processing, classification, clustering and visualization. It contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces (Hall et al., 2009).

The following are WEKA interfaces as shown in figure 15 GUI.

1. Explorer

This is a popular interface that includes data preprocessing, classification, clustering, association, attribute selection and visualization of data

2. Knowledge flow

This is where users layout and connect widgets representing WEKA Components, it allows incremental data processing. WEKA components are selected from the tool bar, position a

layout canvas, and connected into a directed graph to model a complete system that process and analyses data.

3. Experimenter

This GUI allows large scale comparison of predictive performances of learning algorithms. It can be run from both the command line and GUI, is a tool that allows you to perform more than one experiment at a time.

4. Work bench

This is a platform or environment that support and facilitates a range of machine learning activities reducing or removing the need of multiple tools

5. Command line interface

This interface provides a simple command interface that allows execution of WEKA commands for operating systems that do not provide their own Command line interface.



Figure 15: Weka GUI chooser.

The following are steps for experimenting with Waikato environment for knowledge analysis

-WEKA

1. RUN the WEKA application from the device.

2. It shows a list of categories such as explorer, experimenter, knowledge flow, workbench and simple command line interface.

3. In this study the experiments are conducted using the explorer section only, it contains a number of sections such as the pre-process, classification, clustering among others.

4. Extract the data to be mined externally. Data can be imported from the file in various formats such as attribute relation file format (ARFF), comma separated values (CSV) and binary. In this study the researcher used ARFF (attribute relation file format).

5. In this study pre-process tab was used to open the dataset. It gives us detail description of the dataset by displaying all the features in the dataset.

6. The classification and attribute selection menu tabs allows us to use heterogeneous classifiers and feature selection techniques. The ensemble feature selection algorithm is implemented in JAVA programming language with the support of WEKA graphical user interface.

3.2.2 Hyper parameter tuning

In machine learning, a hyper parameter is a parameter whose value is set before the learning process begins; this is primarily used to control the learning process (Table 3). Each classifiers has its own parameter that required (Claesen et al., 2015).the following are some of the parameters for each classifier.

Table 3: Hyper parameter tuning for individual classifiers

Algorithm Technique	Parameters
Naïve Bayes	Data parsing, vectoring, word frequency
Decision tree	Depth, no of bins for interval variables, splitting criterion
Support Vector Machine (SVM)	C regularization, Polynomial degree
Neural Networks	Number of hidden levels, Number of neurons at each level, L1, L2 Regularization, Learning annealing rate Number of hidden levels, number of neurons at each level. No of epochs
K th nearest neighbor	K value, No of neighbor, distance metric, search algorithm

3.3 The ensemble modeling steps

The stages for developing the ensemble model follows a sequential manner ,where the input is the data set provided from UCI dataset and the collected data by the researcher, whereas the output is the ensemble model ,that integrates the optimal features through data processing, feature engineering techniques, dimension reduction techniques and class correlation .This steps are summarized as follows.

- i. Data preprocessing which involves preparing raw messages from its raw format for model training using ensemble pre-processing techniques such as stop wording
- ii. Feature engineering that involves removal of irrelevant and less important features, this is achieved using ensemble feature engineering algorithm and techniques.
- iii. Dimension reduction is used to reduce the time and storage space required, so that it becomes easier to visualize the data when reduced to very low dimensions.

- iv. Class correlation enables an attribute to be clustered based on similarity to the class it belongs to, using correlation based feature selection, chi-square, Gain ratio, Symmetrical Uncertainty among others.
- v. Training and Testing that is done based on the feature engineered methods in order to make the model learn from the data set,
- vi. Clustering of message content that is based on clustering algorithm.
- vii. Prototype design and implementation on android devices will test the working of the model on mobile users.

3.4 Ensemble dataset pre-processing techniques

In machine learning, data set Pre-processing is important to ensure the data is free from irrelevant features (Shams and Mercer, 2013). In the study different preprocessing techniques and their combination were considered, Tokenization, Stop Words removal, stemming, lemmatization and TF-IDF on different individual classifiers NB, SVM, J48, ANN, MNB using 10-fold cross-validation for evaluating the performance accuracy.

The dataset consist of 5634 message instances. The key attribute include, class (nominal), messages (string). According to this study a message belongs to either spam or ham class. Spam has 785 instances whereas ham has 4849 instances , i.e. plain text and attribute relation file format (ARFF) , the amount of sample in each set and the total number of samples. In this study preprocessing techniques considered includes words tokenization, stemming and stop words removal.

Table 4: Data set list

Application	File format	#spam	#Ham	Total
General	Plaintext	785	4849	5634
WEKA	ARFF	785	4849	5634

3.4.1 Word tokenization

When tokenizing text documents one often expresses very high dimensional data. Tens of thousands of dimensions are not easy to handle, therefore feature selection plays a significant role. Document frequency thresholding achieves reductions in dimensionality by excluding terms having very high or very low document frequencies. Terms that occur in almost all documents in a collection do not provide any discriminating information.

Tokenizing process was applied in every word in documents. In this step every punctuation mark or spacing (example: ' -) (\ / = . , ; ! ? .) was transform into delimiter '#' that separate every word as token where every token is transform into lowercase .The purpose of tokenization was to calculate the number of words for feature selection and classification (Cormack, 2008) using the tokenization algorithm 4 mentioned in chapter 2. To make the provided document classifiable using machine learning, a feature extraction was done which involved converting the normal text to a set of features .This was achieved using String To Word vector (STWV) function paired with Inverse document Frequency (IDF) and TF (Term frequency) was used to reflect how important a particular word is to a class, for example if a document contains 13000 words where a word *loan* appeared 10 times. The term frequency (i.e., tf) for *loan* is $(10 / 13000) = 0.000769$, the inverse document frequency (i.e., idf) is calculated as $\log (13,000 / 10) = 3.11$ thus, the Tf-idf weight is the product of these quantities: $0.000769 * 3.11 = 0.0024$.

STWV assumes each word in the document is a feature and the number of occurrences in each instance is the feature value (Mohammed and Omar, 2020).

Table 5: Tokenization extract

SMS	Tokenized SMS
Dear KCB Mpesa customer ,loan is now available at 10%, Get 10000, 20000, 30000 , Call 0788XXXXXX :KCB Bank	#Dear# KCB#Mpesa#customer#loan#is#now #available#at #10%#Get#10000#20000# 30000 #Call#0788XXXXXX#KCB#Bank#

3.4.2 Stemming and lemmatization of words

The goal of stemming is to minimize or rather reduce the word space dimension of a message and also to improve the precision for the classifiers by overcoming the data sparseness problem especially when the training data is small in comparison to the word space dimension. In many studies stemming of words and lower case representation of words has showed good results in information retrieval and filtering (Cormack, 2008) especially for spam filtering for example words such as “availability”, “available” are stemmed to its morphological root “avail”. This is done to reduce the word space and decrease words that have same meaning, in this study several stemming algorithms were considered.

In contrast to stemming, lemmatization aims to obtain the grammatically correct forms of lemmas (Cormack, 2008). In implementation the researcher found out that stemming and lemmatization have big impact on the performance of text classification.

3.4.3 Stop words removal

Stop words are words that are particularly common in a text corpus and thus considered as rather un-informative, they only increase the computation resources required to process a

message .The approach to stop word removal involved a creation of a stop word list by sorting in order all words in the entire text message by frequency. These words are specific to a given language for example in English these words include 'I', 'we' 'are' etc. The stop word list, after changing into a *set* of non-redundant words is then used to delete all those words from the input text file that are ranked among the top n words. The lists of stop words are selected from different sources such as Natural Language Toolkit(NLTK) ,Google stop words , the researchers also created stop words for Swahili and slang language such as “yako”, “sasa” ,”la” , msee,” buda”,vipi,“baadaye”, “ni “ ” among others . This combination of stop words file indicated pretty good results for sentiment NLP classification.

Punctuation removal was also considered, similar to stop words and do not convey special meaning. Hence they are also discarded from the text messages using an NLP library. Additionally all text were converted to lower case before training. Common sense reveals that capital letters and small letters of the same alphabets convey similar meaning regardless and hence the conversion is necessary to ensure that they both are standardized and are not treated as separate entities of the text under consideration. For instance, if we consider a text like "LIPA NA MPESA should convey the same meaning as “lipa na mpesa”.

3.4.4 Word N- gramming modeling

N-gramming is a contiguous sequence of n items from a given sample of text, these items include syllables, letters and words in a message. In this study the n-gram method is part of the tokenize, the algorithm is shown as algorithm 14, N-grams are selected and constructed after the message has been tokenized; this is done by reading the tokens from beginning to the end. Step by step it adds each token as an array between the i^{th} token to the $i^{\text{th}} + n$ token together to build a new N-gram which is stored with the message. In this study the variable i is the current token number in the for loop and n is the desired size of the predicted or expected n -

gram. This loop is iterated beginning from the minimum to the size of the N-grams. More concisely, this n-gram model predicts x_i based on $x_{i-(n-1)}, \dots, x_{i-1}$. In probability terms, this is $p(x_i | x_{i-(n-1)}, \dots, x_{i-1})$. When used for language modeling, independence assumptions are made so that each word depends only on the last $n - 1$ words. This assumption is important because it massively simplifies the problem of estimating the language model from data. In addition, because of the open nature of language, it is common to group words unknown to the language model together. N grams exist in many dimension, a 2-gram (or bigram) is a two-word sequence of words like “send money”, “tuma pesa”, or ”your homework”, and a 3-gram (or trigram) is a three-word sequence of words like “please send money”, or “you are selected”. The N-gram model, like many statistical models, is significantly dependent on the training corpus. As a result, the probabilities often encode particular facts about a given training corpus. The performance of the N-gram model varies with the change in the value of N as per the algorithm 7. In this study bigram and trigram is used, and their performance recorded.

```

Data:OriginalTokens
  Data :n-gram Tokens
Results :n=minigram
While n is smaller or equal to maxgram do
While i+n smaller than or equalto=originalTokens do
n-gramsTokens.add (original Tokens(i) to originalTokens (i-n) )
end
end
originalToken = n-gramTokens;

```

Algorithm 13: N-gram modeling algorithm

3.4.5 Bag of words modeling

The bag-of-words model is a way of representing text data when modeling text with machine learning algorithms. The bag-of-words model (Bow) is used in natural language

processing and information retrieval (IR). In this study the model is represented as the multi set of its words, disregarding grammar and even word order but keeping multiplicity (Sivic et al., 2008). Each word is used as a feature for training an algorithm classifier in the ensemble method , Once a vocabulary has been selected, the occurrence of words in the text documents is binary scored(based on its presence or absence). Additional scoring methods includes counting all the words in the document and calculating the frequency that each word appears in the whole document in table 6. The advantage here is that the Bow leads to a high dimensional feature vector due to large size of Vocabulary V (Brownlee,2017), especially for heterogeneous data as represented in the table as an histogram of words. The table 6 contains five sms message and their target feature (spam or Not spam) class.

1. Win win win
2. please call me
3. KCB loan available
4. Babe advice
5. Free loan, win gambling

Table 6: Bag of words histogram

	Bag of words											
ID	Win	Please	Call	Me	KCB	Loan	available	Babe	advice	Free	Gambling	SPAM
1	3	0	0	0	0	0	0	0	0	0	0	True
2	0	1	1	1	0	0	0	0	0	0	0	False
3	0	0	0	0	1	1	1	0	0	0	0	True
4	0	0	0	0	0	0	0	1	1	0	0	False
5	1	0	0	0	0	1	0	0	0	1	1	True

The same information is further represented as term-document matrix based on the keywords (figure 16) assumptions, a presence of a word in a message is assigned binary 1 and absence of a word is given binary 0. If the vocabulary is kept fixed and not increased with a growing training set the term document matrix is excellent because it will take care of misspelling and word adjustment adjustments by spammers.

$$\begin{pmatrix}
 \text{win} & \text{please} & \text{call} & \text{me} & \text{kcb} & \text{loan} & \text{available} & \text{babe} & \text{advice} & \text{free} & \text{gambling} & \text{loan} \\
 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1
 \end{pmatrix}$$

Figure 16: Term-document matrix.

In a bag of words modeling of natural language processing and information retrieval, the message consists of the number of occurrences of each word in a document. Laplace smoothing is a technique is used to smooth categorical data, it allows the assignment of non-zero probabilities to words which do not occur in the sample. Recent studies have proven that Laplace smoothing is more effective than other probability smoothing methods in several retrieval tasks such as language based modeling (Hazimeh and Zhai, 2015). Feature hashing another embedded fast and space-efficient way of vectorizing features by turning arbitrary features into indices in a vector or matrix (Weinberger et al., 2009), to enhance vectorization. This method has similarities with one hot encoding.

3.5 Sampling procedure and sample size

When conducting research many types of sampling are possible, although researchers in qualitative research usually focus on relatively small samples (Lyell, 1998). Research participants are generally selected because they are able to provide rich descriptions of their experiences and are willing to articulate their experiences, thereby providing information that is rich and which will be able to challenge and enrich the researcher's understanding (Crabtree & Miller, 1992; Hutchinson & Wilson, 1991). Two non-probability sampling approaches were used to select the participants for this study. The sampling method was a combination of judgment and snowball techniques. The researcher specifically selected participants who would be able to contribute to the research topic and who would be willing to share the data required for sms spam. The researcher initially approached potential participants that were known to him. This was done by actively selecting participants who met the criteria for inclusion in the study (Marshall, 1996), i.e. mobile phone users. The sample was then expanded by asking the identified participants to refer other users willing to provide relevant input on the research topic. This is known as snowball sampling (Marshall, 1996). When data reaches a point of

saturation, i.e. when new text stop emerging, the researcher concluded that there is no need for more interviews (Hutchinson & Wilson, 1991; Marshall, 1996; Orbele, 2002).

3.5.2 Data collection

In this research both primary and secondary data is considered since they both complement each other, this helped the researcher to construct a convincing argument. Primary sources are more credible as evidence, and secondary sources show how the work relates to existing research. The dataset collected contains multi-lingual text, the secondary data is collected from public repository , a database of 5574 text messages from UCI(university of California Irvine) Machine Learning repository gathered in 2012 (Hidalgo et al., 2012). It contains a collection of 425 SMS spam messages manually extracted from the Grumble text web site (a UK forum in which cell phone users made public claims about SMS spam), a subset of 3,375 SMS randomly chosen non-spam (ham) messages of the NUS SMS Corpus (NSC), a list of 450 SMS non-spam messages collected from Caroline Tagg PhD Thesis (Hidalgo et al., 2012, Tagg, 2009), and the SMS Spam Corpus v.0.1 Big (1,002 SMS non-spam and 322 spam messages publicly available). The uniqueness of this data is that it allows the researcher to generate new insights from previous research analysis, it also provides longitudinal data analysis. Additionally more primary data were added to this public data from local collected messages that contain English ,Swahili, local and slang languages collected from individual and local repositories inorder to provide the multi-lingual corpus , this makes the total research text messages five thousand six hundred and thirty four.

3.6 Ensemble based feature engineering methods

Feature engineering plays a key role in data analytics. The main idea is to generate multiple diverse feature selectors and combine their outputs (Guan et al., 2014). Little can be achieved if there are few features to represent the underlying data objects, and the quality of results of

those algorithms largely depends on the quality of the available features. In this study filter based feature engineering techniques are considered because they are fast and independent of the classification model, another advantage is it is unbiased. It also allows the algorithms to have a simple structure. Having a simple structure in the filter model generates two critical uses. The algorithm becomes easy to design and fast in execution (Vege, 2012) The ensemble filter based feature engineering methods considered include: Correlation feature selection (CFS) with Best first search , Chi square(CS) with ranker search algorithm, Information Gain (IG), Gain ratio(GR) and Symmetrical Uncertainty (SU) methods using ranker search method. Ranker search algorithm ranks attributes by their individual evaluations while Best first search Searches the space of attribute subsets by greedy hill climbing augmented with a backtracking facility.

3.6.1 Ensemble feature ranking techniques

Ensemble of feature ranking techniques is an approach where many feature ranking lists obtained from corresponding feature ranking are combined to generate a single ranking list. This technique is used to improve the classification performance (Karegowda et al., 2010) of a model.

In General there are two major steps performed in ensemble of feature ranking techniques. The first step is to create a set of n ranking lists using corresponding rankers and the second is to select the merging function for transforming it into one large list. The second step involves the merging of the methods.

There are three types of merging methods: fusion based, selection based, and hybrid (Karegowda et al., 2010) Fusion based makes use of all the information obtained from individual rankers to produce a final results, this is computing intensive. On the other hand selection based methods selects a single ranker from the list to become the final result which

can lead to bias and in hybrid method, the final outcome is obtained after both selection and fusion methods have been used. In this study an ensemble hybrid of several feature ranking techniques is used, this method yields more stable and robust results as per the experiments conducted.

3.6.2 The ensemble feature selection algorithm

This algorithm is based on ensemble hybrid approach technique. It evaluates feature significance or score by checking the presence of a feature in a given ranking list. It also uses a mean function to avoid redundancy and repetition. This algorithm can be re-used or extended to any number of ranking lists.

The algorithm consists of two steps. It begins with creating a set of different ranking lists obtained using the rankers selected in WEKA, and then applies the ensemble hybrid approach to establish a singleton feature ranking list. In our study the ensemble hybridized approach used is a frequency count which is accompanied by mean to resolve any feature detection based on collision. The following are the steps of the algorithm.

Step 1: Select a static or fixed no of features from each ranked list for example $f=10$.

Step 2: count the occurrence of any individual feature in all the ranking lists, where this is the frequency of each feature or attribute. Inorder through sorting of the features based on ascending frequency of average ordering i.e. each feature's score is determined by the average of ranking scores in all the ranking lists.

This sorting is performed in an increasing order of rankings. The input to our algorithm would be a list containing n ranking lists with top features. This algorithm starts with selecting the first feature in the ranking list and then searches for the corresponding feature in the remaining ranking list as per the method. It assigns rank for the feature obtained in each ranking list to

another list L. After searching all the ranking lists the feature count is updated and the average is calculated.

This process is iterated for all the features in all the lists available. The output of this algorithm 13 would be a global list G containing top M features from the list L obtained from the ensemble hybrid method.

Input:

N ranking lists (zero to size-1) and each list has k features for example.

Output:

1. An array L containing features and their rank in every list, count, and mean rank.
2. An ensemble hybrid list G

Initialize L and G to NULL

FOR @ ranking list i.

FOR @ feature in ith ranking list

IF the feature is not in L

Add feature and the rank in list i to L

FOR the list j, j is from i+1 to n-1

IF the feature --- the list j

Add the rank of the feature in list j to L

END IF

END FOR

END IF

END FOR

ENDFOR

FOR each feature in L

Compute the frequency and average rank of the feature

END FOR

Sort the features in L based on their frequency, if same frequency, sort by average/merit rank; select the top M features and assign the features to list G.

Algorithm 14: Ensemble feature selection algorithm

3.6.3 KNN Modeling using Euclidean and Manhattan distancing

Enhanced KNN classifier algorithm is used to classify text by finding the K nearest matches in training data and then using the label of closest matches to predict on the training corpus. This classifier has standard procedures 'fit' for training and 'predict' for predicting on test data. It uses lazy training which means all computation is deferred till prediction. The fit method, adjusts and assign the training data to class variables with Xtrain and Ytrain dimension. The K denotes how many closest neighbors are to be used to make the prediction and it start with k=1, k=2, 3 and son. The other parameter explains the distance type to be used between two texts, the distance are Euclidean and Manhattan distance. In prediction every row of text data, is compared with every row of train data to get similarity score. The asymptotic Big O order of growth is O(m * n) where m = no. of rows in training data and n is no. of rows of test data for which prediction needs to be done.

In Manhattan Distance, the distance between two points x and y is the sum of the lengths of the projections of the line segment between the two points onto the coordinate axes: $d(x, y) = \sum_{i=0}^n |x_i - y_i|$ where x is the first point; y is the second point; and x_i and y_i are the i^{th} component of the first and second point, respectively. On the other hand Euclidean Distance is the distance between two points x and y is the length of the line segment connecting v and u.

It is calculated as: $d(x, y) = \sum_{i=0}^n \sqrt{v_i^2 + u_i^2}$, where x is the first point; y is the second point; and x_i and y_i are the i^{th} component of the first and second point, respectively. These distances provide a method for measuring the deviation between two text messages. In order to be able to compare a single text against a group of various spam messages, it is necessary to place a distance selection rule to obtain a unique value dependent on every distance measure performed. Three different selection rules are applied to enhance performance: (i) Mean

selection rule, which computes the average of the distances to all the members of the spam group; ii) min selection rule, which selects the distance to the nearest spam message; and iii) max selection rule, which returns the distance to the furthest point in the normality representation. The final deviation value of the message under inspection depends on the distance measure computed and the selection rule of the message under inspection depends on the distance measure computed and the selection rule applied. Therefore, when the method inspects a message a final distance value is acquired, which will depend on both the distance measure and the combination metric (Laorden et al., 2011).

3.7 Ensemble training, testing and validation

In contrast to a single classifier, ensemble classifiers can be used to determine whether a message is spam or not spam. The dataset is split by the initial corpora assignments that is split into a training (66%) and a test set (33%). The classifier is trained for every plausible assignment i.e. only spam/ham combinations, being evaluated against the testing set. Then the evaluation results are rounded and set to the majority count. The test set of the corpora is then evaluated by all classifiers. The results are summarized into one by majority count, whatever most of the classifiers assign to the message, is compared to its actual class, yielding in a very stable and better results. The ensemble classification considered include stacking, boosting and bagging which are themselves a type of ensemble method, whose performance can be further improved by inclusion in a heterogeneous ensemble method. In this study the researcher uses face validity method, which shows the extent to which a test or an experiment is subjectively viewed as covering the concept it purports to measure. It refers to the transparency or relevance of a test as it appears to test (Holden & Ronald, 2010). In other words, a test can be said to have face validity if it "looks like" it is going to measure what it is supposed to measure (Gravetter and Wallnau 2016) in real world. The Classifiers are trained

using 10-fold cross-validation, a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. In cross validation step each training split is resampled with replacement 10 times then balanced using under sampling of the majority class. The aim in cross-validation is to ensure that every example from the original dataset has the same chance of appearing in the training and testing set. Cross validation involved the following techniques:-

- **10 -fold cross-validation** that divides the data up into **10** chunks and train 10 times, treating a different chunk as the holdout set each time.
- **Leave-one-out validation**: just like 10-fold cross-validation except that chunks contain a single data point.

3.7.1 Clustering in machine learning

Clustering is the process of grouping similar data items together such that data items that are more similar to each other are placed in one cluster (Bijuraj, 2013). In machine learning and statistics, classification is the problem of identifying to which a set of categories (sub-populations) new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. There is no clustering algorithm that is objectively "correct", but as it is noted, "clustering is in the eye of the beholder (Castro and Vladimir, 2002). In this study the following clustering algorithm considered includes, K means, hierarchical and Cobweb algorithms.

3.7.1.1 K- Means clustering algorithm

K -means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). This clustering algorithm uses quantization, which is popular for cluster analysis in data mining. K-means clustering

aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster (Vassilvitskii & Arthur, 2006).

The objective of this algorithm technique is to find groups/clusters in the data, with the total number of groups represented by value K . The algorithm works repeatedly by assigning each data point to one of K group based on the features that are available or provided. Data points are clustered based on feature similarity as per the algorithm the output of K -means clustering algorithm can be illustrated as follows:-

1. The centroids points of the K variable cluster is used to label new data
2. Each data point is assigned to a single cluster through labeling.

Instead of defining groups before searching through the data, clustering allows you to find and analyze the groups/ clusters that have formed organically. The seeding of K section describes how the number of groups for the clusters can be determined i.e. Each centroid of a group is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of cluster group the message represents. The algorithm repeats or rather iterate between the following two solid steps:

1. Data allocation step:

In this step each centroid defines one of the cluster group and each data point is assigned to its nearby centroid, based on the squared Euclidean distance. More formally, if c_i is the collection of centroids in set C , then each data point x is assigned to a group (MacKay et al, 2003).

2. Upgrading of the centroids step:

Here the centroids are recalculated. This is achieved by taking the average of all the data points assigned to that centroid group (Castro et al., 2002). The algorithm technique then repeats between steps one and two until there are no data points change (stopping criterion) clusters or the sum of the Euclidean distance is minimized, or a given maximum number of iterations is reached. The convergence of this technique is normally guaranteed since it relies on the greedy local optimal approach (satisfiable), meaning that assessing one more experiment run of the algorithm with stochastic starting centroids point may give a better results. A number of other techniques exist for validating K , including cross-validation. In addition, monitoring the distribution of data points across groups provides insight into how the algorithm is splitting the data for each K .

3.7.1.2 Hierarchical clustering

Hierarchical clustering (also called hierarchical cluster analysis or HCA) is a method of cluster analysis which builds a hierarchy of clusters (groups). They fall in two types (Rokach et al., 2005)

Agglomerative: This is a "bottom-up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

Divisive: This is a "top-down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

In general, the merges and splits are determined in a greedy manner. The results of hierarchical clustering (Nielsen, 2016) are usually presented in a dendrogram. The standard algorithm for hierarchical agglomerative clustering (HAC) has a time complexity of $O(N^3)$ and requires $O(N^2)$ memory, which makes it too slow for even medium data sets.

3.7.1.3 Cobweb clustering

COBWEB is an incremental system for hierarchical conceptual clustering (Fisher and Douglas, 1987). COBWEB incrementally organizes observations into a classification tree. The COBWEB algorithm yields a clustering dendrogram called classification tree that characterizes each cluster with a probabilistic description. Cobweb generates hierarchical clustering, where clusters are described probabilistically. COBWEB uses a heuristic evaluation measure called category utility to guide construction of the tree. It incrementally incorporates objects into a classification tree in order to get the highest category utility (Sharma, Bajpai and Litoriya, 2012). The main disadvantage of Cobweb is that it produces a larger training tree that may slow down the time of clustering the training data.

3.7.2 Methods of measurement

When it comes to prediction models many methods are used for comparing results of classifiers or models in order to evaluate their performance. Some relevant methods used in this study are recall, precision, true positive, and true negative, false negative, false negative-rates, accuracy, F-measure, mcc among others. True positives from a classifier are spam classified correctly as such, and false positives would be non-spam classified incorrectly as spam. On the contrary false negatives are spam not classified as such and true negatives is a message that is correctly identified as such. The following table gives a more detail description on these methods of measurements including their evaluation function as described in Table 7, p for precision, r for recall and acc for accuracy, tp for true positive, fp for false positive; tn for true negative and fn for false negative and so on.

Table 7: Evaluation measures for SMS spam filters

Evaluation measure	Evaluation function
Accuracy	$ACC = \frac{TN + TP}{TP + FN + FP + TN}$
Recall	$R = \frac{TP}{TP + FN}$
Precision	$P = \frac{TP}{TP + FP}$
F- measure	$F = \frac{2PR}{P + R}$
Mathew correlation Coefficient (MCC)	$\frac{(TP * TN) - (FN * FP)}{\left(\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}\right)}$

According to this study accuracy, Recall, Precision, F-measure, False positive (FP), False Negative (FN), True Positive (TP) and True Negative (TN) are defined as follows:

- i. Accuracy (acc): This illustrates the part of the messages that are correctly classified. It is necessarily important that the acc is high in order to have a low amount of legit messages being wrongly classified as well as to correctly classify as much spam as possible.

- ii. Recall(R): This is the fraction of the spam messages that are correctly classified as such, recall should be high as possible to capture more spam messages.
- iii. Precision (P): the fraction of the message classified as spam to be actually spam
- iv. F-measure (F): Weighted average of precision and recall.
- v. False Positive Rate (FP): The number of mis-classified non spam text messages (false alarm (Type I error)
- vi. False Negative Rate (FN): The number of mis-classified text messages (false negative = miss (aka Type II error)
- vii. True Positive (TP): The number of spam text messages that are correctly classified as spam
- viii. True Negative (TN): The number of non-spam messages that is correctly classified as non-spam
- ix. Matthews's correlation coefficient (MCC) is a measure of the quality of binary classifications (Boughorbel, 2017). It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes,

Additionally other model reliability metrics were further considered such as Kappa, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) detection metrics. Kappa statistic is used to establish or measure the consensus between predicted and observed categorizations of a dataset (Kaggle, 2014)

Mean absolute error (MAE) and root mean squared error (RMSE), makes an assumption of the predicted values on the test instances, for example test instances may be $\mathbf{p_1, p_2, \dots, p_n}$; and the actual class values are $\mathbf{a_1, a_2, \dots, a_n}$. MAE and RMSE are formulated using the equation Eq (21) and Eq (22).

$$MAE = |p1 - a1| + |p2 - a2| + \dots + |pn - an| \quad \text{Eq (21)}$$

$$RMSE = \sqrt{(p1 - a1)^2 + (p2 - a2)^2 \dots (pn - an)^2} \quad \text{Eq (22)}$$

According to Ian, Witten and Frank (2005) the best model should have a Kappa closer to numeric one, MAEs and RMSEs closer to zero. In addition to the metrics mentioned, time taken to execute the model is also recorded across for comparison purposes.

3.7.3 Confusion matrix

Another important measurement is the confusion matrix in Table 8, it illustrates the accuracy of a model to a classification problem. Given N classes, a confusion matrix is a $m \times n$ matrix, where $C_{i,j}$ indicates the number of tuples from Document (D) that were assigned to class $C_{i,j}$ but where the correct class is C_i . Obviously the best solution will have only zero values outside the diagonal (Goyal and RajniMehta, 2012).

A confusion matrix contains information about actual and predicted classifications done by a classification model. Performance of such systems is commonly evaluated using the data in the matrix. The following table shows the confusion matrix for a binary class. The entries in the confusion matrix have the following meaning in the context of this study.

Table 8: Confusion matrix

Prediction Model		
A	B	Classified as
True Positive	False positive	a= Spam
False Negative	True Negative	b=Ham

These techniques as well as processing time offers the necessary information required to properly compare the different filter parameters as per the hardware requirements and classification rates. Intel® Pentium processor 1.10 GHZ and 4GB RAM specifications was used to conduct all the experiments. One of the major developments in machine learning especially for predicting data is the use of ensemble methods which have proven to give better results than individual predictors, in this study the ensemble methods adopted include bagging, boosting and stacking in order to improve sms spam prediction. several experiments are conducted based on the UCI+ local data and their performance compared and evaluated based on the precision realized, the steps include , getting sms from the database bank, preprocessing, dimension reduction, ensemble classifier design, experiment and analysis , Model prototype design and implementation.

3.7.4 Bagging and boosting with six individual classifiers

First the experiment is conducted with bagging and boosting ensemble model techniques with six classifiers. The individual classifiers includes naïve bayes (NB), Support vector machines (SVM-SOM), Artificial Neural networks (Perceptron), ad boost (adaboostM1), KNN-IBK and

Decision trees (J48) REPTree, this classifiers are evaluated in Table 9 based on the metric methods aforementioned.

Table 9: Ensemble modelling parameters

Parameter	Function
Learning rate	A configurable hyper parameter used in the training of a model e.g. neural network
Max depth	In decision trees, it's the maximum depth of the tree that will be created. It can also be described as the length of the longest path from the tree root to a leaf.
Grid search cross validation	The process of performing hyper parameter tuning in order to determine the optimal values for a given model.
Base learner	The classifier used for building base classifiers (e.g., decision tree).
Size of estimators (leaves)	The size of estimator for picking the “best,” or most likely accurate prediction
Seed value	A value from which learning starts i.e. features required to proceed in ensemble modeling
Decision Threshold	The decision for converting a predicted probability or scoring into a class label
Sub set(bags)	Number of bags as a percentage of training set

3.7.5 Stacking as a meta- classifier with six algorithms

Stacking is methodology concerned with combining multiple classifiers generated by different learning algorithms L_1, \dots, L_N on a single dataset S , which is composed by a feature vector $s_i = (x_i, y_i)$, where the base classifier is trained on complete training set, then the meta-model is trained on the outputs of the base level model as features, the other individual model include Support vector machines(SVM-SMO), Decision tree(J48) , Artificial neural network (ANN-MLP) , multinomial Naïve bayes (MNB) and KNN-IBk .

In this study the stacking process is broken down into two stages namely.

1. Generate the base-level classifier C_1, \dots, C_N • Where $C_i = L_i(S)$ in this case Naïve bayes algorithm is selected as one of base classifier.
2. Train a meta-level classifier to combine the outputs of the base-level classifier

Through stacking the model will learn from the data how to combine the class prediction from the different individual classifier to achieve high precision.

Naïve bayes is the simplest learning method, it is based on bayes theorem and an independence hypothesis that generates statistical classifier that is based on probabilities, highly scalable and fast (Sable and Kalavadekar, 2016). Naïve bayes hyper parameter tuning is done based on the table 9. It is for this reason that makes it a good base learner for sms spam detection.

3.7.6 The ensemble hybrid machine learning model

One common deficiency of most ensemble methods is the lack of comprehensibility and complexity issues. Improving the comprehensibility of ensembles is an important aspect yet largely understudied (Gupta et al., 2019). The major limitation is that it generates a lot of false positives and high misclassification, most spammer can navigate through most of them, using newer attacks, the proposed model solves these problems by smartly combining several classifiers using aforementioned pre-processing and feature engineering methods. Most of

these classifier algorithms have unique strengths in different aspects when it comes to client side SMS spam detection. The model undertakes the following stages:

- i. Ensemble data preprocessing by removing noise and outliers in text data
- ii. Ensemble multi-Feature engineering using enhanced feature selection and extraction techniques
- iii. Using machine learning algorithms to discover class correlation among features
- iv. Ensemble hybrid Classification through clustering
- v. Model prototype development and implementation

3.8 Model prototype development and implementation.

The android based client side SMiShing detection model implementation of this research is done using open source machine learning libraries, Java language, JavaScript, XML on android studio IDE with SQLite database for the backend, However this model can also be implemented using other platforms such as the python programming language The model also includes an additional cloud translation API module for multilingual language processing. The final implementation includes the following main modules.

- Listening of incoming messages from the phone.
- Enhance Text pre-processing
- Application of enhanced Feature engineering methods
- Training and testing of the model.
- Message classification and clustering.

3.8.1 Software testing

As part of software validation, module testing is adopted as a software testing type, it checks individual subprograms, subroutines, classes, or procedures in a program. Instead of testing

whole software program at once, module testing recommends testing the smaller building blocks of the program, Module testing avoid redundant activities and checks. This testing is done using JUnit. JUnit is a testing framework for the Java programming language. JUnit has been important in the development of test-driven development (Gromov et al., 2019), it covers faults and defects for a given software.

3.9 Ethical considerations

Given the importance of ethics in conducting research and the challenges around conducting research, it is important for research participants to protect the dignity and safety of the study (Silverman, 2009). This research followed a formal ethical procedure (gaining written consent from participants) was not required the researcher ensured that research ethics were adhered to during the research process. Several ethical considerations were taken into account to ensure that the study was conducted in an appropriate manner (Babbie & Mouton, 2001). To comply with ethical considerations in conducting research all participants provided verbal consent to participate in the research. The participants therefore willingly participated in the study after they were approached by the researcher (Leedy, 2000; Neuman, 2000) and the research purpose and process was explained to them. While it is common practice to request written consent, Silverman (2009) states that highly formalized ways of securing consent should be avoided in favour of fostering relationships in which ongoing ethical regard for participants is sustained. In this study verbal consent was deemed appropriate.

3.9.1 Chapter summary

This chapter provided a detailed research methodology used in this study, it includes the procedures, algorithm techniques, model flow, processes, ensemble and individual hyperparameter tuning, Methods of measurements to evaluate the model. The model is enhanced based on the recent studies. The performance of the model is optimized based on the techniques such as pre- processing, feature engineering, dimension reduction, training, testing and clustering.

CHAPTER FOUR

DATA ANALYSIS PRESENTATION AND DISCUSSION

4.1 Introduction

This chapter gives analysis, interpretations of the research findings as per the objectives in chapter one. The experiments are presented in tables, figures and graphs. The main objective of this study is to design a hybrid ensemble machine learning model to improve the performance of Sms spam detection. To achieve this, the study is performed on five major Stages:

1. Evaluation of the optimal short message service spam preprocessing techniques
2. Determination of the best features for the ensemble hybrid model using hybrid multi-filter feature engineering.
3. Improving Class content correlation model using ensemble hybrid methods to detect sms spam
4. Classifying and clustering text messages by training and testing the model using heterogeneous classifiers
5. Model prototype implementation and testing on mobile android operating system.

An experiment was done to examine 5634 messages as either spam or not, the data was collected from secondary data sources selected from the study.

These experiments were conducted using WEKA, a data mining tool to test the classification accuracy for text data.

This analysis was done with reference to the five objectives aforementioned in chapter 1. The findings were used to explain the results, conclusions and future work.

4.2 Model Pre- processing techniques

After cleaning and preparing the text messages , different preprocessing techniques were considered, Tokenization, Stop Words removal, stemming, lemmatization and TF-IDF on different individual classifiers NB, SVM, J48 ,ANN , MNB using 10-fold cross-validation for evaluating the performance accuracy.

During the experiments, the preprocessing techniques and all their possible combinations were considered: Stop words removal, stemming, lemmatization, Stop word removal and stemming, stop word removal and lemmatization, stemming and lemmatization, stop word and stemming, Tokenization and stemming, Tokenization + Stop words and Stemming, Tokenization, Stop words ,Stemming and Lemmatization and finally, all the techniques combined. Table 10 exhibits the accuracies achieved by the classifiers without using any preprocessing technique and after involving each preprocessing technique and the possible combinations.

Table 10: Ten fold cross validation scores for evaluating pre-processing techniques

Pre-processing Techniques	10 Fold Cross Validation					
	NB	MNB	SVM	J48	KNN	ANN
Without pre-processing	64	63	71	57	62	65
Tokenization only	68.5	64.5	73.34	59.8	63.5	66.3
Stop words only	81.6	77.6	84.5	68.4	70.2	78.6
Stemming only	74.71	79.7	81.05	69.5	68.6	74.2
Lemmatization only	79.4	80.6	82.5	70	69.4	74.9
TF/IDF only	80.6	76.6	83.5	67.4	69.2	76.6
Tokenization Stop words + Stemming	86.6	84.6	87.5	72.4	72.2	80.6
Tokenization + Stopwords + Lemmatization	88.4	86.2	88.5	73.6	74.3	81.9
Tokenization+ Stop words +TF/IDF	90.5	89.6	90.4	85.7	83.9	89.3
Tokenization+Stopwords+Stemming+ Lemmatization	93.8	94.7	91.6	93.8	89.9	91.8
All combine	95.2	96.9	94.5	95.7	94.9	94.8

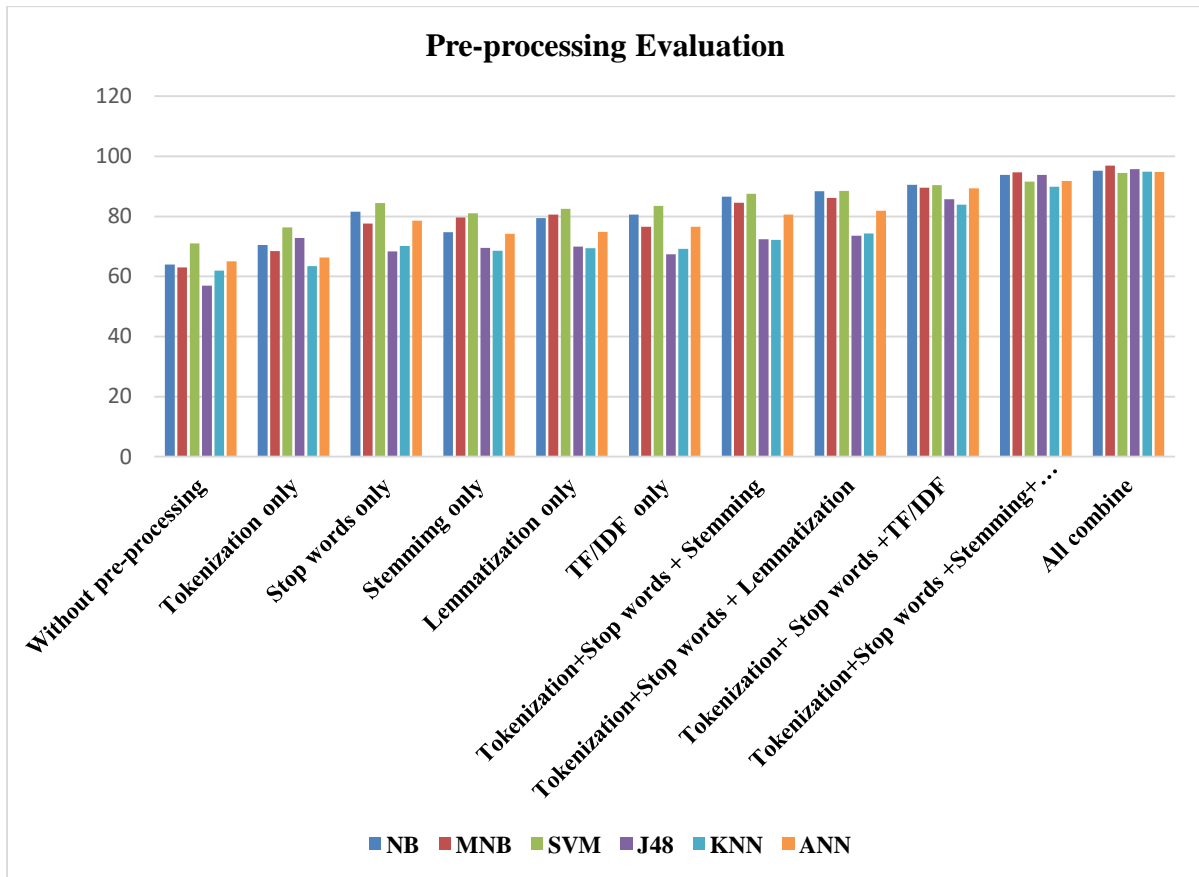


Figure 17: Percentages of the improvement recorded for each classifier

The results obtained prove the effectiveness of the preprocessing techniques selected (+8.3% to +17.6%). Moreover combining more preprocessing techniques led to improved accuracy than using individual (+18.26% to +34.73%), In this experiment, the researcher observe that the average impact of the stop words removal (+13,07%) on the six classifiers had better than that of stemming (+10.96%) , lemmatization (+11.98%) and tokenization (+5.99%). Similarly, the combinations that included stop words, stemming and tokenization yielded a much better performance (+16.9%), stop words, tokenization and TF/IDF is (+18,48%) and stop words removal with lemmatization, stemming and tokenization (+28.9%). In fact, applying exclusively stop words removal is more useful than using stemming and lemmatization combined by (+1.4%). This implies that involving stop word removal is highly recommended

since most multilingual text message contains a lot of irrelevant text, in this case the research found out that the multi-lingual corpus contained an average of 30% stop word content. Thus, removing these stop words maximized the reduction in dimensionality, which enhanced classifiers that suffer badly in high dimensional feature spaces such as the J48. Finally, all the classifiers achieved the best results when all the five preprocessing techniques were combine (an average of +38.7%). On average the accuracy was recorded at 95% (True positives). The performed experiment confirm that the selected preprocessing techniques have a great impact on multi-lingual text classification. Stop words removal has shown to be the most beneficial technique especially when combine with feature techniques.

Table 11: Classification summary per classifier on combine pre -processing techniques

	TP	FP	Precision	Recall	F -measure	MCC	ROC	PRC
Naive bayes	0.952	0.119	0.951	0.952	0.951	0.859	0.957	0.964
J48	0.957	0.210	0.956	0.957	0.955	0.812	0.888	0.939
MNB	0.969	0.065	0.958	0.953	0.952	0.860	0.961	0.969
ANN	0.948	0.174	0.950	0.948	0.946	0.848	0.965	0.974
SVM	0.9450	0.175	0.947	0.945	0.943	0.838	0.885	0.908
KNN	0.949	0.307	0.952	0.949	0.945	0.775	0.821	0.918

Table 11 illustrates the accuracy of data after pre-processing and before feature engineering was implemented, the highest TP was recorded by Multinomial Naïve bayes of 0.969 and a

false positive of 0.065. All classifiers weighted averaged at 0.95 for TP and 0.8 for FP, table 15 shows the same classifiers after feature engineering.

4.3 Enhanced ensemble multi- filter feature engineering methods

On this objective a novel feature engineering method is proposed for the ensemble hybrid modeling. The technique considered include correlation feature selection(CFS) ,Chi square(CS), information gain(IG), Gain ratio (GR) , Symmetrical uncertainty(SU) paired with other dimensional reduction techniques such as BoW modeling, N gramming tokenization, string to word vector(STWV) , stemming , stop words modeling and TF-IDF. The consensus based multi- filter feature selection method averages the output of the ranked important features of information gain (IG), gain ratio (GR), chi-squared (CS), correlation feature selection (CFS) and Symmetric uncertainty (SU). The proposed model removes bias from individual method and is more diverse, stable and robust.

Table 12: Reduction in set of features with attribute selection filter application on the data set

Feature techniques	Features	Top 10 Selected features
No of features without preprocessing + feature selection applied	81,175 tokens	
Information gain + ranker search	1862	114,1730,352,243,1229,821,549,1559,1839,1018
Gain ratio + ranker search	1862	1229,1559,1839,1730,1379,1694,1364,1855,1756,1720
Correlation based + ranker search	71	136,1754,265,1252,572,1583,1863,844,774,1046
Chi square+ Ranker	1862	118,1736,247,1234,554,1565,1845,826,756,1028
Symmetrical uncertainty filter + Ranker	1862	136,1747,1245,265,1576,572,1856,767,1039,1773

The table 12 describe the feature technique and the number of features selected by the technique and the top 10 feature ranked based on the ranking algorithm, Further Table 13 shows the same information in a much detailed version that includes the top ten selected attributes as per ranker algorithm. The ranked decimal column represents how important (correlated) the attribute is to the class based on the feature selection method for example the decimal 0.0584 representing the word claim on information gain column.

The attribute column gives the average position of the attribute in respect to all the selected attributes based on the fold in this case 1229 for the word claim. According to this experiments the top ten common selected words as per the selected methods include call, claim, shinda, call, free, pesa, zawadi, tuma and prize. The average rank for Information Gain, Mutual information, Chi-Square Symmetrical Uncertainty, Correlation Feature is 0.05562, 0.3613, 0.3275, 0.1296 and 0.3275 respectively, the best registered correlation rank in order; mutual information, Chi square, Correlation feature selection, symmetric uncertainty and information gain the average rank difference is between 4% and 30 %.

Table 13: Feature engineering techniques with feature search algorithms

Information Gain		Mutual information		Chi-Square		Symmetrical Uncertainty		Correlation Feature	
Ranked	Attr	Ranked	Attr	Ranked	Attr	Ranked	Attr	Ranked	Attr
0.0927	114 call	0.415	1229 claim	0.431	118 call	0.1772	136 call	0.431	136 call
0.0669	1730 txt	0.39	1559 zawadi	0.386	1736 shin	0.1744	1747 txt	0.386	1754 txt
0.0601	352 click	0.363	1839 www	0.358	247 free	0.1615	1245 claim	0.358	265 free
0.0584	243 free	0.361	1730 txt	0.354	1234 claim	0.1414	265 free	0.354	1252 claim
0.0584	1229 claim	0.357	1379 won	0.322	554 mobile	0.1278	1576 prize	0.322	572 mobile
0.0512	821 toll	0.351	1694 Pesa	0.309	1565 prize	0.1252	572 mobile	0.309	1583 prize
0.0461	549 mobile	0.349	1364 guarant	0.305	1845 www	0.1206	1856 www	0.305	1863 www
0.0445	1559 prize	0.349	1855 mobile	0.278	826 tuma	0.0907	767 stop	0.278	844 to
0.0421	1839 www	0.341	1756 uk	0.274	756 stop	0.0903	1039 won	0.274	774 stop
0.0358	1018 your	0.337	1720 tone	0.258	1028 please	0.0869	1773 uk	0.258	986 won

In table 14 the average merit tells us how important that attribute is (i.e. the higher the number the better), the attribute in the table 13 were mapped into table 14 for further analysis, this is averaged over the folds of the cross validation .The average rank is the average of its ranking throughout the cross validation and the numbers following “+ -“are the standard deviation as per table 14 for different feature engineering methods.

Table 14: Attribute Selection with 10 fold cross-validation

Information Gain		Mutual information		Chi-Square		Symmetrical Uncertainty		Correlation Feature	
Avg merit	Avg rank	Avg merit	Avg rank	Avg merit	Avg rank	Avg merit	Avg rank	Avg merit	Avg rank
0.093 +- 0	1 +- 0	0.415 +-1 +- 0 0.002		0.431 +-1 +- 0 0.007		0.177 +-1 +- 0 0.005		0.431 +-1 +- 0 0.007	
0.067 +- 0.002	2 +- 0	0.39 +-2 +- 0 0.002		0.386 +-2 +- 0 0.007		0.174 +-1.8 +- 0.4 0.005		0.386 +-2 +- 0 0.007	
0.06 +- 0.001	3.5 +- 0.5	0.363 +-3.4 +- 0.49 0.006		0.358 +- 3.4 0.008 0.49		+ -0.161 +-3 +- 0 0.003		0.358 +- 3.4 +- 0.008 0.49	
0.058 +- 0.002	4.2 +- 0.98	0.361 +-4.3 +- 1.42 0.008		0.354 +-3.6 +- 0.49 0.004		0.141 +-4 +- 0 0.006		0.354 +-3.6 +- 0.49 0.004	
0.058 +- 0.001	4.3 +- 0.64	0.357 +-4.6 +- 0.66 0.002		0.322 +-5.1 +- 0.3 0.007		0.128 +- 5.3 +- 0.46 0.002		0.322 +-5.1 +- 0.3 0.007	
0.051 +- 0.002	6 +- 0	0.349 +-7 +- 0.45 0.003		0.309 +-6.1 +- 0.54 0.003		0.125 +- 6 +- 0.77 0.005		0.309 +-6.1 +- 0.54 0.003	
0.046 +- 0.002	7.1 +- 0.3	0.349 +-7.1 +- 1.22 0.004		0.305 +-6.8 +- 0.4 0.004		0.121 +-6.7 +- 0.46 0.003		0.305 +-6.8 +- 0.4 0.004	
0.044 +- 0.001	7.9 +- 0.3	0.351 +-7.1 +- 1.14 0.002		0.278 +- 8.4 0.005 0.49		+ -0.309 +-6.1 +- 0.54 0.003		0.278 +- 8.4 +- 0.005 0.49	
0.042 +- 0.001	9 +- 0	0.332 +-13 +- 2.05 0.004		0.274 +- 8.9 0.005 0.83		+ -0.274 +- 8.9 +- 0.83 0.005		0.274 +- 8.9 +- 0.005 0.83	
0.036 +- 0.002	10.4 +- 0.66	0.036 +-10.4 +- 0.66 0.002		0.268 +-10.2 0.007 1.17		+ -0.268 +-10.2 +- 1.17 0.007		0.268 +-10.2 +- 0.007 1.17	

This strategy evaluates the worth of an attribute by repeatedly sampling an instance and then accessing the distance to the nearest instance of the same class in comparison to a different class (Demisse et al., 2017). According to the attributes selected for sms spam detection, the average merit values for the selected attributes ranged from 0.4 to 0.9 as shown in Table 14, the higher the merit score the better the correlation to the class hence the method.

4.3.1 Ranking the retained feature sets by feature selection method

The number of features retained in the data set is compared across the five feature selection methods and the feature selection method which yielded in retention of least number of features is ranked with rank 1 and next least number of features is ranked rank 2 and so on, In cases where same number of features are retained (this is repeated for execution time and other measures) same rank is given to both however the next rank is jumped for the next lower valued feature retention. The lowest number of retained features is ranked as 1 because the algorithm runs more efficiently due to less computation, CFS method recorded the least features in this case. Table 15 shows the ranks based on the number of features retained post application of the various attribute selection filters

Table 15: Rankings based on retained reduced feature sets

Rankings based on retained reduced feature sets				
Chi-square(CS)	Information gain(IG)	Gain ratio(GR)	Correlation Selection (CFS)	symmetric uncertainty(SU)
2	2	2	1	2

Correlation feature selection was ranked 1 since it had the least amount of features retained, all the other four methods were given ranked 2 because they retained same amount of features as shown in table 15, this ranking is useful since it gives an indication on which method is appropriate for the ensemble model optimization.

Table 16: Accuracy performance of feature selection methods with individual classifiers.

Feature selection method	NB	J48	SVM	IBK	ANN	MNB	Rank
CFS	98	97	98	94	95	96	3
SU	97	96	97	94	96	97	4
GR	97	96	98	95	96	98	5
IG	97	97	98	96	97	97	1
CS	98	95	98	95	97	98	2

Comparing accuracy before Feature selection and after , the researcher observed an improvement in accuracy in most classifiers .Table 16 show the results for each feature selection method (rows) and the classifiers(Columns) .For all the filters used , accuracies obtained improved better than the accuracies obtained with the same models without any filters applied. The researcher found out that the accuracies recorded were almost similar for Chi Squared filter (CS), and Information Gain filter (IG), CFS and SU. Generally CFS improved on accuracy by 2.4 % CS by 2.8% by GR by 1.7% by IG 3% SU by 2.2%. According to table 16 the filter method with the highest average accuracy was assigned rank 1 ,second best rank 2 and where the accuracy is shared same rank is given and the next one is skipped by 1.

The researcher noticed that most filters scored close values in terms of accuracy, Therefore accuracy alone is not the sole determinant in selecting the best feature method. The time taken to build the model, feature retained, Kappa, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) were also considered in determining the effectiveness of these methods.

Table 17: Classifiers execution time as per the feature selection methods

Feature selection method	NB	J48	SVM	IBK	ANN	MNB	Rank
CFS	3.82	94.3	6.71	0.01	0.2	0.06	1
SU	3.15	96.31	5.71	0.02	1	0.05	5
GR	2.71	98.12	5.51	0.15	0.17	0.03	4
IG	2.81	96.81	5.87	0.02	1	0.01	1
CS	2.6	96.41	5.51	0.01	2	0.02	3

The average execution time for the filters was also recorded, the optimal execution time was recorded by IG and CFS and was assigned rank 1 followed by CS ranked as 3, GR ranked 4 and SU ranked 5 as shown in table 17 the average time taken by the methods was recorded at 17.6 seconds. The worst time on specific classifier was recorded by J48 decision tree with an average of 96.9 seconds the best recorded by multinomial naïve Bayes with an average 0.034 seconds.

Table 18 illustrates kappa statistic for the feature methods. Kappa statistic is used to measure the degree of agreement for categorical data. It is generally robust measure than simple percentage agreement calculation as it takes into account the possibility of the agreement occurring by chance. When kappa statistic value is closer to 1, then a model is said to be more accurate (Mchugh,2012).

Table 18: KAPPA statistic metric per feature model on classifiers

KAPPA statistic metric per feature model on a classifier							
Feature selection method	NB	J48	SVM	IBK	ANN	MNB	Rank
CFS	0.8772	0.8067	0.9294	0.7528	0.9110	0.9339	5
SU	0.8931	0.8167	0.9308	0.7528	0.91	0.9339	4
GR	0.8772	0.8067	0.9294	0.7788	0.8922	0.9002	3
IG	0.9772	0.928	0.9156	0.9881	0.9922	0.9002	1
CS	0.948	0.918	0.9294	0.9552	0.9252	0.9339	2

The highest Kappa across model is given rank 1 and lowest kappa is given rank 5 as shown in table 18. In situations of same kappa, same rank is assigned to both however the next rank is skipped for purpose of assigning the next rank. Highest valued kappa is given the rank 1 because the closer the kappa statistic is to 1 the better model. The best KAPPA was recorded by information gain of 0.95, the least KAPPA recorded by Correlation feature selection of 0.87,

on average most of the feature selection performed fairly with cumulative average agreement of 0.89.

In considering the mean absolute error the lowest mean absolute error is given rank 1 and highest mean absolute error is given rank 5 , for RMSE the lowest root mean squared error across each model is given rank 1 and highest root mean squared error is given rank 5 as shown in table 19 and 20 respectively.

Table 19: MAE statistic metric per feature model on a classifier

MAE statistic metric per feature model on a classifier							
Feature Method	NB	J48	SVM	IBK	ANN	MNB	Rank
CFS	0.032	0.0646	0.0165	0.0605	0.0208	0.0214	4
SU	0.0285	0.0581	0.0162	0.0605	0.0208	0.0214	2
GR	0.032	0.0646	0.0165	0.0584	0.0254	0.0248	5
IG	0.032	0.0293	0.0197	0.0584	0.0254	0.0248	1
CS	0.041	0.0982	0.0165	0.0139	0.0174	0.0214	3

The best MAE was recorded by IG of 0.031 % .SU 0.034 % CFS 0.035 % GR 0.036 % and CS 0.037 % , and on average the MAE is about 0.034 % , these values are closer to 0, which concludes that most of these feature methods are good for improving model performance.

Table 20: RMSE statistic metric per feature model on a classifier

RMSE statistic metric per feature model on s classifier							
Feature selection method	NB	J48	SVM	IBK	ANN	MNB	Rank
CFS	0.1574	0.2011	0.1285	0.2228	0.1441	0.116	3
SU	0.1711	0.2293	0.1285	0.2594	0.1319	0.116	5
GR	0.1574	0.2011	0.1285	0.2138	0.1593	0.1495	4
IG	0.1574	0.0982	0.1404	0.2138	0.1593	0.1495	2
CS	0.1475	0.1973	0.1271	0.2228	0.1441	0.116	1

The paramount RSME was recorded by CFS as 0.15 % and on average RMSE is approximately 0.16 % for rest of the methods, this value is also close to 0, Lower values of RMSE indicate better fit of the model. This is a good measure of how accurate the model predicts the response, this is an important criterion for fit, especially for prediction purposes. Table 21 shows the performance of different feature selection methods on the data set with the following algorithms Naïve bayes (NB), J48 tree, Multinomial naïve bayes (MNB), Support vector

machines (SVM), ANN and IBK in terms of accuracy of classification for both spam and Non spam classes.

Different ranks are assigned to the filters based on diverse parameters such as execution time, KAPPA, MAE, RMSE and features retained, in order to arrive at non biased conclusions. To achieve this the researcher computed final rankings from the various individual ranks by adding the ranks obtained across all models. The summation was computed across for CS, GR, IG, CFS and SU attribute selection filters. The lowest aggregate among the obtained aggregates is assigned a rank of 1 and the highest aggregate is ranked with possible rank of 5. The cumulative rank aggregation metric of information gain (IG) is 8 and this is an improvement by 5 ranks compared to the aggregated metric of Chi Square (CS). Overall Information Gain feature selection filter's rank aggregation metric is better by 5, 8, 14, 15 ranks when compared to the rank aggregation metrics of CS, CFS, SU and GR respectively. Through these quantitative measurements, the researcher adopted an ensemble filter based feature ranking of information gain (IG), Chi-Square (CS) and CFS feature selection methods as per the proposed feature selection algorithm 6. This ensemble feature ranking was adopted for this study to build the ensemble hybrid model. Table 21 shows the overall accuracy comparison of the proposed method that gave an average accuracy of 96.7%. The feature combination methods presents a slight increase in classification accuracy by 1.38 % as compared to previous accuracy before feature engineering recorded as 95.33%.

Table 21: Accuracy performance of feature selection with the different classifiers

Feature method	NB	J48	SVM	IBK	ANN	MNB	Rank
CFS	98	97	98	94	95	96	3
SU	97	96	97	94	96	97	4
GR	97	96	98	95	96	98	5
IG	97	97	98	96	97	97	1
CS	98	95	98	95	97	98	2

4.3.2 Experiment of bagging and boosting with six classifiers

After the feature selection method, An experiment was further conducted (table 29) with bagging and boosting ensemble method on the classifiers using 10 fold cross validation to determine the overall accuracy, true positive(tp), false positive(fp) and kappa statistic of the model.

Table 22 Bagging and Boosting performance evaluation

Classifier	Accuracy		True positive		False positive		Kappa	
	Bagging	Boosting	Bagging	Boosting	Bagging	Boosting	Bagging	Boosting
J48	97	98	0.97	0.98	0.116	0.128	0.8190	0.8839
NB	96.6	97	0.966	0.971	0.116	0.098	0.8587	0.8654
REPTree	96.5	96.8	0.966	0.968	0.146	0.134	0.8522	0.8786
IBK	93.628	94.5	0.936	0.945	0.388	0.295	0.6735	0.7214
SVM	98.2	98.3	0.982	0.983	0.088	0.074	0.9235	0.9345
ANN	98.5	97.6038	0.985	0.976	0.081	0.072	0.9235	0.9005

Generally most algorithm gave good results for both bagging and boosting, however boosting gave much better results, both bagging and boosting suffered relatively high computational time .The highest accuracy recorded was 99 % by SVM and the lowest being 94% by IBK. When adaboostm1 algorithm is implemented with J48 as a weak classifier it achieved the best accuracy of 98% , with true positive of 0.98% and false positive 0.128 % Naïve bayes achieved the best TP of 97% . IBK recorded the highest false positive of 0.388 this is because IBk is a lazy classifier. On these two ensemble methods (bagging and boosting),There was slight improvement especially in bagging for example ANN improved with 1% .Boosting was able to reduce the false positives between 0.003% and 0.001 % . The highest recorded

Kappa statistic was recorded by SVM of 0.9345 and the lowest was recorded by IBK 0.6735. In conclusion bagging improved the stability and accuracy of the classifiers while boosting converted weak learners such as IBK to strong ones by improving on their false positives, however one greatest disadvantage of bagging is that it introduces a loss of interpretability of the model. The resultant model experienced lots of bias when the proper procedure is ignored. From the experiment, despite bagging being improving on accuracy, it is computationally expensive, this discouraged its use in certain instances. On the other hand boosting is sensitive to outliers since every classifier is obliged to fix the errors in the predecessors. Thus, this method is too dependent on outliers, in addition to the difficulty in detecting outliers from natural variations in text messages (Kannan et al., 2017).

4.3.3 Multi-classifier using stacking with six machine learning algorithms

In stacking the individual classification models are trained based on the complete training set; then, the meta-classifier is fitted based on the outputs meta-features of the individual classification models in the ensemble. The meta-classifier can either be trained on the predicted class labels or probabilities from the ensemble. This was achieved with six algorithms, Naïve bayes, Support vector machines (SVM-SMO), Decision tree (J48), Artificial neural network (ANN), multinomial Naïve bayes (MNB) and KNN-IBk. The base classifier considered includes Naive Bayes. The other classifiers learns from the classification output given by the base classifiers after feature stage. The next stage classifies the test data using Meta classifier. Several combinations of these algorithms were used for stacking using 10 fold cross validation to determine the optimum combination. Naïve bayes is one of the most used algorithm for text classification and specifically for text classification. 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 (Saritas et al., 2019). Naïve bayes is among the popular method for text categorization especially in the problem of documents classification that a word belongs to one category or the other document categorization as used in areas like spam sports or politics among others. With word frequencies as the features and appropriate pre-processing, it a very competitive method in this domain as compared to more advanced classifiers such as SVM (Rennie, Shih, Teevan and Karger, 2003).

Table 23: Performance of Naïve bayes as a base learner and stacking with six classifier

Stacking	TP	FP	Precision	F-M	MCC	ROC	PRC	Execution
NB	0.964	0.117	0.964	0.964	0.848	0.976	0.987	3.09 sec
NB+ SVM	0.982	0.077	0.982	0.982	0.930	0.949	0.970	61.45 sec
NB+ J48	0.975	0.079	0.975	0.975	0.896	0.967	0.975	902.32 sec
NB+ ANN	0.979	0.105	0.978	0.978	0.908	0.938	0.67	1.67 sec
NB+IBK	0.976	0.112	0.975	0.975	0.895	0.991	0.994	16.43 sec
NB+ ANN+SVM	0.983	0.064	0.983	0.983	0.927	0.960	0.997	62.62 sec
NB + ANN+	0.975	0.079	0.975	0.975	0.896	0.967	0.975	475.88 sec
SVM+J48								
NB+ANN+IBK	0.975	0.064	0.976	0.976	0.899	0.976	0.978	443.88 sec
SMO+ J48+ MNB								

Table 23 shows the accuracy FP,TP, MCC ROC for the stacking experiment using the propose feature selection algorithm 6. Naïve bayes as the base classifier registered a True positive of 0.964 and false positive of 0.117 , F measure of 96.45 , precision of 96.4 % , ROC of 0.976 , all these recorded a time of 3.09 sec, However when naïve bayes was stacked with SVM ,True positive improved with 0.019 and false positive reduced with 0.117%, the execution time increased to 61.45 sec. When ANN was added the TP remained the same but false positive reduced further by 0.013%, the execution time increased to 62.62 sec i.e. an increment of 1.17 seconds. The computational time significantly increased every time J48 classifier was stacked with the base classifier regardless of the order of classifiers. The highest accuracy was recorded at 98.3 when Naïve bayes, ANN SVM were stacked, the lowest False positive was recorded at 0.064 when Naïve bayes ANN IBK, SVM J48 and MNB. The highest ROC and PRC was recorded by IBK of 0.991 and 0.994 respectively. A ROC (Receiver Operating Characteristics) analysis based approach was approved to evaluate the performance of different classifiers. In ROC every classifier, TP (True Positive) and FP (False Positive) are calculated and mapped to a two dimensional space with FP on the x-axis and TP on the y-axis. The most efficient classifiers should lie on the convex hull of this ROC plot since they represent the most efficient TP and FP trade off as shown in table 30 . NB, ANN, IBK, SVM, J48 and MNB stacked recorded a ROC curve of 0.976 and PRC of 0.978.. C4.5 (J48) is an algorithm used to generate a decision tree developed by Ross Quinlan (1993). C4.5 is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier. A significant advantage of a J48 is that it forces the consideration of all possible outcomes of a decision and traces each path to a conclusion (Onik et al., 2015). It creates a comprehensive analysis of the consequences along each branch and identifies decision nodes that need further analysis. The main

disadvantage of J48 is a small change in the data can cause a large change in the structure of the decision tree causing instability. Training a J48 decision tree is relatively expensive as complexity and time taken is more it is this reason that J48 recorded high execution time despite recording an improved accuracy. The high execution time was due to the huge size of the tree generated that led to a dense tree and overfitting of data. J48 recorded an execution time of 902.32 second TP of 0.975 and FP of 0.075, this FP was the second best lowest false positive recorded. Support-vector machines (SVM) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (Chang and Lin, 2011). SVM Works well with unstructured and semi structured data like text, Images and trees. The kernel trick is real strength of SVM and in this study poly-kernel was used. SVM models have generalization in practice, the risk of over-fitting is low, however it involves long training time for large datasets but when compared with J48, SVM cross validation time is much less. The weighted True positive of SVM was recorded of 0.982 and false positive of 0.077. ANN is a machine learning algorithm based on the model of a human neuron. It is an information processing technique that works like the way human brain processes information (Chen et al., (2019). ANN includes a large number of connected processing units that work together to process information,. Neural Networks have the ability to learn by themselves and produce the output that is not limited to the input provided to them. The input is stored in its own networks instead of a database, hence the loss of data does not affect its working. In this study Neural network paired with Word2Vector algorithm gave better results in terms of accuracy. It recorded a TP 0.979 and FP 0.105. Instance based learner (IBk) is a simple, supervised machine learning algorithm that can

be used to solve both classification and regression problems. It's easy to implement and understand, but has a major drawback of becoming significantly slower as the size of that data in use grows (Dwivedi and Arya, 2016). IBk algorithm does not build a model, instead it generates a prediction for a test instance just-in-time. The IBk algorithm uses a distance measure to locate k “close” instances in the training data for each test instance and uses those selected instances to make a prediction. IBk is a lazy learner because it doesn't learn a discriminative function from the training data but “memorizes” the training dataset instead (Ghareb et al., 2016), therefore no training time (skipping training) which means its execution time is low, but this comes with a cost i.e. the prediction step in IBk- K-NN is relatively expensive. Each time we want to make a prediction, K-NN is searching for the nearest neighbor(s) in the entire training set. This explains why when IBK was stacked on the model the execution time reduced by 32 seconds but true positive also reduced by 0.008%.

Table 24: Stratified 10 fold cross validation summary for the stacked model.

Correctly Classified Instances	5536	98.2606 %
Incorrectly Classified Instances	98	1.7394 %
Kappa statistic	0.9268	
Mean absolute error	0.0181	
Root mean squared error	0.13	
Relative absolute error	7.5369 %	
Root relative squared error	37.5297 %	
Total Number of Instances	5634	

In this study it is clear that combining classifiers using stacking algorithm was observed to give a better accuracy compared to bagging and boosting since it provide more diversity of classifiers and also recorded high true positive and low false positive. A stacked model of Naïve bayes, Artificial Neural Network and Support vector machine recorded the most optimal solution with highest precision of 98.3% and a low false positive of 0.064 %.

Table 25: Detailed accuracy of the ensemble hybrid stacked model

	TP Rate	FP Rate	Precision	Recall	F- Measure	MCC	ROC	PRC- Area	Class
	0.992	0.073	0.988	0.992	0.990	0.927	0.960	0.987	Ham
	0.927	0.008	0.947	0.927	0.937	0.927	0.959	0.912	Spam
W.Avg	0.983	0.064	0.982	0.983	0.983	0.927	0.960	0.977	

This model also recorded a Kappa statistics of 0.9268, mean absolute error of 0.0181, MCC of **0.927** , Root mean squared error 0.13 and ROC of 0.960 , the ROC Curve is shown in figure 18 . The table 24 and 25 gives the summary of this adopted model.

The confusion matrix for this model is described in Table 26, it recorded correctly classified at 5536 instances and incorrectly classified as 98 instances which is 98.2606 % and 1.7394 % respectively.

Table 26: Confusion matrix

Predicted		
A	B	Classified as
4808	41	a= NOT spam (Ham)
57	728	b=SPAM

The confusion matrix in Table 26 recorded correctly classified at 5536(4808+728) instances and incorrectly classified as 98 (41+57), the integer 4808 are the Non spam messages that were correctly classified and 57 were misclassified instances.

For Spam messages 728 messages were classified correctly (TP) and 41 was misclassified which is cumulatively 98.2606 % and 1.7394 % respectively.

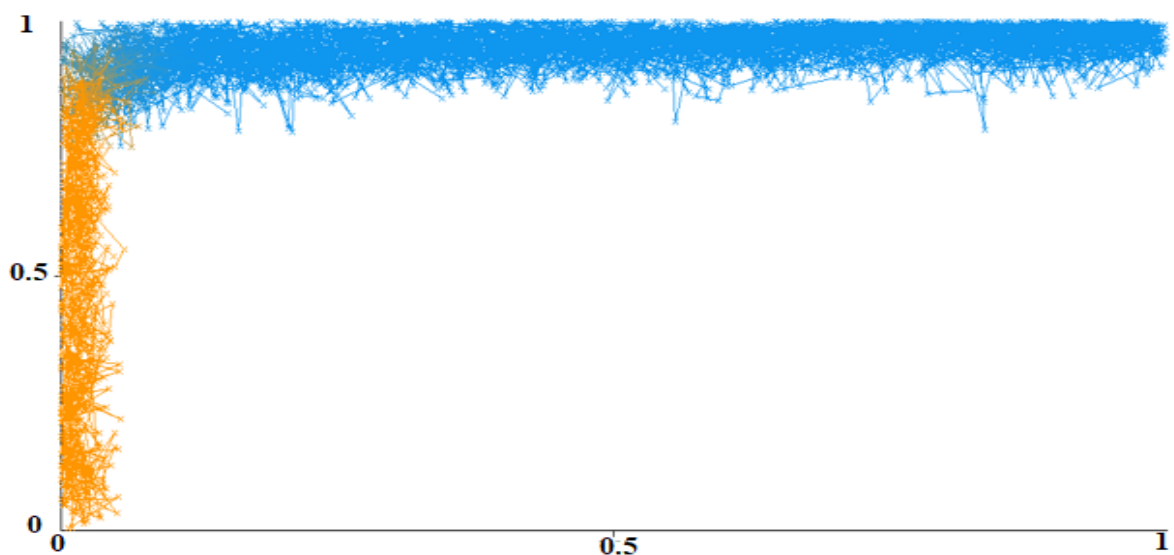


Figure 18: Receiver Operating Curve

ROC graphs in Figure 18 is a two-dimensional graph in which TP (true positives) rate is plotted on the Y axis and FP (false positive) rate is plotted on the X axis. A ROC graph depicts relative tradeoffs between benefits (true positives) and costs (false positives). To compare classifiers studies have reduced ROC performance to a single scalar value representing expected performance. A common method was to calculate the area under the ROC curve. Since the Area Under this Curve is a portion of the area of the unit square, its value will always be between 0 and 1.0. However, because random guessing produces the diagonal line between (0, 0) and (1, 1), No classifier should have an AUC less than 0.5 (Bradley, 1997; Hanley and McNeil, 1982), The stacked model presented a ROC of 0.960 .This gave an important statistical property since the AUC of the model is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. This Graph is very useful tool for visualizing and evaluating the model. It provided a richer measure of classification performance than scalar measures such as accuracy, error rate or error cost, because they decouple the classifier performance from the class of spam or Not Spam.

4.4 Improving learning using clustering techniques

Clustering is the process of using unsupervised machine learning algorithm to improve learning for supervised algorithms. Therefore, all the 785 spam messages from the dataset were further tested, the 785 spam messages are categorized into 12 clearly defined groups

G_i , where $i = 0 \dots 11$, the categories includes competitions, Chat, claims, dating, prizes, services, banking , ringtones, credentials , news , links and monetary

Table 27: Clustering of 785 spam messages using three algorithms

Cluster algorithm	Prediction results	Processing time (sec)
K means	G0: 212 items G1: 2 item G2: 15 items G3: 2 items G4: 6 items G5: 2 items G6: 501 items G7: 2 items G8: 2 items G9: 3 items G10: 3 items G11: 35 items	2.8
Hierarchical	G0: 767 items G1: 2 items G2: 2 items G3: 2 items G4: 2 items G5: 2 items G6: 2 items G7: 2 items G8: 2 items G9: 2 items G10: 2 items G11: 2 items	1
Cobweb	G0: 785 items	11

The findings of the clustering as per the stacked shows that hierarchical algorithm took 1 second , k means 2.8 and cobweb 11 seconds , the prediction results shows that cobweb algorithm had only one cluster of 785 , this shows bias on the algorithm. Hierarchical though it took less amount of time also recorded bias since most items were categorized with same cluster, numerically indicated as 2. In conclusion, K-Means algorithm is the best suited to cluster the 785 spam messages as per the 12 groups in table 27. Further , When K means Clustering algorithm was used on all the data (Test mode split 66% train, remainder 34% for

testing), and the parameters used include : K means ++ for choosing the initial values (or "seeds") , Euclidean distance function (Arthur and Sergei , 2007) .

Initially a seed of 10 was used for making the initial assignment of instances to clusters. Canopy periodic pruning rate was used in order to allow just one pass over the randomly selected data, Euclidean distance function was used to assign a correct label for an instance. Table 28 indicates the centroid of each cluster as well as statistics on the number and percentage of instances assigned to different clusters. Cluster centroids are the mean vectors for each cluster (so, each dimension value in the centroid represents the mean value for that dimension in the cluster), Thus, centroids were used to characterize the clusters. The number of iterations performed was 2 and the clustered sum of squared errors is 64123.522727272044.

Initial starting points (random) was random based on the seeding. The centroid for cluster 0 shows that this is a segment of cases representing random messages number with the mean value for example message 158 has a mean value of 1.647025 , message 262 has mean value of 1.536989 .For example cluster 1 message the 15 has a mean of 2.678482 ,a summary of this data is given below .

Cluster 0: {158 1.647025,262 1.536989,414 3.439982,668 2.585925,680 1.162856,747 2.408525}

Cluster 1: {15 2.678482,60 3.439982,163 2.023831,174 2.198029,215 2.530443,226 2.134972,243 1.334411,247 2.785331,331 4.027284,342 2.785331,344 2.285886,445 1.998623,450 1.751394,470 3.066378,482 2.408525,503 3.010896,731 1.32968}.

Table 28: Cluster centroids for selected words

Attribute	Full Data	0 (spam)	1 (ham)
	(5634.0	(785)	(4849.0)
Sportpesa	0.0048	0.008	0
Usilipe	0.0048	0.008	0
Stop	0.0522	0.0644	0.034
Ameumwa	0.0121	0.0121	0
August	0.0153	0.0153	0
Bank	0.0213	0.0213	0
Goodnight	0.0086	0.0086	0
Money	0.0562	0.0536	0.6785

Clustering results based on classes is shows that the time taken to build the model on full training data ,It took 5.55 seconds , Furthermore, the K- means algorithm preferred clusters of approximately similar size, as they will always assign an object to the nearest centric which leads to incorrectly cut borders in between the clusters , this is not surprising as the algorithm optimizes class centers and not borders (Yang and Pedersen (1997). Figure 19 shows an

example of clustering the x axis is the class (spam and ham) and y axis is the distribution of one of the attribute “mshindi”



Figure 19: word clustering

4.5 Model comparison with existing studies

The model was further compared with recent existing studies Choudhary et al.,(2017), this study used Naïve bayes(NB) , Decision tree (DT)and regression trees(RT) this single classifier recorded accuracies of 94.1% ,96 % and 96.5 % respectively , TP of 0.941, 0.96, and 0.965. The proposed study was also compare with Almeida (2011) , this particular study used linear SVM that recorded an overall accuracy of 97.6%, compare to the proposed model of 98.2 %.

Bottazzi et al (2015) method used J48 and Bayes Net individual classifiers that recorded an overall accuracy of 89.2% and 78% respectively. The summary of these comparison is shown in Table 29 and a graph in figure 20.

Table 29: Comparison of various model with the stacked model

Study order	Data Set	Method	Accuracy	TP	FP
1	Sms Spam collection UCI+ local	NB+ANN+SVM	98.2%	0.983	0.064
2	Sms Spam collection	NB	94.1%	0.941	0.077
		DT	96 %	0.96	0.133
		RT	96.5%	0.965	0.102
3	Sms Spam	Linear SVM	97.6%	0.96	0.18
4	Sms Spam collection	J48	89.2%	0.144	0.92
		Bayes Net	78%	0.304	0.781
5	Sms Spam	SMO,NB	96%	0.326	0.782

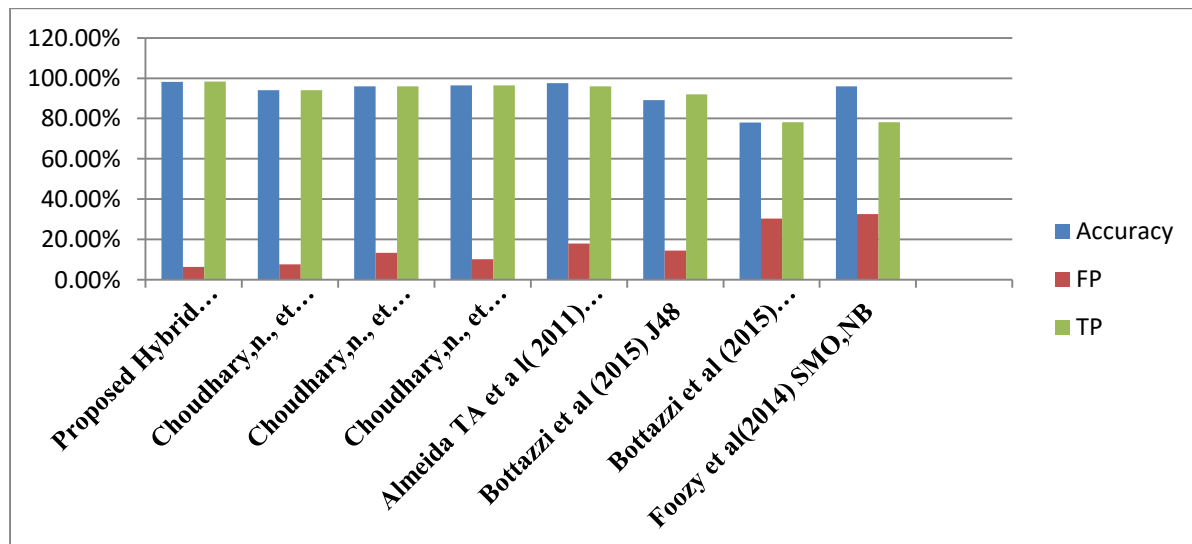


Figure 20: A comparison graph for related studies.

4.6 Prototype design and implementation

In this section an overview of the ensemble hybrid client side sms spam detection modules are presented as per the model. After having evaluated the model. The researcher implemented it as per the reasons aforementioned previously. The application architecture is shown in figure 21. The architectures involves the initial stage of receiving incoming raw messages which are pre preprocessing (stop words, tokenization and stemming), TF and IDF, training and testing. To achieve this implementation it was necessary to comprehend well not only how the hybridized ensemble model is designed, but also an in depth pre-processing and feature engineering procedures, the steps are illustrated below. The top-level pseudo code developed in Java language. The coding developed in this research work consisted of four modules. The model is implemented at the client- side portable android smart device GSM capabilities. The user must allow the application to listen to incoming messages through phone settings.

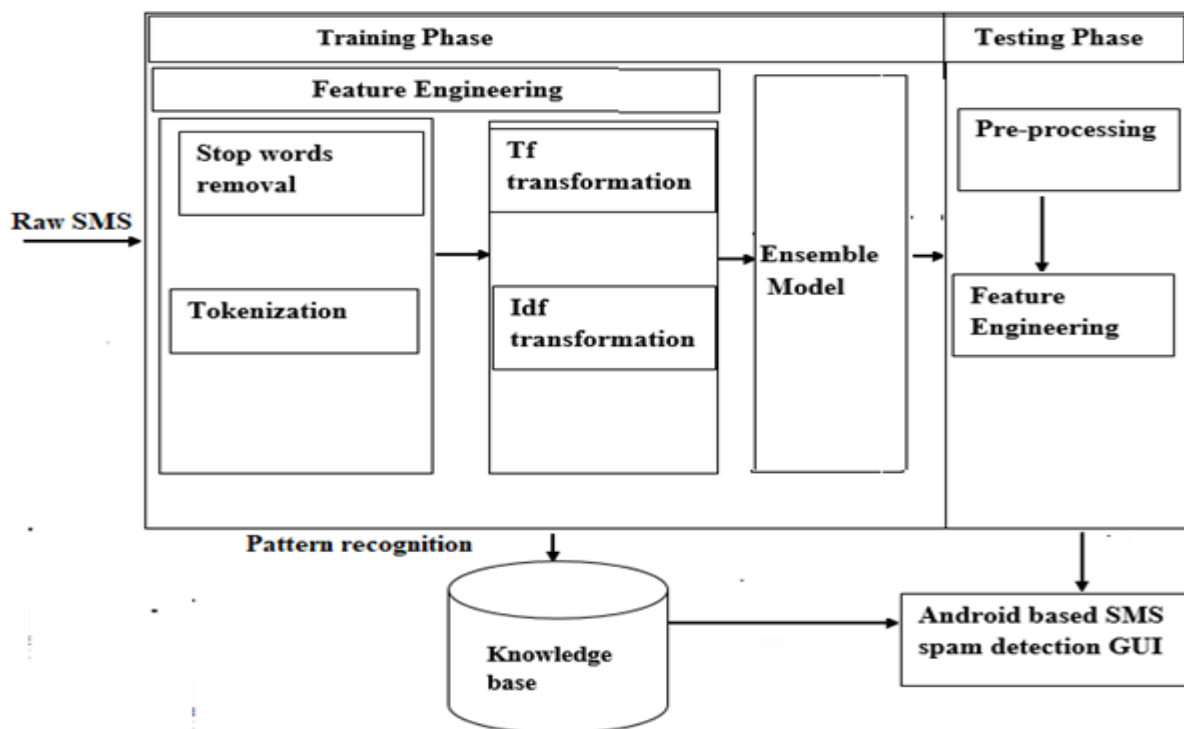


Figure 21: Android based prototype architecture.

The GUI is developed using JAVA and android studio Integrated Development Environment (IDE) with SQLite database which contains the messages for training and testing .The system GUI conforms to user interface design principles that's satisfies the following user interface design principles

- Clear
- Consistent
- Efficient
- Responsive
- Reliable

The incoming text message is first translated into English from Swahili text messages when required, if the message is in English then there is no need of translation. Figure 22 Shows how this is done, If a message is detected as SPAM the user has an option of saving it or dismissing it, If the save option is selected the user can view the saved message in a different folder for future reference which also include the Spam contacts associated with the Spam messages received.

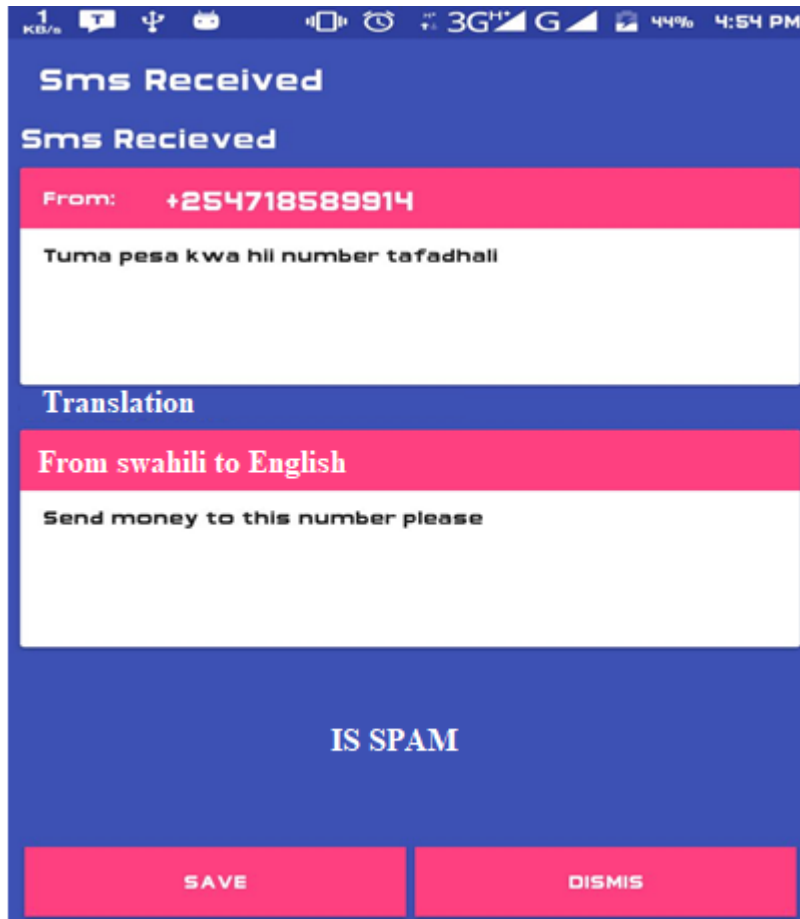


Figure 22: The translation example.

Other prototype capabilities include training of data that involves selecting of an external file that contain list of messages label as spam or not, and testing of data from the back end as shown in figure 23,24 and 25

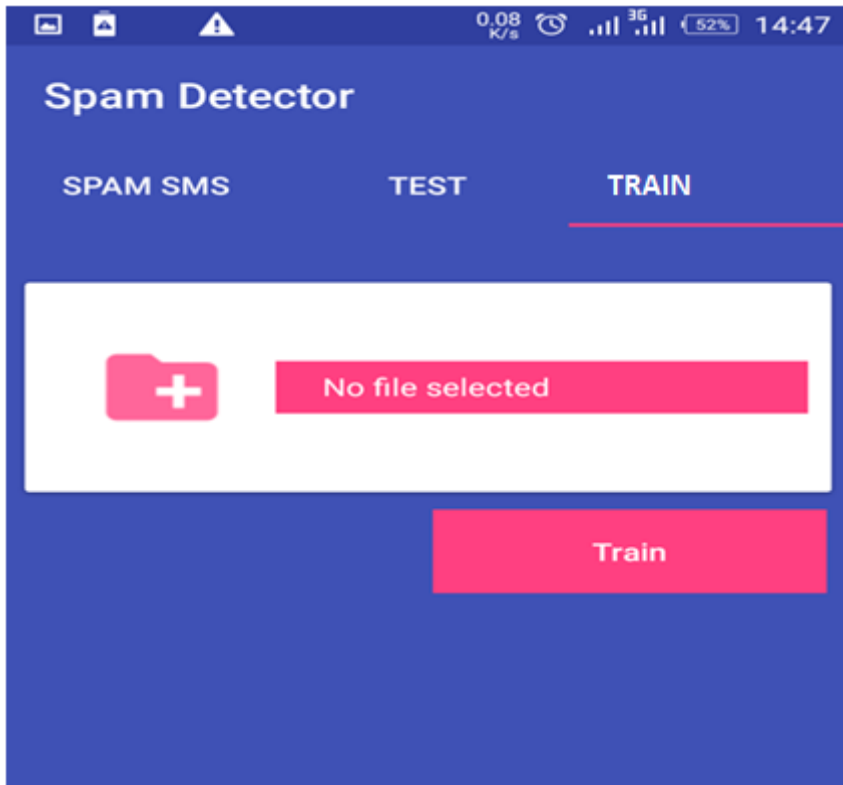


Figure 23: Training module

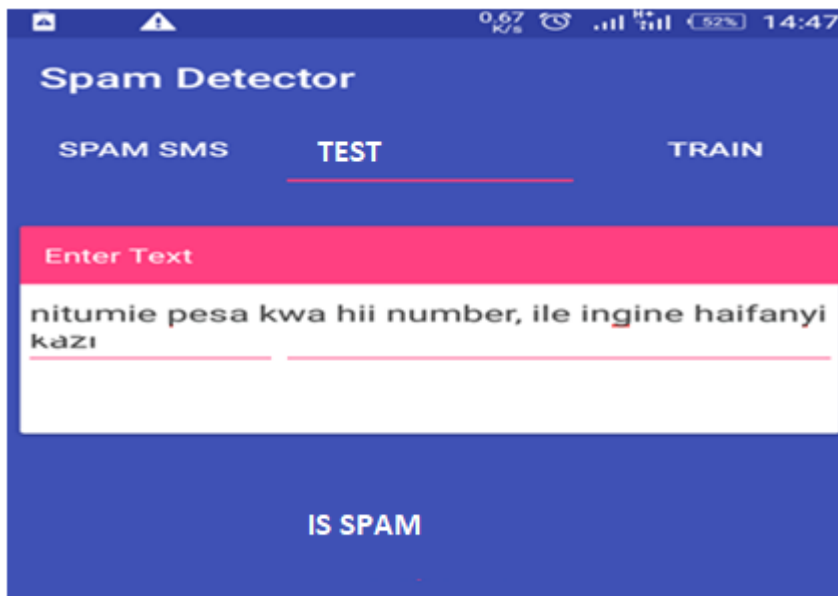


Figure 24: Testing module for a spam SMS

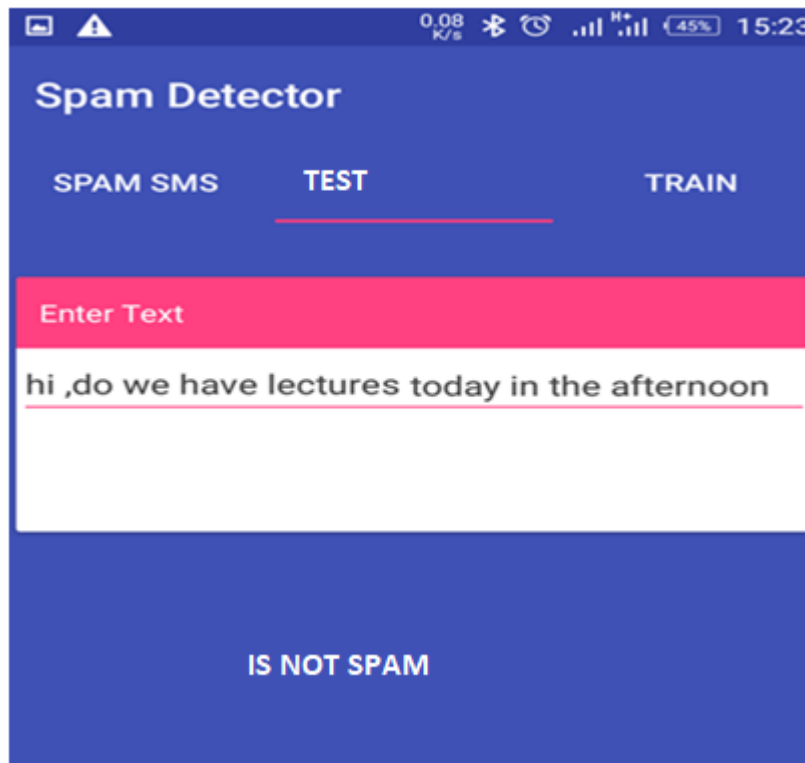


Figure 25: Testing module for non-spam SMS

4.6.1 Software development tools

The language used for implementation is Java using JDK version 8 for android devices with SQLite Database. Android is an open-source, Linux-based operating system for mobile devices such as smartphones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies. There are two major steps here; the pre-processing and the classification step that includes training module. In the training module the initial step involves loading the training data. The pre-processing step it involves creating a feature vector template and then building the feature vectors for the training data as per the methods described in chapter three. The first part of this consists of preprocessing of the incoming messages as per the techniques aforementioned, and then performing a feature selection over all the tokens. The top n features with the highest feature engineering score will

be selected as valid features for the model, n in this case is the number of features required as per the algorithm proposed.

Another important part of pre-processing is matching the tokens from a message to a valid feature in the feature template, for example when a token from a message matches a valid feature, the feature count will be increased /incremented by one, and starting from zero if there are no tokens matching that particular feature. After all the tokens have been checked a feature vector for the current message is constructed; this is an iterative process that involves all the messages for training. Lastly is to receive the classified feature vectors, which are built from the training data as per the ensemble methods.

4.6.2 Coding modules

The Top-level coding was developed using Java programming language. The coding developed in this thesis work consists of four modules (see appendix I for more codes) viz., main module, word stopping, stemming/lemmatization, Tokenization, TF/IDF and the vector space modeling followed the steps below:-

4.6.2.1 Main module

The main module consists of the following steps

Step 1 : Reading each word from each message.

Step 2 : If the scanned word is a language stop word then remove the stop word. The following is a code extract for Swahili language.

```
public void rereadStopWords(final InputStream inputStream, final Charset encoding) {  
  
    try {
```

```

final BufferedReader in = new BufferedReader(new InputStreamReader(inputStream,
        encoding));
try {
    String line;
    while ((line = in.readLine()) != null) {
        line = line.replaceAll("\\.|.*", "").trim();
        if (line.length() == 0) {
            continue;
        }
        for (final String w : line.split("\\s+")) {
            stopwords.add(w.toLowerCase(Locale.SWAHILI));
        }
    }
} finally {
    in.close();
}
} catch (final IOException e) {
    throw new RuntimeException(e);
} } }

```

Step 3 : Perform stemming/lemmatization based on the rules of the stemming algorithm.

```

public static void main(String[] args) {
    char[] w = new char[];
    Stemmer s = new Stemmer();

```

```

for (int i = 0; i < args.length; i++)
    try {
        FileInputStream in = new FileInputStream(args[i]);
        try {
            while (true) {
                int ch = in.read();
                if (Character.isLetter((char) ch)) {
                    int j = 0;
                    while (true) {
                        ch = Character.toLowerCase((char) ch);
                        w[j] = (char) ch;
                        if (j < 500) j++;
                        ch = in.read();
                        if (!Character.isLetter((char) ch)) {
                            /* to test add(char ch) */
                            for (int c = 0; c < j; c++) s.add(w[c]);
                            /* or, to test add(char[] w, int j) */
                            /* s.add(w, j); */
                            s.stem();
                            {
                                String u;
                                /* and now, to test toString() : */
                                u = s.toString();
                                /* to test getResultBuffer(), getResultLength() : */

```

```

        /* u = new String(s.getResultBuffer(), 0, s.getResultLength()); */
        System.out.print(u);
    }
    break;
}
}
}
}
if (ch < 0) break;
System.out.print((char) ch);
}
} catch (IOException e) {
    System.out.println("error reading " + args[i]);
    break;
}
} catch (FileNotFoundException e) {
    System.out.println("file " + args[i] + " not found");
    break;
}
}
}
}
}

```

Step 4 : Build the vocabulary and calculate the TF –IDF

```

import java.util.Arrays;

import java.util.List;

```

```

/**
 * @author Andrew Kipkebut
 */

public class TFIDF{

    /**
     * @param doc list of strings
     * @param term String represents a term
     * @return term frequency of term in document
     */

    public double tf(List<String> doc, String term) {

        double result = 0;

        for (String word : doc) {

            if (term.equalsIgnoreCase(word))

                result++;

        }

        return result / doc.size();

    }

    /**
     * @param docs list of list of strings represents the dataset
     * @param term String represents a term
     * @return the inverse term frequency of term in document
     */

```

```

public double idf(List<List<String>> docs, String term) {

    double n = 0;

    for (List<String> doc : docs) {

        for (String word : doc) {

            if (term.equalsIgnoreCase(word)) {

                n++;

                break;

            }

        }

    }

    return Math.log(docs.size() / n);

}

/**
 * @param doc a text document
 * @param term
 * @return the TF-IDF of term
 */

public double tfIdf(List<String> doc, List<List<String>> docs, String term) {

    return tf(doc, term) * idf(docs, term);

}

public static void main(String[] args) {

```

```

List<String> doc1 = Arrays.asList("WIN", "MPESA", "PAY", "MPESA", "FREE",
"WIN");

List<List<String>> documents = Arrays.asList(doc1);

TFIDFCalculator calculator = new TFIDF();

double tfidf = calculator.tfIdf(doc1, documents, "MPESA");

System.out.println("TF-IDF (MPESA) = " + tfidf);

} }

```

Step 5 : Cluster the message using K-means clustering algorithm

Step 6 : Translate the message

Step 7: Classify the test document using Ensemble hybrid model

4.6.2.2 Stopping module

The stopping module consists of the following steps

Step 1 : Check if the word in the main module is present in the stop list file

Step 2 : If present, then remove the word.

Step 3 : Else do not remove.

Step 4 : Check if the data is a number or any special symbol

Step 5 : If so, remove that word.

4.6.2.3 Stemming module

The stemming module consists of the following steps

Step 1 : If the word is not stopped, then check if a root word exists for that word by various rules provided by the algorithm.

Step 2 : If a root word exists, then replace all the occurrences of that word with the root word.

4.6.2.4 Vector space model module

The vector space module consist of the following steps

Step 1 : Check if the word is already present in the vocabulary list.

Step 2 : If not, insert this word into a new node and update the document number and frequency in the corresponding node.

Step 3 : If the word is already present, and if it is appearing for the first time in the document, then create a new node with the document number and it's corresponding frequency.

Step 4 : Else if the word is appearing again in the same document then increment the frequency.

Step 5 : Calculate the inverse document frequency (Idf) for each term(word) by the formula $Idf = \log (N/dft)$, where N is the total number of documents and dft is the number of documents that the term has occurred in.

Step 6: Calculate the Tf - idf of each word in each document by the formula

$Tf - idf = Frequency * idf.$

4.6.3 Prototype module testing

Testing is a very important process in any design and development of a software. It uncovers all the bugs generated by the software to make the application a successful product. In this

thesis the prototype testing was done as per the modules using JUnit a module framework for Java based applications. It is an automation framework for module testing as well as user interface. It contains annotations such as @Test, @Before, @After etc. This process is an important phase in software development lifecycle since it serves as the “Quality Gate” for the android application, the test summary report is an important deliverable which is prepared at the end of a Testing project. Further several metrics were used to help understand the test execution results, the status of test cases, defects among others. Defect Summary-Severity wise; Defect Distribution-Function/Module wise; Defect rejection ratio (DRR) and Defect leakage ratio(DLR) were also included as part of the software test report that including the use of Charts/Graphs for better visual representation.

Table 30: Test cases plan vs executed report

Tests cases planned	Test cases executed	Test cases passed	Test cases failed
25	20	18	2

The tests planned and tests executed report on table 30 allowed the researcher to optimally track the testing progress as per the test cases plan, number of executions, number of passed and failed test, the test plans were executed successfully and from this confirmed the usefulness of the machine learning model developed as per the objectives in chapter one.

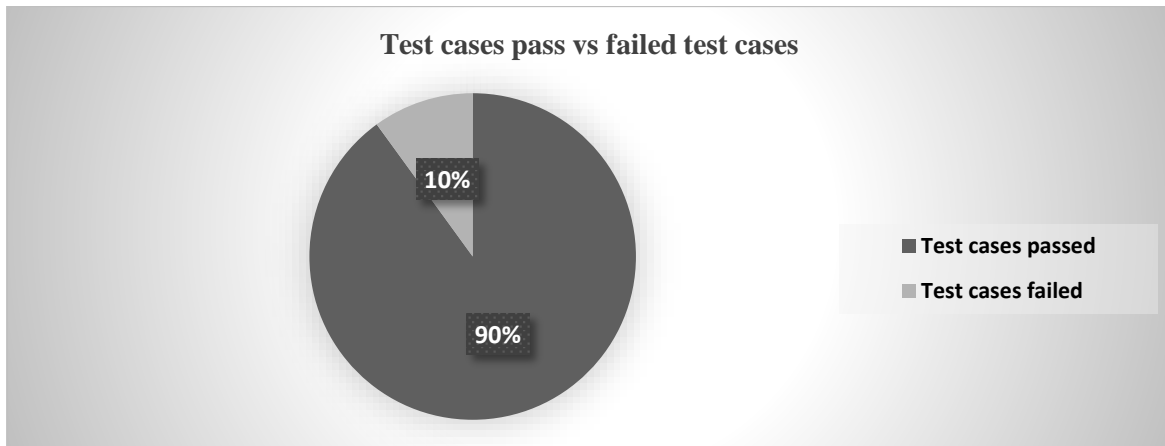


Figure 26: Test Cases vs. failed test cases

In general there were a total of 25 test cases plan as per the KLOC of the application, 20 test case were executed successfully, 18 (90%) of them passed the test and 2 (10%) of them failed the test. The 10 % in the failed test cases was due to server and network issues, and also from unresponsive scripts as illustrated in figure 26, these were resolved by modifying the ActivityTestRule and then re-executing the failed test cases again.

Table 31: No of defects identified by status and severity

	Critical	Major	Medium	Cosmetic	Total
Closed	8	4	6	0	18
Open	0	0	0	3	3
				Total	21

In table 31 and figure 27 the defects opened (New, Assigned, Reopened, and Blocked) vs. closed report (Resolved, Closed, and canceled) displays the number of defects that were opened compared with the number of Defects that were closed. This report helps to determine the rate at which defects are being opened compared with the rate at which defects are being closed. In

general there were 8 (44%) closed critical, 4 (22%) major, 3 (16.6%) medium and zero cosmetic defects .All Critical defects were closed since they represented those features that are most important to the system to function e.g. the training module. The cosmetic test case that represented 100 % of the open defect remained open since it only represented the aesthetic value of the project rather than the functionality.

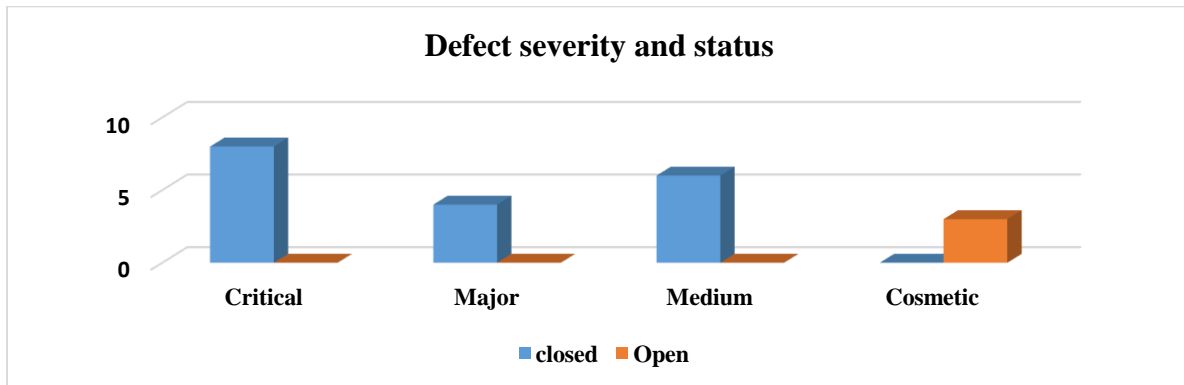


Figure 27: Bar chart representing defect severity

Table 32: Module defects distribution

	Main module	Pre-processing module	Feature selection module	Training module	Testing Module	Total
Critical	3	2	2	1	0	8
Major	0	1	1	1	1	4
Medium	1	2	1	1	1	6
Cosmetic	0	0	1	0	2	3
Total	4	5	5	3	4	21

In general the module defect for the main module registered a total of 4 defects (21%) , preprocessing module 5 defects (23%) , feature selection module 5 defects (23%) , training module 3 defects (13%) and testing module 4 defects (21%) of the total defects for critical , major , medium and cosmetic as shown in figure 28 . The module defect distribution report is important since it displays the number of defects by status, helping the researcher track the progress of defects for the prototype and also in identifying and prioritizing the defects.

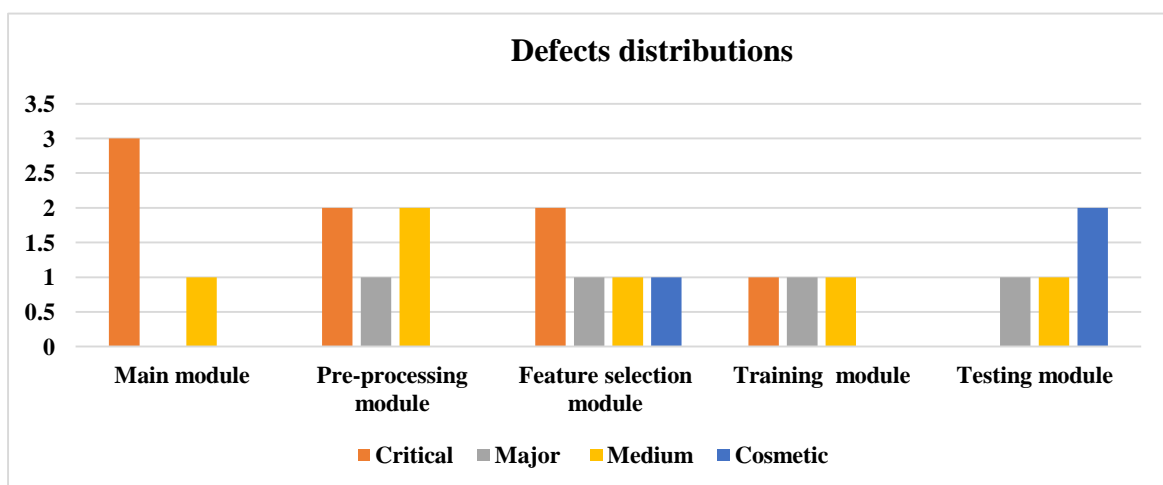


Figure 28: Defects distributions

After identifying the defects the next step was fixing the defects as per priority , once a defect has been resolved and verified, the defect status is changed to closed ,another important metric is to measure and evaluate the quality of a test execution - Defect rejection ratio (DRR) $=(\text{number of defects rejected}/\text{total number of defects raised}) \times 100$ and defect leakage ratio(DLR) $=(\text{number of defects missed } / \text{total defects of the application}) \times 100$ DRR was recorded as 0.0952 (9.52%) and DLR of 0.1423 (14.23%). The smaller value of DRR and DLR is, the better quality of test execution done (Mesquita et al.,(2016).

4.7 Chapter summary

In this chapter a new proposed ensemble hybrid model is presented that is based on machine learning algorithms, It starts with building the prototype using the enhanced preprocessing techniques i.e. Removal of noise using stop words, tokenization, TF/IDF, lemmatization and stemming techniques. Further dimensional reduction and feature selection methods are proposed that provide statistical correlation between attributes and the class. The enhanced pre-processing and feature selection methods was used to develop an ensemble hybrid stacked model based on the selected classifiers. The experiments conducted on the ensemble method produced a better classification performance as compared to other existing model presented in chapter three. This improvement was achieved because of the preprocessing techniques and the enhanced feature selection methods adopted. The prototype implementation and testing conducted on android device also proved that the generalized model is usable and can handle diverse text messages through multi-lingual processing.

CHAPTER FIVE

SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

5.1 Summary

The study was aimed at enhancing performance in short message service spam detection using ensemble hybrid machine learning techniques, new methods are proposed to accomplish the philosophy of providing a complete and optimal ensemble hybrid method. The specific contributions are:

5.1.1 To evaluate the optimal text preprocessing techniques for SMS spam

Often messages have a lot of abbreviations, slang and idioms that may affect the filters accuracy, spam filters may have their performance seriously impacted when employed to classify this kind of messages. The main objective of message preprocessing is preparing raw messages from a database for automatic analysis .to achieve this objective an enhanced ensemble preprocessing techniques were adopted that it includes String To Word vectorization, stop word, tokenization and stemming. This methods significantly made the data much easier for automated analysis using WEKA, this same data can also be handle by other tools such as Sklearn python libraries among others. The multi-lingual stop word created from external file was able to remove words that were meaningless and did not contribute to spam detection both for English, Swahili and slang language. To demonstrate the effect of pre-processing, this was tested with several classifier such as NB, J48, IBk, SVM, ANN, not only did this methods improved accuracy but also reduced the processing time.

5.1.2 To enhance the features engineering methods for ensemble learning

Message texting is personal and dynamic, on the other hand features can be large for SMS. In this study Features set was selected from correlational content based feature selection methods. This methods select features that are highly correlated with a given class (spam or

not spam) and are uncorrelated with each other. Most redundant features are removed because of high correlation with the residual feature set. It involves deleting irrelevant and less important features. To achieve this objective, the main idea was to use multiple diverse feature selectors and aggregate their outputs to achieve an optimal ensemble model. In SMS spam detection, little can be achieved if there are few features to represent the underlying data objects, and also the quality of the results of those algorithms largely depends on the quality of the available features, in other words all selected features (words in this case) are important to spam classification. In this study filter based feature engineering techniques were considered because their advantage is mentioned in chapter three. The feature selection methods aggregated in this study included Information gain (IG), Chi-square (CS), Mutual information (MI), and symmetric uncertainty (SU) and Correlation Feature selection (CFS). These methods were tested on the data set and the results indicated that the proposed feature selection technique has less features, less training time, more accuracy, precision and less errors from outliers, especially when paired with the preprocessing techniques as per the conceptual framework presented in chapter three. This ensemble feature selection method was tested with six classifiers NB, J48, IBk, SVM, MNB and ANN. Another important observation is that the True positive improved and false positive were reduced significantly when compared with the same classifiers without the feature selection.

5.1.3 To Enhance class correlation using ensemble hybrid of classifiers

As mentioned earlier ensemble learning in its primary definition is a machine learning style or a paradigm where diverse learners are trained to solve a given problem. In contrast to individual machine learning methods that try to solve a problem through learning one hypothesis from training data, ensemble methods usually use a set of hypotheses and then merge them to solve a given problem. The underlying principle of ensemble learning is an acknowledgment that in

real-world situations, each model has limitations and is prone to errors. Given that each single model will have these 'limitations' the aim of ensemble learning is to manage their strengths and weaknesses, leading to the best possible decision being taken overall. Several theoretical studies and empirical results have shown that the accuracy of an ensemble can meaningfully exceed that of a single model. Ensemble methods work best when the predictors are as independent of one another as possible. One way to get diverse classifiers is to train those using very different algorithms. This increases the chance that they will make very different types of errors, improving the ensembles moreover the performance of the ensemble can only improve upon that of the best base classifier if the ensemble has a sufficient pool of accurate and diverse classifiers, and so successful selection methods must balance these two requirements (accuracy and diversity).

In this study three ensemble techniques were considered bagging, boosting and stacking in order to improve classification precision. Several classifiers were used in this purpose using 10 fold cross validation. Generally the application of stacking outperformed bagging and boosting. In the study the stacking process was broken down into two stages namely: Generating the base-level classifier and Training a meta-level classifier to combine the outputs of the base-level classifier. It was clear that combining classifiers using stacking algorithm was observed to give a better accuracy compared to bagging and boosting since it provided more diversity of classifiers by recording high true positive and low false positive. A stacked model of Naïve bayes, Artificial Neural Network and Support vector machine recorded the most optimal solution with highest precision of 98.3% and a low false positive of 0.064 %. This model was adopted in this study and was used to implement the prototype, moreover it was compared with other studies, and it showed better results in terms of TP and FP.

5.1.4 To Cluster short messages service (SMS) as spam

To achieve this objective three clustering algorithms were considered, K means, hierarchical and cobweb. The findings shows that hierarchical algorithm took 1 second , k means 2.8 and cobweb 11 seconds , the prediction results shows that cobweb algorithm had only one cluster of 785 . Hierarchical though it took less amount of time also recorded bias since most items were categorized with same cluster. K-Means algorithm is the best suited to cluster the 785 spam messages.

5.1.5 To design and test the prototype on android platform

To achieve this objective the Top-level coding was developed using Java programming language. The coding developed in this work consisted of four modules main module, word stopping, stemming/lemmatization, Tokenization and Feature engineering. Testing is a very important process in any design and development of a software. It uncovers all the bugs generated by the software to make the application a successful product. In this study the prototype testing was done in five different stages, unit testing, module testing, integration testing, system testing and acceptance testing.

5.2 Conclusion

This study has proposed and developed a client side ensemble hybrid sms spam detection system which uses enhance ensemble preprocessing and feature selection techniques by adopting a stacking model. The study has also reviewed related literature on recent works .The idea of an ensemble method is due to changing tact in sms fraud and also to provide diversity while improving accuracy and reducing processing times.

SMS spam filtering is an important issue in mobile security and machine learning. The researcher further concludes that current mobile phones are able to filter SMS spam

automatically using the proposed model without depending on another computer or server for support or a large amount of data in advance. This independent and generalized filtering model obtained reasonable accuracy, improved TP, low FP, low storage consumption, and has acceptable processing times.

Based on the analysis of the tests done in this thesis it can be concluded that:-

1. The methods used in this research are equally well for SMS classification with average accuracy of over 98%.
2. The performance Quality of the ensemble model is based on datasets used in training, therefore it is important to conduct a thorough data preprocessing.
3. The model has shown highest precision in Sms Spam detection. By looking at the words that are present within the message, the classifier was able to correctly classify the message as either Spam or Not.
4. Using a portable model such as this mobile users can detect Spam messages using their phones therefore reducing fraud.
5. The proposed model being a generalized one can also be implemented using other frameworks such as Scikit learn and python programming.

From this study the researcher concludes that the main contribution of the study are:

The proposal of a unique client side ensemble hybrid preprocessing, feature engineering and prototype design and implementation for multi- lingual SMS spam classification and detection .The model, through the experiments conducted has shown tremendous results when compared

with recent studies. This model was evaluated using renowned metrics such as accuracy, True positives, false positives, precision, and recall, F-measure, MCC, ROC, PRC, MAE and RMSE.

5.3 Recommendations

- i. To speed up the training and testing, the researcher recommends thorough preprocessing of data. especially when handling large data set is highly recommended
- ii. An additional Profiling module may be added to profile frequent sms Spam phone numbers used, to help detect these social engineering act.
- iii. Client side detection may not be enough. It would be interesting to have both client and server side detection mechanism from the service providers. This will further help in reducing the damages of Sms spam from both ends.
- iv. For future research direction, this study recommends a creation of more general multi-lingual and standard research dataset that is publicly available for use by researchers

5.4 Areas of further research

This study focused on SMS spam detection using ensemble machine learning algorithm. However there are almost infinite text classification ways to detect SMS Spam that needs to be researched on, this is not limited to hybrid models combinations. Further the following elements of a message classification need to be further looked at:-

5.4.1 Data set

In this study the accuracy of the model can be improved with considering large data set and restrict the algorithm model to ignore normal dictionary words and instead use frequently used spam words in any language including deep Slang language (ghetto language) and local vernacular languages (Njemanze, 2012).

5.4.2 Data set imbalance

A classification data set with skewed class proportions is called imbalanced. Classes that make up a large proportion of the data set are called majority classes. Those that make up a smaller proportion are minority classes. The only thing that was implemented to account for the imbalance was an adjusted performance metric. It could be interesting to see newer methods to handle imbalance data especially those that enhances performance.

5.4.3 Toleration for class misclassification

Classifying a spam message as not spam is dangerous mistake than classifying a ham message as spam, therefore in classification, wrong class predictions are not as wrong as others. It would be interesting to have a way of determining which misclassifications can be tolerated and which ones cannot. This will definitely improve the model performance even further.

5.4.4 Dimension reduction

Further study on dimensional reduction techniques such as principal component analysis linear discriminant analysis and generalized discriminant analysis with performance feature transformation can be further studied to verify if the model's performance can be improved further not only in terms of accuracy but also other evaluation metrics.

REFERENCES

- Almomani, A., Gupta, B. B., Wan, T. C., Altaher, A., & Manickam, S. (2013). Phishing dynamic evolving neural fuzzy framework for online detection zero-day phishing email. *arXiv preprint arXiv:1302.0629*.
- Arif, M. H., Li, J., Iqbal, M., & Liu, K. (2018). Sentiment analysis and spam detection in short informal text using learning classifier systems. *Soft Computing*, 22(21), 7281-7291.
- Almeida T. A., Almeida J, &Yamakami, A. (2013). Spam filtering: How the dimensionality reduction affects the accuracy of Naive Bayes classifiers. *Journal of Internet Services and Applications*, 1(3), 183–200.
- Alpaydin, Ethem (2020). *Introduction to Machine Learning*. MIT Press. p. 9. ISBN 978-0-262-01243
- Anshul Goyal, A., & Mehta, R. (2012). Performance comparison of Naïve Bayes and J48 classification algorithms. *International Journal of Applied Engineering Research*, 7(11), 1-5.
- Auq M., bin Abdullah, M. F. A, Kang, K., & Choi, D.-J. (2010). A Survey of Preventing, Blocking and Filtering Short Message Services (SMS) Spam. *In Proceedings of International Conference on Computer and Electrical Engineering* (pp. 462{466). IEEE volume 1.
- Aziz, R., Verma, C., & Srivastava, N. (2016). A fuzzy based feature selection from independent component subspace for machine learning classification of microarray data. *Genomics data*, 8, 4-15.

- Bach, C., & Gunnarsson, J. (2010). *Extraction of trends in SMS text*. Department of Computer Science, Faculty of Engineering, LTH, Lund University.
- Banu, M. N., & Banu, S. M. (2013). A comprehensive study of phishing attacks. *International Journal of Computer Science and Information Technologies*, 4(6), 783-786.
- Basnet, R., Mukkamala, S., & Sung, A. H. (2008). Detection of phishing attacks: A machine learning approach. In *Soft computing applications in industry* (pp. 373-383). Springer, Berlin, Heidelberg.
- Basnet, R. B., Sung, A. H., & Liu, Q. (2012, June). Feature selection for improved phishing detection. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (pp. 252-261). Springer, Berlin, Heidelberg.
- Bahassine, S., Madani, A., Al-Sarem, M., & Kissi, M. (2020). Feature selection using an improved Chi-square for Arabic text classification. *Journal of King Saud University-Computer and Information Sciences*, 32(2), 225-231.
- Boughorbel, S., Jarray, F., & El-Anbari, M. (2017). Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. *PloS one*, 12(6), e0177678.
- Bach, C., & Gunnarsson, J. (2010). *Extraction of trends in SMS text*. Department of Computer Science, Faculty of Engineering, LTH, Lund University.
- Berrar, D. (2019). Cross-Validation.
- Bickel, B., Nichols, J., Haspelmath, M., Dryer, M. S., Gil, D., & Comrie, B. (2005). *Fusion of selected inflectional formatives*.
- Bijuraj, L. V. (2013). Clustering and its Applications. In *Proceedings of National Conference on New Horizons in IT-NCNHIT* (Vol. 1, pp. 169-172).

- Bikel, D., & Zitouni, I. (2013). *Multilingual Natural Language Processing Applications*. Edited by IBM Press Pearson Upper Saddle River, ibmpressbooks.com ISBN-13, 978-0..
- Bottazzi, G., Casalicchio, E., Cingolani, D., Marturana, F., & Piu, M. (2015, October). MP-shield: A framework for phishing detection in mobile devices. In *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing* (pp. 1977-1983). IEEE.
- Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7), 1145-1159.
- Brown, G., & Kuncheva, L. I. (2010, April). “good” and “bad” diversity in majority vote ensembles. In *International workshop on multiple classifier systems* (pp. 124-133). Springer, Berlin, Heidelberg.
- Brownlee, J. (2013). A tour of machine learning algorithms. *Machine Learning Mastery*, 25.
- Brownlee, J. (2017). *Introduction to time series forecasting with python: how to prepare data and develop models to predict the future*. Machine Learning Mastery.
- Cambria, E., Poria, S., & Gelbukh, A. (2017). IP National, Sentiment Analysis Is a Big Suitcase. *IEEE Intell. Syst*, 74-80.
- Cohen, S. B., & Smith, N. A. (2007, June). Joint morphological and syntactic disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)* (pp. 208-217).
- Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1), 16-28.

- Chang, M., Do, Q., & Roth, D. (2007). Multilingual dependency parsing: A pipeline approach. *Amsterdam studies in the theory and history of linguistic science series* 4, 292, 55.
- Chao, G., Luo, Y., & Ding, W. (2019). Recent advances in supervised dimension reduction: A survey. *Machine learning and knowledge extraction*, 1(1), 341-358.
- Chen, Y. Y., Lin, Y. H., Kung, C. C., Chung, M. H., & Yen, I. (2019). Design and implementation of cloud analytics-assisted smart power meters considering advanced artificial intelligence as edge analytics in demand-side management for smart homes. *Sensors*, 19(9), 2047.
- Cheng, J., & Greiner, R. (2013). Comparing Bayesian network classifiers. *arXiv preprint arXiv:1301.6684*.
- Choudhary, N., & Jain, A. K. (2017, March). Towards filtering of SMS spam messages using machine learning based technique. In *International Conference on Advanced Informatics for Computing Research* (pp. 18-30). Springer, Singapore.
- Cai, J., Tang, Y., & Hu, R. (2008, July). Spam filter for short messages using winnow. In *2008 International Conference on Advanced Language Processing and Web Information Technology* (pp. 454-459). IEEE.
- Cai, L., & Hofmann, T. (2003, July). Text categorization by boosting automatically extracted concepts. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (pp. 182-189).
- Cai, J., Luo, J., Wang, S., & Yang, S. (2018). Feature selection in machine learning: A new perspective. *Neurocomputing*, 300, 70-79.
- Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1), 16-28.

- Chiroma, H., Shuib, N. L. M., Muaz, S. A., Abubakar, A. I., Ila, L. B., & Maitama, J. Z. (2015). A review of the applications of bio-inspired flower pollination algorithm. *Procedia Computer Science*, 62, 435-441.
- Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press.
- Cortes , Robles-Velasco, A., Cortés, P., Muñuzuri, J., & Onieva, L. (2020). Prediction of pipe failures in water supply networks using logistic regression and support vector classification. *Reliability Engineering & System Safety*, 196, 106754.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms*. MIT press.
- Chorgha, S. P., & Shekokar, N. (2016, August). A survey on anti-phishing techniques in mobile phones. In *2016 International Conference on Inventive Computation Technologies (ICICT)* (Vol. 2, pp. 1-5). IEEE.
- Claesen, M., & De Moor, B. (2015). Hyperparameter search in machine learning. *arXiv preprint arXiv:1502.02127*.
- Delany, S. J., Buckley, M., & Greene, D. (2012). SMS spam filtering: Methods and data. *Expert Systems with Applications*, 39(10), 9899-9908.
- Deng, W.-W., & Peng, H. (2006). Research on a Naive Bayesian Based Short Message Filtering System. In *Proceedings of the International Conference on Machine Learning and Cybernetics* (pp. 1233–1237).IEEE.
- Dietterich, T., Domingos, P., Mitchell, T., Page, D., & Shavlik, J. (2016). Learning Bayesian Networks (part 3). Terms via iOS and Android Based Devices. *International Journal of Interactive Mobile Technologies*, 11(3).

- Doak, J. (1992). An evaluation of feature selection methods and their application to computer security. *Technical Report CSE-92-18*.
- Dunham, M. H. (2006). *Data mining: Introductory and advanced topics*. Pearson Education India.
- Dwivedi, S. K., & Arya, C. (2016, March). Automatic text classification in information retrieval: A survey. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies* (pp. 1-6).
- ElKah, A., & Zeroual, I. (2021). The effects of pre-processing techniques on Arabic text classification. *International Journal*, 10(1).
- Emigh, A. (2006). Online identity theft: Phishing technology, chokepoints and countermeasures. *Identity Theft Technology Council*.
- Elman, J. L. (2014). Finding structure in time. *Cognitive science* 14(2):179-211.
- Fisher, Douglas H. (1987). Improving inference through conceptual clustering. *Proceedings of the 1987 AAAI Conferences*. AAAI Conference. Seattle Washington. pp. 461–465.
- Foozy, C.F.M. & Ahmad, Rabiah & Abdollah, Mohd (2014). A framework for SMS spam and phishing detection in Malay language: A case study. *International Review on Computers and Software*. 9. 1248-1255.
- Foozy, CikFeresa Mohd, Rabiah Ahmad, and Mohd FaizalAbdollah (2013). Phishing detection taxonomy for mobile device. *International Journal of Computer Science Issues* (IJCSI) 10.1 338-344.
- Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780), 1612.

- Friendly, M. (2008). A brief history of data visualization. *Handbook of data visualization*, 15-56.
- Gajjala, R., & Tetteh, D. (2014). Relax, you've got M-PESA: Leisure as empowerment. *Information Technologies & International Development*, 10(3), pp-31.
- Gao, W., & Sebastiani, F. (2016). From classification to quantification in tweet sentiment analysis. *Social Network Analysis and Mining*, 6(1), 19.
- Giveki, D., Salimi, H., Bahmanyar, G., & Khademian, Y. (2012). Automatic detection of diabetes diagnosis using feature weighted support vector machines based on mutual information and modified cuckoo search. *arXiv preprint arXiv:1201.2173*.
- Ghareb, A. S., Bakar, A. A., & Hamdan, A. R. (2016). Hybrid feature selection based on enhanced genetic algorithm for text categorization. *Expert Systems with Applications*, 49, 31-47.
- Goel, D., & Jain, A. K. (2017). Smishing-Classifer: A Novel Framework for Detection of Smishing Attack in Mobile Environment. In *International Conference on Next Generation Computing Technologies* (pp. 502-512). Springer, Singapore.
- Gómez Hidalgo, J. M., Bringas, G. C., Sáenz, E. P., & García, F. C. (2006, October). Content based SMS spam filtering. In *Proceedings of the 2006 ACM symposium on Document engineering* (pp. 107-114).
- Goyal, A., & Mehta, R. (2012). Performance comparison of Naïve Bayes and J48 classification algorithms. *International Journal of Applied Engineering Research*, b7(11), 1-5.
- Gravetter, F. J., & Wallnau, L. B. (2016). edition 10. *Statistics for the behavioral sciences*. Boston. Cengage Learning.

- Gromov, M. L., Prokopenko, S. A., Shabaldina, N. V., & Laputenko, A. V. (2019, June). Model Based JUnit Testing. In *2019 20th International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM)* (pp. 139-142). IEEE.
- Gupta, M., Bakliwal, A., Agarwal, S., & Mehndiratta, P. (2018, August). A comparative study of spam SMS detection using machine learning classifiers. In *2018 Eleventh International Conference on Contemporary Computing (IC3)* (pp. 1-7). IEEE.
- Gupta, V., Mehta, A., Goel, A., Dixit, U., & Pandey, A. C. (2019). Spam detection using ensemble learning. In *Harmony search and nature inspired optimization algorithms* (pp. 661-668). Springer, Singapore.
- Hagos, T. (2018). Android studio. In *Learn Android Studio 3* (pp. 5-17). Apress, Berkeley, CA.
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, *11*(1), 10-18.
- Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. (2009). Unsupervised learning. In *The elements of statistical learning*, pp. 485-585. Springer, New York, NY.
- Hazimeh, H., & Zhai, C. (2015, September). Axiomatic analysis of smoothing methods in language models for pseudo-relevance feedback. In *Proceedings of the 2015 international conference on the theory of information retrieval* (pp. 141-150).
- Holden, R. R. (2010). Face validity. *The corsini encyclopedia of psychology*, 1-2.
- Hamsapriya, T., & Renuka, M. D. K. (2010). Email classification for spam detection using word stemming. *International journal of computer applications*, *1*(5), 45-47.

- Hamon, J. (2013). *Optimisation combinatoire pour la sélection de variables en régression en grande dimension: Application en génétique animale* (Doctoral dissertation, Université des Sciences et Technologie de Lille-Lille I).
- Hidalgo, J. M. G., Almeida, T. A., & Yamakami, A. (2012, December). On the validity of a new SMS spam collection. In *2012 11th International Conference on Machine Learning and Applications* (Vol. 2, pp. 240-245). IEEE.
- Hingmire, S., Chougule, S., Palshikar, G. K., & Chakraborti, S. (2013, July). Document classification by topic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 877-880).
- Jain, A. K., & Gupta, B. B. (2018). Rule-based framework for detection of smishing messages in mobile environment. *Procedia Computer Science*, *125*, 617-623.
- Jain, V., & Sharma, A. (2013). The consumers preferred operating system: Android or iOS. *International Journal of Business Management and Research (IJBMR)*, *3*(4), 29-40.
- Joe, I., & Shim, H. (2010, December). An SMS spam filtering system using support vector machine. In *International Conference on Future Generation Information Technology* (pp. 577-584). Springer, Berlin, Heidelberg.
- Jianqiang, Z., & Xiaolin, G. (2017). Comparison research on text pre-processing methods on twitter sentiment analysis. *IEEE Access*, *5*, 2870-2879.
- Joo, J. W., Moon, S. Y., Singh, S., & Park, J. H. (2017). S-Detector: an enhanced security model for detecting Smishing attack for mobile computing. *Telecommunication Systems*, *66*(1), 29-38.

- Joshi, N., & Srivastava, S. (2014). Improving classification accuracy using ensemble learning technique (using different decision trees). *International journal of computer science and mobile computing*, 3(5), 727-732.
- Junaid, M. B., & Farooq, M. (2011, July). Using evolutionary learning classifiers to do MobileSpam (SMS) filtering. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation* (pp. 1795-1802).
- Jurafsky, D., & Manning, C. (2012). Natural language processing. *Instructor*, 212(998), 3482.
- Juanchaiyaphum, J., Arch-int, N., Arch-int, S., & Saiyod, S. (2015). A novel lightweight hybrid intrusion detection method using a combination of data mining techniques. *International Journal of Security and Its Applications*, 9(4), 91-106.
- Laorden, C., Ugarte-Pedrero, X., Santos, I., Sanz, B., & Bringas, P. G. (2011, September). Enhancing scalability in anomaly-based email spam filtering. In *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference* (pp. 13-22).
- Lee, C., & Lee, G. (2006). Information gain and divergence-based feature selection for machine learning-based text categorization. *Information processing & management*, 42(1), 155-165.
- Lee, D. G., & Rim, H. C. (2004, February). Towards language-independent sentence boundary detection. In *International Conference on Intelligent Text Processing and Computational Linguistics* (pp. 142-145). Springer, Berlin, Heidelberg.
- Liu, J. Y., Zhao, Y. H., Zhang, Z. X., Wang, Y. H., Yuan, X. M., Hu, L., & Dong, Z. J. (2012). Spam short messages detection via mining social networks. *Journal of Computer Science and Technology*, 27(3), 506-514.

- Lota, L. N., & Hossain, B. M. (2017). A systematic literature review on sms spam detection techniques. *International Journal of Information Technology and Computer Science*, 9(7), 42-50.
- Kannan, S., Gurusamy, V., Vijayarani, S., Ilamathi, J., Nithya, M., Kannan, S., & Gurusamy, V. (2014). Preprocessing techniques for text mining. *International Journal of Computer Science & Communication Networks*, 5(1), 7-16.
- Kannan, R., Woo, H., Aggarwal, C. C., & Park, H. (2017, June). Outlier detection for text data. In *Proceedings of the 2017 siam international conference on data mining* (pp. 489-497). Society for Industrial and Applied Mathematics.
- Karegowda, A. G., Manjunath, A. S., & Jayaram, M. A. (2010). Comparative study of attribute selection using gain ratio and correlation based feature selection. *International Journal of Information Technology and Knowledge Management*, 2(2), 271-277.
- Kang, J.-Y., Yoon, J., Kim, Y (2013). Phishing/Pharming Examples and Countermeasure Analysis. *The Korean Institute of Information Scientists and Engineers* 12(2), 171–180.
- Karami, A., & Zhou, L. (2014). Improving static SMS spam detection by using new content-based features.
- Kim Y. (2015). Convolutional neural networks for sentence classification arXiv preprint *arXiv:1408.5882*.
- Kim, S. E., Jo, J. T., & Choi, S. H. (2015). SMS spam filtering using keyword frequency ratio. *International Journal of Security and Its Applications*, 9(1), 329-336.

- Kumar, V., & Minz, S. (2014). Feature selection: a literature review. *SmartCR*, 4(3), 211-229.
- Laudon, K. C., & Traver, C. G. (2013). E-commerce. Pearson.
- Lee, J., & Kim, D. W. (2013). Feature selection for multi-label classification using multivariate mutual information. *Pattern Recognition Letters*, 34(3), 349-357.
- Litan, A. (2004). Phishing attack victims likely targets for identity theft.
- Manandhar, P., & Aung, Z. (2014). Towards practical anomaly-based intrusion detection by outlier mining on TCP packets. In *International Conference on Database and Expert Systems Applications* (pp. 164-173). Springer, Cham.
- Mahesh, B. (2020). Machine Learning Algorithms-A Review. *International Journal of Science and Research (IJSR).[Internet]*, 9, 381-386.
- Mahmoud, T. M., & Mahfouz, A. M. (2012). SMS spam filtering technique based on artificial immune system. *International Journal of Computer Science Issues (IJCSI)*, 9(2), 589.
- Martin, H. (2017). The indeterminacy of word segmentation and the nature of morphology and syntax. *Folia linguistica*, 51(s1000), 31-80.
- Mathew, K., & Issac, B. (2011, December). Intelligent spam classification for mobile text message. In *Proceedings of 2011 International Conference on Computer Science and Network Technology* (Vol. 1, pp. 101-105). IEEE.
- Mesquita, D. P., Rocha, L. S., Gomes, J. P. P., & Neto, A. R. R. (2016). Classification with reject option for software defect prediction. *Applied Soft Computing*, 49, 1085-1093.
- Meistrell, M. L., & Spackman, K. A. (1989). Evaluation of neural network performance by ROC analysis: Examples from the biotechnology domain. *Computer Methods and Programs in Biomedicine*, 32, 73-80.

- Milian, M. (2009). Why text messages are limited to 160 characters. *Los Angeles Times*.
- Mirzaei, N., Malek, S., Păsăreanu, C. S., Esfahani, N., & Mahmood, R. (2012). Testing android apps through symbolic execution. *ACM SIGSOFT Software Engineering Notes*, 37(6), 1-5.
- Mishra, S., & Soni, D. (2020). Smishing Detector: A security model to detect smishing through SMS content analysis and URL behavior analysis. *Future Generation Computer Systems*, 108, 803-815
- McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochemia medica: Biochemia medica*, 22(3), 276-282.
- Mohaddes Deylami, H., Muniyandi, R. C., Ardekani, I., & Sarrafzadeh, H. (2016). Taxonomy of malware detection techniques.
- Mohammed, M., & Omar, N. (2020). Question classification based on Bloom's taxonomy cognitive domain using modified TF-IDF and word2vec. *PloS one*, 15(3), e0230442.
- Moran, K., Bernal-Cárdenas, C., Curcio, M., Bonett, R., & Poshyvanyk, D. (2018). Machine learning-based prototyping of graphical user interfaces for mobile apps. *IEEE Transactions on Software Engineering*, 46(2), 196-221.
- Morgado, I. C., & Paiva, A. C. (2019). The iMPAcT tool for Android testing. *Proceedings of the ACM on Human-Computer Interaction*, 3(EICS), 1-23.
- Moral, C., de Antonio, A., Imbert, R., & Ramírez, J. (2014). A survey of stemming algorithms in information retrieval. *Information Research: An International Electronic Journal*, 19(1), n1.

- Nayak, A. S., Kanive, A. P., Chandavekar, N., & Balasubramani, R. (2016). Survey on pre-processing techniques for text mining. *International Journal of Engineering and Computer Science*, 5(6), 16875-16879.
- Naseem, U., Razzak, I., & Eklund, P. W. (2020). A survey of pre-processing techniques to improve short-text quality: a case study on hate speech detection on twitter. *Multimedia Tools and Applications*, 1-28.
- Nielsen, F. (2016). Hierarchical clustering. In *Introduction to HPC with MPI for Data Science* (pp. 195-211). Springer, Cham.
- Nofal, A. A. D. M., & Bani-Ahmad, S. (2010). Classification based on association-rule mining techniques: a general survey and empirical comparative evaluation. *Ubiquitous Computing and Communication (UBICC) Journal*, 5(3).
- Noureldien, N. A., & Yousif, I. M. (2016). Accuracy of machine learning algorithms in detecting DoS attacks types. *Science and Technology*, 6(4), 89-92.
- Novac, O. C., Novac, M., Gordan, C., Berczes, T., & Bujdosó, G. (2017). Comparative study of Google Android, Apple iOS and Microsoft Windows phone mobile operating systems. In *2017 14th International Conference on Engineering of Modern Electric Systems (EMES)* (pp. 154-159). IEEE.
- Nuruzzaman, M. T., Lee, C., & Choi, D. (2011, August). Independent and personal SMS spam filtering. In *2011 IEEE 11th International Conference on Computer and Information Technology* (pp. 429-435). IEEE.

- Njemanze, Q. U. (2012). The SMS Style of Communication: Implications of Language Usage Among Nigerian University Students' Communication. *Journal of Communication, 3(1), 17-23.*
- Onashoga, A. S., Abayomi-Alli, O. O., Sodiya, A. S., & Ojo, D. A. (2015). An Adaptive and Collaborative Server-Side SMS Spam Filtering Scheme Using Artificial Immune System. *Information Security Journal: A Global Perspective, 24(4-6), 133-145.*
- Okediran, O. O., Arulogun, O. T., Ganiyu, R. A., & Oyeleye, C. A. (2014). Mobile operating systems and application development platforms: A survey. *International Journal of Advanced Networking and Applications, 6(1), 2195.*
- Onan, A., Korukoğlu, S., & Bulut, H. (2016). Ensemble of keyword extraction methods and classifiers in text classification. *Expert Systems with Applications, 57, 232-247.*
- Othman, N. F., & Din, W. I. S. W. (2019). Youtube spam detection framework using naive bayes and logistic regression. *Indonesian Journal of Electrical Engineering and Computer Science, 14(3), 1508-1517.*
- Onik, A. R., Haq, N. F., Alam, L., & Mamun, T. I. (2015). An analytical comparison on filter feature extraction method in data mining using J48 classifier. *International Journal of Computer Applications, 124(13).*
- Ortega, J. H. J. C., Lagman, A. C., Natividad, L. R. Q., Bantug, E. T., Resurreccion, M. R., & Manalo, L. O. (2020). Analysis of Performance of Classification Algorithms in Mushroom Poisonous Detection using Confusion Matrix Analysis. *International Journal, 9(1.3).*
- Pannu, P., & Tomar, Y. A. (2010). *ICT4D information communication technology for development.* IK International Pvt Ltd.

- Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500), 2323-2326.
- Richerzhagen, B., Stingl, D., Ruckert, J., & Steinmetz, R. (2015, August). Simonstrator: Simulation and prototyping platform for distributed mobile applications. In *The 8th EAI International Conference on Simulation Tools and Techniques (ACM SIMUTOOLS 2015)* (pp. 99-108).
- Raschka, S. (2014). Naive bayes and text classification i-introduction and theory. *arXiv preprint arXiv:1410.5329*.
- Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the 20th international conference on machine learning (icml-03)* (pp. 616-623).
- Rokach, L., & Maimon, O. Z. (2007). *Data mining with decision trees: theory and applications* (Vol. 69). World scientific.
- Saghapour, E., Kermani, S., & Sehhati, M. (2017). A novel feature ranking method for prediction of cancer stages using proteomics data. *PLoS One*, 12(9), e0184203.
- Saritas, M. M., & Yasar, A. (2019). Performance analysis of ANN and Naive Bayes classification algorithm for data classification. *International Journal of Intelligent Systems and Applications in Engineering*, 7(2), 88-91.
- Sharma, N., Bajpai, A., & Litoriya, M. R. (2012). Comparison the various clustering algorithms of weka tools. *facilities*, 4(7), 78-80.
- Silva, R. M., Alberto, T. C., Almeida, T. A., & Yamakami, A. (2017). Towards filtering undesired short text messages using an online learning approach with semantic indexing. *Expert Systems with Applications*, 83, 314-325.

- Sivic, J., & Zisserman, A. (2008). Efficient visual search of videos cast as text retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 31(4), 591-606.
- Singh, J., & Gupta, V. (2019). A novel unsupervised corpus-based stemming technique using lexicon and corpus statistics. *Knowledge-Based Systems*, 180, 147-162.
- Shafi'I, M. A., Abd Latiff, M. S., Chiroma, H., Osho, O., Abdul-Salaam, G., Abubakar, A. I., & Herawan, T. (2017). A review on mobile SMS spam filtering techniques. *IEEE Access*, 5, 15650-15666.
- Serrano, J. M. B., Palancar, J. H., & Cumplido, R. (2014, November). The evaluation of ordered features for sms spam filtering. In *Iberoamerican Congress on Pattern Recognition* (pp. 383-390). Springer, Cham.
- Shams, R., & Mercer, R. E. (2013, December). Classifying spam emails using text and readability features. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*(pp. 657-666). *IEEE*.
- Sharma, A., & Dey, S. (2013). A boosted SVM based sentiment analysis approach for online opinionated text. In *Proceedings of the 2013 research in adaptive and convergent systems* (pp. 28-34).
- Sonowal, G., & Kuppusamy, K. S. (2018). SmiDCA: An anti-smishing model with machine learning approach. *The Computer Journal*, 61(8), 1143-1157.
- Sohn, D. N., Lee, J. T., & Rim, H. C. (2009, August). The contribution of stylistic information to content-based mobile spam filtering. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers* (pp. 321-324).
- Sumathi, S., & Sivanandam, S. N. (2006). Introduction to data mining and its applications (Vol. 29). Springer.

- Sable, S., & Kalavadekar, P. N. (2016). SMS Classification Based on Naïve Bayes Classifier and Semi-supervised Learning. *International Journal of Innovations in Engineering Research and Technology*, 3(7).
- Saif, H., He, Y., & Alani, H. (2012). Alleviating data sparsity for twitter sentiment analysis. *CEUR Workshop Proceedings* (CEUR-WS. org).
- Stamatatos, E., Fakotakis, N., & Kokkinakis, G. (1999). Automatic extraction of rules for sentence boundary disambiguation. In *Proceedings of the Workshop on Machine Learning in Human Language Technology* (pp. 88-92).
- Sumathi, S., & Sivanandam, S. N. (2006). *Introduction to data mining and its applications* (Vol. 29). Springer.
- Tan, P. N., Steinbach, M., & Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.
- Tahir, S. B., Jalal, A., & Batool, M. (2020, February). Wearable sensors for activity analysis using SMO-based random forest over smart home and sports datasets. In *2020 3rd International Conference on Advancements in Computational Sciences (ICACS)* (pp. 1-6). IEEE.
- Tagg, C. (2009). *A corpus linguistics study of SMS text messaging* (Doctoral dissertation, University of Birmingham)
- Uysal, A. K., Gunal, S., Ergin, S., & Gunal, E. S. (2013). The impact of feature extraction and selection on SMS spam filtering. *Elektronika ir Elektrotechnika*, 19(5), 67-72.

- Vassilvitskii, S., & Arthur, D. (2006). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (pp. 1027-1035).
- Vijayarani, S., Ilamathi, M. J., & Nithya, M. (2015). Preprocessing techniques for text mining- an overview. *International Journal of Computer Science & Communication Networks*, 5(1), 7-16.
- Vipin, K., & Sonajharia, M. (2014). Feature selection: a literature review. *Smart Comput. Rev*, 4(3), 211-229.
- Vege, S. H. (2012). Ensemble of feature selection techniques for high dimensional data.
- Wang, A. H. (2010). Don't follow me - spam detection in twitter. In S. K. Katsikas, & P. Samarati (Eds.), *SECRYPT* (pp. 142-151).SciTePress.
- Wang, H., Fan, W., Yu, P. S., & Han, J. (2003, August). Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 226-235).
- Weinberger, K., Dasgupta, A., Langford, J., Smola, A., & Attenberg, J. (2009, June). Feature hashing for large scale multitask learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 1113-1120).
- Wang, C., Zhang, Y., Chen, X., Liu, Z., Shi, L., Chen, G., ... & Lu, W. (2010). A behavior-based SMS antispam system. *IBM Journal of Research and Development*, 54(6), 3-1.
- Wijaya, M. R., Saptono, R., & Douwes, A. (2016). The Effect of Best First and Spreadsubsample on Selection of a Feature Wrapper With Naive Bayes Classifier for The Classification of The Ratio Inpatients. *Scientific Journal of Informatics*, 3(2), 41-50.

- Wu, N., Wu, M., & Chen, S. (2008). Real-time monitoring and filtering system for mobile SMS. In *Proceedings of 3rd IEEE Conference on Industrial Electronics and Applications* (pp. 1319-1324).
- Xiang, Y., Chowdhury, M., & Ali, S. (2004). Filtering mobile spam by support vector machine. In N. Debnath (Ed.), *Proceedings of the Third International Conference on Computer Sciences, Software Engineering, Information Technology, E-Business and Applications* (pp. 1-4).
- Xing, E. P., Jordan, M. I., & Karp, R. M. (2001, June). Feature selection for high-dimensional genomic microarray data. In *Icml* (Vol. 1, pp. 601-608).
- Xu, S., Li, Y., & Wang, Z. (2017). Bayesian multinomial Naïve Bayes classifier to text classification. In *Advanced multimedia and ubiquitous engineering* (pp. 347-352). Springer, Singapore.
- Yadav, K., Kumaraguru, P., Goyal, A., Gupta, A., Naik, V. (2011) Smsassassin: crowdsourcing driven mobile-based system for SMS spam filtering. In: *12th Workshop on Mobile Computing Systems and Applications*, pp. 1-6. ACM, New York .
- Yan, G., Eidenbenz, S., & Galli, E. (2009, September). Sms-watchdog: Profiling social behaviors of sms users for anomaly detection. In *International Workshop on Recent Advances in Intrusion Detection* (pp. 202-223). Springer, Berlin, Heidelberg.
- Yang, Y., & Pedersen, J. O. (1997). A comparative study on feature selection for text categorization
- Yardi, S., Romero, D., & Schoenebeck, G. (2010). Detecting spam in a twitter network. *First monday*.
- Yoon, H. G., Kim, H., Kim, C. O., & Song, M. (2016). Opinion polarity detection in Twitter data combining shrinkage regression and topic modeling. *Journal of Informetrics*, 10(2), 634-644.

- Zainal, K., Sulaiman, N. F., & Jali, M. Z. (2015). An analysis of various algorithms for text spam classification and clustering using RapidMiner and Weka. *International Journal of Computer Science and Information Security*, 13(3), 66.
- Zein, S., Salleh, N., & Grundy, J. (2016). A systematic mapping study of mobile application testing techniques. *Journal of Systems and Software*, 117, 334-356.
- Zhou, Z. H. (2021). Ensemble learning. In *Machine Learning* (pp. 181-210). Springer, Singapore.
- Zou, W. Y., Socher, R., Cer, D., & Manning, C. D. (2013, October). Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1393-1398).
- Zheng, Y., Hatakka, M., Sahay, S., & Andersson, A. (2018). Conceptualizing development in information and communication technology for development (ICT4D).
- Zhou, Z. H. (2009). Ensemble Learning. *Encyclopedia of biometrics*, 1, 270-273.
- Zhang, J., & Wang, Y. (2012). A real-time automatic detection of phishing URLs. In: 2012 2nd International Conference on Computer Science and Network Technology (ICCSNT), IEEE (pp. 1212–1216).
- Zhang, X. D. (2020). Machine learning. In *A Matrix Algebra Approach to Artificial Intelligence* (pp. 223-440). Springer, Singapore.
- Zhang, H., Jiang, L., & Yu, L. (2021). Attribute and instance weighted naive Bayes. *Pattern Recognition*, 111, 107674.

APPENDICES

Appendix I: JAVA programming code extract

```
/**  
 * Classifier, this class does most of the classification using Naive Bayes  
 */  
  
package Weka.api;  
  
import weka.classifiers.classifier;  
  
import weka.classifiers.j48;  
  
import weka.classifiers.Bayes.NaiveBayes;  
  
import weka.classifiers.Meta.Stacking;  
  
import android.content.Context;  
  
import android.database.Cursor;  
  
import android.text.TextUtils;  
  
import java.util.Arrays;  
  
import java.util.HashSet;  
  
import java.util.Set;  
  
public class CombineModels  
  
    private static final int STRENGTH = 1;  
  
    private static final int HAM = 0;  
  
    private static final int SPAM = 1;  
  
    private static final double PrC = 0.5;  
  
    Context;  
  
    DBHelper mDb;
```

```

public Classifier(Context context){
    this.context = context;
    mDb = new DBHelper(context);
}

public boolean isSpam(String sms) {

    float [] categoryWordProbability = { 1, 1 };

    String [] words = splitWord(sms);

    int[] allWords = { mDb.allWordCount(HAM), mDb.allWordCount(SPAM)};

    for(int c=0;c<words.length;c++) {
        if (words[c].length() < 3)
            continue;

        for (int i=0;i<2;i++) {

            int allWordsInCategory = allWords[i];

            // Get occurrence of this word in category

            int wordInCategory = mDb.thisWordCount(words[c], i);

            // Get occurrence of this word both categories

            int n = mDb.wordCount(words[c]);

            // Corrected probability

            float probability = (float) wordInCategory/allWordsInCategory;

            categoryWordProbability[i] *= (float) (STRENGTH*PrC +
n*probability)/(STRENGTH + n);

        }
    }
}

```

```

    }

    return (categoryWordProbability[SPAM] > categoryWordProbability[HAM]) ?
        true : false;
    }

public Cursor getLikelySpamMessages(){
    return mDb.getLikelySpamMessages();
}

public void addSpamSMS(String from, String sms){
    mDb.addSpamSMS(from, sms);
}

public String[] splitWord(String sms) {
    // Split words
    String [] words = TextUtils.split(sms, "\\W+");

    // unique
    Set<String> temp = new HashSet<String>(Arrays.asList(words));
    String[] unique = temp.toArray(new String[temp.size()]);

    return unique;
}
}

/**

```

```
* Constants
```

```
*/
```

```
public class Constants {  
  
    public static final String [] CATEGORIES = {"ham_count", "spam_count"};  
  
    public static final String KEY_SPAM_C = "spam_count";  
  
    public static final String KEY_HAM_C = "ham_count";  
  
    public static final String KEY_WORD = "word";  
  
    public static final String KEY_ID = "id";  
  
    public static final String KEY_FROM = "sender_id";  
  
    public static final String KEY_BODY = "body";  
  
    public static final String KEY_DATE = "date";  
  
}
```

```
//WEKA JAVA code extract
```

```
import weka.classifiers.Evaluation;  
import weka.classifiers.bayes.NaiveBayes;  
import weka.classifiers.bayes.NaiveBayesMultinomial;  
import weka.classifiers.Neuralnets;  
import weka.classifiers.meta.FilteredClassifier;  
import weka.classifiers.Stacking  
import weka.classifiers.Evaluation;  
import weka.core.Instances;  
import weka.core.Instance;  
import weka.core.converters.ConverterUtils.DataSource;  
import weka.core.Attribute;  
import weka.core.DenseInstance;  
import weka.core.converters.ArffSaver;  
import weka.classifiers.meta.FilteredClassifier;  
import weka.filters.unsupervised.attribute.StringToWordVector;  
import java.io.File;  
import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
import java.util.List;
```

```

import java.util.ArrayList;
public class WekaClassifier {
private FilteredClassifier classifier;
private Instances trainData;
private Instances testData;
private ArrayList<Attribute> fvWekaAttributes;

WekaClassifier(){
classifier = new FilteredClassifier();
Attribute attribute_text = new Attribute("text", (List<String>) null);

// Declare the label attribute along with its values
ArrayList<String> classAttributeValues = new ArrayList<String>();
classAttributeValues.add("spam");
classAttributeValues.add("ham");
Attribute classAttribute = new Attribute("label", classAttributeValues);

// Declare the feature vector
fvWekaAttributes = new ArrayList<Attribute>();
fvWekaAttributes.add(classAttribute);
fvWekaAttributes.add(attribute_text);

}
public Instances load (String filename) throws IOException
{

// // Declare text attribute
// Attribute attribute_text = new Attribute("text", (List<String>) null);

// // Declare the label attribute along with its values
// ArrayList<String> classAttributeValues = new ArrayList<String>();
// classAttributeValues.add("spam");
// classAttributeValues.add("ham");
// Attribute classAttribute = new Attribute("label", classAttributeValues);

// // Declare the feature vector
// ArrayList<Attribute> fvWekaAttributes = new ArrayList<Attribute>();
// fvWekaAttributes.add(classAttribute);
// fvWekaAttributes.add(attribute_text);

/* Create an empty training set
name the relation "Rel". set initial capacity of 10*
*/
Instances dataset = new Instances("Rel", fvWekaAttributes, 10);

// Set class index

```

```

dataset.setClassIndex(0);

// read text file, parse data and add to instance
try(BufferedReader br = new BufferedReader(new FileReader(filename))) {
for(String line; (line = br.readLine()) != null; ) {
try{

// split at first occurrence of n no. of words
String parts[] = line.split("\\s+",2);

// basic validation
if (!parts[0].isEmpty() && !parts[1].isEmpty()){

DenseInstance row = new DenseInstance(2);

row.setValue(fvWekaAttributes.get(0), parts[0]);

row.setValue(fvWekaAttributes.get(1), parts[1]);
// add row to instances

dataset.add(row);}
//
}
catch (ArrayIndexOutOfBoundsException e){

        System.out.println("invalid row");
}

}

}
catch (IOException e){
e.printStackTrace();
}
return dataset;

}

public void prepare() throws Exception{
trainData = load("data/train.txt");
testData = load("data/test.txt");
}
public void transform(){

// create the filter and set the attribute to be transformed from text into a feature vector (the
last one)
StringToWordVector filter = new StringToWordVector();
filter.setAttributeIndices("last");

```



```

classifier.setFilter(filter);
classifier.setClassifier(new NaiveBayes());
    }
public void fit() throws Exception classifier.buildClassifier(trainData);
    }
public String classify(String text) throws Exception {

Instances newDataset = new Instances("testdata", fvWekaAttributes, 1);
newDataset.setClassIndex(0);

DenseInstance newInstance = new DenseInstance(2);
newInstance.setDataset(newDataset);

newInstance.setValue(fvWekaAttributes.get(1), text);

double pred = classifier.classifyInstance(newInstance);

System.out.println("===== Classified instance =====");
System.out.println("Class predicted: " + trainData.classAttribute().value((int) pred));
    return
trainData.classAttribute().value((int) pred);

// try {
// DenseInstance instance = new DenseInstance(2);
// instance.setValue(new Attribute("text", (List<String>) null), text);
// double pred = classifier.classifyInstance(instance);
// System.out.println("===== Classified instance =====");
// System.out.println("Class predicted: " + trainData.classAttribute().value((int) pred));
//     return
trainData.classAttribute().value((int) pred);
// }

// catch (Exception e) {
// System.out.println("Problem found when classifying the text");
// }
// return "";
    }
public String evaluate() throws Exception{
Evaluation eval = new Evaluation(testData);
eval.evaluateModel(classifier, testData);
System.out.println(eval.toSummaryString());
return eval.toSummaryString();
    }

public void saveArff(Instances dataset, String filename) throws IOException{
try
{
// initialize
ArffSaver arffSaverInstance = new ArffSaver();

```

```

arffSaverInstance.setInstances(dataset);
arffSaverInstance.setFile(new File(filename));
arffSaverInstance.writeBatch();
    }
catch (IOException e)
    e.printStackTrace();
    }
}

public static void main(String[] args) throws
Exception{

WekaClassifier wt = new WekaClassifier();
wt.prepare();
wt.transform();
wt.fit();
wt.evaluate();
wt.classify("Nitumie hii pesa kwa ile number , hii nyingine Haifanyi kazi");

// Instances trainData = wt.load("data/train.txt");
// System.out.println(trainData);

// // create the filter and set the attribute to be transformed from text into a feature vector (the
// last one)
// StringToWordVector filter = new StringToWordVector();
// filter.setAttributeIndices("last");

// FilteredClassifier classifier = new FilteredClassifier();
// classifier.setFilter(filter);
// classifier.setClassifier(new NaiveBayes());

// classifier.buildClassifier(trainData);
// System.out.println(WekaTransformer.classify("I said its okay. Sorry"));
/*
Now that we create and trained a classifier, let's test it. To do so, we need an evaluation
module (weka.classifiers.Evaluation) to which we feed our testing set
*/
// Instances testData =
wt.transform("data/test.txt");
// evaluation set
// Evaluation eval = new Evaluation(testData);
// eval.evaluateModel(classifier, testData);
//
System.out.println(eval.toSummaryString());

}}

```

Appendix II: Data set in (ARFF format)

@relation smsspam

@attribute class {spam,ham}

@attribute messages string

@data

spam,"You have won yourself ksh 100, 000 for being a loyal Equity bank customer. Click <http://equitybanknig-plc.com/> for more details."

ham,"Please call me . Thank you."

spam,"SAFARICOM NGURUMA IMBAMBE PROMOTIONS: Dear subscriber, Your phone number has WON Ksh100,000/= Call us NOW.. 0788XXXXXX/ ,Do Not send money."

spam,"Dear jumia shopper your purchase last month won ksh 10000 via jumia card go to www.jumia.co.ke to claim"

spam,"Hello ! I just got 500 credit and 10GB internet free from here:<http://winprize.net/kenya>"

spam," Dear Equitel Customer,Your equitel number has expired,SMS your details to 0788XXXXXX for upgrade"

spam,"We,ve been hired 2 shorten ur life ,4 Details mpesa 20,000 to be pardon ."

spam,"hi, earn real money online by working as a part time job easily ,earn up to 2500KSHs weekly .Call our customer care now!!!"

spam,"Have you seen the doctor about your healness."

spam,"Nguruma Ibambe! na safaricom nambari yako imezawadia 100,000 na EQUITY BANK NA M-KESHO kwa maelezo zaidi piga -729XXXXXX OR (0787XXXXXX)"

spam,"This is Major Omonge, forward the names of your nephew ad his Id number, few Army recruit chances remaining.."

ham,"I love you Baby gal."

ham,"I tried calling you , please call me back ",9,0,0,0,0,0,0,0,0

spam,"Hello,You have won yourself ksh 15,000/- plus a Techno phone of ksh 9,000/- for using MPESA Service.Go MPESA Menu.",19,0,0,0,1,0,0,0,1

spam,"THIS IS IT !karibu VUNA TATU where sh10 win SH3,500; 20 ni SH 7000.try now! Paybil 326326; accno weka 3luckynumbers; na Amt 10 au zaidi. Match &win every 30 min",30,0,0,1,1,0,0,0,0

spam,"Adidas is giving away 5000 free pair of shoes and 900 T shirts to celebrateits 93rd anniversary.Get your free shoes at :<http://adidas.com>",22,0,0,0,0,0,0,0,0

spam,"Sister Jayne tafadhali mtoto ameumwa na nyoka na mahali niko sina credit tafadhali nitumie credit ya 50 nipigie mtu was BODABODA nimpeleke hosipitali halaka.",24,0,0,0,0,0,0,0,0

spam,"Earn real money online by working as a part time job easily ,earn up to 2500 dollars weekly guranteed join link at <http://oncemore.club/?id=101362>",23,0,1,0,0,0,0,0,0

spam,"hi,ulipata kazi? Kama bado sms job to 40224 for all vacancies or click <http://bit.ly/2ErvoB> to join chamas uchangiwe pesa ya bizna of fees.inform classmates",24,0,0,0,0,0,0,0,0

spam,"OLX is hiring -no experience required SMS sending job work from home work 3-5 hours daily on mobile and earn 500-600 kshs.",22,0,0,0,0,0,0,0,0

spam,"Dear KCB Mpesa customer ,loan are now available at 10%, Get 10,000, 20,000, 30,000 , Call 0788XXXXXX :KCB Bank",19,1,0,0,0,1,1,0,1

ham,"Dear customer your IUC number 2009450548 is due on 23/08/2018.Pay via MPESA paybill 463655,ensure that your decoder is on.",19,1,0,1,0,0,1,0,1

spam,"Nitumie hiyo pesa kwa hii number ile ingine haifanyi kazi",10,0,0,1,0,0,0,0,0

spam,"Habari , mtoto wako ameumwa na nyoka shuleni, Tafadhali tuma shilingi elfu tano tumpeleka district hospital ile asife.",18,0,0,0,0,0,0,0,0

ham,"Dear classmates. We have a class from 10-1 pm today",10,1,0,0,0,0,0,0,0

spam,"Lotto , umeshinda shilingi milioni tano!!, tupigie simu",8,0,0,0,0,0,0,0,0

ham,"Say hi to your Dad, loving Aunty",7,0,0,0,0,0,0,0,0

ham,"I love you honey, call me when you get a chance",11,0,0,0,0,0,0,0,0

spam,"Your account has been hacked , please send us your MPESA PIN to unlock .SAFARICOM MPESA",16,0,0,0,0,0,0,0,1

spam,"send money to this number , the other one is not functional",12,0,1,1,0,0,0,0,0

ham,"We can meet tomorrow for the meeting, Ag Secretary",9,0,0,0,0,0,0,0,0

spam,"Dear customer soft loan available from Equity at 6%, 8%, call 0715XXXXXXX to get the best,No paying anything.",19,0,0,0,0,0,0,1,0,0

spam,"Send your contributions to mama ken burial through this number.",10,0,0,1,0,0,0,0,0

spam,"M-PESA BK12LY038 confirmed you have received Ksh2,350 from PETER SANG 254703275802 on 28/5/18 at 03:22 PM Sender M-PESA",18,0,0,0,0,0,0,0,0

ham,"May you have a blessed Sunday my brother.",8,0,0,0,0,0,0,0,0

spam,"we also buy and sale airtime and bonga points, Hurry while stock last!!!!!!",13,0,0,0,0,0,0,0,0

ham,"I love you hun",4,0,0,0,0,0,0,0,0

ham, "I will spoil you in bed as well :)",9,0,0,0,0,0,0,0,0

ham,"I'm going for bath will msg you next",8,0,0,0,0,0,0,0,0

spam,"Grab yourself the best safaricom internet bundles in town call 0715XXXXXXX for more details , Thank you.",17,0,0,0,0,0,0,0,0

spam,"This is the 3rd time we have tried to contact you, You have won yourself a Smart Tv courtesy of SamSang , Call Now.",24,0,0,0,1,0,0,0,0

ham,"Are you around town , we talk.",7,0,0,0,0,0,0,0,0

ham,"Please don't text me anymore. I have nothing else to say",11,0,0,0,0,0,0,0,0

spam,"Entry in 2 a wkly comp to win FA Cup final tkts 21st May 2016. Text FA to 87121 to receive entry question",23,0,0,0,1,0,0,0,0

spam,"WINNER!! As a valued network customer you have been selected to receivea 20,000 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.",26,0,0,0,1,0,1,0,0

ham,"Even my friend Jenny does not like to speak to me. They treat me like HIV patent.",17,0,0,0,0,0,0,0,0

spam,"Have yo had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030",31,0,0,0,0,0,0,0,0

ham,"I'm gonna be home soon and i don't want to talk about this stuff anymore tonight,I've cried enough today",21,0,0,0,0,0,0,0,0

spam,"SIX chances to win CASH! From 1000 to 20,000 Kshs txt ,CSH11 and send to 87575. Cost 10/day, 6days, 16+ Tand Cs apply Reply HL 4 info.",27,0,0,0,1,0,0,0,0

spam,"URGENT! You have won a 1 week FREE membership in our £100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net",23,0,0,0,1,0,0,0,0

ham,"I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise." ,27,0,0,0,1,0,0,0,0

ham,"You have been wonderful and a blessing at all times.",10,0,0,0,1,0,0,0,0

spam,"Russia 2018 SPAIN VS BRAZIL - dont miss the goals/team news. Txt ur national team to 87077 eg ENGLAND to 87077 .",22,0,0,0,0,0,0,0,0

ham,"Oh k...i'm watching here",4,0,0,0,0,0,0,0,0

ham,"Just forced myself to eat a slice. I'm really not hungry tho. This sucks. Mark is getting worried.",18,0,0,0,0,0,0,0,0

ham,"He got in at 2 and was v apologetic. n had fallen out and she was actin like spoilt child",20,0,0,0,0,0,0,0,0

spam,"Thanks for your subscription to SKIZA tune will be charged 500/month Please confirm by replying YES or NO. If you reply NO you will not be charged.",27,0,0,0,0,0,0,0,0

ham,"Oops, I'll let you know when my roommate's done",9,0,0,0,0,0,0,0,0

ham,"Hello! How's you and how did sato going? ",8,0,0,0,0,0,0,0,0

spam,"Mater run ,We tried to call you .your reply to our sms for a free nokia mobile + free camcorder. Please call now 0773XXXXXX - for delivery tomorrow",28,0,0,0,0,0,0,0,0

ham,"WHO ARE YOU SEEING?",4,0,0,0,0,0,0,0,0

ham,"Great! I hope you like your man well endowed. I am ",11,0,0,0,0,0,0,0,0

ham,"Didn't you get hep B immunisation in Kenya?",8,0,0,0,0,0,0,0,0

ham,"Fair enough, anything going on?",5,0,0,0,0,0,0,0,0

ham,"Yeah hopefully, if Andrew can't do it I could maybe ask around a a bit for a plot.",18,0,0,0,0,0,0,0,0

ham,"U don't know how stubborn I am. I didn't even want to go to the hospital.",16,0,0,0,0,0,0,0,0

ham,"What you think about me. First time you saw me in class.",12,0,0,0,0,0,0,0,0

ham,"Sorry, I'll call later in meeting.",6,0,0,0,0,0,0,0,0

ham,"Your gonna have to pick up a 100 hot dog at Naiva for yourself on your way home. I can't even move. Pain is killing me.",26,0,0,0,0,0,0,0,0

ham,"Ha ha ha good joke. Girls are situation seekers.",9,0,0,0,0,0,0,0,0

ham,"Sorry my roommates took forever, it ok if I come by now!!",12,0,0,0,0,0,0,0,0

spam,"As a valued customer, I am pleased to advise you that following recent review of your Mob No. you are awarded with a Ksh 1500 Bonus Prize, call 09066364589",29,0,0,0,1,0,1,0,0

ham,"Finished class where are you.",5,0,0,0,0,0,0,0,0

ham,"where are you?how did you performed?",6,0,0,0,0,0,0,0,0

ham,"U can call me now...",5,0,0,0,0,0,0,0,0

ham,"Thats cool. i am a gentleman and will treat you with dignity and respect.",14,0,0,0,0,0,0,0,0

ham,"Its not the same here. Still looking for a job. How much do Nurses earn there.",16,0,0,0,0,0,0,0,0

ham,"Sorry, I'll call later",4,0,0,0,0,0,0,0,0

ham,"Ok i am on the way to home" ,8,0,0,0,0,0,0,0,0

ham,"I call you later, don't have network",7,0,0,0,0,0,0,0,0

ham, "Yes I started to send requests to make it but pain came back so I'm back in bed",18,0,0,0,0,0,0,0,0

ham,"Ulinunua ile shamba eldoret",4,0,0,0,0,0,0,0,0

spam,"Please call our customer service representative on 0800 169 6031 between 10am-9pm as you have WON a guaranteed £1000 cash or £5000 prize!",23,0,0,0,1,0,1,0,0

ham,"Please don't text me anymore. I have nothing else to say.",11,0,0,0,0,0,0,0,0

ham,"I'm still looking for a car to buy. And have not gone 4the driving test yet.",16,0,0,0,0,0,0,0,0

spam,"Get our free sport pesa jackpot prediction and anlysis @
www.kadenge.com",11,0,0,0,0,0,0,0,0

spam,"You are a winner U have been specially selected 2 receive 100000 or a 4* holiday
(flights inc) speak to a live operator 2 claim 0719XXXXXX." ,26,0,0,0,1,0,0,0,0

spam,"URGENT! Your Mobile No. was awarded 2000 Bonus Caller Prize on 5/03/18 This is
our final try to contact U! Call from Landline 09064019788,24",0,0,0,0,0,0,0,0,0

spam,"Todays airtel numbers ending 548 are selected to receive a 20k award. If you have a
match please call 08712300220 quoting claim code 4041 standard rates app",27,0,0,0,0,0,0,0,0

ham,"Are you there in the room?." ,6,0,0,0,0,0,0,0,0

spam,"Todays Quiz Wkly Q! Win a top Sony DVD player if u know which country mugambe
is from ? Txt ansr to 82277." ,23,0,0,0,1,0,0,0,0

ham,"Sir, I am Waiting for your mail." ,7,0,0,0,0,0,0,0,0

spam,"FreeMsg Why haven't you replied to my text? I'm Randy, sexy, female and live local.
Luv to hear from u." ,20,0,0,0,0,0,0,0,0

ham,"Whats the Lecturer name who is taking the class for us",11,0,0,0,0,0,0,0,0

ham,"OK..i deleted my contact thats why?" ,6,0,0,0,0,0,0,0,0

spam,"call our customer service representative on FREEPHONE 0808 145 4742 between 9am-
11pm as you have WON a guaranteed £1000 cash or £5000 prize!" ,23,0,0,0,1,0,1,0,0

ham,"Are you unique enough? Find out from 30th August.
www.areyouunique.co.uk" ,10,0,0,0,0,0,0,0,0

ham,"Hi :)finally i completed the course:)" ,6,0,0,0,0,0,0,0,0

ham,"How are you doing? Hope you've settled in for the new school year. Just wishin you a
gr8 day",19,0,0,0,0,0,0,0,0

i)



e-ISSN: 2582-5208

International Research Journal of Modernization in Engineering Technology and Science

Volume:02/Issue:08/August-2020

Impact Factor- 5.354

www.irjmets.com

TOWARDS MULTILINGUAL FEATURE ENGINEERING FOR SMS SPAM DETECTION

Kipkebut Andrew*¹, Thiga Moses *², Okumu Elizabeth*³

*¹PG student, Computer science and I.T, Kabarak, Nakuru, Kenya .

*² Dr., Computer science and I.T, Kabarak, Nakuru, Kenya.

*³Dr., Computer science and I.T , Kabarak, Nakuru, Kenya.

ABSTRACT

Billions of money is lost by mobile phone users every day due to SMS spam, a social engineering skill attempting to obtain sensitive information such as passwords, Personal identification numbers and other private data by masquerading as a trustworthy entity through Short message service. The design of efficient feature engineering techniques is the key to reducing these financial losses. Most machine learning classifiers solutions today produce less accurate predictions and are inefficient due to dynamic nature of spamming. It is in this background that the study proposes an ensemble feature engineering techniques for sms spam, that can be used for multilingual natural language processing, data training, validation and testing of a model. The contributors of data include UCI database and local repositories that contain mixture English and Swahili messages. Machine learning and data mining experiments are conducted using WEKA tool and the results and discussions are presented in form of descriptive statistics. This novel approach recorded an overall satisfiable accuracy of 99%.

KEYWORDS: Algorithm, Detection, ensemble, Feature engineering, Machine learning, Sms.

Android Based Multi-Lingual SMS Spam Prototype Design and Development

KIPKEBUT ANDREW¹, THIGA MOSES², OKUMU ELIZABETH³

^{1, 2, 3} School of Science and Engineering, Kabarak University -Kenya

Abstract- Most spammers are constantly developing new sophisticated methods, rendering previous techniques obsolete. A thoughtful deficiency in most sms spam detection methods is lack of satisfying accuracy, reliability, low performance and comprehensibility especially when individual classifiers are used, these remains important aspects to be considered for an optimal model development. Sms spam detection using machine learning techniques is a new approach especially in ubiquitous computing devices such as mobile phones, moreover the design of short message spam detection techniques in a mobile platform is challenging task due to the non-stationary distribution of the data and the multi-lingual nature of text messages from users. It is in this background that the research proposes a multi-stage ensemble hybrid prototype sms spam detection model for a mobile environment using machine learning techniques. It involves enhanced use of pre-processing techniques, content-based feature engineering techniques, multilingual natural language processing, data training and testing. The effectiveness of the proposed prototype is empirically validated using ensemble classification methods that gave an overall classification accuracy of 98.2606%.

Indexed Terms— Algorithm, Detection, Ensemble Feature engineering, Machine learning, SMS.

types of mobile operating systems [2].According [3] Android's percentage share in the market is increasing at an alarming rate, Google android is rapidly taking its place in the eyes of today's youth and every person today wants affordable and the best operating system which Android guarantees to provide to its users. Dollah et al [4] research on mobile device ownership, the research indicated that 159 out of 225 respondents (70.4%) had Android based device for their mobile phones followed by others (19%), Apple iPhone (11.1%), and Windows Phone (2.2%). The least was Blackberry mobile phone with a percentage of (1.3%) only. The possible factors that led to the high ownership rate of Android based mobile phones may be attributed to the competitive price of these devices. However, in lieu of this finding, the simplicity, reliability and functionality may be best attributed to others, such as, Apple iPhone and windows Phone. A research conducted by [5] on the increasing market penetration of mobile devices, such as smartphones and tablets, poses additional challenges on the design of distributed systems. Due to the heterogeneous environment consisting of both, mobile and fixed devices, a multitude of effects on different scales need to be considered. Microscopic effects, such as an individual user's interaction with the device, as well as macroscopic effects, such as scalability with the number of users have an impact on the system's performance. The combined evaluation of micro- and

Appendix IV: Letter of Introduction



INSTITUTE OF POST GRADUATE STUDIES

Private Bag - 20157
KABARAK, KENYA
E-mail: directorpostgraduate@kabarak.ac.ke

Tel: 0203511275
Fax: 254-51-343012
www.kabarak.ac.ke

30th October, 2017

Ministry of Higher Education Science and Technology,
National Council for Science, Technology & Innovation,
P.O. Box 30623 - 00100,

Dear Sir/Madam,

RE: RESEARCH BY ANDREW KIPKEBUT- GDS/M/0206/1/16

The above named is a student at Kabarak University taking PHD Degree in IT Security. He is carrying out research entitled "*A Client side Smishing Detection framework using machine learning algorithms for naïve M-commerce users in Kenya.*"

The information obtained in the course of this research will be used for academic purposes only and will be treated with utmost confidentiality.

Please provide the necessary assistance.

Thank you.

Yours faithfully

A handwritten signature in black ink, appearing to be 'Dr. Betty J. Tikoko'.

Dr. Betty J. Tikoko
DIRECTOR - (POST-GRADUATE STUDIES)



Kabarak University Moral Code

As members of Kabarak University family, we purpose at all times and in all places, to set apart in one's heart, Jesus as Lord. (1 Peter 3:15)

Appendix V: Research permit

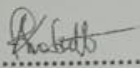
THIS IS TO CERTIFY THAT:
MR. ANDREW KIPROP KIPKEBUT
of **KABARAK UNIVERSITY, 0-20100**
Nakuru, has been permitted to conduct
research in *Baringo , Nairobi, Nakuru ,*
Uasin-Gishu Counties

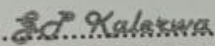
Permit No : **NACOSTI/P/18/91928/20830**
Date Of Issue : **21st February, 2018**
Fee Received : **Ksh 2000**

on the topic: **CLIENT SIDE SMISHING
DETECTION FRAMEWORK FOR NAIVE
M-COMMERCE USERS IN KENYA**



for the period ending:
20th February, 2019


.....
**Applicant's
Signature**

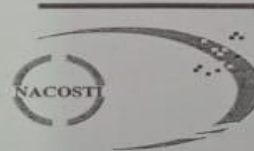
.....

**Director General
National Commission for Science,
Technology & Innovation**

CONDITIONS

1. This permit is valid for the proposed research,
for the specified period.
2. The Licensee shall retain all intellectual property
rights and any rights thereunder are
transferable.
3. The Licensee shall report to the Commission,
the Licensee shall submit a progress report.
4. The Licensee shall report to the County Director of
Health and County Governor in the area of
research before commencement of the research.
5. The Licensee shall, before commencement of the research,
obtain necessary permits, filming and collection of specimens
and to further permissions from relevant
agencies.
6. The Commission does not give authority to transfer
this permit to other persons.
7. The Licensee shall submit two (2) hard copies and
soft copy of their final report.
8. The Commission reserves the right to modify the
terms and conditions of this Licence including its cancellation
without prior notice.



REPUBLIC OF KENYA



National Commission for Science,
Technology and Innovation

**RESEARCH CLEARANCE
PERMIT**

Serial No.A 17592

CONDITIONS: see back page