CATEGORISATION OF SEXUAL REPRODUCTIVE HEALTH SHORT MESSAGES TEXTS INTO THEMATIC AREAS USING TEXT MINING

(A CASE STUDY FOR THE ZAMBIA U-REPORT)

BY

Tobias Makai

A Dissertation submitted to the University of Zambia in partial fulfilment of the requirements of the degree in Masters of Science in Computer Science.

THE UNIVERSITY OF ZAMBIA,

LUSAKA

2024

DECLARATION

I, the undersigned, declare that this has not previously been submitted in candidature for any degree. The dissertation is the result of my own work and investigations, except where otherwise stated. Other sources are acknowledged by given explicit references. A complete list of references is appended.

Signature: …….……

Date: …….

CERTIFICATE OF APPROVAL

This document by **Tobias Makai** is approved as fulfilling the requirements for the award of the degree of **Master of Science in Computer Science** of the University of Zambia.

Examiner's Signature: …… Date: ……

Dr. Mayumbo Nyirenda, HOD - Department of Computer Science, Supervisor

Examiner's Signature: …… Date: ……...…....

Examiner's Signature: …… Date: …….....…….

DEDICATION

This dissertation is dedicated to my amazing wife and our unborn children. Niza, you are an incredible partner, and your unwavering support and understanding have been crucial to getting me through this journey. You have always been there to push me when I needed it and to celebrate my wins with me. This, I could not have done without you.

To our future children, I hope this dissertation can serve as a model for you to aim much higher.

Thank you for letting me share this moment with you all.

ABSTRACT

In Zambia, the U-report platform was launched to improve young people's understanding of sexual reproductive health and access remote counseling services. This free texting program is run by the National HIV/AIDS/STI/TB Council supported by UNICEF. Millions of messages have been exchanged, but categorizing them manually to see what topics adolescents frequently ask about is challenging. This research explored using computers, to automatically classify these messages into different subject areas, making it faster to identify knowledge gaps in sexual health and related domains. The study investigated how U-report messages are currently categorized and built a system to automatically sort them into different topics. The new system was compared to the manual method currently used and it was found that the automated system is faster and more accurate, giving an accuracy score of above seventy percent. In the first stage of the research, administrators of the platform were interviewed to understand how they categorize messages. This helped identify the different categories they use. Then, a classification model was trained to sort messages into these categories using machine learning. This model has the potential to significantly improve how quickly and accurately messages are categorized on the Zambia U-report SMS system.

Keywords—Feature Extraction, Machine Learning, Natural Language Processing (NLP), Text Classification, Text Mining, Text Extraction.

ACKNOWLEDGEMENT

I begin by expressing my deepest gratitude to God Almighty for renewing my strength and providing the necessary guidance and inspiration throughout my dissertation journey. I could not have reached this point without His unwavering grace and mercy.

I am profoundly thankful to my supervisor, Dr. Mayumbo Nyirenda for his guidance and support during the research journey have been instrumental. His knowledge and feedback were invaluable in shaping my work.

I wish to also convey my appreciation to my wife, Niza Sinkala Makai, for her support and understanding at every stage of this journey. She has been my pillar of strength, always encouraging me to keep going even when I felt like giving up. I am grateful for her patience and for enduring some of my late nights, which sometimes meant sleeping alone.

I am very grateful to the Zambia U-report team members who provided me with invaluable information and support to complete some crucial tasks. In particular, I am grateful to Grace Chansa, Allan Msimuko, Gift Chimpampwe, and Dee Machona for their contributions, which were instrumental in the success of my research.

Gratitude to Andre Lesa for his unwavering commitment to ensuring that I had all the necessary permissions, linkages, and information from the National HIV/AIDS/STI/TB Council (NAC). His support was essential in enabling me to conduct my research.

Finally, my sincere thanks to NAC management for allowing me to pursue this study. Without their cooperation and support, this research would not have been possible.

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF FIGURES

CHAPTER ONE:

INTRODUCTION TO THE RESEARCH

1.1 INTRODUCTION

Growing up in many African countries including Zambia, is as exciting as it is challenging. Transitioning from childhood to adulthood can be challenging for many adolescents, presenting a range of issues from psychosocial to physical. Among the psychosocial challenges faced are unprotected sex, peer pressure, early marriages, unwanted pregnancies, sexual abuse, unsafe abortions, and drug and alcohol abuse and high HIV prevalence rates, all exacerbated by inadequate Sexual Reproductive Health (SRH) knowledge. The transitional period between childhoods to adulthood shapes the behaviors of many young people with a direct bearing on their future. For this reason, it is imperative that young people are exposed to information that can help them make informed and responsible choices in life. Preparing young people for the challenges they will face as they grow is difficult, as many parents do not openly talk to their children about critical issues such as reproductive health and sexuality [1], [2], [3]. This situation has led to numerous adolescents turning to unreliable sources like social media and peers for information, due to limited or no access to counseling and youth-friendly resource centers, which are only available in certain locations across the country. The localities and limited operating hours for counselling and resource centers present a challenge of lack of timeous access to the much-needed factual sexual reproductive health information, counselling and guidance for young people. Evidence indicates young people being at a high risk of HIV exposure, making it crucial to engage with them in efforts to end AIDS [4].

In response to the aforementioned phenomenon in Zambia, the U-report platform was created by the National HIV/AIDS/STI/TB Council (NAC). This was done with support from UNICEF. This interactive Short Message Service (SMS) system is used to disseminate SRH information to young people subscribed to the platform, aiming to address the information gap. The objective of the system is to foster behavioral change by giving young subscribers access to essential knowledge on sexual reproductive health topics. The system allows people to ask about particular concerns by sending a text message on their mobile devices [5]. Additionally, the platform empowers a team of counselors to address messages that require personalized responses beyond what can be handled

automatically. This ensures that users receive accurate information to help them make informed decisions. Furthermore, the system provides capabilities to run targeted polls and campaigns through SMS.

In other words, Zambia U-report is a youth-led SMS-based HIV/AIDS response initiative platform accessible via the short code 878. Motivated by a comparable initiative by UNICEF Uganda, Zambia U-report was established to expedite HIV prevention efforts targeting adolescents and young adults, contributing to the goal of achieving an HIV-free generation in Zambia [5], [6], [7]. The effort, which has so far attracted over 200,000 young people, is filling the gap caused by the lack of timeous access to factual SRH information and counselling services for adolescents. Since its launch in 2012 and nationwide expansion in 2014, Zambia U-report has, over seven years, amassed a vast repository of textual data, comprising a total of 5,987,040 SMS text messages [8]. However, both NAC and UNICEF have underutilized this data for assessing impact and making decisions, primarily due to challenges in categorizing the messages. This data can be leveraged to identify the most frequently asked questions and emerging concerns related to sexual reproductive health, among other topics. It can serve as a knowledge bank for SRH, inform a counseling curriculum centered around these frequently asked questions, and help develop an intervention package for high-impact HIV services. The data is currently used to identity which key areas or problems affect the users. Being textual information, the pool of SMS text messages is unstructured and currently difficult to analyze.

This dissertation presents the automation of categorizing Zambia U-report text messages into key thematic areas for decision-making, utilizing text mining techniques.

1.2 PROBLEM STATEMENT

Since its launch, the Zambia U-report system has recorded more five million interactions between users and the platform, involving over 200,000 subscribers. Given humongous volume of SMS text messages, the National HIV/AIDS/STI/TB Council (NAC) faces challenges in effectively categorizing U-report data into relevant thematic areas to:

i. Track knowledge gaps among adolescents, and

ii. Find out and track emerging issues and figure out which topics beyond sexual reproductive health, subscribers would be interested to discuss.

Currently, the process of categorizing text messages on the platform is conducted manually requiring NAC officers to read through all messages one after another categorizing them into a chosen list of thematic areas. This process is **hectic** and **time-consuming** especially that it is the counsellors hired for the sole purpose of providing counselling services that are given the extra task of manually categorizing the text messages.

1.3 AIM

The aim of the research is to develop a prototype that classifies U-report users' SMS messages automatically using existing machine learning and text mining techniques to enhance analysis of the textual data.

1.4 RESEARCH QUESTIONS

To guide and direct the focus of this study, the following research questions are raised;

i. How is information currently categorized into key thematic areas on the Zambia U-report system?

ii. How can the manual process of categorizing U-report information into key thematic areas be automated?

iii. How does the performance and effectiveness of the automated system compare to the manual process?

1.5 OBJECTIVES

In order to realize the aim above, the following objectives will have to be met during the study;

i. To understand and identify the model currently used for categorizing information into key thematic areas on the Zambia U-report system.

ii. To develop a prototype that automates the categorization of information on the U-report platform into key thematic areas.

iii. To evaluate and compare the performance and effectiveness of the developed prototype with the existing manual system.

1.6 SIGNIFICANCE OF RESEARCH

With the objective of transforming the HIV response among Zambian adolescents through mobile technology [6], this study aims to enable the automatic classification of accumulated U-report textual data into important themes. This will significantly decrease the time required for U-report counselors to categorize these messages manually. The automation will enable decision makers to categorize data for analysis by simply running a classification model created as a result of this study. This will take away the extra task of categorizing data from counsellors so that they can focus only on providing psychosocial counselling which is their primary role.

The study will also reveal important information around how the current categorization process is done, detailing how long it takes counsellors to categorize the text messages from the moment an SMS text message is seen to when it is allocated a thematic area. This information will establish a solid background to the creation of a classification model. Learning the current model will also provide a basis for comparison between the created classifier and the current model culminating into an evidence-based solution to the aforementioned problem.

Ultimately, this research provides a proof of concept to improve the tracking and analysis of SMS texts received on the Zambia U-report system. It will provide a faster and more efficient method for analysis, enabling NAC and UNICEF officers to accurately identify and report on key knowledge gaps and emerging issues concerning HIV, Sexually Transmitted Infections (STIs), and other aspects of Sexual Reproductive Health among young people.

1.7 SCOPE OF RESEARCH

The solution was developed as an independent module that can be integrated into the existing system, specifically to categorize U-report SMS texts.

1.8 RESEARCH CONTRIBUTION

As evidenced by the extensive literature reviewed and documented in the next chapter, this research is the first attempt at looking into the automation of categorizing reproductive health text messages into key thematic areas through text mining.

This study enhances the body of knowledge by offering an automated text categorization model tailored for sexual reproductive health textual data. It utilizes machine learning methods and text mining techniques to address the challenges of the hectic and time-consuming manual categorization process.

The study also contributes to society by providing a model for automatic sexual reproductive health text messages' classification which NAC can use for their categorization needs, in their quest to track the most frequently discussed topics by adolescents through the U-report platform. The study adds value to NAC and UNICEF's U-report information analysis needs for decision making around sexual reproductive health knowledge gaps in Zambia. It significantly reduces the time required to categorize accumulated textual data on the U-report platform into key topics, thereby improving the analysis process.

1.9 ORGANIZATION OF THE THESIS

Chapter one introduces the aim of the research, describes the platform being used as a case study and details the problem statement, research objectives, justification and scope of the study.

Chapter two is Literature Review. The chapter provides a theoretical analysis of existing text mining techniques and review of related works.

Chapter three describes the methods used to understand the current categorization model and implement a prototype for an automated categorization process.

Chapter four presents the results of the study.

Chapter five provides a discussion and conclusions drawn from the results presented in chapter four.

1.10 SUMMARY

The chapter opens with an introduction to the research which provides background information describing the Zambia U-report platform, how and why it was created followed by the problem statement stating the area of concern to which the research proposes a solution. The aim of the study, its research questions, objectives, scope, significance or justification and contribution to the body of knowledge are then established.

The chapter establishes that the method of categorizing textual data on the U-report platform is currently manual whereby counsellors hired to provide counselling services are given an extra task of manually categorizing text messages, a process that is not only hectic but also time-consuming. Thus, it is established in the chapter that the aim of the study was to automate how SMS messages are categorized on the U-report platform into key themes. This will enhance analysis using existing data, alongside machine learning and text mining techniques.

Text mining, also referred to as Intelligent Text Analysis, Text Data Mining, or Knowledge-Discovery in Text (KDT), generally involves the process of deriving meaningful and insightful information from unstructured text [9]. U-report has a large collection of text-based messages. Textual data is unclear, unstructured, and challenging to manipulate [10]. Therefore, this study employs machine learning and text mining techniques to extract valuable information from a large collection of text messages. Machine learning is a field where algorithms are designed to learn from data, allowing them to perform better over time [11]. We employed supervised machine learning methods to construct a prototype for classifying SMS texts automatically. Text classification, a core component of text mining, has traditionally required the manual creation of classification systems through knowledge-engineering techniques. This typically involves the manual definition of logical rules that embody expert knowledge on categorizing documents into designated categories [12].

CHAPTER TWO:

LITERATURE REVIEW

2.1 INTRODUCTION

As outlined in the previous chapter, this research focuses on creating an automated system to classify incoming SMS messages from U-report subscribers in Zambia. This system will categorize the messages into key thematic areas to make analysis easier. This study uses Computer Science tools and methods to analyze the huge collection of U-report text messages collected over time. This will improve how the information is understood and used to make decisions based on data. Being a text classification problem, we looked at various areas of computer science that deal with automated text analysis. These include Artificial Intelligence (AI), Text Mining (TM), Machine Learning (ML), and Natural Language Processing (NLP). Our goal was to leverage these fields to automate tasks like information extraction, labeling, categorization, summarization, and topic tracing. By identifying patterns within the text data, we aim to improve decision-making processes.

This chapter dives into the theory behind the technologies we mentioned earlier. It explains their relevance and application to the problem at hand. The chapter further provides an elaborate review of approaches that can be adopted for our problem and finally an overview of similar problems and approaches used to solve them.

2.2 RELATED TECHNOLOGIES

2.2.1 ARTIFICIAL INTELLIGENCE

AI or Artificial Intelligence refers to the creation and analysis of computational agents. Computational agents are essentially programs designed to act intelligently [13]. This intelligence manifests in various ways, such as learning from experience, adapting to new situations, and performing tasks similar to humans [14]. For this study, the human-like tasks involve processing a large volume of text messages, analyzing each one sequentially, and assigning an appropriate category to each message. Artificial Intelligence is a strategic technology that aligns with related to the solution profile for the problem at hand and encompasses multiple techniques. Specific

Artificial Intelligence techniques commonly utilized for automating text categorization including Text Mining, Machine Learning, and Natural Language Processing are discussed in detail below.

2.2.2 TEXT MINING

Fueled by the ever-growing amount of text data from social media, medical records, news sources and other internet sources, Text Mining (TM) has emerged as a powerful tool for uncovering valuable insights from this unstructured information [15]. TM is a technique that allows us to analyze and extract meaning from this vast amount of raw text data [9]. Currently, the internet is the primary source of textual documents, and the volume of textual information accessible continues to grow. It is estimated that around 80% of an organization's information is stored in unstructured formats, such as reports, emails, opinions, and news. This indicates that about 90% of global data exists in unstructured formats [16]. This effort is fueled by a massive dataset of unstructured texts on the Zambia U-report system, which includes over five million messages [8]. Due to its unstructured form, managing and processing textual data for decision-making is difficult. Thus, text mining aims to facilitate automatic extraction of information from a collection of texts. Alternative terms like Intelligent Text Analysis, Text Data Mining, or Knowledge-Discovery in Text (KDT) are also used to describe the process of text mining [9]. Figure 1 illustrates the overall process involved in text mining.

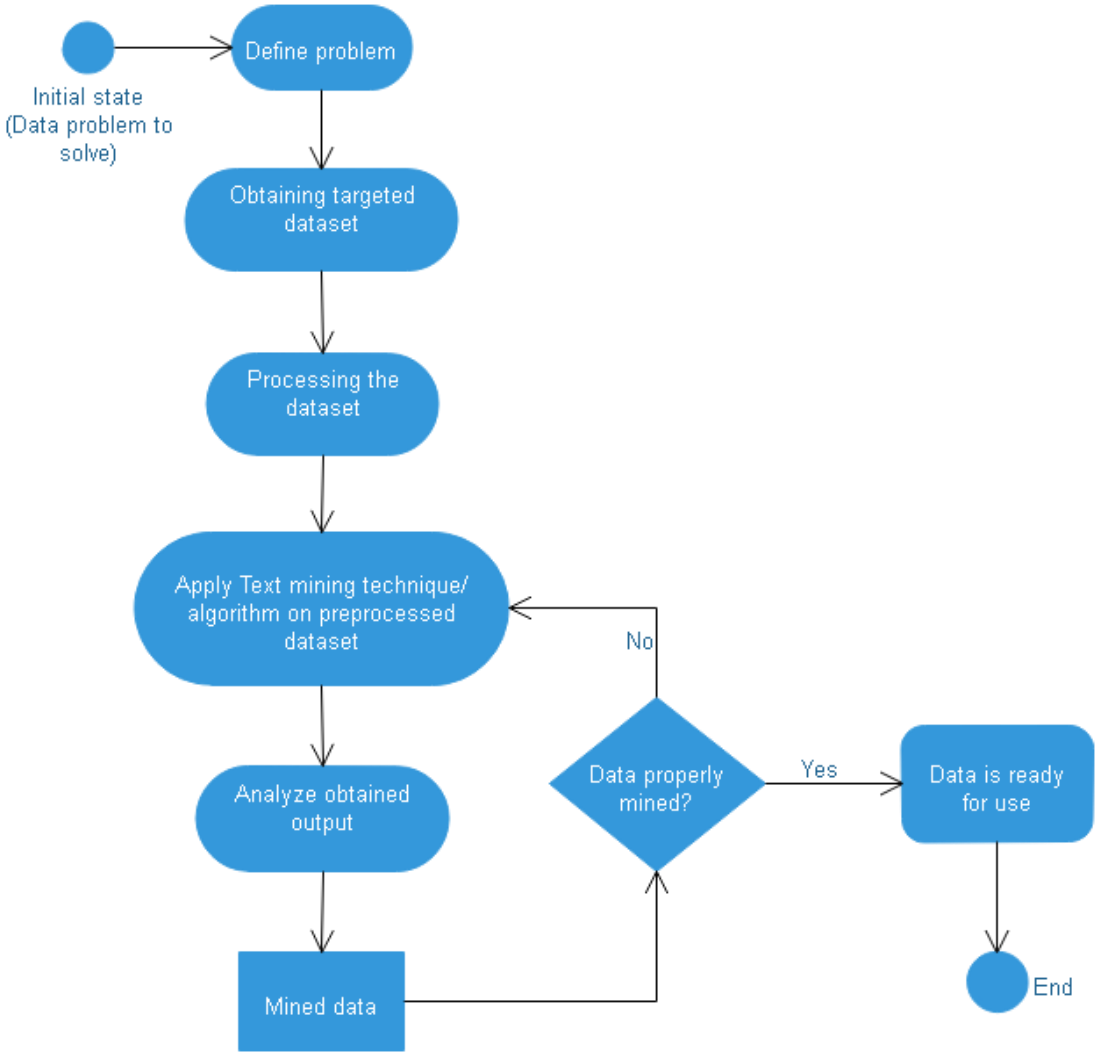


Figure 1: The general text mining process.

The next section dives deep into the various tools and methods used in text mining, along with the different ways they are applied to extract valuable information from textual data. We explore the range of techniques available and how they are utilized within the field of text mining.

2.2.2.1 TEXT MINING TECHNIQUES

Text mining tackles the challenge of extracting knowledge from text data through a series of steps. These include finding specific information (information retrieval), extracting key details (information extraction), sorting text into categories (text categorization), condensing large

amounts of text (text summarization), grouping similar texts together (text clustering), and finally, presenting the information visually (visualization).

I. INFORMATION RETRIEVAL

Information Retrieval (IR) involves the process of locating materials, typically unstructured text documents, that meet specific information needs from within extensive collections, usually stored digitally [17]. It is defined in [18], as the process of retrieving information resources that are relevant to a specific information need from a collection of such resources. Automated information retrieval systems are employed to mitigate the effects of information overload. Information retrieval primarily aims to enhance access to information rather than analyze it or uncover hidden patterns, which are the main objectives of text mining. Numerous universities and public libraries employ information retrieval systems to facilitate access to books, journals, and other documents. The most common applications of information retrieval involve handling textual data, with web search engines being the most prominent examples [15], [18], [19].

To meet an information retrieval task, user information needs are entered into information retrieval systems as natural language text queries. A query is a formal request for information from a database. In an information retrieval process, a query identifies several objects that may match a particular search weighted in accordance to relevance. Since textual information retrieval involves processing both document sources and queries expressed in natural language, several textual operations are performed in addition to the classic steps of retrieval. The user inputs a textual query (usually keywords) into the system. This query gets broken down and adjusted using special text-handling techniques. These techniques are also used to organize the information the system has access to. Finally, the system transforms the adjusted query into a format it can understand to find the answer. The query is run against a document source, such as a text database, to retrieve a collection of relevant documents. Rapid query processing is enabled by an index structure previously constructed from the documents in the document source. The retrieved documents are then organized, with each document ranked based on its estimated relevance to the user's needs. The user subsequently reviews this set of ranked documents to extract useful information [19].

Figure 2 provides a high-level depiction of a typical information retrieval process, as described above;

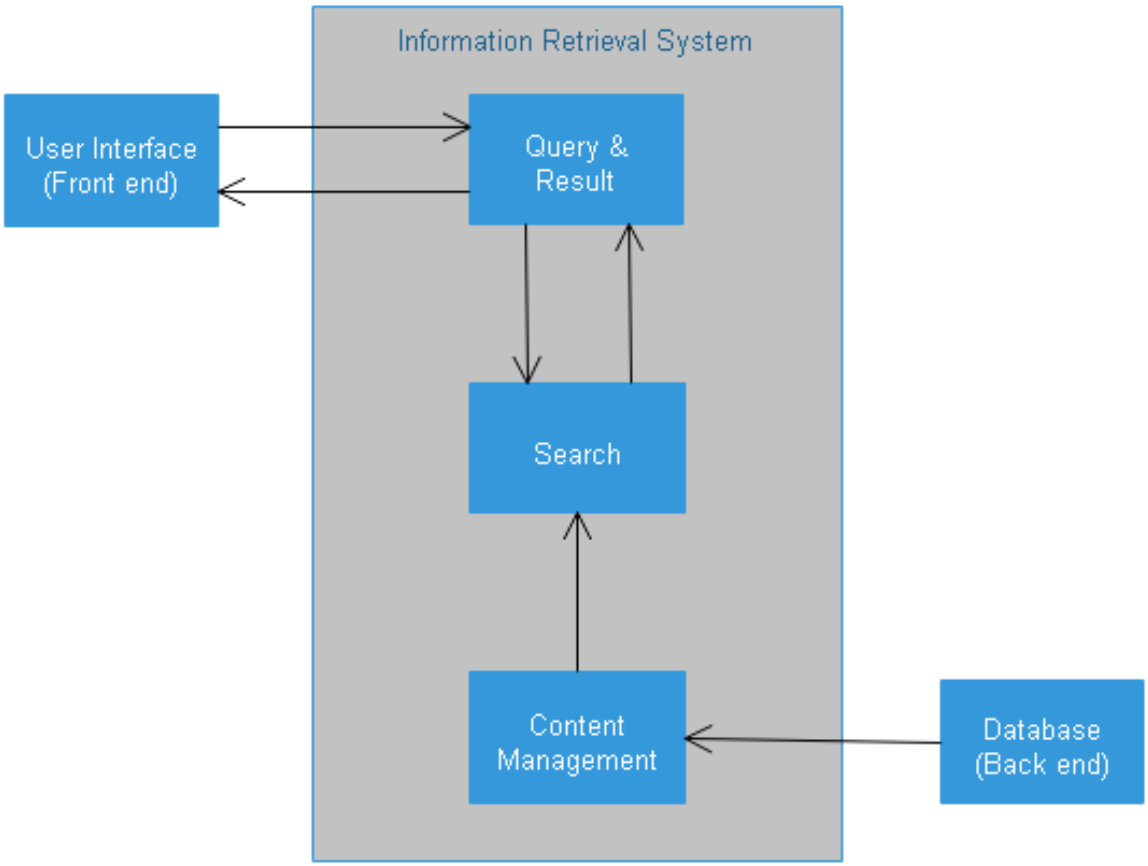


Figure 2: Information retrieval process

II. INFORMATION EXTRACTION

Information extraction (IE) involves scanning text to identify information of specific interests, which includes extracting entities, relationships, and, more complexly, events detailing who did what to whom, along with when and where [20]. The core concept of information extraction involves using computers to automatically pull out pre-defined types of information from human language text. It seeks to process text in natural language to identify instances of certain classes of objects or events, as well as the relationships between them. Information extraction is one of the core technology-premises for text mining. The process uses relationships within the textual dataset to extract small segments of text, such as people, places, times, dates, and addresses, by matching patterns. This technique yields remarkable results in fulfilling information needs when applied to large volumes of text, especially after converting the unstructured textual dataset into a structured

format [21] - [23]. In essence, the objective of information extraction is to identify phrases within the dataset and determine the relationships among them. This technique in many instances where the overall goal is to analyze unstructured textual data, serves as an initial step [10].

Figure 3 below illustrates the information extraction process.

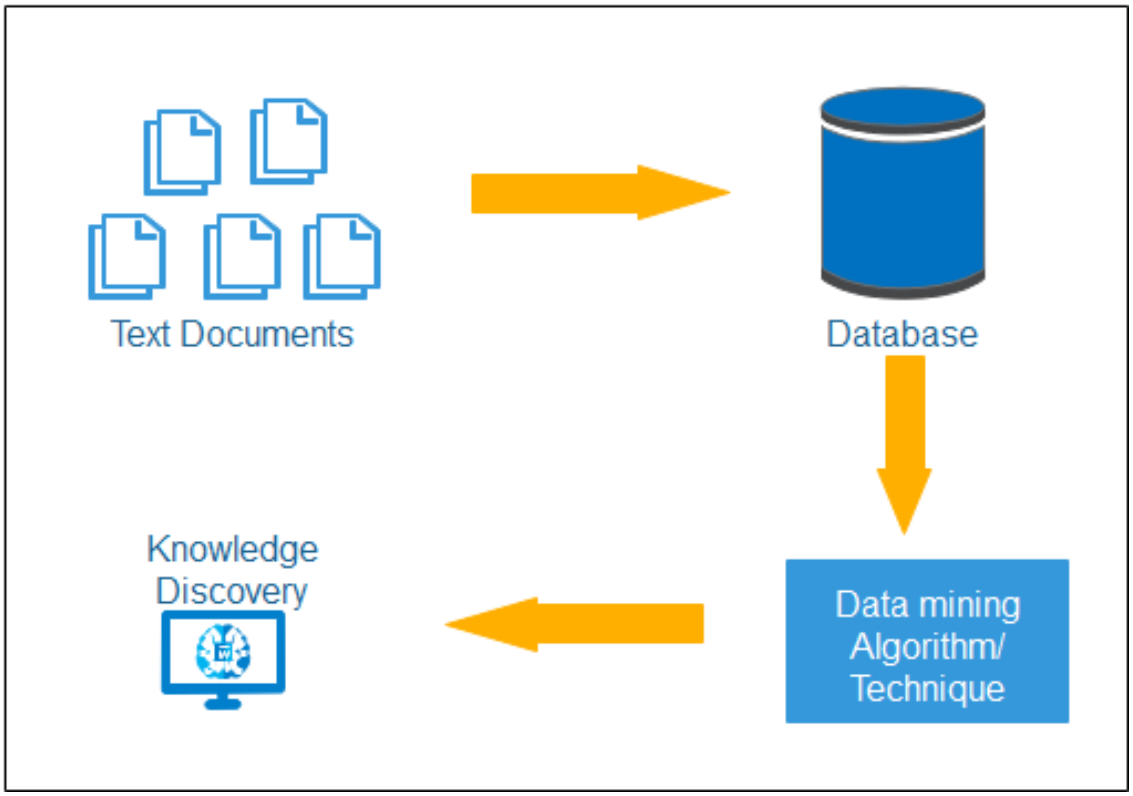


Figure 3: Information extraction process

III. TEXT CATEGORIZATION

Text Categorization, also known as Text Classification (TC), is a crucial component of text mining. It involves the process of manually constructing automatic text categorization systems using computer algorithms. This entails manually defining relevant algorithmic functions that encapsulate expert knowledge on classification of documents or textual data into specific categories [12]. Categorizing unstructured text into predefined groups based on its inherent content is an essential aspect of numerous real-world applications [24]. These applications include sorting emails into folders, identifying topics, and other information management activities, such as categorizing a collection of Short Message Service (SMS) text messages—a challenge we aim to address in this study.

The process of Text Classification involves several sub-processes, presented in Figure 4.

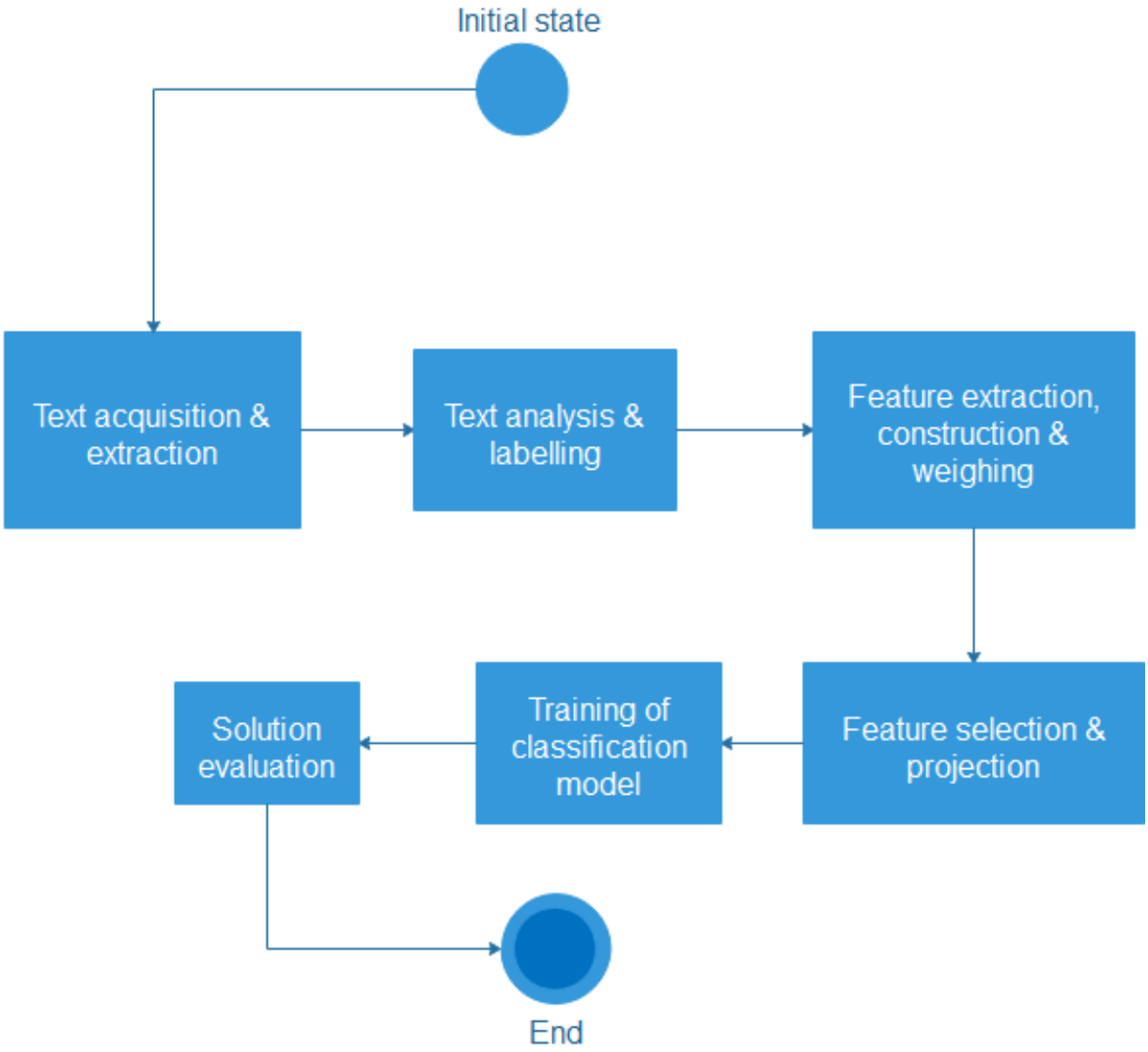


Figure 4: Text categorization process

Details of each stage shown in Figure 4 above are described below;

a) Text Acquisition and Extraction

Text categorization begins with the acquisition of a textual data from a source or various sources commonly being a table or tables in open databases or flat files. Other data sources may include the internet, research activities, survey results, measurements from physical devices such as sensors etc. Text acquisition and extraction is the task of obtaining a dataset

that represents a physical or business process [25]. This stage is sometimes referred to as document collection.

b) Text Analysis and Labelling

At this stage, the dataset undergoes preprocessing to create a representation that is required by the chosen learning method. The preprocessing step includes, among other tasks, dividing the dataset into two subsets for training and testing the categorization model, respectively. Each collection or row of texts in the training dataset is then labeled, meaning a relevant topic is assigned to it. In many cases, this exercise is done by domain experts. Text labelling is basically manual categorization of text but for the sole purpose of training a computer model. It is an indispensable stage of data preprocessing [26].

There are several methods for data labeling, including internal labeling, outsourcing, crowdsourcing, using specialized outsourcing companies, synthetic labeling, and data programming [27]. Internal labelling involves assignment of the labelling task to an in-house team of experts – either domain experts or data scientists, outsourcing involves searching for freelancers across various recruitment, freelance, and social networking sites to perform the task of labeling, crowdsourcing involves cooperation with freelancers from crowdsourcing platforms such as Amazon Mechanical Turk or Clickworker while specialized outsourcing companies involves hiring an external team for a specific project which in our case is labelling. Synthetic labeling involves creating data that mimics real data based on key parameters defined by the user. Data programming involves writing a computer program or scripts that can automatically label data.

The result of the categorization phase is a labeled dataset to be utilized in subsequent steps.

c) Feature Extraction, Construction and Weighting

At this stage, a labelled dataset undergoes feature extraction, construction and weighting for text representation. A feature is referred to as a word, numbers or symbol in a message. Turning text documents into a format computer can understand is a classic text classification task. It involves changing the content of the documents for these reasons:

i. To reduce processing time when executing a machine learning algorithm.

ii. To decrease the storage space required for saving the dataset on a hard disk or for loading it into a computer's main memory.

iii. To facilitate the creation of more insightful visualization tools.

iv. To mitigate problems associated with the curse of dimensionality.

In text classification, documents are depicted using the vector space model, which treats the content of documents as a Bag of Words (BOW) while disregarding their syntactic and grammatical structures. The BOW model is formed as a two-dimensional vector. The first dimension represents the terms found within the documents, and the second dimension refers to the documents themselves. In this model, each entry contains a value that signifies the discriminative importance of a term in the context of a classification task [27] - [32].

Feature extraction entails creating a new set of features from the original set by applying a functional mapping, while feature construction involves revealing hidden relationships among features and enlarging the feature space by generating new features. Methods that employ feature extraction seek to decrease the dimensionality of datasets by combining features.

Feature weighting can be seen as an extension of feature selection. Imagine a dataset Y comprising n entities, each described by the same set of features $V = \{v_1, v_2, ..., v_m\}$. In this scenario, a feature weighting algorithm is designed to assign a weight wy to each feature $v \in V$, usually within the range [0, 1]. This weight w_y reflects the importance of feature v in relation to the specific problem at hand [32].

The objective of this stage is to create a representation that meets the requirements of the chosen learning method for the classification model.

d) Feature Selection and Projection

Feature selection involves selecting a subset of the original features, with the goal of effectively minimizing the feature space according to a specific criterion. Dimensionality

reduction aims to trim down the number of features to be modeled while preserving the content of individual messages [27]. In a dataset Y consisting of n entities y_i, each characterized by the same set of features $V = \{v_1, v_2, ..., v_m\}$, feature selection algorithms use the constraint $w_y \in \{0, 1\}$ for each feature $v \in V$. Here, if $w_y = 1$, the feature v is included in the selected subset; if w_y equals 0, it is excluded [32], [33].

After constructing and appropriately weighting the features from the dataset using the chosen feature representation algorithm, the feature selection method reduces the number of features. These are then mapped onto a lower-dimensional space to attain the optimal representation of the data [25].

e) Training of Classification Model

At this stage, a supervised machine learning approach, detailed in the section below, is employed to train a classification function that identifies a target concept. A portion of the dataset is reserved to assess the performance of the classifier on a dataset previously unseen. The dataset is randomly divided, with a larger percentage allocated as the training set and a smaller percentage as the testing set. The hyper-parameter tuning process is conducted using cross-validation on the training data. The final model is then fitted to this data and evaluated using entirely unseen data to obtain an evaluation metric that is as unbiased as possible [34]. There exist multiple methods of training a classification function such as the under listed;

i. Naive Bayes
ii. Support Vector Machines
iii. Regression-Based Classifiers
iv. Decision Trees
v. Random Forest
vi. K-nearest Neighbor
vii. Rule-based Classifiers

The result of this phase is a fully-fledged, testable classifier. For a classification model to work well on a dataset, the new dataset needs to be formatted in the same way as the training dataset [25].

f) Solution Evaluation

The evaluation stage is final and borders on the tasks of measuring performance and results of the trained classifier. The evaluation procedure assesses the performance of the text classification process in terms of accuracy, precision, recall, and F1-score. Accuracy is calculated as the proportion of correct predictions relative to the total number of instances evaluated. Precision measures the accuracy of the labels assigned by the classifier, while recall measures the fraction of truly relevant labels that the classifier accurately identified. The F1-Score represents the harmonic mean between precision and recall. It serves as a core indicator of the effectiveness of the categorization model [25], [34], [35].

In this phase, we validate whether the text classifier delivers the expected results.

2.2.3 MACHINE LEARNING

Machine learning is defined as the examination of algorithms that enhance their performance autonomously through experience [11]. It is a practical approach within AI and data processing that allows programs to continuously improve through analysis of historical data. Machine learning utilizes computing to develop systems that are capable of learning from data through training. Over time, these systems can learn and enhance their capabilities, refining a model that enables them predict outcomes from past learning experiences [36].

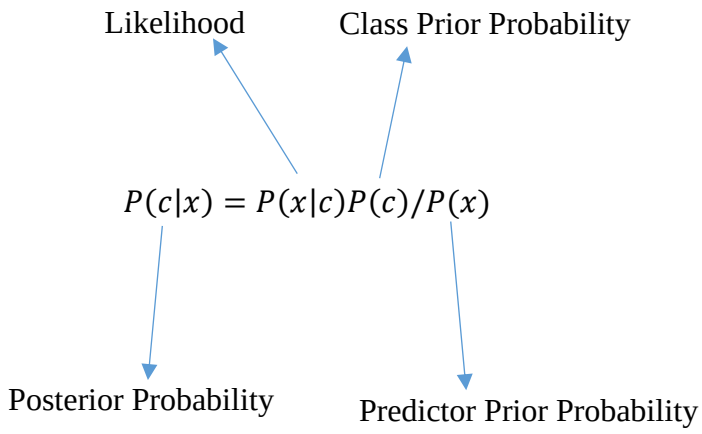
There are four categories of Machine learning, namely; supervised, unsupervised, semi-supervised, and reinforcement learning. In supervised learning, algorithms are trained to construct models that capture the relationships between a set of input features and a target output variable. This enables the generation of predictions for unseen data instances by leveraging the learned relationships embodied within the model. Unsupervised learning involves training computers with unlabeled data, allowing them to discover patterns within the data. Unsupervised learning algorithms employ pattern detection and descriptive modeling techniques to analyze input data, mining for rules, identifying patterns, and grouping data points. This helps derive meaningful

insights and provides a clearer description of the data to users. Semi-supervised learning takes advantage of components of both supervised and unsupervised learning. It is particularly effective for building models when labels are missing from most observations but are available in a few. In reinforcement learning, a model interacts with its surroundings, and learns to take actions that lead to the most positive outcomes. This method allows software agents to independently ascertain the best course of action within a given context, thereby improving their efficacy [37].

In this study, we are developing a model that can automatically categorize text data. Supervised machine learning allows us to achieve this by training the model on existing labeled text data. This training helps the model identify key features that differentiate various categories, enabling it to make accurate classifications for new text data [38]. Numerous text classification algorithms have been employed by researchers to address problems in text classification. This study explores a variety of algorithms including Naive Bayes, Support Vector Machines, Regression-Based Classifiers, Decision Trees, Random Forest, K-nearest Neighbor, and Rule-based Classifiers.

I. NAÏVE BAYES

Naïve Bayes is a commonly employed and simple classification algorithm in machine learning, founded on Bayes' Theorem and predicated on the assumption that predictors are independent of one another [39]. Bayesian classifiers determine the most probable class for a given example, which is characterized by its feature vector. Naive Bayes classifiers represent the most basic instance of Bayesian network models for classification tasks. Their defining characteristic is the assumption of strong conditional independence among all features, given the class label. The algorithm determines the probability of specified outcomes under the conditional independence assumption that each word in a text is independent of the rest. In simpler terms, the impact of a specific feature value (x) on the probability of belonging to a particular class (c) is solely dependent on the class itself and not influenced by the values of any other features in the data [40] - [43].



$$P(c|X) = P(x_1|c) \times P(x_2|c) \times P(x_n|c) \times P(c) \quad (1)$$

Where;

- *P(c|x) is the posterior probability of class (target) given predictor (attribute).*
- *P(c) is the prior probability of class.*
- *P(x|c) is the likelihood which is the probability of predictor given class.*
- *P(x) is the prior probability of predictor.*

As shown in Equation (1), the Bayes' theorem offers a method for computing the posterior probability, *P(c|x),* from *P(c), P(x),* and *P(x|c).* Naïve Bayes Classifiers are probabilistic in nature and are known for their scalability [42]. Naïve Bayes helps in text categorization by combining a bunch of different features into creating a model that calculates the likelihood of some text belonging to a particular category resulting into a classifier that is not only fast but relatively effective and easy to implement – it often outperforms more sophisticated classification methods.

II. SUPPORT VECTOR MACHINE

For supervised learning text classification assignments, Support Vector Machine (SVM) are a most suited choice [39]. SVM functions as a computational algorithm that masters the art of distinguishing data points in a multi-dimensional space through the guidance of labeled examples. For example, an SVM can be trained to identify fraudulent credit card transactions by analyzing

numerous examples of both fraudulent and non-fraudulent credit card activities [46]. Classification is achieved by plotting the dataset features as individual coordinates on an n-dimensional space and performed by finding the hyper-plane providing a clear distinction between two classes. A Support Vector Machine is a theoretical classifier that uses labeled training data to form a hyperplane in a multi-dimensional space to achieve strong generalization to new data points by maximizing the margin of classification [47]. For the training data, we possess a collection of input vectors, each designated as x_i. Each vector comprises several elements, known as *features*. Each of these input vectors is matched with a corresponding label, denoted as y_i, forming m such pairs that are indexed from $i = 1$ to m. To understand this, we consider an example problem in disease surveillance that involves classifying cases of a specific cancer into relapse and non-relapse categories using genetic data for prediction. In this scenario, instances of relapse are marked as $y_i = +1$, while non-relapse cases are labeled as $y_i = -1$. The corresponding x_i are input vectors that encode the genetic data from each patient i. These datasets are represented as labeled data points within an input space, illustrated in Figure 5. For two distinctly separated classes, the learning task entails identifying a directed hyperplane—an oriented hyperplane where data points on one side are labeled $y_i = +1$, and those on the opposite side are labeled $y_i = -1$.

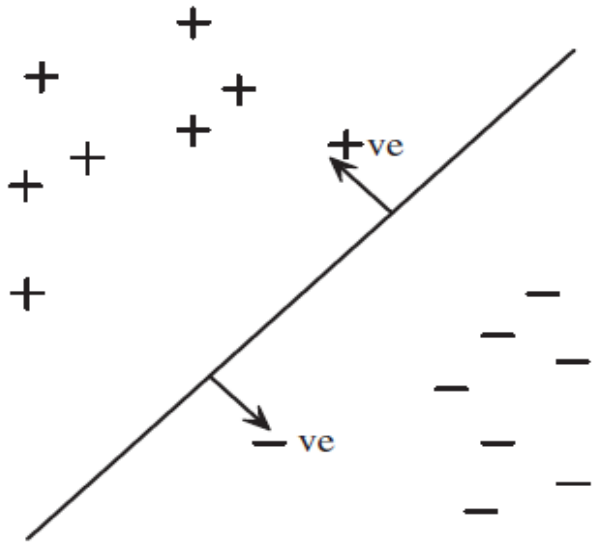


Figure 5: An illustration of the Support Vector Machine Classifier.

In SVMs, the formulated hyperplane is designed to maximize the distance between the nearest labeled points, known as support vectors, from each class. These points, lying closest to the

hyperplane on both sides, have the greatest impact on its placement and are thus termed 'support vectors'. The equation defining this separating hyperplane is presented as follows:

$$w.x + b = 0 \quad (2)$$

Where;

- *w is the weights that determine hyperplane orientation*
- *. denotes the inner or scalar product*
- *x are points located within the hyperplane and the normal to the hyperplane*
- *b is the bias or offset of the hyperplane from the origin in input space*

As shown in Figure 5, the decision function of a classifier includes the expression w.x + b. The corresponding separating hyperplane, defined by w.x + b = 0, appears as a line on this 2-dimensional plot. This hyperplane segregates the two classes of data. Points on one side are labeled as positive or yi = +1 (where w.x + b ≥ 0), and points on the other side are labeled as negative or yi = −1 (where w.x + b < 0). Distinct from other classification techniques, Support Vector Machines necessitate the inclusion of both positive and negative training sets. This is essential for identifying the decision surface that most adeptly divides positive from negative data within the n-dimensional space — termed the hyperplane. Support vectors are the data points that are closest to this decision surface [12].

According to [48], Support Vector Machines tend to perform well for text categorization problems owing to their ability to generalize into high dimensions.

III. LOGISTIC REGRESSION

Logistic regression is frequently used in the social sciences to analyze outcomes that are inherently or usually represented by binary variables [49]. Logistic regression is designed to discover the most effective Boolean combinations of initial predictors that capture the variations in the outcome variable. This approach identifies variables and their interactions that are connected to the response and/or possess predictive capabilities. [50]. Logistic regression employs a linear regression model with a sigmoid function to map the linear combination of independent variables to a probability

distribution between 0 and 1 for a categorical dependent variable. The model coefficients, which link predictors to the target, are derived using Maximum Likelihood Estimation (MLE). This method consistently estimates the parameters of a model by identifying values that maximize the likelihood of the observed sample [51].

According to [52], Logistic regression offers several advantages over traditional linear regression. It doesn't require a strict linear relationship between variables, can handle non-normal error distributions, and is less sensitive to the measurement scale of independent variables. This makes it a more robust tool for analyzing certain types of data. Logistic regression can handle non-linear relationships by applying a non-linear log transformation to the linear regression equation. The error terms' distribution does not require multivariate normality, but meeting this condition can enhance the robustness of the solution. The error variance may vary and be heteroscedastic across different levels of the independent variables. Logistic regression is sufficiently versatile to manage both continuous and discrete data as independent variables. However, logistic regression requires larger sample sizes because the maximum likelihood estimates it uses are generally less efficient than the ordinary least squares method used in linear regression. The basic logistic function is defined by the following formula:

$$y = \frac{e^x}{1+ e^x} = \frac{1}{1+ e^x} \quad (3)$$

The function, represented graphically in Figure 6, can be extended for flexibility purposes and be defined by the formula:

$$y = \frac{e^{\propto + \beta x}}{1+ e^{\propto + \beta x}} = \frac{1}{1+ e^{-(\propto + \beta x)}} \quad (4)$$

α and β represent the intercept and slope of the logistic function, respectively. Logistic regression is used to fit these parameters, α and β, which are the regression coefficients. Figure 6 illustrates the logistic function when α and β are set to 0 and 1, respectively.

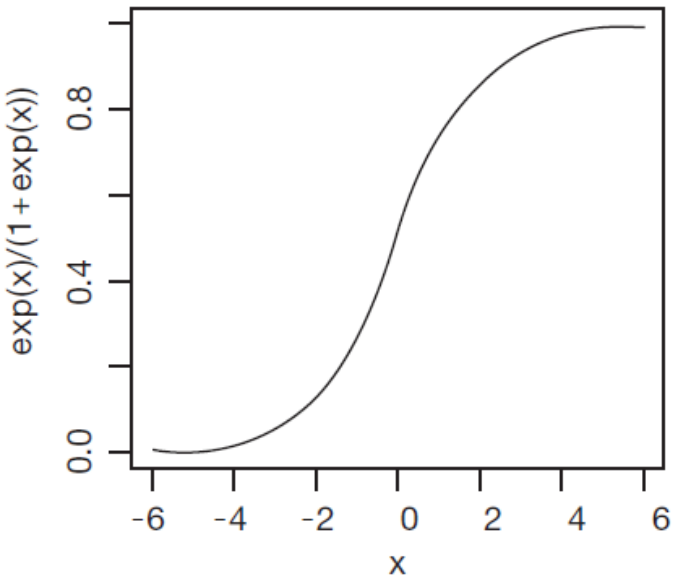


Figure 6: The logistic curve where α = 0 and β = 1.

A logistic curve begins with gradual, linear growth, transitions into exponential growth, and eventually stabilizes. The logistic function transforms an 'S'-shaped curve into a nearly straight line, expanding the scale of the proportion from 0 to 1 to range between -∞ and +∞:

$$logit(y) = \ln(odds) = \left(\frac{P}{1-P}\right) = \propto + \beta x \qquad (5)$$

Here, p represents the probability of the outcome of interest, α is the intercept parameter, β is a regression coefficient, and x is a predictor variable [51], [52].

The logistic regression classifier can handle any number of numerical and/or categorical variables.

IV. DECISION TREES

Decision Trees are a non-parametric supervised learning technique used for both classification and regression. The goal is to build a model that predicts the value of a target variable by learning straightforward decision rules derived from the data features [53]. Decision trees, use a recursive partitioning strategy, effectively uncover crucial attributes and hidden patterns in large datasets for classification and predictive modeling [54]. According to [55], a decision tree is a hierarchical approach to supervised learning where the local region is pinpointed through a series of recursive splits, achieved in a relatively small number of steps. The effectiveness of decision trees is seen in

their ability to deliver straightforward representations that are easily comprehensible to both humans and computers, coupled with their robust capability to tackle complex problems [56]. Decision trees are recognized as highly efficient and effective machine learning techniques, widely applied to solve real-world challenges within the realm of artificial intelligence.

A decision tree algorithm is a visual model that outlines all potential outcomes of a decision based on certain conditions. It comprises a Root Node, Leaf Nodes, and Branches. The primary operations involved in managing a decision tree are Splitting and Pruning. The Root Node encapsulates the entire dataset or a subset, which is then divided into two or more homogeneous sets. A Leaf Node indicates a segment of the dataset that cannot be further divided. A Branch, also referred to as a Sub-tree, emerges when a decision tree or node undergoes splitting. Splitting is the act of dividing the Root Node or a Sub-node into various segments based on specific criteria, while pruning involves the removal of extraneous branches from the tree.

In a decision tree, each node conducts a test with discrete results that label the branches. For any given input, a test is performed at each node, and a branch is chosen based on the test's outcome. This sequence begins at the root node and continues recursively until reaching a leaf node. At this leaf node, the value present is the output of the decision tree [55]. The input for a decision tree construction algorithm consists of a set of classically labeled examples. Decision trees function as Boolean operators where the input is an object or the characteristics of a situation, and the output is a binary "yes" or "no" decision. In a nutshell, Root nodes correspond to property tests, Leaf nodes to Boolean values and Branches to possible values of testing attributes [57].

For example, considering the Iris Flower Dataset [58] that was first introduced in [59] containing data representing 150 flowers with each instance belonging to one of the three flower species, namely; Virginica, Versicolor, and Setosa. Decision trees can be employed to determine the species of each Iris Flower instance based on a set of learned features or rules, which in this scenario are the measurements of various parts of the flower. These measurements encompass lengths and widths of petals and sepals [60]. Figure 7 displays the decision tree derived from the classification example provided. In this tree, instances of the Setosa species are identified by a petal length of 2.35 cm or less. Versicolor species are recognized by a petal length greater than

2.35 cm and a petal width of 1.2 cm or less, while Virginia species are distinguished by a petal length greater than 2.35 cm and a petal width exceeding 1.2 cm.

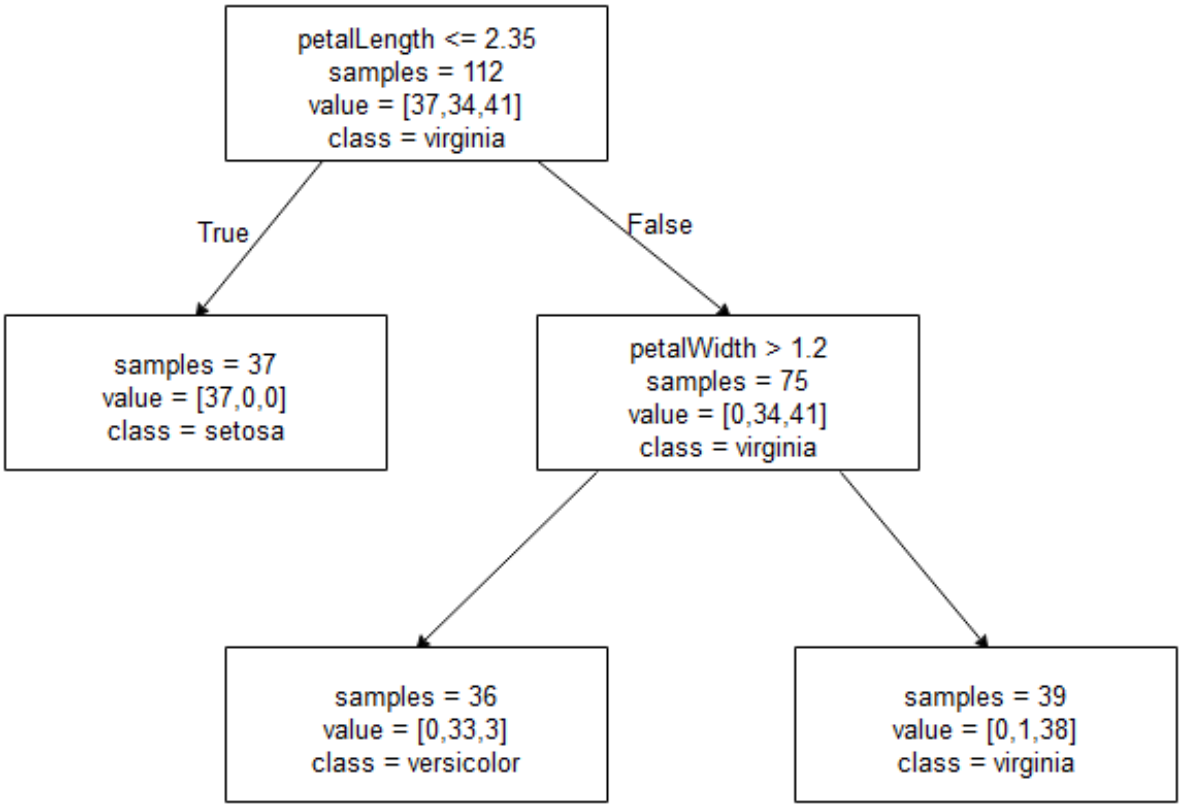


Figure 7: Example of a decision tree resulting from a classification task aimed at identifying rules that predict the species of a flower based on measurements of specific parts of the flower.

V. RANDOM FOREST

Random forests combine multiple, randomized decision trees to improve prediction accuracy and combat overfitting [61]. Overfitting is a statistical term describing a modeling error that arises when a function fits a specific dataset too closely [62]. Random Forest (RF) classifiers are based on decision trees. Although single decision trees are straightforward to visualize and comprehend because they replicate human decision-making processes, they tend to perform poorly in generalizing to new, unseen samples [63]. Therefore, Random Forest classifiers come in play as a combination of multiple decision trees to augment generalization. Random Forest Classifiers offer notable enhancements in classification accuracy by constructing an ensemble of decision trees and aggregating their predictions to select the most popular class [64].

Random Forest combines several decision tree models, each trained on a randomly selected subset of the data, using replacement to ensure variability. The final model is derived by averaging the predictions from these individual trees or by a majority vote. An example of a Random Forest model is depicted in Figure 8.

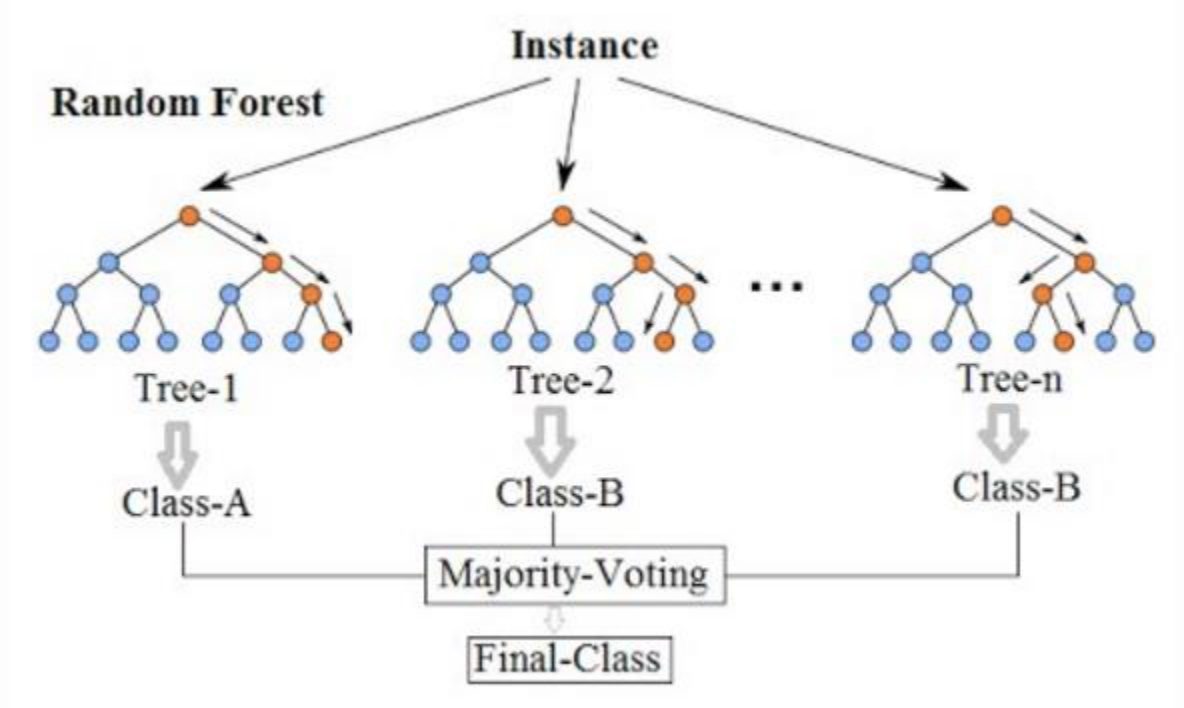


Figure 8: A simplified Random Forest example framework.

VI. K-NEAREST NEIGHBOR

K-Nearest Neighbor (k-NN) is a nonparametric learning algorithm that is highly sensitive to the choice of distance function, due to its inherent vulnerability to irrelevant features [66]. The k-NN classifier stands as one of the most straightforward machine learning algorithms one can use. It classifies new instances based on their similarity to existing training patterns. A class is determined by the most frequently occurring class among its k-nearest neighbors. This is achieved by calculating the Euclidean distance between the new pattern to be classified and all other patterns in the training dataset to measure similarity [67]. Depending on the particular problem, k-NN techniques can sometimes show limited effectiveness in terms of classification accuracy.

K-nearest neighbors of a test document x are identified from all training documents and categorized based on the classifications of the k-neighbors. The score for each category is derived

from the similarity between x and each neighboring document. If multiple k-nearest neighbor documents fall into the same category, their scores are summed to represent the overall similarity score for that category in relation to the test document *x*. The algorithm then assigns to the test document *x* the candidate category with the highest score by ranking the scores of all candidate categories [68]. Expressed below is the k-NN decision rule:

$$f(x) = argsMaxScore(x_i, C_j) = \sum_{d_j \in kNN} sim(x_1, d_i) y(d_i, C_j) \quad (6)$$

Where;

- *f(x) is the label assigned to the test document x*
- *Score(x, C_j) is the score of the candidate category C_j with respect to x*
- *sim (x, d_i) is the similarity between x and the training document d_i*
- *y(d_i, C_j) € {0, 1} is the binary category value of the training document di with respect to C_j (y = 1 indicates document di is part of category C_j, or y = 0)*

Despite their ease of implementation, k-NN classifiers often lack in the area of accuracy because they do not tackle sensitivity to noisy data. K-NN classifiers are also constrained by data storage requirements, particularly in large search problems where finding the nearest neighbors is computationally demanding [69].

VII. RULE-BASED CLASSIFIERS

Rule-based classifiers operate using a series of IF-THEN rules for classification. Each rule consists of conditions that test the attributes of instances. A rule determines an instance if the instance's attributes satisfy the conditions specified in the rule, determining the probability distribution over the applicable class or classes for those instances. The classifier constructs the set of rules from the training dataset and based on the modelled rules, the test instance is classified [70] - [73].

Rules are expressed as:

*IF **condition** THEN **decision**.*

The algorithm predicts an appropriate class for an instance of a dataset according to the decision For example, shown in the table below is a simple illustration of instances of a dataset of living organisms classified using a rule-based classifier.

Table 2.1: Example rule-based classification for selected organisms

Name	**Blood type**	**Give birth**	**Can fly**	**Live in water**	**Class**
Grizzly bear	warm	yes	no	no	mammal
Hawk	warm	no	yes	no	bird

R1: IF (GiveBirth = “yes”) and (CanFly = “no”) THEN (Class = “mammal”) (7)

R2: IF (GiveBirth = “no”) and (CanFly = “Yes”) THEN (Class = “bird”) (8)

Where;

- *R is the rule*
- *GiveBirth and CanFly are conditions*
- *Class is the decision*

While they make modification for new data easy and tailoring rules to specific cases possible, Rule-based classifiers present a challenge of high computational costs.

2.2.4 NATURAL LANGUAGE PROCESSING

Natural Language Processing (NLP) is a multidisciplinary field that equips computers to understand and process human language in all its spoken and written forms [89]. NLP involves the translation human language into data that computers can process for knowledge about the world [90], involving the automated handling of text in languages like English or Swahili. The scope of NLP tackles a spectrum of tasks, from fundamental steps like sentence and word segmentation to

advanced functionalities like sentiment analysis and uncovering opinions within text data [91]. Automatic text categorization can be accomplished through the execution of various NLP tasks.

Natural Language Processing (NLP) is divided into two main categories: Natural Language Understanding (NLU) and Natural Language Generation (NLG). NLU involves deciphering the intended semantic meaning from various possible interpretations of a natural language expression [92], while NLG focuses on converting structured data into natural language. Limited by a lack of artificial intelligence, traditional algorithms struggle with natural language. Every new document or word set would require a custom algorithm, hindering scalability and adaptation [93]. This is why Machine Learning (ML) bridges the gap, allowing NLP tasks to handle new information. Building on the ML techniques mentioned earlier, NLP can now extract meaning and adapt to new text through learned rules. Natural Language Processing (NLP) relies on two crucial steps for text categorization: getting the text ready for analysis and using AI's language capabilities [94]. The first step involves data cleaning and preprocessing.

This includes various processes like breaking the text into smaller units (segmentation and tokenization), reducing words to their base forms (stemming and lemmatization), identifying the grammatical role of words (part-of-speech tagging), analyzing sentence structure (dependency parsing), recognizing named entities (people, places, organizations), and even discovering underlying themes (topic modeling). These steps essentially prepare the text for AI to understand its meaning and perform categorization effectively.

I. TEXT SEGMENTATION AND TOKENIZATION

This stage entails a Machine Learning model dividing a given paragraph into distinct sentences. Segmentation involves dividing text into segments to establish a structured format [95] that facilitates text processing tasks like information extraction and summarization, allowing each segment to yield meaningful insights. This is an intuitive activity, just like human beings process the meaning of paragraphs from one line to the next [93].

Tokenization involves splitting raw text into smaller pieces, such as words and phrases, known as tokens [96]. The algorithm for tokenization can be as simple as identifying a word [93] every time a space is encountered. The tokens are categorized into various types i.e. nouns, numbers, verbs,

punctuation marks and so on. For example, given the sentence “Jesus fed 5000 people.” each element would be separated and categorized as illustrated in Figure 9.

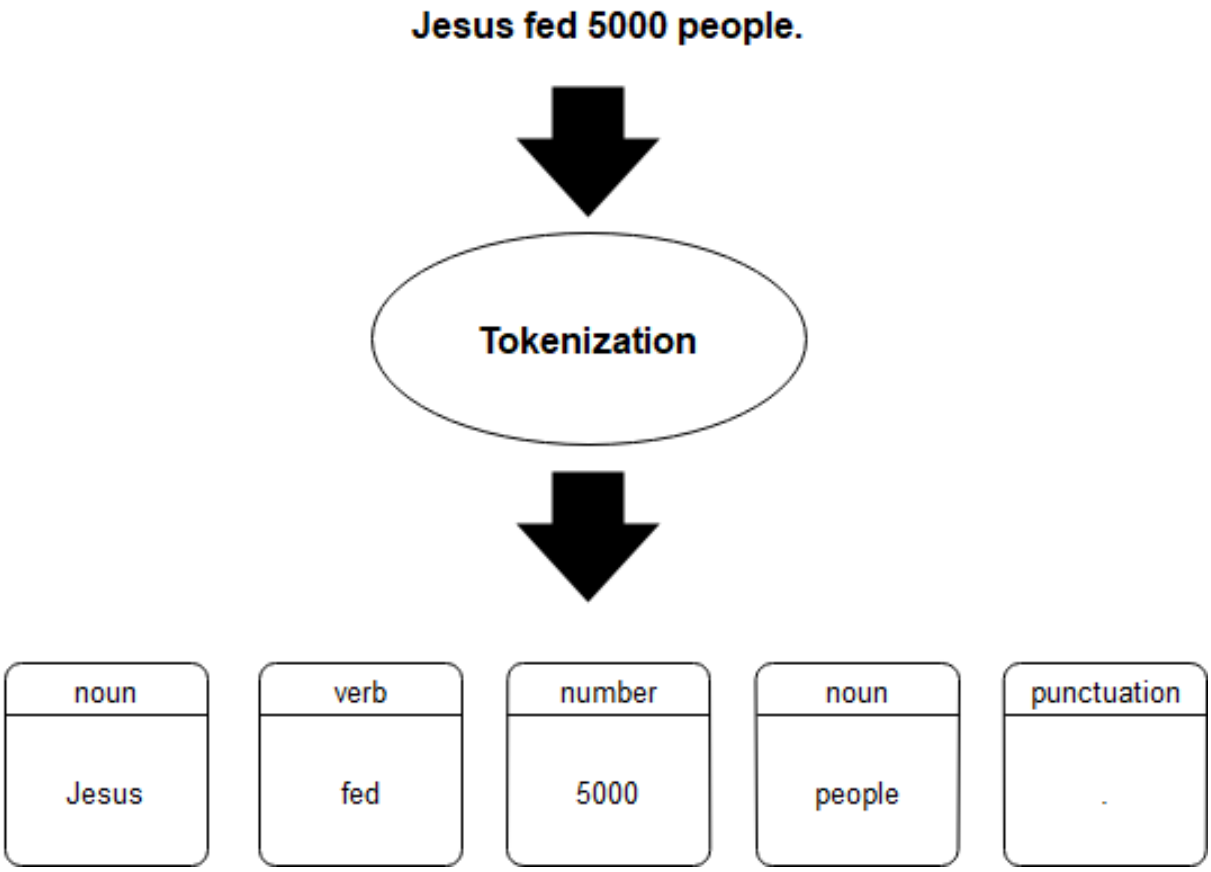


Figure 9: Example of Tokenization

II. STEMMING

Stemming is the process of reducing words to their root form [94], which improves text recall during indexing and searching in information retrieval tasks. Typically, stemming involves stripping suffixes and prefixes from words before they are assigned to an index [97]. This broadens the scope of the word, improving match potential and even accommodating spelling variations [94]. Figure 10 illustrates stemming.

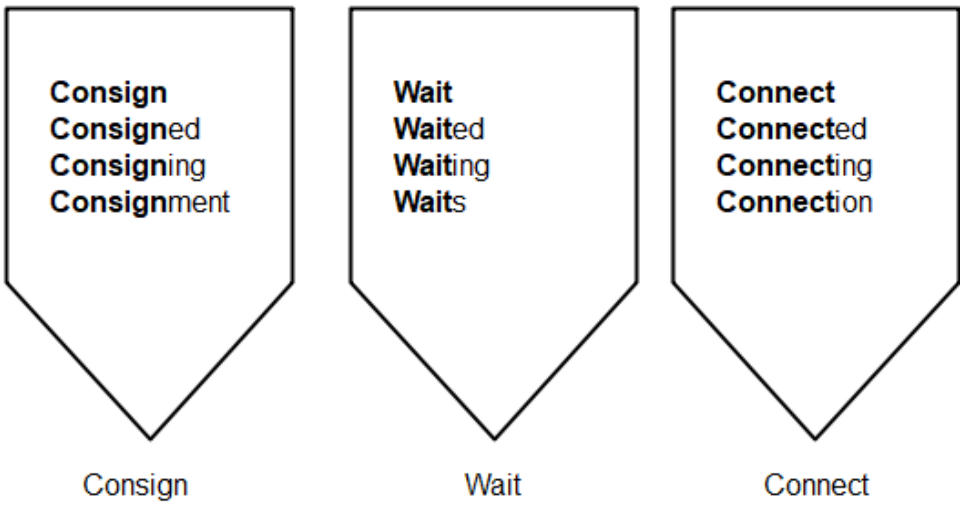


Figure 10: Example of Stemming

III. LEMMATIZATION

Similar to stemming [94], Lemmatization It involves reducing the word form to its root form [97]. For example, an algorithm can lemmatize the words “geese” and “better” can be lemmatized to “goose” and “good” respectively. Unlike stemming, lemmatization focuses on finding root words whose meaning is mostly similar to, but clearer than the given word. Figure 11 illustrates this step.

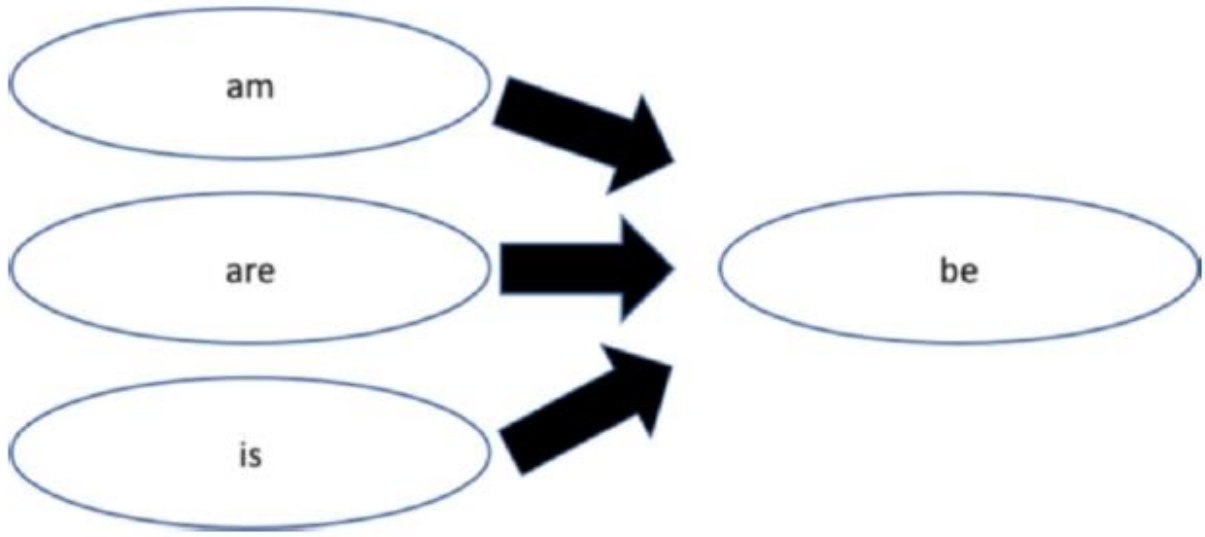


Figure 11: Example of Lemmatization

The steps for understanding and generating language are initiated once the text has been transformed into a format that is processable by a computer. These include; tagging parts-of-speech (POS), dependency parsing, named entity recognition and topic modelling.

IV. TAGGING PARTS-OF-SPEECH (POS)

This step entails determining the part of speech for each word in the text. An algorithm analyzes the text and assigns each word to its appropriate grammatical category, such as nouns, verbs, adverbs, etc. [94]. Recognizing the parts of speech of words aids the machine learning model in understanding their function within the sentence [93]. Unlike humans, who can intuitively grasp a word's meaning within the context of a sentence, machine learning algorithms require extensive text data annotated with accurate tags for each word's meaning and grammatical forms. This requirement is a fundamental aspect of Artificial Intelligence (AI) and deep learning.

V. DEPENDENCY PARSING

This step focuses on examining the text through its constituent words and phrases. It entails the automatic analysis of the dependency structure within a given sentence [99]. Each sentence can be decomposed into sub-phrases that fall into various grammatical categories, such as a noun phrase (NP), which consists of nouns that function as the subject or object of a verb [94]. In this process, an algorithm establishes how words in the text dataset relate to one another based on the rules governing the natural language being processed.

For example, in the phrase 'rainy weather,' the word 'rainy' modifies the noun 'weather.' Thus, a dependency relationship is established where 'weather' is the head, and 'rainy' is the dependent or child [100].

VI. NAMED ENTITY RECOGNITION

Words that represent important entities are identified during this process. For example, from the sentence ‘the University of Zambia published new rules today,’ humans can understand that ‘University of Zambia’ is the name of a learning institution and should be referred to as an entity. However, for a machine to handle and recognize that entity is quite challenging [101]. Here, a machine learning algorithm is employed to categorize entities into predefined classes such as Person, Location, Organization, Money, Percent, Date, Time, among others. To accurately identify and classify these entities, the model needs to be trained with relevant data [102].

VII. TOPIC MODELLING

The goal of this step is to discover underlying patterns and clusters within the text. Topic modeling dives into a collection of documents and uncovers hidden thematic structures using unsupervised

machine learning [103]. Initially, the dataset is assigned random topics, and then the machine learning model iteratively searches for matches [94]. The Latent Dirichlet Allocation (LDA) algorithm is commonly used to accomplish this task.

Leveraging advancements in Machine Learning (ML) and Natural Language Processing (NLP) – as explored in this chapter – researchers are actively tackling text mining hurdles like automated text categorization within the Computer Science domain.

2.3 RELATED WORK

SRH SMS text message categorization into thematic areas exhibits significant overlap with established text classification tasks like language identification, topic modeling, sentiment analysis, profanity detection, and opinion mining. By leveraging these existing methodologies and adapting them to the specific domain of SRH terminology, efficient automation of message categorization into topics becomes achievable.

Similar to this study's text message classification, Rosa and Ellen [104] explored machine learning for categorizing "micro-text" chat entries in military chat rooms. They tested four classifiers: SVM, k-NN, Rocchio, and Naive Bayes. Their dataset, like yours, involved thousands of messages (19,898) categorized into specific groups (filler, binary, etc.). Their findings suggest k-NN and SVM performed well, indicating their potential effectiveness in other text classification tasks.

Balabantaray et al. [105] also leveraged machine learning for text classification, similar to our study. They built an emotion classifier using SVM to categorize tweets into six emotions (sadness, anger, etc.) – expanding upon a prior system with just positive, negative, and neutral categories. This aligns with our sentiment analysis goals. Their work, like ours, focuses on classifying textual data with machine learning. They used a dataset of 8,150 tweets and achieved 73.24% accuracy in identifying emotions from text, demonstrating the potential of SVM for this task.

Weng et al. [106] developed a supervised machine learning-based Natural Language Processing (NLP) pipeline to create classifiers tailored for specific medical subdomains. These classifiers categorize clinical notes into specific medical subdomains using two distinct datasets. The initial dataset includes anonymized clinical notes from various transcriptionists and clinical users, which are publicly accessible and sourced from the Integrating Data for Analysis, Anonymization, and

Sharing (iDASH) data repository which can be found at MedicalTranscriptionSamples.com. The second dataset includes clinical notes from Massachusetts General Hospital (MGH), obtained from the Research Patient Data Registry (RPDR) of the Partners HealthCare system [107]. For each dataset, Weng et al. employed various data representation methods, weighting strategies, and supervised learning algorithms to construct classifiers. These tools aid clinicians by timely redirecting patients' unresolved issues to the appropriate medical specialties and experts, based solely on the content of the clinical notes. The study demonstrates the effectiveness of a supervised learning-based NLP approach in developing text classifiers for the medical field.

Y. Hayashida et al. [108] undertook a project to automate the categorization of conversation flows in counseling sessions using Support Vector Machine (SVM) for text data classification. They utilized a dataset containing dialogues between novice counselors and clients. The primary aim of the research was to improve the traditional process where a supervisor reviews verbatim transcripts from counseling sessions to provide guidance to beginner counselors. Y. Hayashida et al. developed an SVM-based category classification model, which successfully achieved an accuracy rate of 63.5%.

Similar to our study's classification task, Poulin et al. [109] employed supervised machine learning for text analysis. They built models to identify suicide risk in veterans' medical records (unstructured clinical notes) using a large national VA dataset. Their approach achieved over 60% accuracy in identifying potential suicide risk from textual data, showcasing the potential of machine learning for such tasks.

Koopman et al. [110] investigated automating disease classification from written death certificates. They combined machine learning and rule-based techniques to identify four specific diseases (pneumonia, influenza, HIV, and diabetes,) in the text. This approach aimed to streamline disease surveillance and enable real-time monitoring.

2.4 SUMMARY

This chapter provides a theoretical overview of various text classification approaches. It delves into relevant Computer Science fields such as Artificial Intelligence, highlighting its application in problems like the automatic categorization of short messages related to sexual reproductive health into thematic areas. The literature reviewed confirms that tasks such as automatic text

classification necessitate a form of intelligence. This chapter delves into the core concepts of text mining, exploring essential tasks like classification, information retrieval, summarization, and visualization. It also explores how text is processed and prepared for analysis, including tasks like categorization and clustering. Additionally, it offers a comprehensive theoretical review of text classification, outlining the steps involved in a text classification cycle.

The chapter outlines machine learning as a segment of artificial intelligence dedicated to creating and training algorithms that enhance their performance by gaining experience. To identify an algorithm that aligns with the objectives of the study, the chapter examines various algorithms including Naive Bayes, Support Vector Machines, Regression-Based Classifiers, Decision Trees, Random Forest, K-nearest Neighbor, and Rule-based Classifiers.

The chapter also outlines Natural Language Processing (NLP), detailing its two primary categories and the extensive stages involved, such as text cleaning, preprocessing, generation, and understanding. It discusses specific processes including text segmentation and tokenization, stemming, lemmatization, parts-of-speech (POS) tagging, dependency parsing, named entity recognition, and topic modeling.

The chapter concludes with concise reviews of relevant research in the text classification field related to the study.

CHAPTER THREE:

METHODOLOGY

3.1 INTRODUCTION

This chapter outlines the methods used to develop a text classification model for automatically categorizing short message texts from Zambia U-report into specific topics related to sexual and reproductive health. The methodology involved the underlisted key steps;

i. Understanding the current model.

ii. Data extracting and understanding.

iii. Data pre-processing and preparation.

iv. Text classification modelling.

v. Model evaluation.

To accomplish the aforementioned goals, an exploratory research approach was utilized, incorporating, observation, document analysis, and semi-structured interviews.

The chapter is organized into sections that correspond to each of the steps listed above, detailing how each step was accomplished in the automation process of the Zambia U-report sexual reproductive health messages classification process.

3.2 UNDERSTANDING THE CURRENT TEXT CATEGORIZATION PROCESS

This study investigated existing methods for categorizing U-report SMS messages into thematic areas, employing an exploratory research approach. This involved conducting interviews with NAC experts responsible for these categorizations, and observing their processes to gain a comprehensive understanding of the methods employed.

To direct the interviews, we developed an interview guide, centered on the research objectives to ascertain the existing model of categorization of text-based data. This approach addressed the

research question concerning the categorization process of this information. To maintain unbiased data, two out of the three counselors were independently interviewed. Additionally, the tools they utilized for categorizing textual data were meticulously observed, along with the list of common categories that have been adopted over time.

3.3 DATA EXTRACTION AND UNDERSTANDING

For data analysis, Zambia U-report data dump containing all messages (over 5.9 million incoming and outgoing SMS) from 2012 to 2019 was acquired from NAC in raw SQL format.

To access and extract the text messages, the researchers imported the database dump into a local server using WampServer, a web development platform that allows for the creation of dynamic web applications [111]. However, due to the file's large size of 1.45GB and MySQL's limitations, the database dump was imported into the MySQL database management system using the command line accessed from the MySQL console. Several commands were used, to import the data and count the number of text messages in the database. These include;

1. **use** ureport
2. **source** C:\Users\Tobsmak\Documents\Ureportdump.sql
3. **SELECT COUNT** (text) **FROM** rapidsms_httprouter_message

Where;

- **use** is the command specifying the database in MySQL into which the Zambia U-report database dump was to be imported.
- **ureport** is the name of the MySQL database into which the Zambia U-report database dump was imported.
- **source** is the command for loading the dump file.
- **C:\Users\Tobsmak\Documents\Ureportdump.sql** is the path to where the dump file was located on the local computer.

More queries were written as follows;

- SELECT COUNT(text) FROM rapidsms_httprouter_message WHERE direction = 'I' and SELECT COUNT(text) FROM rapidsms_httprouter_message WHERE direction = 'O', to count the number of incoming and outgoing messages.
- SELECT id, text, date INTO OUTFILE 'ureport_text.csv' FIELDS SEPARATED BY ',' ENCLOSED BY '"' ROWS TERMINATED BY '\n' FROM rapidsms_httprouter_message WHERE direction = 'I", which extracts the dataset into a Comma Separated Values (CSV) file for easy readability and data preparation.

Where;

- **text** is the actual SMS messages texts field.
- **rapidsms_httprouter_message** is the SMS messages texts table in the ureport database.
- **direction** is the SMS messages texts direction field with values indicating whether the message is incoming or outgoing.
- **I** is the value indicating that a particular message is incoming.
- **O** is the value indicating that a particular message is outgoing.

Incoming SMS messages texts categorization being the focus of this study, the following query was used to extract the dataset into a Comma Separated Values (CSV) file for easy readability and data preparation;

SELECT id, text, date

INTO OUTFILE 'ureport_text.csv'

FIELDS SEPARATED BY ','

ENCLOSURE CHARACTER '"'

RECORDS TERMINATED BY '\n'

FROM rapidsms_httprouter_message *WHERE* direction = 'I';

Where;

- **id** is the primary identification field for the SMS messages texts in the messages table.
- **text** is the SMS messages texts field.
- **ureport_text.csv** is the file name for the result set.
- **rapidsms_httprouter_message** is the SMS messages texts table in the ureport database.
- **I** is the value indicating the direction of a particular message as incoming.

A subset of the dataset was exported into a CSV file using Microsoft Excel to facilitate further analysis.

3.4 DATA PREPARATION AND PREPROCESSING

Following successful data extraction and storage in CSV format, the phase of data pre-processing began. This stage is critical for text classification tasks, as it transforms raw textual data into a machine learning-friendly representation. The data preparation process in this study involved the following steps:

i. Data sampling and labelling

Labeling each data point is a critical component of the data preparation phase, particularly in supervised learning where algorithms are trained to identify patterns and make predictions based on these labels. Using random sampling without replacement, a random selection of 50,877 incoming text messages was chosen from the original dataset to be used for further analysis. This sampling method ensures that each element in a finite population has an equal chance of selection, and once selected, it is removed from the pool and cannot be chosen again [122]. The messages were manually categorized into thematic areas with assistance from domain experts. This labeling process not only prepared the data for algorithm training but also served as a training exercise for the researcher to enhance their labeling skills. The annotation of the initial 1,770 texts was done by this researcher to refine their annotation proficiency.

To train the initial machine learning model, two counselors, considered experts in the field (domain experts), independently labeled the same set of U-report text messages. The model's accuracy in predicting message categories was then evaluated. To evaluate their own categorization skills against the experts, the researcher labeled a separate copy of the same dataset. Before proceeding, the researcher's labeling accuracy was assessed by comparing their labels with those generated by the trained model.

An independent dataset consisting of 4,567 U-report SMS text messages, collected between July 1st and September 31st, 2020, and previously labeled by experts as documented in one of their reports, was utilized to benchmark the trained model. This was done to verify the consistency of the researcher's labeling with that of the experts. The approach was that text messages were categorized using the model and topic predictions were compared against expert labels to assess if the researcher's labelling of the messages as consistent as domain experts.

More messages to make the training dataset were labeled after this researcher's labelling accuracy was measured to be consistent with the results produced by the trained classifier application on the previously expert labelled data. The researcher with help from domain experts labeled 50,877 text messages to make the training dataset. This number of messages was extracted in bits based on availability of annotators and the amount of data they could label.

The initial step in labeling the text messages was deciding which categories within which the dataset falls. Fourteen categories within the broader theme of sexual reproductive health namely; cervical cancer, circumcision, condoms, HIV, masturbation, other, prevention, registration/platform issues, sex and SRH, STIs, symptoms, testing, transmission and treatment.

The labeling stage was essential for the success of the machine learning process and set the groundwork for the subsequent data cleaning stage.

ii. Text cleaning

Text cleaning pre-processes the data to improve model performance by eliminating irrelevant noise. The dataset contained various impurities, including punctuation, numbers, inconsistent capitalization, special characters, and stop words (common words like "a," "the," etc.). Removing these elements improves the model's ability to focus on significant information. Additionally,

lemmatization, a dimensionality reduction technique, was applied. Lemmatization simplifies words to their root form (like changing "running" to "run"). This helps the model group words with similar meanings, improving its grasp of the text data [116]. This simplifies the analysis process and makes it easier to find patterns and relationships in text. Figure 12 shows a snippet of the code that was used to create data cleaning function.

```
def clean_text(text):
    """
    text: a string
    return: modified initial string
    """
    lemmatizer = WordNetLemmatizer()  # initialize lemmatizer
    text = text.lower()  # lowercase text
    text = REPLACE_BY_SPACE_RE.sub(' ', text)  # replace REPLACE_BY_SPACE_RE symbols by space in text
    text = text.replace('x', '')  # Remove the XXXX values
    text = REMOVE_NUM.sub('', text)
    text = BAD_SYMBOLS_RE.sub('', text)  # delete symbols which are in BAD_SYMBOLS_RE from text
    #text = ' '.join(word for word in text.split() if word not in STOPWORDS)  # delete stopwords from text
    text = ' '.join([lemmatizer.lemmatize(word, get_wordnet_pos(tag)) for word, tag in pos_tag(text.split()) if
                     word not in STOPWORDS])  # lemmatize words and delete stopwords from text
    return text

def get_wordnet_pos(tag):
    """Map POS tag to first character used by wordnet_lemmatizer"""
    tag = tag[0].upper()
    tag_dict = {"J": nltk.corpus.wordnet.ADJ,
                "N": nltk.corpus.wordnet.NOUN,
                "V": nltk.corpus.wordnet.VERB,
                "R": nltk.corpus.wordnet.ADV}
    return tag_dict.get(tag, nltk.corpus.wordnet.NOUN)
```

Figure 12: Data cleaning function

Effective text analysis requires clean data. Text cleaning techniques, like removing noise and applying lemmatization, improve data accuracy and quality. In this study, text cleaning significantly enhanced the text classification model's performance by reducing the number of unique words the machine learning algorithm needed to handle.

iii. Feature extraction and data splitting

After the text was cleaned, we moved to extract features from it in order to represent the text as numerical data. Common techniques used for feature extraction include bag-of-words, n-grams, word embeddings, etc. To prepare the cleaned text data for the machine learning model, a technique called TF-IDF was used. TF-IDF assigns a weight to each word, considering how often it appears in a single message (term frequency) and how rare it is overall in the entire dataset

(inverse document frequency). Words that are important in a specific message but uncommon overall get higher weights. This creates a numerical representation of the text that the machine learning model can understand and use for categorization.

The final step involved splitting the data into two sets: training and testing. This split is essential to prevent the model from memorizing the training data too much (overfitting) and ensure it can accurately categorize new messages it hasn't seen before (generalization). Figures 13 and 14 show the code used for this process.

```
# Import necessary libraries
from sklearn.feature_extraction.text import TfidfVectorizer
# sublinear_df is set to True to use a logarithmic form for frequency
# min_df is the minimum numbers of documents a word must be present in to be kept
# norm is set to l2, to ensure all our feature vectors have a euclidian norm of 1
# ngram_range is set to (1, 2) to indicate that we want to consider both unigrams and bigrams
# stop_words is set to "english" to remove all common pronouns ("a", "the", ...) to reduce the number of noisy features

# Define the vectorizer with specific parameters
cv2 = TfidfVectorizer(sublinear_tf= True, norm='l2', min_df=10, ngram_range=(1, 2), stop_words='english')

#Fit and transform the training data using the vectorizer
X_traincv = cv2.fit_transform(x_train)

# Transform the test data using the vectorizer
x_testcv = cv2.transform(x_test)
```

Figure 13: Feature extraction

```
# Import necessary libraries
from sklearn.model_selection import train_test_split, cross_val_score

# Split the data into train and test sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

Figure 14: Data splitting

This code snippet shuffles the data and splits it into training and testing sets. First, it mixes up the order of the data points. Then, it separates the text messages (often called features) from their corresponding categories (called targets). Ultimately, it splits the data into two sections: a larger training set, 80% of the data, which is used to train the machine learning model, and a smaller testing set, 20% of the data, employed to assess the model's performance on new, unseen data.

Before feeding the text data to the machine learning model, it needs to be converted into a format the model can understand. Figure 13 shows some code that performs this conversion. The code

uses a technique called TF-IDF to transform the words in each message into numbers. Numbers are easier for the model to process than words. The code stores these transformed messages in two sets: one for training (X_traincv) and another for testing (x_testcv). These transformed messages become the model's main source of information for learning how to categorize new messages.

3.5 TEXT CLASSIFICATION MODELLING

Once the data was prepared, a text classification model was built. This model, written in Python, uses machine learning algorithms from a free and open-source library called Scikit-learn to automatically sort the messages into categories.

To create the model, different machine learning algorithms were evaluated. These included K-Nearest Neighbors, Decision Trees, Naive Bayes, Support Vector Machines (SVM) in two variants, Random Forests, and Stochastic Gradient Descent. All these algorithms are known to perform well in text classification tasks. The goal was to find an algorithm that could precisely categorize U-report text-based data into the predefined topics.

Various machine learning algorithms were trained on the labeled training data (80% of the total messages). Each algorithm's ability to predict message categories was then assessed. The process was repeated to identify which classifier algorithm was the most accurate, the most performing classifier was then saved for further use. The remaining 20% of the data served as the test set to assess the final model's performance on unseen messages. Once the classifier achieved satisfactory accuracy, it was employed to classify 206,625 messages that had not been previously seen by the model. The unseen messages were also chosen using Uniform Random Sampling without replacement. The size of the dataset was chosen to ensure computer resources such as the RAM and processor were kept optimal.

Using the command line interface, we executed the training, testing, and saving of the model to ensure its accessibility and reusability. This method guaranteed the model's adaptability, making it simple to deploy and use. This approach aligned with our second objective.

3.6 MODEL EVALUATION AND DEPLOYMENT

The final stage involved rigorously evaluating the trained model's performance. Just like in any machine learning project, assessment and deployment are critical steps. A thorough evaluation was necessary to confirm the model's effectiveness on unseen data, not just the test set. Only after a successful evaluation could the model be confidently deployed for real-world use.

Several machine learning models were trained using the identical training dataset. Prediction accuracy of each algorithm was assessed through multiple iterations to select the best-performing model, which was then saved. To gauge the model's overall performance, researchers employed several evaluation metrics: accuracy, precision, recall, and F1 score. These metrics provide a comprehensive picture of how well the model is classifying messages.

Following the assessment of the model's performance, the subsequent step involved deploying the model for practical use on new datasets. The deployment process entailed confirming that the model is ready for production use. Using this model, over 200 thousand messages were successfully classified. The automated process was finally compared against the current manual classification process to assess its effectiveness and performance. This comparison helps to prove whether the model performs better than the current manual process meeting our third objective.

3.7 SUMMARY

This chapter details how a machine learning model was built to automatically classify U-report SMS messages about sexual and reproductive health into different categories. It covers several crucial phases, including examining existing classification methods, extracting and analyzing text data, preparing and preprocessing data, modeling for text classification, and evaluating and deploying the model. This chapter asserts that an exploratory research approach to achieve the study's objectives.

The chapter also details the process of extracting and comprehending text data. It explains how a copy of U-report data dump, containing a collection of more than 5.9 million SMS text messages was acquired. From this, a dataset of incoming messages was further extracted for additional analysis, labeling, and usage.

The chapter explains how the extracted text messages underwent preparation and preprocessing for classification. It is explained that text messages were labelled and cleaned. The cleaning involved the automatic removal of irrelevant or unnecessary words and characters. After cleaning the text data, the next step involved converting it into a format that a machine learning model can understand. This process, called feature extraction, assigns numbers to words based on their importance in a message and how rare they are overall in the data. This reduces the number of unique elements the model needs to handle. To achieve this, a technique named TF-IDF was used. The data was finally divided into two sets: training and testing. The training set is utilized to train the model, while the testing set helps in evaluating the model's ability to categorize new messages it has not previously encountered.

The chapter also describes the specific steps taken to construct the text classifier using machine learning tools. They experimented with different approaches, like finding similar messages (K-Nearest Neighbors), making decisions based on a series of simple rules (Decision Trees), and using probabilities (Naive Bayes). They also explored more advanced techniques like Support Vector Machines and Random Forests. The goal was to find the best approach for accurately sorting the messages into the right categories. The finalized model successfully classified 206,625 previously unseen messages, achieving a satisfactory level of accuracy.

Finally discussed in this chapter, were the steps in the text classification process. This includes evaluation and deployment of trained models. We assessed the model's performance using metrics such as accuracy, precision, recall, and the F1 score. After multiple iterations, the most effective model was selected based on these evaluations. The model was saved and deployed for use with the new dataset. Performance of the automated process was compared with the current manual classification process to assess its effectiveness and performance. This comparison helped determine if the model is performing better than the current manual process.

CHAPTER FOUR:

RESULTS AND DISCUSSION

4.1 INTRODUCTION

This chapter delves into the results of the study. We evaluate the performance of the machine learning model designed to automatically categorize incoming U-report SMS messages into specific topics. This approach leverages existing data, machine learning and text mining techniques, to enhance data analysis and extract valuable information.

The chapter dives into the research findings, unfolding in several sections. The first section examines the current manual process for classifying information on the Zambia U-report platform by topic. It also sheds light on the drawbacks inherent in this manual approach.

The subsequent section focuses on evaluating the researcher's ability to accurately and consistently label text messages following training. This section compares the researcher's labeled dataset with that of domain experts to determine the accuracy and consistency of the labeling process and establish whether the researcher learned how to annotate the text-based data.

To gauge the machine learning classifier’s effectiveness in classifying U-report text-based data, another section compares the human-labeled dataset with the model's predictions. This analysis sheds light on the model's accuracy in automating message categorization.

The chapter further explores the final model training process, including the selection of algorithms and the training dataset. To identify the most effective model, the chapter analyzes the prediction accuracy achieved by each algorithm. Additionally, it presents a comprehensive performance report for the best-performing model, encompassing precision, recall, F1 score, and support metrics.

To further illustrate the model's performance, this chapter includes a confusion matrix, a visual tool for understanding prediction accuracy. The chapter also analyzes the final model's predictions and compares the time efficiency of manual versus machine learning categorization for U-report messages. This analysis highlights the significant time savings offered by the machine learning model, making it a clear advantage over the manual approach.

4.2 CURRENT CATEGORISATION MODEL

At present, the classification of text is conducted manually. This is done by NAC counselors who categorize each message by recording a list of selected categories and marking them with a pen and notepad as each message is received and analyzed under a specific topic. The result of this process is a tally sheet. The counsellor assesses the message once it is received and counts it as part of one of the listed categories immediately after responding to it. Counting a particular text message as part of a given topic is purely determined by the counsellor based on the core theme of the message in accordance to their expert assessment.

The current categorization process entails utilizing a tally sheet, as depicted in Figure 15. The tally sheet comprises a set of classes, and every SMS message is computed as belonging to one of the classes.

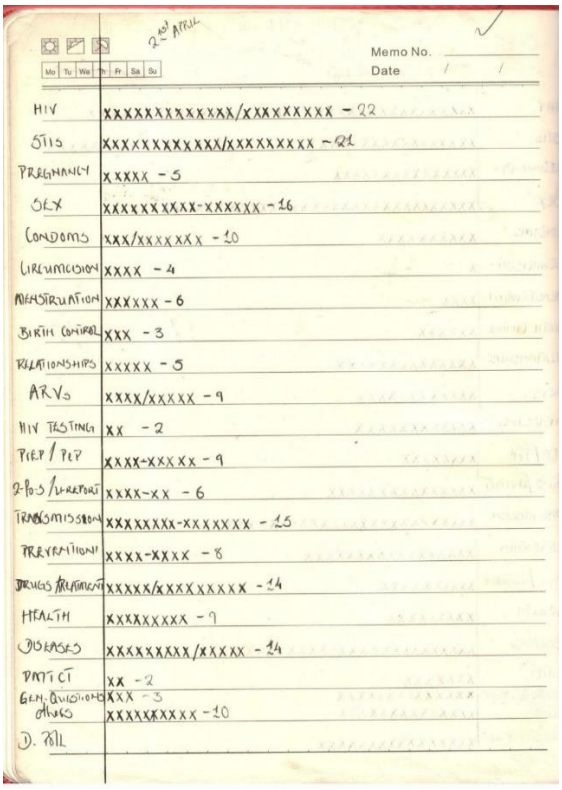


Figure 15: A tally sheet of topics for u-report messages dated 21st April, 2019.

As shown in Figure 18, there exists a classification named 'Others' for messages that do not fit into any of the specified categories. This 'Others' category assists the experts in identifying new domains for similar but frequently received messages without any specified category, aiding in the

discovery of emerging issues. Despite being a strenuous task, counsellors have to respond to U-report subscribers' messages while simultaneously categorizing them into their respective groups. At the end of each month, all counsellors convene to combine the tallies in each category to create a monthly report. he report is subsequently presented to a designated committee of stakeholders, known as the U-report Core Group, which is chaired and hosted by the NAC for purposes of making decisions. Losing, damaging, or misplacing the notepad would disrupt accurate reporting. Counselors rely on it to track messages responded to throughout the month and reclassify them if needed. This manual system presents challenges for data reliability and efficiency. Without filters to aid in searching, retrieving old messages on the platform is consistently challenging. Figure 16 dhows the report generated in April 2019, representing the combined tallies from both counselors.

	1st - 7th April	7th - 20th April	21st - 28th April
HIV	17	37	22
STIs	14	42	21
Pregnancy	4	15	5
Sex	11	34	16
Condoms	5	10	10
Circumcision	2	1	4
Menstruation	2	4	6
Birth Control	2	6	3
Relationships	3	14	5
ARVs	7	12	9
HIV Testing	4	12	2
PrEP/ PeP	1	8	9
2-for-5/ U-Report	12	15	6
Transmission	15	28	15
Prevention	6	17	8
Drugs/Treatment	1	10	14
Health	6	9	9
Diseases	9	24	14
PMTCT	3	7	2
Gen Questions	13	14	3
Others	4	14	10
Masturbation	1	16	-

Figure 16: Summary report for April 2019 based on manual assignment of topis to U-report SMSs.

Counselors typically take about 15 minutes for the first U-report message each month, and 5 minutes for subsequent messages. This breakdown reflects the initial time investment (10 minutes) in recording categories, crafting a response (3 minutes), and message categorization (2 minutes). Analyzing messages for categorization takes the most time initially. Factors like unclear language,

local dialects, distractions, or technical issues can occasionally extend this timeframe. These durations are detailed in Table 4.1.

Table 4.1: Average duration each counselor spends responding to the first message

ACTIVITY	**COUNSELLOR 1**	**COUNSELLOR 2**	**AVERAGE**
Listing topics	10.5mins	9.8mins	10.15mins
Responding to SMS text message	2.7mins	3.4mins	3.05mins
Categorizing the message	**1.7mins**	**2.4mins**	**2.05mins**
Total			**15.25mins**

The current flow described above is illustrated in Figure 17 below as a key result to achieving our first objective which is "*to understand and identify the model currently used for categorizing information into key thematic areas on the U-report platform.*"

Figure 17: Workflow for manual classification of U-report messages.

The study provides an important finding to measure the time and effort involved in the manual process. Sticking with the manual method to categorize 5,987,040 U-report messages (averaging 2.05 minutes per message) would translate to 204,557 hours of work. This highlights the potential time savings that machine learning could bring. This translates into approximately 8523.217days, that is; 23 years, 128 days assuming that the counsellors worked 24/7. Going by International Labour Standards (ILO) [117] on working hours, which is 48 hours a week, categorizing all the 5,987,040 would take approximately 82 years.

The research brought to light the challenging and time-consuming nature of classifying a humongous quantity of SMS texts through the current manual method. The outcomes suggest that this undertaking would demand an impractical amount of time, even with the counsellors working tirelessly. The limitations of the manual approach expose the need for more efficient and accurate U-report SMS text categorization. Machine learning algorithms offer a promising solution, potentially accelerating and improving the entire process.

This analysis of the current approach confirms the achievement of our first objective: to understand and identify the model currently used for categorizing information into key thematic areas on the U-report platform.

4.3 DATA SAMPLING AND LABELLING

This section outlines the results of the data sampling and labeling process for the Zambia U-report SMS messages. As highlighted in Chapter 3, the labelling approach was such that it involved training the researcher on how to label the messages and later measuring the researcher's ability to label the messages like the U-report counsellors who are the domain experts in order to establish that the training indeed took place before the researcher could continue labelling more messages to reduce on the labelling time in the data preparation process.

To evaluate the researcher's ability to categorize messages compared to the domain experts, a set of 1,770 messages was co-labeled by both parties. Separate machine learning models were then trained on each labeled dataset, and their performance was analyzed. This assessment ensured the researcher's labeling accuracy before proceeding with labeling additional messages.

Additionally, the ability of the researcher to accurately and consistently annotate SMS texts was compared to that of the counselors (domain experts). The results, shown in Table 4.2, demonstrate that the researcher's annotations were consistent with the experts. Indicating that training significantly enhanced the researcher's capability to label U-report text messages.

Table 4.2: Assessing the researcher's proficiency in labeling text messages to match the accuracy of domain experts following training

ALGORITHM	**ACCURACY**		
	EXPERT 1	**EXPERT 2**	**RESEARCHER**
K-Nearest Neighbor	0.487759	0.487759	0.463277
Decision Tree	0.525424	0.521657	0.527307
Multinomial Naive Bayes	0.514124	0.506591	0.483992
Support Vector Classification (SVC)	0.563089	0.568738	0.553672
Support Vector Machines (SVM)	**0.581921**	**0.587571**	**0.580038**
Random Forest Classifier	0.548023	0.572505	0.542373
Stochastic Gradient Descent (SGD) Classifier	0.574388	0.566855	0.563089

We tested the SVM model's performance on a dataset of messages already categorized by experts for a 2020 NAC report. The aim of this evaluation was to ascertain whether the model could categorize messages as accurately as domain experts. The model's performance was evaluated using a dataset of U-report messages from July 1st to September 30th, 2020. This dataset included a new topic, "Covid-19," which wasn't part of the original training data. Figure 18 illustrates how closely the model's predictions for various topics (condoms, testing, prevention, STIs, etc.) aligned with the manual categorizations made by experts on the same dataset. This consistency is further explored in Figure 19.

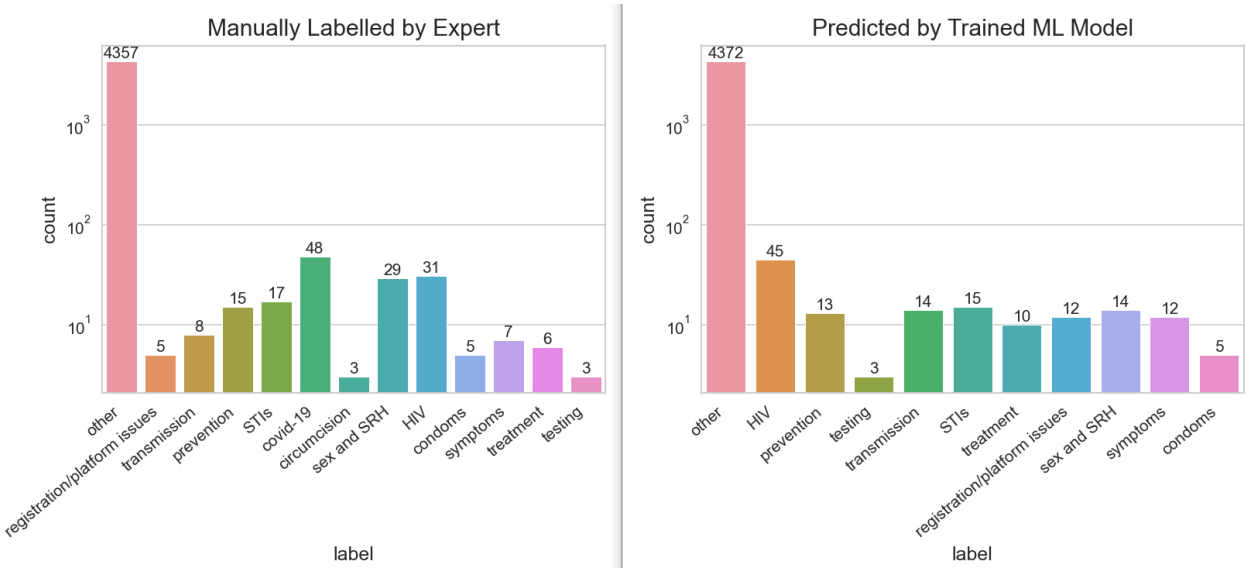


Figure 18:Manually labelled results vs prediction results from machine learning model trained on researcher-labeled dataset.

message	predictions	expert label
thank	0	other
hiv spread	3	HIV
question covid going prevent infiniteve disease	6	prevention
went first covid said didnt know epectmy turm finaly came medical person got smear noseyou imagine suppriseand occuri went chelstone clinic something ever covid test mandatoryi found interestingwill touch paulphiri	11	covid-19
yet circumcised se one use comdoms contract hiv aids	12	transmission
thanks bt u stop invit us workshop u report zambia need learn alot hiv aids stis gbv se reproductive health	8	sex and SRH
yearscan male se	1	registration/platform issues
afternoon causes tiny pimple like bubbles around dick head weeks circumcisionnote emergency	9	STIs
signs siferis sti	5	symptoms
allowed use two condoms sex	2	condoms

LEGEND

Cervical cancer – 0
Circumcision – 1
Condoms – 2
HIV – 3
Masturbation – 4
Other – 5
Prevention – 6
Registration/platform issues – 7
Sex and SRH – 8
STIs – 9
Symptoms – 10
Testing – 11
Transmission – 12
Treatment – 13

Figure 19: Comparison between Manually Assigned Labels by Domain Experts and Researcher-Labeled Training Data Predictions.

In general, these results emphasize the efficacy of the training in enabling the researcher, a non-expert annotator, to attain a level of consistency similar to that of domain experts in labeling text messages. This result is significant for text classification tasks. It suggests that trained non-experts, in addition to domain experts, can effectively contribute to labeling data. This combined approach has the potential to streamline the process and reduce the financial burden of developing machine learning models for text classification.

4.4 FINAL CLASSIFIER TRAINING AND SELECTION

This section evaluates the prediction accuracy of each algorithm trained during the selection process for the final model. The model development involved choosing a suitable algorithm and training it with a preprocessed dataset. This study explored several machine learning algorithms well-suited for text classification tasks. Figure 20 illustrates the definitive model training and selection process. We utilized a training dataset consisting of 50,877 U-report messages that were collaboratively labeled by the researcher and SRH experts. The resulting dataset is depicted in Figure 21.

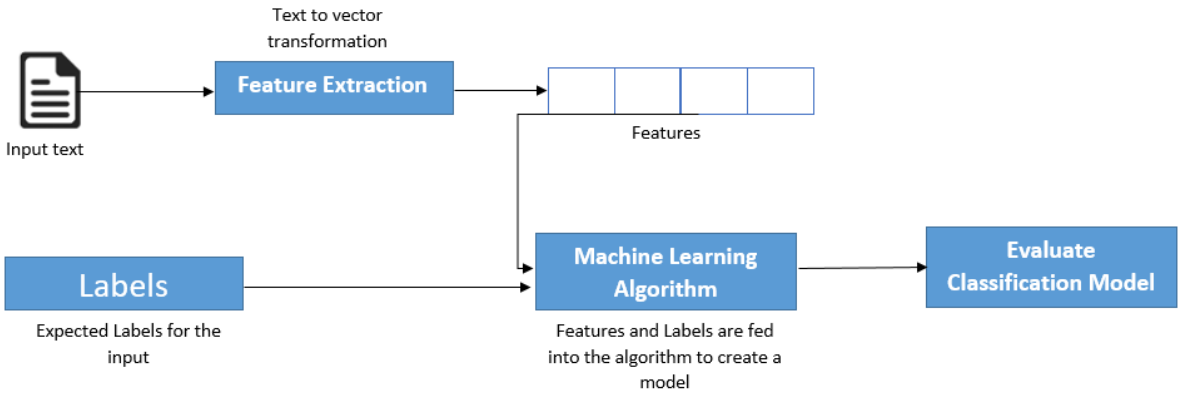


Figure 20: Process for Training and selecting the model (described in chapter 3).

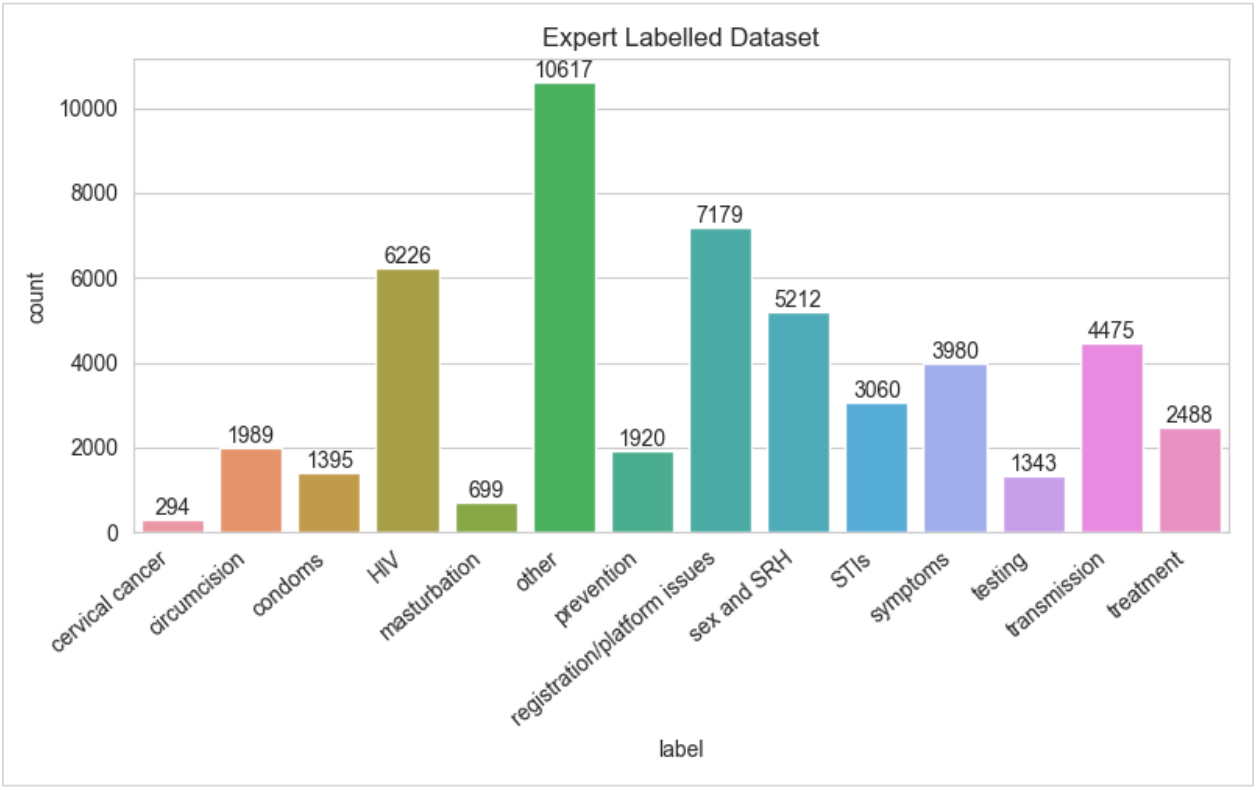


Figure 21: Annotated dataset.

Our primary objective was to build a text classifier that accurately sorts U-report text-based data into their designated thematic categories. We evaluated the predictive accuracy of various machine learning algorithms (K-Nearest Neighbor, Decision Tree, etc. – as mentioned earlier). The algorithm with the strongest performance was chosen as the final model for classifier. Figure 22

illustrates the evaluation of each algorithm's prediction accuracy, which guided the selection process. Table 4.3, presented later, summarizes these results for easy comparison.

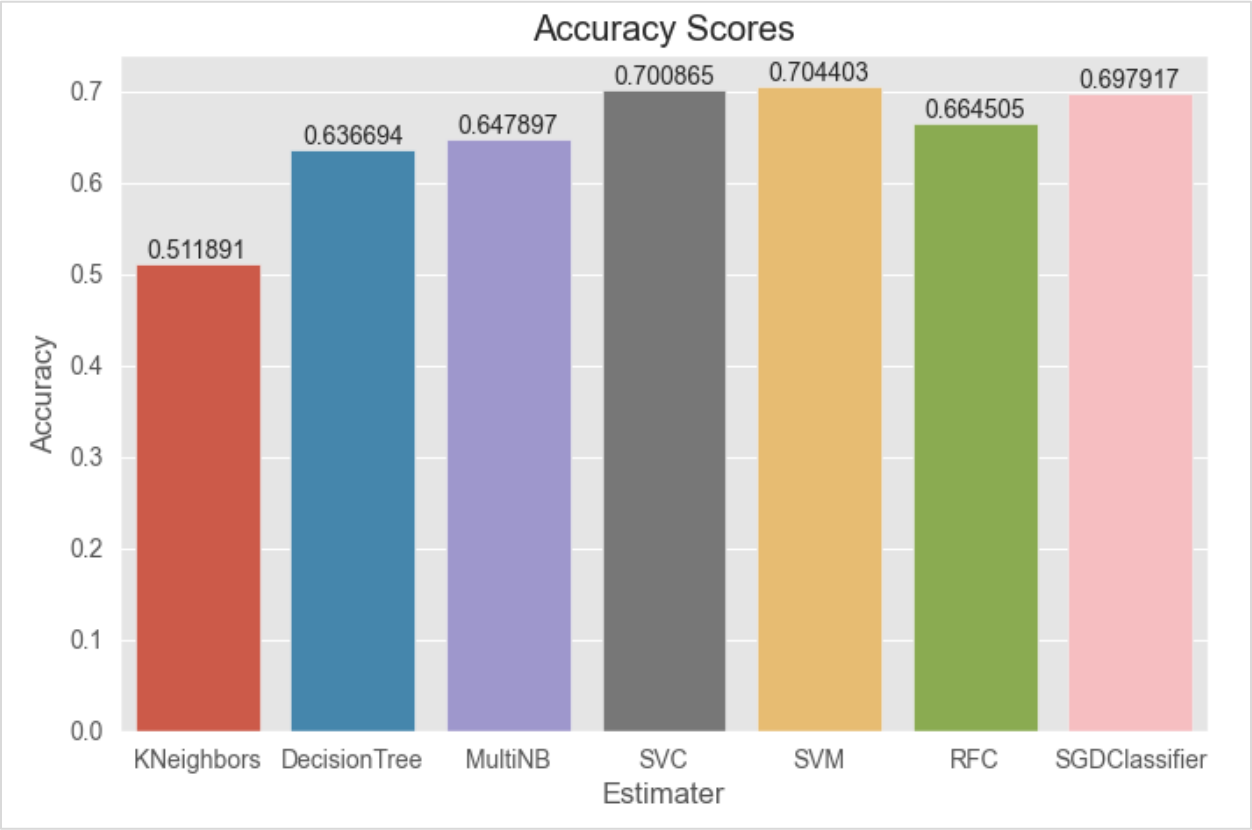


Figure 22: Algorithm Prediction Accuracy Evaluation

Table 4.3: Algorithm Prediction Accuracy Comparison

ALGORITHM	**ACCURACY**
K-NEAREST NEIGHBOR	51.2%
DECISION TREE	63.7%
MULTINOMIAL NAIVE BAYES	64.8%
SUPPORT VECTOR CLASSIFICATION (SVC)	70.1%
SUPPORT VECTOR MACHINES (SVM)	**70.4%**
RANDOM FOREST CLASSIFIER	66.5%
STOCHASTIC GRADIENT DESCENT (SGD) CLASSIFIER	69.8%

Our evaluation identified SVM (Support Vector Machine) as the most effective algorithm for classifying U-report messages, achieving a 70.4% accuracy rate (as shown in Table 4.3). This

aligns with SVM's known efficiency in text classification tasks. While Decision Tree (63.7%) and Multinomial Naive Bayes (64.8%) also performed well, K-Nearest Neighbor (51.2%) proved less suitable. Based on this analysis, SVM was chosen as the final classification model.

4.5 THE TEXT CLASSIFICATION PROTOTYPE

The final Support Vector Machines (SVM) model, along with the trained TfidfVectorizer, was saved to disk for future use, ensuring the preservation of its state and accessibility for subsequent analysis or deployment. By saving both the model and vectorizer, the prototype enables easy reuse of the trained components.

The prototype was designed to be accessible through the command line interface (CLI), allowing users to execute the entire process, including model training, testing, and prediction, using simple command-line commands and integrated development environments (IDE). The design ensures that the model can be easily deployed and utilized in various environments, including production systems. The full source code for the Prototype has been included in this dissertation as **Appendix B**.

4.6 FINAL MACHINE LEARNING MODEL PERFORMANCE ANALYSIS

This section dives into the final model's performance. It provides a detailed report that evaluates the precision, recall, F1 score, and support metrics for each classified category. We also examine the model's predictions and compare the time efficiency of machine learning versus manual categorization for U-report messages. A confusion matrix is included to visualize the model's performance. Finally, the section explores the significant time savings and efficiency gains offered by the machine learning classifier, highlighting its advantages in comparison with the manual approach. This analysis provides valuable insights into the model's effectiveness in accurately and efficiently classifying U-report SMS messages.

Table 4.4 presents a detailed categorization report for the SVM model, identified as the best model for classifying U-report messages into multiple topics (see previous section for details on model selection). This table analyzes various metrics for each category, including precision, recall, and F1-score. Additionally, it provides the overall accuracy, macro-average, and weighted-average.

Below is a breakdown of the key metrics used:

- **Precision:** This metric reflects the proportion of correctly categorized messages out of all messages the model classified into a specific category. In simpler terms, it signifies the proportion of true positive predictions relative to all positive predictions made for that category [118].
- **Recall:** Recall measures the fraction of messages accurately identified by the model in a specific category, in relation to the total number of messages that truly belong to that category. It essentially measures how well the model finds all the relevant messages for a specific category [118].
- **F1-score:** The F1-score provides a balanced perspective by taking into account both precision and recall. It represents the harmonic mean of these two metrics, delivering a singular score that reflects both dimensions of the model's performance [119].

The table also features macro-average and weighted-average metrics. The macro-average computes the unweighted mean of performance metrics across all categories, treating each category with equal importance [122]. Weighted-average takes into account the number of messages in each category when calculating the mean, giving more weight to categories with a larger number of instances [122].

The report details various categories including HIV, transmission, symptoms, registration/platform issues, circumcision, sex and SRH, other, STIs, prevention, condoms, cervical cancer, treatment, masturbation, and testing. For each of these categories, the report provides the number of instances (messages) associated with each category, along with precision, recall, and F1-score metrics for each.

Table 4.4: Best Model Analysis Report

CATEGORY	**PRECISION**	**RECALL**	**F1-SCORE**	**SUPPORT**
HIV	0.51	0.62	0.56	60
Transmission	0.80	0.73	0.76	384
Symptoms	0.63	0.79	0.70	256
Registration/platform issues	0.66	0.72	0.69	1237
Circumcision	0.81	0.61	0.69	149
Sex and SRH	0.67	0.82	0.74	2167

Other	0.65	0.58	0.61	345
STIs	0.94	0.81	0.87	1497
Prevention	0.66	0.59	0.62	1031
Condoms	0.63	0.59	0.61	643
Cervical cancer	0.66	0.62	0.64	793
Treatment	0.68	0.65	0.66	266
Masturbation	0.66	0.62	0.64	848
Testing	0.72	0.61	0.66	500
Accuracy			0.70	10176
Macro average	0.69	0.67	0.68	10176
weighted average	0.71	0.70	0.70	10176

The report reveals a promising performance for the SVM model. It reached an overall accuracy of 70%, successfully classifying seven out of every ten U-report messages into their appropriate categories. Furthermore, the macro-average and weighted-average F1-scores of 68% and 70%, respectively, indicate consistent performance across the various message categories.

The category with the highest precision score was STIs (94%). This suggests the model excelled at accurately identifying messages related to sexually transmitted infections. The category with the highest recall score was sex and SRH (82%), indicating that the model was able to identify a high percentage of messages related to sex and SRH out of all the messages that truly belonged to this category in the dataset.

In addition to the categorization report (Table 4.4), a confusion matrix (Figure 23) is provided for a visual depiction of the model's performance. This matrix breaks down the model's predictions for each category, showing how many messages were correctly categorized (true positives and true negatives) and how many were incorrectly classified (false positives and false negatives) [123]. Generally, a higher number of true positives and negatives indicates stronger model performance. Figure 23 provides a detailed view of the SVM model's performance across different categories, enabling a more thorough examination of its strengths and weaknesses. This breakdown is crucial for calculating metrics like precision, recall, and F1-score, which were previously discussed.

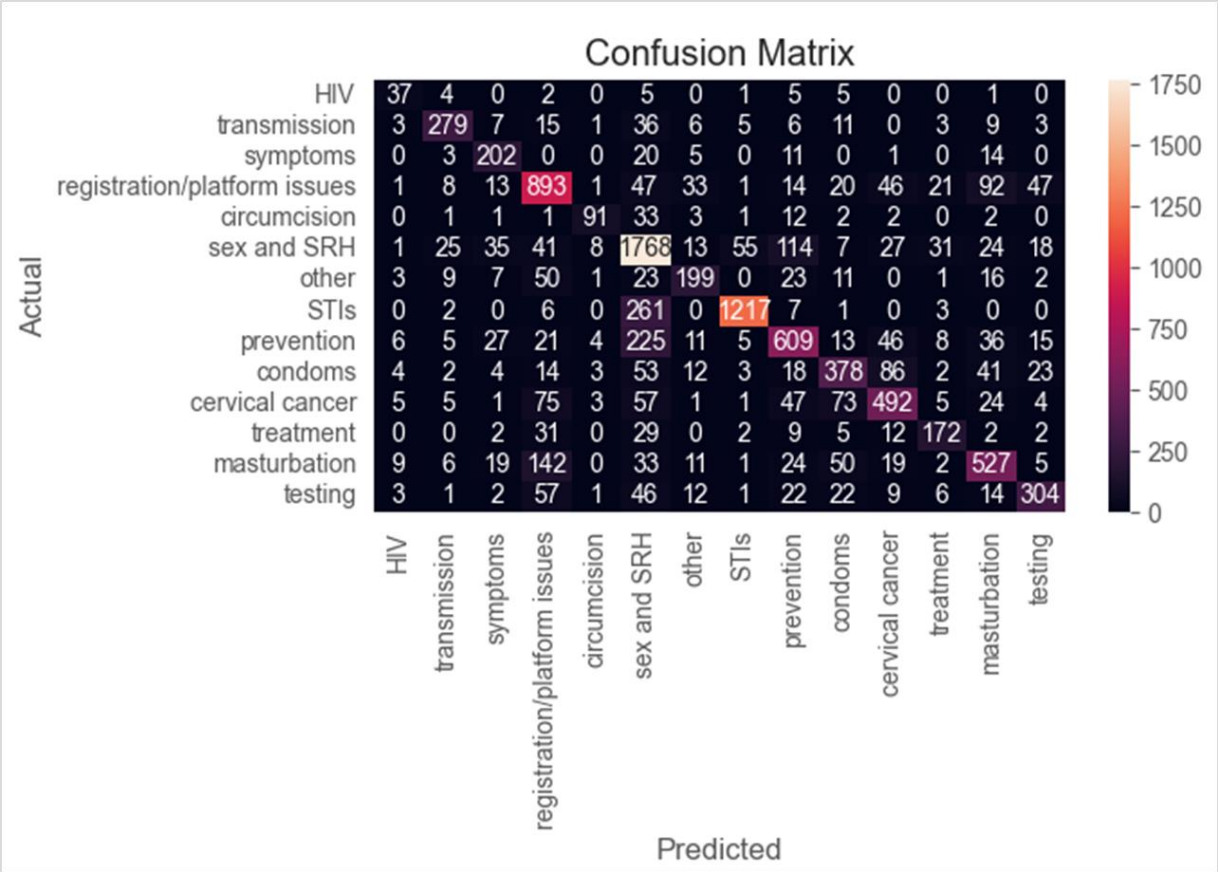


Figure 23: Confusion matrix

The Figure depicts this confusion matrix, which analyzes the categorization of messages for the 14 thematic areas examined in this study. The rows of the matrix represent the actual categories to which messages belong, while the columns represent the categories the model predicted the messages belonged to. Each cell within the matrix shows how many messages were assigned to a specific category by the model.

By delving deeper into this matrix, we can determine four key metrics that provide a more granular understanding of the model's strengths and weaknesses. True positives represent messages that truly belong to a category, and the model correctly classified them as such. Conversely, true negatives are messages that don't belong to a category, and the model accurately identified them that way. On the other hand, false positives are messages that the model incorrectly assigned to a category they don't belong to, and false negatives are messages that truly belong to a category but were missed by the model and assigned to a different category. Examining these counts within the

confusion matrix allows for a more comprehensive evaluation of the model's performance and helps identify areas for potential improvement in future iterations.

Both the detailed categorization report and the confusion matrix provide strong evidence for the SVM model's effectiveness in classifying U-report messages into their designated thematic areas. The model achieved a commendable overall accuracy of 70%, demonstrating consistent performance across the various message categories.

The SVM model's generalizability was evaluated using a fresh dataset of 206,625 U-report messages. This new dataset, spanning November 2012 to July 2019, was independent of the data used to train the model. Figure 24 illustrates the category distribution within this new dataset.

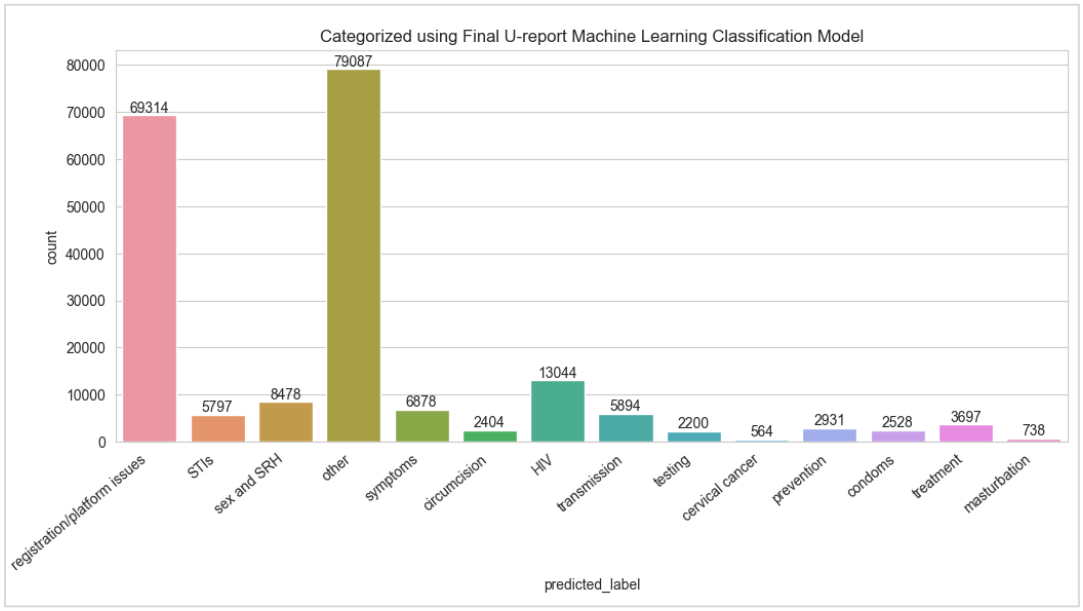


Figure 24: Category Distribution in New SMS Dataset (n=206, 625).

Additionally, Figure 24 organizes the resulting categories by year, displaying how messages are distributed across categories annually throughout the specified period.

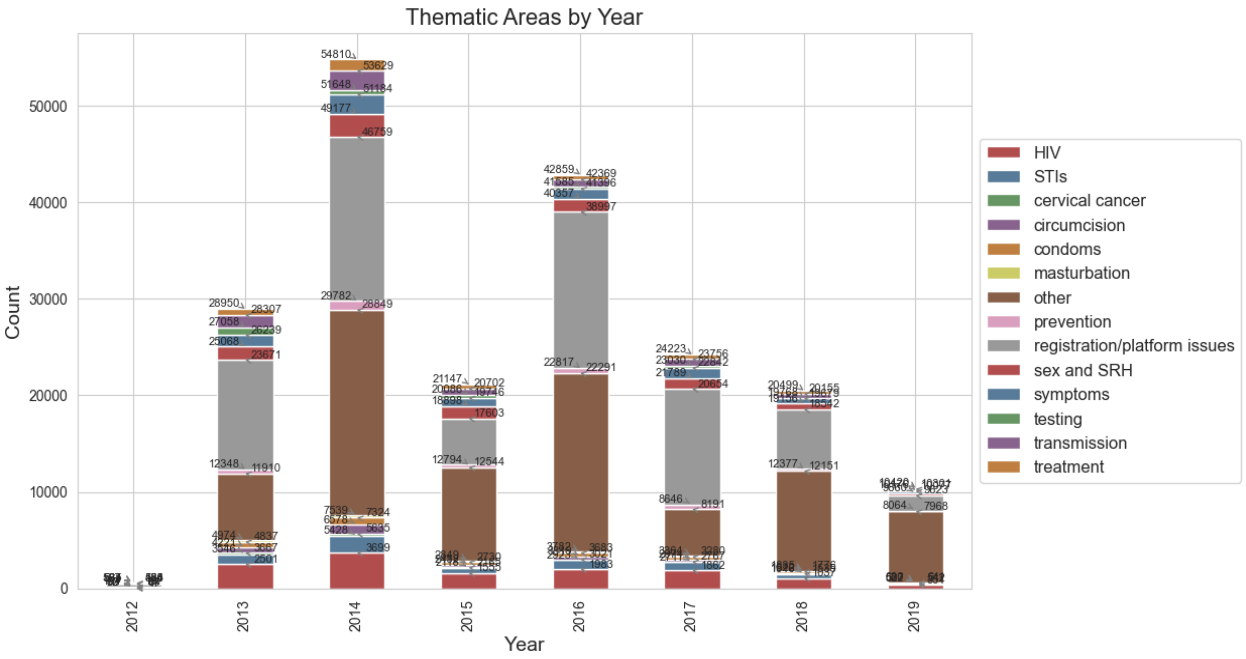


Figure 25: Breakdown of categorized U-report SMSs by year

Manually classifying the massive dataset of 206,625 SMS messages would take 2.82 years if annotators worked according to international labor standards and calculations presented earlier. In contrast, the SVM model, once trained, accomplished this task in 6.4 minutes. This translates to a processing speed of 18.5 milliseconds per message. Even for significantly larger datasets, like the 5,987,040-message set, the automated model would process the messages exceptionally in 3.1 hours. This huge reduction in processing time compared to manual methods highlights the SVM model's superior speed and scalability for categorizing U-report SMS messages.

In addition to being faster, the SVM model offers a crucial advantage: consistent categorization. Unlike manual categorization, which can be prone to human error and variability, the model delivers consistent results. This consistency, coupled with its exceptional processing speed, renders the trained SVM model a significantly more efficient and effective solution for classifying large sets of unstructured text data.

4.6 SUMMARY

This chapter explores the creation, deployment, and assessment of an automated U-report message classification model. Designed to improve the classification of incoming text messages from subscribers to the U-report SMS platform according to predefined thematic areas, this model

leverages text mining techniques and machine learning algorithms. By automating this process, the model aims to significantly improve data analysis efficiency and unlock deeper insights from the wealth of U-report data.

The chapter begins by examining the drawbacks of the existing manual categorization process, highlighting its time-intensive nature and the possibility of inconsistencies, which underscores the necessity for more efficient methods like machine learning algorithms. It then discusses the data preparation process, which includes evaluating the researcher's ability to accurately and consistently label text messages after receiving training as an additional annotator. Furthermore, the chapter compares the manually labeled dataset with the predictions made by the trained SVM classifier to assess its accuracy and effectiveness.

Various machine learning algorithms were evaluated during the training phase, with the Support Vector Machine (SVM) emerging as the optimal choice due to its impressive 70.4% accuracy. The SVM's performance was then thoroughly analyzed using a categorization report that included metrics like precision, recall, F1-score, and support. To further enhance understanding, a confusion matrix was employed to provide a visual depiction of how effective the model is.

To assess the model's generalizability, a new dataset of 206,625 U-report messages was used. The model not only performed well on this unseen data but significantly outperformed the manual categorization process. While manual categorization would have taken a staggering 2.82 years, the automated model completed the task in a mere 6.4 minutes. This remarkable speed demonstrates the model's exceptional efficiency and scalability for handling large volumes of U-report SMS messages.

This chapter presented the development, implementation, and evaluation of a machine learning model for classifying U-report SMS messages. The model attained a high accuracy rate of 70.4% and markedly surpassed manual categorization in terms of speed and consistency. The chapter also explored the use of trained non-expert annotators for labeling tasks during model training. This approach offers a valuable strategy for improving the efficiency and cost-effectiveness of developing text classification models, ultimately leading to better analysis and interpretation of large, unstructured datasets like U-report SMS messages.

CHAPTER FIVE:

CONCLUSION AND RECOMMENDATIONS

This research demonstrates that supervised machine learning offers a practical and efficient solution for automating SMS message classification on the Zambia U-report platform. The developed machine learning model achieved a prediction accuracy of 70.4% for thematic areas, and it processed data much faster than the existing manual method. The findings demonstrate that employing supervised machine learning can drastically reduce the time required to categorize U-report SMS text messages. Tasks that would ordinarily take 82 years to complete manually can now be accomplished in just 3 hours using the model.

This research offers the following key findings and recommendations:

1. **Proof of Concept**: This study serves as a proof of concept, demonstrating the effectiveness of supervised machine learning in automating the categorization of the vast collection of SMS messages on the Zambia U-report platform. This approach significantly improves the categorization process, enabling faster analysis of text messages.
2. **Model Performance:** The final machine learning model achieved a promising accuracy of 70.4% in classifying thematic areas for Zambian U-report SMS messages. This, combined with the significant improvement in processing speed compared to manual methods, highlights the immense potential of machine learning for automating text classification tasks.
3. **Reduced Categorization Time:** This study demonstrates the power of supervised machine learning in reducing processing time for U-report message categorization, as evidenced by the reduction from 82 years to just 3 hours in this study. This substantial decrease in processing time can enable staff at NAC to dedicate more time to higher-value tasks and make quicker decisions based on the insights derived from the categorized data.
4. **Continuous Model Improvement:** For potentially further improvement in the machine learning model's performance, ongoing retraining efforts and the continual expansion of the training dataset are recommended. This approach will enable the model to more effectively recognize new patterns and trends in incoming text messages, resulting in more precise categorizations.

5. **Future Work:** As a subsequent step, it is necessary to leverage the model to create a software application that includes a graphical user interface. This application will lead to a more efficient Zambia U-report platform overall, allowing for faster data processing and improved functionality.

This study demonstrates the effectiveness of supervised machine learning in automating message categorization for the Zambia U-report platform. By continuously refining the model and integrating it into a user-friendly software application, NAC can leverage machine learning's potential to streamline data analysis and support decision-making.

The study aimed to automatically classify the textual data on U-report. This goal was met by first learning the current processes and appreciating its bottlenecks, developing a prototype that automates the current process and assessing the potential improvements made in terms of efficiency and speed by the automated process. The study ultimately addresses the challenges of the current hectic and time-consuming manual categorization process

REFERENCES

[1] Planned Parenthood Association of Zambia, "Passport to Health," 2018.

[2] World Health Organization Africa, "Report on the regional meeting to take stock of the progress made in adolescent Sexual and Reproductive Health and Rights, in the 20 years since the International Conference on Population and Development, and on the Opportunities and Challenges in moving the agenda forward," Visualizing the Problems and Generating Solutions for Adolescent Health in the African Region, Congo Brazzaville, 2015, p. 12. Available: https://www.afro.who.int/sites/default/files/2018-05/ASRH-%20AFRO%20-%20AH%20workshop%20report.pdf

[3] Ministry of Health, *Adolescent Health Strategic Plan 2011 to 2015*, Lusaka Zambia, 2011. Available: https://zambia.unfpa.org/sites/default/files/pub-pdf/ZambiaAdolescentHealthStrategicPlan2011-2015.pdf

[4] UNDP, National HIV/AIDS/STI/TB Council of Zambia, *Zambia: Legal Environment Assessment for HIV, TB and Sexual and Reproductive Health & Rights*," Lusaka Zambia, p. 58. 2019. Available: https://www.nac.org.zm/sites/default/files/publications/Zambia%20Legal%20Environment%20Assessment%20for%20HIV%2C%20TB%20and%20Sexual%20and%20Reproductive%20Health%20%26%20Rights%20-%20UNDP%20in%20Zambia.pdf

[5] UNICEF Zambia, "PROMISING PRACTICE: UNICEF Zambia U Report SMS pilot", 2012.

[6] Unicefstories, "Revolutionizing HIV response among adolescents in Zambia through the use of mobile phones," 2014. Available: http://unicefstories.org/2014/01/02/revolutionizing-hiv-response-among-adolescents-in-zambia-through-the-use-of-mobile-phones/

[7] Unicefstories, "UReport providing counseling services on HIV and STIs," 2014. Available: http://unicefstories.org/2014/05/16/ureport-providing-counseling-services-on-hiv-and-stis/

[8] Zambia U-Report Database December 2012 to July 2019. Accessed on: Jul., 20, 2019. [Online]. Available: https://ureport-backup.s3-eu-west-1.amazonaws.com/1563073201+-+Sunday+14+July+2019+%40+0300/1563073201+-+Sunday+14+July+2019+%40+0300+-+rapidsms.sql.gz.

[9] V. Gupta and G. S. Lehal, "A Survey of Text Mining Techniques and Applications." *Journal of Emerging Technologies in Web Intelligence,* Vol. 1, No. 1, Aug 2009.

[10] N. Yogapreethi and S.Maheswari, "A Review on Text Mining in Data Mining." *International Journal on Soft Computing (IJSC)* Vol.7, No. 2/3, Aug., 2016.

[11] M. A. Hall, "Correlation-based feature Selection for Machine Learning", Ph.D dissertation, Dept. of Computer Sc., Univ. of Waikato, Hamilton, 1999. Accessed on: Feb., 10, 2019. [Online]. Available:https://www.lri.fr/~pierres/donn%E9es/save/these/articles/lpr-queue/hall99correlationbased.pdf

[12] V. Korde and C. N. Mahender, "Text Classifications and Classfiiers – A Survey", *IJAIA*, Vol. 3, No. 2, Mar., 2012.

[13] D. L. Poole and A. K. Mackworth, "What is Artificial Intelligence?" in *Artificial Intelligence: Foundations of Computational Agents*, 2nd ed., Cambridge University Press, 2017, pp. 9.

[14] Y. Duan, J. S. Edward and Y. K. Dwivedi, "Artificial intelligence for decision making in the era of Big Data – evolution, challenges and research agenda" *International Journal of Information Management,* Vol. 48, 14 Jan., pp. 63-71, 2019.

[15] M. Allahyari et al., "A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques," 28 Jul., 2017. Accessed: Apri1 20, [Online] Avalaible: arXiv:1707.02919v2

[16] S. Dang and P. H. Ahmad, "A Review of Text Mining Techniques Associated with Various Application Areas", *IJSR* Vol. 4, No. 2, Feb., 2015.

[17] C. D. Manning, P. Raghavan and H. Schutze, "Boolean retrieval" in *Introduction to Information Retrieval*. New York: Cambridge University Press, 2008, pp. 1 – 17

[18] *What is INFORMATION RETRIEVAL? What does INFORMATION RETRIEVAL mean?*, Jan. 20, 2017. Accessed on: Mar. 17, 2020. [Video file]. Available: https://www.youtube.com/watch?v=Y0CZmsel5Rs

[19] S. Ceri, A. Bozzon, M. Brambilla, E. D. Valle, P. Fraternali and S. Quarteroni. "The Information Retrieval Process" in *Web Information Retrieval*. Berlin, Heidelberg: Data-Centric Systems and Applications. Springer, 2013.

[20] J. R. Hobbs and E. Rilo. "Information Extraction" in *Handbook of Natural Language Processing*. 6000 Broken Sound Parkway NW, Suite 300 Boca Raton, FL 33487-2742: Chapman & Hall/CRC Taylor & Francis Group, 2010 pp. 511 - 532.

[21] D. C. Wimalasuriya and D. Dou, "Ontology-based information extraction: An introduction and a survey of current approaches", *Journal of Information Science (JIS)* Vol. 36, No.3, 19 Mar., 2010, pp. 306 - 323.

[22] S. J. Russell and P. Norvig, "Information Extraction" in *Artificial Intelligence: A Modern Approach*, 3rd ed., Edinburgh Gate Harlow Essex CM20 2JE England: Pearson Education Limited pp. 873-882, 2016.

[23] A. Kaushik and S. Naithani, "A Comprehensive Study of Text Mining Approach", *IJCSNS* Vol.16, No.2, Feb., 2016.

[24] G. Wei, X. Gao and S. Wu, "Study of text classification methods for data sets with huge features," In Proc. 2nd International Conference on Industrial and Information Systems, 2010.

[25] M. M. Mironczuk and J. Protasiewicz "A Recent Overview of the State-of-the-Art Elements of Text Classification," *Expert Systems with Applications*, 2018, Accepted Manuscript, doi: 10.1016/j.eswa.2018.03.058

[26] altexsoft, *Labelling approaches*, 29 Mar., 2018, Accessed: 10 February 2019. [online]. Available: https://www.altexsoft.com/blog/datascience/how-to-organize-data-labeling-for-machine-learning-approaches-and-tools/

[27] M. W. Berry and J. Kogan, "Data preprocessing" in *Text Mining: Applications and Theory*, The Atrium, Southern Gate, Chichester West Sussex, PO19 8SQ, United Kingdom: John Wiley & Sons Ltd, pp. 45-56, 2010.

[28] Alsmadi, I., Hoon, G.K. Term weighting scheme for short-text classification: Twitter corpuses. *Neural Computing & Applications*, Vol. 31, 9 Jan., 2018, pp. 3819-3831, Available: https://doi.org/10.1007/s00521-017-3298-8

[29] T. Pavlenko, "On feature selection, curse-of-dimensionality and error probability in discriminant analysis," *Journal Statistical Planning Inference*, Vol.115, No.2, 2003 pp. 565-584

[30] E. Chavez, G. Navarro, "Probabilistic proximity search: Fighting the curse of dimensionality in metric spaces," *Information Processing Letters*, Vol. 85, No.1, 2003, pp. 39-46

[31] V. Pestov, "On the geometry of similarity search: dimensionality curse and concentration of measure," *Information Processing Letters*, Vol. 73, No.1, 2000, pp. 47-51

[32] D. Pandaya, R. C. Amorimb and P. Lanea, "Feature weighting as a tool for unsupervised feature selection," *Information Processing Letters*, Vol. 129, Jan., 2018, pp. 44-52

[33] K. Lillywhite, D. Lee, B. Tippetts and J. Archibald, "A feature construction method for general object recognition," *Pattern Recognition*, Vol. 46, No.12, Dec., 2013, pp. 3300-3314

[34] M. F. Zafra, *Towards Data Science*, June 16, 2019. Accessed on: Nov. 18, 2019. [Online]. Available on: https://towardsdatascience.com/text-classification-in-python-dd95d264c802

[35] A. Montejo-Raez, "Automatic Text Categorization of documents in the High Energy Physics domain", M.S thesis, Universidad de Granada, 2005. Accessed on: October, 7, 2019. [Online]. Available: https://hera.ugr.es/tesisugr/15903837.pdf

[36] J. Bell, "What is machine learning?" in *Machine Learning: Hands-On for Developers and Technical Professionals*, Hoboken, New Jersey: John Wiley & Sons, Inc., pp. 1 – 16, 2015

[37] D. Fumo, *Types of Machine Learning Algorithms You Should Know*, 15 Jun., 2017. Accessed: Nov. 4 2019. [online]. Available: https://towardsdatascience.com/types-of-machine-learning-algorithms-you-should-know-953a08248861

[38] A. Lundborg, "Text classification of short messages", M.S. thesis, Dept. of Computer Sc., Lund Univ., Scania, 2017. Accessed on: Jan. 18, 2020. [online]. Available: http://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=8928009&fileOId=8928011

[39] S. Bansal, *A Comprehensive Guide to Understand and Implement Text Classification in Python*, 23 Apr., 2013. Accessed: 10 November 2019. [online]. Available: https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/

[40] I. Rish, *An empirical study of the naive Bayes classifier*, Accessed: 30 October, 2019. [online]. Available: https://www.cc.gatech.edu/~isbell/reading/papers/Rish.pdf

[41] *Naïve Bayesian* [online]. Available: https://www.saedsayad.com/naive_bayesian.htm

[42] P. Chandrasekar and K. Qian, "The Impact of Data Preprocessing On the Performance of Naïve Bayes Classifier," In Proc. IEEE 40th Annual Computer Software and Applications Conference, 2016

[43] L. Jianga, Z. Caia, H. Zhang and Dianhong WangcNaive, "Bayes text classifiers: a locally weighted learning approach," *Journal of Experimental & Theoretical Artificial Intelligence*, 2012

[44] L. Serrano, *Naive Bayes classifier: A friendly approach*, Feb. 10, 2019. Accessed on: Mar. 28, 2020. [Video file]. Available: https://www.youtube.com/watch?v=Q8l0Vip5YUw

[45] L. Jiang, Z. Cai, H. Zhang and D. Wang, "Naive Bayes text classifiers: a locally weighted learning approach," *Journal of Experimental & Theoretical Artificial Intelligence*, Vol. 25, No. 2, Jun., 2013, pp.273-286

[46] W. S. Noble, "What is a support vector machine?," *Nature Biotechnology*, Vol.24, No.12, 2006, pp.1565-1567

[47] C. Campbell and Y. Ying, "Learning with Support Vector Machines" in *Synthesis Lectures on Artificial Intelligence and Machine Learning*, Morgan & Claypool Publishers, 2011, pp.1-21

[48] Joachims, T. "Text categorization with support vector machines: learning with many relevant features". In Proceedings of ECML-98, 10th European Conference on Machine Learning (Chemnitz, DE), 1998, pp. 137–142

[49] C. Mood, "Logistic Regression: Why We Cannot Do What We Think We Can Do, and What We Can Do About It," *European Sociological Review*, Vol. 26, No.1, 2009, pp. 67–82

[50] M. Banerjee, C. Filson, R. Xia and D. C. Miller, "Logic Regression for Provider Effects on Kidney Cancer Treatment Delivery," *Computational and Mathematical Methods in Medicine*, 2014, pp. 1–9.

[51] *Classification with Logistic Regression*, Aug. 27, 2014. Accessed on: Mar. 28, 2020. [Video file]. Available: https://www.youtube.com/watch?v=BrmS1LFw-dQ

[52] H. Park, "An Introduction to Logistic Regression: From Basic Concepts to Interpretation with Particular Attention to Nursing Domain," *J Korean Acad Nurs*, Vol. 43, No. 2, Apr. 2013, pp.154-164

[53] Scikit-learn, *Decision Trees*, Accessed on: Apr. 15, 2020. [Online]. Available: https://scikit-learn.org/stable/modules/tree.html

[54] A. J. Myles, R. N. Feudale, Y. Liu, N. A. Woody and S. D. Brown, "An introduction to decision tree modeling," *J. Chemometrics*, Vol.18, Sep. 2004, pp.275-285

[55] E. Alpaydın, “Decision Trees,” in *Introduction to Machine Learning.*, 2nd ed., The MIT Press, Cambridge, Massachusetts, London, England, 2010, pp. 185 – 208

[56] A. Trabelsi, Z. Elouedi and E. Lefevre “Decision tree classifiers for evidential attribute values and class labels,” *Fuzzy Sets and Systems*, Vol. 366, 1 Jul. 2019, pp. 46-62. Available: https://doi.org/10.1016/j.fss.2018.11.006

[57] Q. Dai, C. Zhang and H. Wu “Research of Decision Tree Classification Algorithm in Data Mining,” *International Journal of Database Theory and Application*, Vol. 9, No. 5, 2016, pp.1-8 Available: http://dx.doi.org/10.14257/ijdta.2016.9.5.01

[58] *Iris Species: Classify iris plants into three species in this classic dataset*, Accessed on: May. 2, 2020. [Online]. Avalaible: https://www.kaggle.com/uciml/iris

[59] R.A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems," *Annals of Human Genetics*, Vol. 7, No. 2, 1936.

[60] K. Collins-Thompson, *Decision Tree Example,* Accessed on: May. 2, 2020. [Online]. Available: https://www.coursera.org/lecture/python-machine-learning/decision-trees-Zj96A

[61] Scikit-learn, *Random Forest Classifier*, Accessed on May. 4, 2020. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[62] *What is Over-fitting?* Accessed on: May. 4, 2020. [Online]. Available: https://corporatefinanceinstitute.com/resources/knowledge/other/overfitting/

[63] S. Glander, *Machine Learning Basics - Random Forest*, Oct. 30, 2018. Accessed on: May 5, 2020. [Online]. Available: https://www.shirin-glander.de/2018/10/ml_basics_rf/?fbclid=IwAR0HTNbbkX0XJ9IOs-19Pycc-D6V1RbOFY-qCAI_kTW4Xl2uxlPhp2v_53U

[64] L. Breiman, “Random Forests,” *Machine Learning*, Vol. 45, 2001, pp. 5–32. Available: https://doi.org/10.1023/A:1010933404324

[65] Wikimedia, *Random Forest Simplified*, Accessed on: May 5, 2020. [Online]. Available: https://commons.wikimedia.org/wiki/File:Random_forest_diagram_complete.png

[66] Y. Chen and Y. Hao, “A Feature Weighted Support Vector Machine and K-Nearest Neighbor Algorithm for Stock Market Indices Prediction,” *Expert Systems With Applications*, 2017 doi: 10.1016/j.eswa.2017.02.044

[67] L. A. Teixeira and A. L. Inácio de Oliveira, “A method for automatic stock trading combining technical analysis and nearest neighbor classification,” *Expert Systems with Applications*, Vol. 37, No. 10, Oct. 2010, pp. 6885–6890

[68] S. Jiang, G. Pang, M. Wu and L. Kuang, “An improved K-nearest-neighbor algorithm for text categorization,” *Expert Systems with Applications*, Vol. 39, 2012, pp.1503–1509

[69] K. Kowsari, K. J. Meimandi, M. Heidarysafa, S. Mendu, L. Barnes and D. Brown, "Text Classification Algorithms: A Survey," *information*, Vol. 10, No. 4, Apr. 2019. Available: doi:10.3390/info10040150

[70] M. Mohammed, M. B. Khan and E. B. M. Bashier, "Introduction to Rule-Based Classifiers," in *Machine Learning: Algorithms and Applications* CRC Press, Taylor & Francis Group, 6000, Broken Sound Parkway NW, Suite 300, Boca Raton, FL, USA, 2017, pp.53-72

[71] D. M. Farid, M. A. Al-Mamun, B. Manderick and A. Nowe, "An adaptive rule-based classifier for mining big biological data," *Expert Systems With Applications* Vol. 64, 2016, pp. 305–316

[72] D. Meese, *Rule-based Classifiers*, Accessed on: May 7, 2020. [Online]. Available: https://slideplayer.com/slide/9610087/

[73] V. K. Vijayan, K.R. Bindu and L. Parameswaran, "A Comprehensive Study of Text Classification Algorithms," In Proc. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017

[74] A. Khan, B. Baharudin, L. H. Lee and K. khan, "A Review of Machine Learning Algorithms for Text-Documents Classification," *Journal of Advances in Information Technology (JAIT)*, Vol. 1, No. 1, Feb. 2010

[75] J. Rocchio, "Relevance feedback in information retrieval," in *The SMART Retrieval System: Experiments in Automatic Document Processing*, Prentice Hall, Inc. 1971

[76] *Rocchio classification*, Accessed on: May 7, 2020. [Online]. Available: https://nlp.stanford.edu/IR-book/html/htmledition/rocchio-classification-1.html

[77] Thamarai Selvi. S, Karthikeyan. P, Vincent. A, Abinaya. V, Neeraja. G and Deepika. R, "Text Categorization using Rocchio Algorithm and Random Forest Algorithm," In Proc. 2016 IEEE Eighth International Conference on Advanced Computing (ICoAC), 2016

[78] M. Negnevitsky, "Artificial Neural Networks," in *Artificial Intelligence*, 2nd ed., Essex, England, 2005, pp. 165 - 213.

[79] R. Manikandan and R. Sivakumar "Machine learning algorithms for text-documents classification: A review," *International Journal of Academic Research and Development (IJARD)* Vol. 3, No. 2, Mar. 2018, pp. 384-389

[80] A. J. C. Trappey, F. Hsu, C. V. Trappey, C. Lin, "Development of a patent document classification and search platform using a back-propagation network", *Expert Systems with Applications*, Vol. 31, 2006, pp. 755–765

[81] simplilearn, *What is Perceptron: A Beginners Tutorial for Perceptron*, Accessed on: May 7, 2020. [Online]. Available: https://www.simplilearn.com/what-is-perceptron-tutorial#:~:text=A%20Perceptron%20is%20an%20algorithm,learn%20only%20linearly%20separable%20patterns

[82] H. T. Ng, W. B. Goh and K. L. Low, “Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization,” In Proc. 20th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 1997, pp. 67-73

[83] M. E. Ruiz and P. Srinivasan, “Automatic Text Categorization Using Neural Network,” In Proc. 8th ASIS SIG/CR Workshop on Classification Research,1998, pp. 59-72

[84] M. Ghiassi, M. Olschimke, B. Moon and P. Arnaudo “Automated text classification using a dynamic artificial neural network model,” *Expert Systems with Applications*, Vol. 39, 2012, pp. 10967–10976

[85] Machine Learnings, *Text Classification using Neural Networks*, Accessed on: May 7, 2020. [Online]. Available: https://machinelearnings.co/text-classification-using-neural-networks-f5cd7b8765c6

[86] G. Brown, “Ensemble Learning” in *Encyclopedia of Machine Learning*, C.Sammut & G.I.Webb (Eds.), Springer Press, 2010

[87] S. Lahmiri, S. Bekiros, A. Giakoumelou and F. Bezzina, “Performance assessment of ensemble learning systems in financial data classification,” *Intell Sys Acc Fin Mgmt*, Vol. 27, No. 1, Jan. 2020

[88] M. A. King, “Ensemble Learning Techniques for Structured and Unstructured Data,” Ph.D dissertation, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, United States, 2015

[89] X. Lu, “Natural Language Processing and Intelligent Computer-Assisted Language Learning (ICALL),” in *The TESOL Encyclopedia of English Language Teaching*, J. I. Liontas, 1st ed, John Wiley & Sons, Inc., 2018

[90] H. Lane, C. Howard and H. M. Hapke, “Natural language vs. programming language,” in *Natural Language Processing: Understanding, analyzing, and generating text with Python*, Shelter Island, NY, USA, Manning Publications Co., 2019, pp. 3-30

[91] D. Maynard, K. Bontcheva and I. Augenstein, “Introduction,” in *Natural Language Processing for the Semantic Web: Synthesis Lectures on the Semantic Web: Theory and Technology*, Morgan & Claypool, Vol. 15, 2017, pp. 1-7

[92] S. B. B. Priyadarshini, A. B. Bagjadab and B. K. Mishra, “A Brief Overview of Natural Language Processing and Artificial Intelligence” in *Natural Language Processing in Artificial Intelligence*, B. K. Mishra and R. Kumar, Palm Bay, Florida 32905, USA, AAP Inc., 2020

[93] Editorial Team, *A Quick Guide to Natural Language Processing (NLP)*, AI and Intelligent Automation, 2019, Accessed on: Jun 15, 2020. [online]. Available: https://www.intelligentautomation.network/learning-ml/articles/a-basic-guide-to-natural-language-processing-nlp

[94] T. Taulli, "Natural Language Processing (NLP): How Computers Talk," in *Artificial Intelligence Basics: A Non-Technical Introduction*, Monrovia, CA., USA, Apress, 2019, pp. 103-124

[95] O. Koshorek, A. Cohen, N. Mor, M. Rotman and J. Berant, "Text Segmentation as a Supervised Learning Task" In Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 2, 2018

[96] S. Chakravarthy, *Tokenization for Natural Language Processing*, towards data science, 2020, Accessed on: Jul 5, 2020. [online]. Available: https://towardsdatascience.com/tokenization-for-natural-language-processing-a179a891bad4

[97] A. G. Jivani, "A Comparative Study of Stemming Algorithms," *IJCTA*, Vol. 2, No. 6, Nov. 2011, pp. 1930-1938

[98] S. Raj, Natural Language Processing for Chatbots, in *Building Chatbots with Python*, Berkeley, CA., USA, Apress, 2019, Available: https://doi.org/10.1007/978-1-4842-4096-0_2

[99] S. Kübler, R. McDonald and J. Nivre, "Dependency Parsing," *Synthesis Lectures on Human Language Technologies*, Vol. 2, No. 1, Jan. 2009, pp. 1-19

[100] A. Sharma, *How Part-of-Speech Tag, Dependency and Constituency Parsing Aid In Understanding Text Data?*, Analytics Vidhya, Accessed on: Jun 15, 2020. [online]. Available: https://www.analyticsvidhya.com/blog/2020/07/part-of-speechpos-tagging-dependency-parsing-and-constituency-parsing-in-nlp/

[101] J. Thanaki, "Feature Engineering and NLP Algorithms," in *Python Natural Language Processing: Explore NLP with machine learning and deep learning techniques*, Birmingham, UK, Packt Publishing, 2017, pp. 102-148

[102] C. Marshall, *What is named entity recognition (NER) and how can I use it?*, super.AI - AI & human data labeling, AI model training & deployment, 2019, Accessed on: Jun 17, 2020. [online]. Available: https://medium.com/mysuperai/what-is-named-entity-recognition-ner-and-how-can-i-use-it-2b68cf6f545d

[103] P. Dwivedi, *NLP: Extracting the main topics from your dataset using LDA in minutes*, towards data science, 2018, Accessed on: Jun 17, 2020. [online]. Available: https://towardsdatascience.com/nlp-extracting-the-main-topics-from-your-dataset-using-lda-in-minutes-21486f5aa925

[104] K. D. Rosa and J. Ellen, "Text Classification Methodologies Applied to Micro-text in Military Chat," In Proc. International Conference on Machine Learning and Applications, 2019

[105] R. C. Balabantaray, M. Mohammad and N. Sharma, "Multi-class twitter emotion classification: A new approach," *IJAIS* Vol. 4, No.1, Sep 2012

[106] W. H. Weng, K. B. Wagholikar, A. T. McCray, P. Szolovits and H. C. Chueh, "Medical subdomain classification of clinical notes using a machine learning-based natural language processing approach," *BMC Medical Informatics and Decision Making*, Vol. 17, No. 1, Dec 2017

[107] S. N. Murphy and H. C. Chueh, “A Security architecture for query tools used to access large biomedical databases,” In Proc. AMIA Symp. 2002, 2002, pp. 552–6

[108] Y. Hayashida, T. Uetsuji, Y. Ebara and K. Koyamada, "Category Classification of Text Data with Machine Learning Technique for Visualizing Flow of Conversation in Counseling," *2017 Nicograph International (NicoInt)*, Kyoto, Japan, 2017, pp. 37-40, doi: 10.1109/NICOInt.2017.35.

[109] C. Poulin et al., “Predicting the Risk of Suicide by Analyzing the Text of Clinical Notes,” *PLoS ONE*, Vol. 9, No. 1, Jan. 2014. doi:10.1371/journal.pone.0085733

[110] B. Koopman et al., “Automatic classification of diseases from free-text death certificates for real-time surveillance,” *BMC Medical Informatics and Decision Making*, Vol. 15, No. 53, Jul. 2015

[111] Sourceforge, *WampServer*, Accessed on: Jul. 7, 2020. [Online]. Available: https://sourceforge.net/projects/wampserver/

[112] Montgomery, Douglas C., George C. Runger, and Norma F. Hubele. "Engineering Statistics." *John Wiley & Sons*, 2018.

[113] MySQL, *Reloading SQL-Format Backups*, Accessed on: Jul. 7, 2020. [Online]. Available: https://sourceforge.net/projects/wampserver/

[114] *An introduction to machine learning with scikit-learn* [online]. Available: https://scikit-learn.org/stable/tutorial/basic/tutorial.html

[115] *Stopwords with NLTK* [online]. Available: https://pythonprogramming.net/stop-words-nltk-tutorial/

[116] Lovins, J.B. "A Comparative Study of Stemming Algorithms for Information Retrieval," *ACM Computing Surveys*, Vol. 4, No. 1, pp. 61-73, 1968.

[117] International Labour Organization, "Convention Concerning the Reduction of Hours of Work to Forty Per Week," 30 June 1930. [Online]. Available: https://www.ilo.org/dyn/normlex/en/f?p=NORMLEXPUB:12100:0::NO::P12100_INSTRUMENT_ID:312175. [Accessed: Feb 1, 2022].

[118] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, Vol. 45, No. 4, pp. 427-437, 2009.

[119] V. Van Asch, "Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews," in Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, 2002, pp. 417-424.

[120] C. D. Manning, P. Raghavan, and H. Schütze, "Introduction to Information Retrieval," Cambridge University Press, 2008.

[121] J. Han, M. Kamber, and J. Pei, "Data Mining: Concepts and Techniques," 3rd ed., Morgan Kaufmann Publishers, 2011.

[122] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” Journal of Machine Learning Research, Vol. 12, pp. 2825–2830, 2011

[123] S. Raschka and V. Mirjalili, "Python Machine Learning," 3rd ed. Birmingham, United Kingdom: Packt Publishing Ltd., 2019, p. 182.

APPENDICES

APPENDIX A: RESEARCH INTERVIEW GUIDE

THE UNIVERSITY OF ZAMBIA
SCHOOL OF NATURAL SCIENCES
COMPUTER SCIENCE DEPARTMENT

RESEARCH INTERVIEW GUIDE

Research Topic: Categorization of Sexual Reproductive Health Short Messages Texts into Thematic Areas Using Text Mining.

Introduction

This document is an interview guide for the collection of data to gain useful insight into the current processes for the classification of SMS messages on the Zambia U-Report platform into key themes. Participants chosen to take part in the interview are psychosocial counsellors hired by the National HIV/AIDS/STI/TB Council for the purpose of providing counselling services to adolescents through the U-report System and have the extra task of manually categorizing the messages into key topics.

All questions in this guide are focused around the first research question for this study, which is; "***How is information currently categorized into key thematic areas on the Zambia U-report system?"***

The guiding questions are to be asked in no particular order and some can be skipped if already covered in earlier responses. More questions that are not part of the guiding list of questions can be asked either as follow-up questions or whenever the interviewer sees the need.

Guiding questions

1. Interview will start with an introduction of who the researcher is and what the research is about. (A detailed background to the study is given)
2. Interviewer asks participant's basic demographic information as a way to break the ice and make the participant feel comfortable as well as for analysis.
3. What is your role at the National HIV/AIDS/STI/TB Council?
   (Ask interviewee to state their job title and state, in general, their duties at NAC in relation to the Zambia U-Report Platform)
4. How does Zambia U-Report relate to the tasks listed in 3?
5. How do you categorize the pool of messages on the Zambia U-Report platform into key thematic areas?
6. What is the purpose of categorizing the messages on the platform into key thematic areas?
7. What criteria is used when choosing the categories (thematic areas)?

8. At what point is a message slotted into a chosen category?
9. How frequent is the categorization of messages done?
   (Is it done hourly, weekly, monthly, quarterly etc)
10. How do you use the information resulting from the categorization of messages?
11. What does the National HIV/AIDS/TB/STI Council (NAC) use the information resulting from the categorization of messages for?
12. What tools, items or materials are used when categorizing the messages into key thematic areas? (probe interviewee to provide a list of these tools, items or materials)
13. What challenges (if any) do you encounter and/or potential dangers do you foresee affecting the categorization process when using the tools/materials/items listed in your earlier response? (refer to the response to question 12)
14. What challenges (if any) do you encounter and/or potential dangers do you foresee affecting the resulting categories when using the tools/materials/items listed in your earlier response? (refer to response to question 12)
15. What is being done or what workaround has been adopted in response to the challenges listed in your earlier response? (refer to responses to questions 13 and 14)
16. How many counsellors perform the task of categorizing the messages into key thematic areas?
17. How are the category tallies from each counsellor aggregated into totals for each category?
18. How does NAC satisfy itself that the categories arrived at are not subjective?

Closing

1. Interviewer asks interviewee if they have any more information to share, questions or comments.
2. Interviewer thanks interviewee for their participation.
3. Interview ends.

APPENDIX B: SOURCE CODE

1. Initial model training

python

```
# Start time for performance tracking

start_time = dt.now()

# Load and prepare the dataset

dataset_path = '1771_labeled_text_messages.csv'

dataset = pd.read_csv(dataset_path, encoding='cp1252')

dataset = dataset[['message', 'label']]

dataset.dropna(subset=["message"], inplace=True)

dataset['category_id'], label_mapping = dataset['label'].factorize()

relevant_labels = ['cervical cancer', 'circumcision', 'condoms', 'HIV', 'masturbation', 'other', 'prevention', 'registration/platform issues', 'sex and SRH', 'STIs', 'symptoms', 'testing', 'transmission', 'treatment']

dataset = dataset[dataset['label'].isin(relevant_labels)]

# Text cleaning function

def sanitize_message(content):

    content = content.lower()

    replace_special_chars = re.compile('[/(){}\[\]\|@,;]').sub(' ', content)

    remove_unwanted_symbols = re.compile('[^0-9a-z #+_]').sub('', replace_special_chars)

    eliminate_digits = re.compile('[\d+]').sub('', remove_unwanted_symbols)

    filtered_words = ' '.join(word for word in eliminate_digits.split() if word not in set(stopwords.words('english')))

    return filtered_words

dataset["message"] = dataset["message"].apply(sanitize_message)

# Visualization of label distribution

sns.set_style('whitegrid')

plt.figure(figsize=(8, 5))

count_plot = sns.countplot(x='label', data=dataset)

count_plot.set_xticklabels(count_plot.get_xticklabels(), rotation=40, ha="right")
```

```
for container in count_plot.containers:
    count_plot.bar_label(container)
plt.tight_layout()
plt.show()
# Shuffle and split the dataset
dataset = dataset.sample(frac=1)
x_train, x_test, y_train, y_test = train_test_split(dataset["message"], dataset["category_id"], test_size=0.2, random_state=0)
# Text vectorization
text_transformer = TfidfVectorizer(sublinear_tf=True, norm='l2', min_doc_freq=10, ngram_span=(1, 2), exclude_stopwords='english')
train_text_features = text_transformer.fit_transform(training_data)
test_text_features = text_transformer.transform(testing_data)
# Model selection and training
classifier_parameters = {
    KNeighborsClassifier(): {
        "neighbor_count": [25, 30, 35, 45],
        "weight_scheme": ['uniform', 'distance'],
        "leaf_dimensions": [25, 30, 35]
    },
    DecisionTreeClassifier(): {
        "split_criterion": ['gini', 'entropy'],
        "node_splitter": ['best', 'random'],
        "tree_depth": [None, 90, 95, 100],
        "feature_selection": [None, "auto", "sqrt", "log2"],
        "seed": [42]
    },
    MultinomialNB(): {
        "prior_fitting": [True, False]
```

```
},
LinearSVC(): {
    "loss_function": ['hinge', 'squared_hinge'],
    "classification_strategy": ['ovr', 'crammer_singer'],
    "intercept_fitting": [True, False],
    "seed": [42],
    "iteration_limit": [900, 1000, 1100]
},
SVC(): {
    'penalty_parameter': [0.1, 1, 10, 100, 1000],
    'smoothing_factor': [1, 0.1, 0.01, 0.001, 0.0001],
    'kernel_type': ['rbf']
},
RandomForestClassifier(): {
    "split_criterion": ['gini', 'entropy'],
    "use_bootstrap": [True, False],
    "tree_depth": [85, 90, 95, 100],
    "feature_selection": ['sqrt', 'log2'],
    "tree_count": [60, 80, 90],
    "seed": [42]
},
SGDClassifier(): {
    "loss_function": ['hinge', 'log', 'perceptron'],
    "regularization": ['l2', 'l1'],
    "learning_rate": [0.0001, 0.0003, 0.0010],
    "stop_early": [True],
    "iteration_limit": [1000, 1500],
    "seed": [42]
```

```
    }
}
# Ignore convergence and future warnings
simplefilter("ignore", category=ConvergenceWarning)
simplefilter(action='ignore', category=FutureWarning)
# Model evaluation
best_accuracy = 0
best_model = None
for model, params in classifier_parameters.items():
    grid_search = GridSearchCV(estimator=model, param_grid=params, cv=3, n_jobs=1)
    grid_search.fit(X_train_vect, y_train)
    predicted = grid_search.predict(X_test_vect)
    accuracy = accuracy_score(y_test, predicted)
    if accuracy > best_accuracy:
        best_accuracy = accuracy
        best_model = grid_search
# Model performance visualization
accuracy_scores = [accuracy_score(y_test, model.predict(X_test_vect)) for model in classifier_parameters]
performance_df = pd.DataFrame({"Model": list(model_params.keys()), "Accuracy": accuracy_scores})
sns.barplot(x='Model', y='Accuracy', data=performance_df)
plt.title('Model Accuracy Comparison')
plt.xticks(rotation=45)
plt.show()
# Runtime calculation
total_runtime_seconds = (dt.now() - start_time).seconds
print(f'Total runtime: {total_runtime_seconds / 60:.1f} minutes')
```

2. Final model training and saving

python

```
# Start time for performance tracking
start_time = dt.now()
# Load and prepare the dataset
dataset_path = 'final_ureport_training_dataset.csv'
dataset = pd.read_csv(dataset_path, encoding='cp1252')[['message', 'label']]
dataset.dropna(subset=["message"], inplace=True)
dataset['category_id'], label_mapping = dataset['label'].factorize()
relevant_labels = ['cervical cancer', 'circumcision', 'condoms', 'HIV', 'masturbation', 'other', 'prevention', 'registration/platform issues', 'sex and SRH', 'STIs', 'symptoms', 'testing', 'transmission', 'treatment']
dataset = dataset[dataset['label'].isin(relevant_labels)]
# Text cleaning function
def sanitize_message(content):
    content = content.lower()
    replace_special_chars = re.compile('[/(){}\[\]\|@,;]').sub(' ', content)
    remove_unwanted_symbols = re.compile('[^0-9a-z #+_]').sub('', replace_special_chars)
    eliminate_digits = re.compile('[\d+]').sub('', remove_unwanted_symbols)
    filtered_words = ' '.join(word for word in eliminate_digits.split() if word not in set(stopwords.words('english')))
    return filtered_words
def get_wordnet_pos(tag):
    tag = tag[0].upper()
    tag_dict = {"J": nltk.corpus.wordnet.ADJ, "N": nltk.corpus.wordnet.NOUN, "V": nltk.corpus.wordnet.VERB, "R": nltk.corpus.wordnet.ADV}
    return tag_dict.get(tag, nltk.corpus.wordnet.NOUN)
dataset["message"] = dataset["message"].apply(sanitize_message)
# Visualization of label distribution
sns.set_style('whitegrid')
plt.figure(figsize=(8, 5))
```

```
label_count_plot = sns.countplot(x='label', data=dataset)
label_count_plot.set_xticklabels(label_count_plot.get_xticklabels(), rotation=40, ha="right")
for container in label_count_plot.containers:
    label_count_plot.bar_label(container)
plt.title('Expert Labelled Dataset')
plt.tight_layout()
plt.show()
# Shuffle and split the dataset
dataset = dataset.sample(frac=1)
x_train, x_test, y_train, y_test = train_test_split(dataset["message"], dataset["category_id"], test_size=0.2, random_state=42)
# Text vectorization
text_transformer = TfidfVectorizer(sublinear_tf=True, norm='l2', min_doc_freq=10, ngram_span=(1, 2), exclude_stopwords='english')
train_text_features = text_transformer.fit_transform(training_data)
test_text_features = text_transformer.transform(testing_data)
# Model selection and training
classifier_parameters = {
    KNeighborsClassifier(): {
        "neighbor_count": [25, 30, 35, 45],
        "weight_scheme": ['uniform', 'distance'],
        "leaf_dimensions": [25, 30, 35]
    },
    DecisionTreeClassifier(): {
        "split_criterion": ['gini', 'entropy'],
        "node_splitter": ['best', 'random'],
        "tree_depth": [None, 90, 95, 100],
        "feature_selection": [None, "auto", "sqrt", "log2"],
        "seed": [42]
    },
    MultinomialNB(): {
```

```
        "prior_fitting": [True, False]
    },
    LinearSVC(): {
        "loss_function": ['hinge', 'squared_hinge'],
        "classification_strategy": ['ovr', 'crammer_singer'],
        "intercept_fitting": [True, False],
        "seed": [42],
        "iteration_limit": [900, 1000, 1100]
    },
    SVC(): {
        'penalty_parameter': [0.1, 1, 10, 100, 1000],
        'smoothing_factor': [1, 0.1, 0.01, 0.001, 0.0001],
        'kernel_type': ['rbf']
    },
    RandomForestClassifier(): {
        "split_criterion": ['gini', 'entropy'],
        "use_bootstrap": [True, False],
        "tree_depth": [85, 90, 95, 100],
        "feature_selection": ['sqrt', 'log2'],
        "tree_count": [60, 80, 90],
        "seed": [42]
    },
    SGDClassifier(): {
        "loss_function": ['hinge', 'log', 'perceptron'],
        "regularization": ['l2', 'l1'],
        "learning_rate": [0.0001, 0.0003, 0.0010],
        "stop_early": [True],
        "iteration_limit": [1000, 1500],
        "seed": [42]
    }
}
```

```
#Model evaluation
best_model, best_accuracy = None, 0
for model, params in classifier_parameters:
    grid_search = GridSearchCV(estimator=model, param_grid=params, cv=3, n_jobs=1)
    grid_search.fit(X_train_vect, y_train)
    predicted = grid_search.predict(X_test_vect)
    accuracy = accuracy_score(y_test, predicted)
    if accuracy > best_accuracy:
        best_accuracy = accuracy
        best_model = grid_search
# Model performance visualization
model_names = [type(model).__name__ for model, _ in classifier_parameters]
accuracy_scores = [accuracy_score(y_test, model.predict(X_test_vect)) for model, _ in classifier_parameters]
performance_df = pd.DataFrame({"Model": model_names, "Accuracy": accuracy_scores})
sns.barplot(x='Model', y='Accuracy', data=performance_df)
plt.title('Model Accuracy Comparison')
plt.xticks(rotation=45)
plt.show()
# Total runtime calculation
total_runtime_seconds = (dt.now() - start_time).seconds
print(f'Total runtime: {total_runtime_seconds / 60:.1f} minutes')
# Save the vectorizer and model
joblib.dump(vectorizer, "final_vectorizer_7.joblib")
joblib.dump(best_model, 'final_ureport_model_7.joblib')
```

3. Predicting unseen dataset using final model

python

```
# Start timer for performance tracking
start_time = dt.now()
```

```
# Load the trained model and vectorizer
saved_model = joblib.load("final_ureport_model_7.joblib")
vectorizer = joblib.load("final_vectorizer_7.joblib")

# Load and preprocess the dataset
df=pd.read_csv("Ureport_dataset_to_be_machine_labelled_206625texts.csv",
encoding="cp1252")
df = df[df['message'].notnull() & df['message'].str.strip() != ""]

# Text cleaning function
def sanitize_message(content):
    content = content.lower()
    replace_special_chars = re.compile('[/(){}\[\]\|@,;]').sub(' ', content)
    remove_unwanted_symbols = re.compile('[^0-9a-z #+_]').sub('', replace_special_chars)
    eliminate_digits = re.compile('[\d+]').sub('', remove_unwanted_symbols)
    filtered_words = ' '.join(word for word in eliminate_digits.split() if word not in
set(stopwords.words('english')))
    return filtered_words

def get_wordnet_pos(tag):
    tag_dict = {"J": nltk.corpus.wordnet.ADJ, "N": nltk.corpus.wordnet.NOUN, "V":
nltk.corpus.wordnet.VERB, "R": nltk.corpus.wordnet.ADV}
    return tag_dict.get(tag[0].upper(), nltk.corpus.wordnet.NOUN)

df['message'] = df['message'].apply(sanitize_message)

# Transform text data and predict using the pre-trained model
X = vectorizer.transform(df['message'])
predictions = saved_model.predict(X)

# Mapping dictionary for predictions
```

```
label_map = {i: label for i, label in enumerate(['cervical cancer', 'circumcision', 'condoms', 'HIV', 'masturbation', 'other', 'prevention', 'registration/platform issues', 'sex and SRH', 'STIs', 'symptoms', 'testing', 'transmission', 'treatment'])}
df['predicted_label'] = [label_map[value] for value in predictions]

# Save the classified data
df.to_csv("Ureport_text_data_classified.csv", index=False)

# Performance tracking
total_runtime_seconds = (dt.now() - start_time).seconds
print(f'Total runtime: {total_runtime_seconds / 60:.1f} minutes')

# Load classified data for visualization
classified_df = pd.read_csv("Ureport_text_data_classified.csv")
sns.set_style('whitegrid')
plt.figure(figsize=(10, 6))
predicted_label_plot = sns.countplot(x='predicted_label', data=classified_df)
predicted_label_plot.set_xticklabels(predicted_label_plot.get_xticklabels(), rotation=40, ha="right")
for bar_container in predicted_label_plot.containers:
    predicted_label_plot.bar_label(bar_container)
plt.title('Categorized using Final U-report Machine Learning Classification Model')
plt.tight_layout()
plt.show()
```