An Analysis of Search Query Evolution in Document Classification and Clustering

Haddela Kankanamalage Prasanna Sumathipala

A thesis submitted in partial fulfilment of the requirements of
Sheffield Hallam University
for the degree of Doctor of Philosophy

January 2023

Candidate Declaration

I hereby declare that:

1. I have not been enrolled for another award of the University, or other academic or professional organisation, whilst undertaking my research degree.

2. None of the material contained in the thesis has been used in any other submission for an academic award.

3. I am aware of and understand the University's policy on plagiarism and certify that this thesis is my own work. The use of all published or other sources of material consulted have been properly and fully acknowledged.

4. The work undertaken towards the thesis has been conducted in accordance with the SHU Principles of Integrity in Research and the SHU Research Ethics Policy.

5. The word count of the thesis is 40183.

Name	*Haddela Kankanamalage Prasanna Sumathipala*
Date	*January 2023*
Award	*PhD*
Faculty	*Department of Computing*
Director(s) of Studies	*Dr. Laurence Hirsch*

Acknowledgment

Undertaking doctoral studies is a profound endeavor that demands unwavering commitment. It necessitates intense desire, dedication, and a strong sense of purpose from the student. However, achieving such a formidable undertaking is not possible without the ceaseless support and encouragement of numerous individuals. Today, I wish to express my heartfelt gratitude to those who have supported me in countless ways, making the completion of this thesis attainable.

I extend my sincere appreciation to my academic advisor, Dr. Laurence Hirsch, whose enduring support, guidance, and patience were invaluable in facilitating the successful completion of this thesis. I am indebted to him for taking me under his wing when my initial academic mentor retired. His guidance and mentorship not only gave me hope but also shaped me into a researcher of integrity. I am also thankful to my former academic mentor, Dr. Keith Burley, for affording me the opportunity to embark on this research journey and for his support during the preliminary stages of the study. My profound gratitude also goes to my supervisory team, including Dr. Teresa Brunsdon from the University of Warwick, as well as Dr. Jotham Gaudoin and Dr. James Baldwin from Sheffield Hallam University.

I also wish to express my gratitude to the administration of SLIIT and my esteemed colleagues for their multifaceted support during my academic pursuit. In particular, I am grateful to Professor Lalith Gamage, Professor Mahesha Kapurubandara, and Professor Chandimal Jayawardena for their guidance and unwavering support. I am deeply appreciative of my cherished friends whose encouragement has been instrumental throughout my academic journey. My heartfelt thanks extend to my dear friends in Sheffield who provided me with a supportive community during my visits to the city.

Above all, I am profoundly grateful to my beloved family for being my wellspring of courage throughout this significant achievement. My gratitude knows no bounds for my dear wife, Enoka Niroshani, whose profound understanding and patience supported me as I pursued my lifelong ambition. I will forever hold a debt of gratitude to my parents for everything they have done for me. I am appreciative of my beloved daughter and son, Thisuni and Genuka, for their understanding and unwavering support. It is my sincerest hope that I have made them all proud.

Abstract

With the increasing use of data analytics in decision-making processes today, the analysis of document collections for various purposes has become a widely accepted area of research. Document classification and clustering are two intensely investigated and active areas of research due to the complex nature of the problem and its impact on society.

However, many of the popular methods developed to classify and cluster documents with high accuracy lack explanation to end users, which affects the trustworthiness of certain applications among them. Therefore, it is crucial to improve explainable classification and clustering methods.

One approach that has shown promise in this regard is the evolved search query (eSQ), a genetic algorithm (GA)-based approach for classification and clustering. GA-based methods excel at finding highly optimized solutions for complex problems, and eSQ has utilized this capability to develop classification and clustering methods that are also human interpretable.

The primary focus of this study is to analyse the eSQ approach to document classification and clustering with an emphasis on explainability. The investigation covers three perspectives of the eSQ-based methods: explainability, document classification, and document clustering. This thesis presents a taxonomy for classification based on human friendliness, empirical observations on the performance of eSQ classifiers using different feature selection methods, the effectiveness of eSQ classifiers for Sinhala documents, and the performance of eSQ clustering for Sinhala documents.

The research contributes significantly by categorizing popular classification methods using the new taxonomy, integrating feature selection methods into eSQ classifiers, enhancing Apache Lucene by incorporating the Sinhala language with basic pre-processing tools, and improving eSQ hybrid single word clustering methods. Notably, the eSQ-based classification and clustering methods demonstrate superior performance when document categories overlap.

Table of Contents

List of Figures

List of Tables

List of Publications

- Laurence Hirsch, Alessandro Di Nuovo and Prasanna Haddela, “Document Clustering with Evolved Single Word Search Queries”, The proceedings of IEEE Congress on Evolutionary Computation (IEEE CEC 2021), Krakow, Poland, Jun 28 to July 01, 2021.
- Prasanna S. Haddela, Laurence Hirsch, Teresa Brunsdon and Jotham Gaudoin, “Use of Interpretable Evolved Search Query Classifiers for Sinhala Documents”, The proceedings of International Conference on Future Technology Conference (FTC 2020), Published in Springer Advances in Intelligent Systems and Computing (AISC) Series, Vancouver, Canada, November 5-6, 2020.
- Prasanna S. Haddela, Laurence Hirsch, Teresa Brunsdon and Jotham Gaudoin, “Human Friendliness of Classifiers: A Review”, The proceedings of International Conference on Emerging Technologies and in Data Mining and Information Security (IEMIS 2020), Published in Springer Advances in Intelligent Systems and Computing (AISC) Series, Kolkata, India, July 2- 4, 2020.

Chapter 1: Introduction

1.1 Background

The expansion of Artificial Intelligence (AI) has been fuelled by the nearly limitless amount of accessible data, inexpensive data storage, and the development of less expensive yet more powerful computation than ever before. The trend of digital transformation of businesses, new scientific research and advances, and social communication platforms have all contributed to the creation of a significant amount of digital data for many good reasons. With this background and due to the competitiveness of modern society, it became necessary to create a wide range of intelligent applications in every industry, with text analytics emerging as a subset of this trend.

Documents related to many industries have been transformed into soft text data for several obvious reasons. In the past, data platforms were used for processing, storage, and communication. With the rise of AI applications, however, text analytics and discovering hidden insights in massive document collections have become quite popular. Core computing methods in text analytics include document classification and clustering. So, it is crucial and beneficial to create extensive methods for document classification and clustering to leverage the power of text analytics (Zhang et al., 2023).

We found that many research attempts have been initiated and have contributed to text analytics by developing the required techniques and tools. But most of the work has been limited to English and a few other languages. In particular, there are few resources for many other languages (which we call “low-resource languages”) in the digital world. They lack pre-labelled datasets for model building, as well as fundamental text analytics tools such as stop-word lists, stemming algorithms, etc. This becomes a barrier to taking full advantage of technological development for those communities at present. Therefore, one of the objectives of this research is to contribute to low-resource languages.

A few years back, it was hard to find research attempts towards Explainable Artificial Intelligence (XAI). However, we had a particular interest in contributing to human-friendly technological development, assuming that it is an important aspect for some real-world applications. It was one of the main reasons to study human-interpretable document classification and clustering methods.

In the early stages of the research, we discovered that Evolved Search Query (eSQ) classifiers (Hirsch & Brunsdon, 2018) had the potential to enhance classification capabilities while promoting low-resource languages and human-friendly technical advancement. As a result, this research assists in improving eSQ classifiers and eSQ clustering methods while also benefiting the Sinhala language, one of the low-resource languages.

1.2 Motivation

Machine Learning (ML), a subfield of AI research, has been rapidly developing in recent years. In Pugliese et al. (2021), authors of the journal listed approximately 16,339 publications relevant to machine learning that were published between 2018 and 2020. But the majority of research efforts are focused on figuring out how to make machine-learning techniques more accurate and efficient. Although this has resulted in improvements in accuracy, most of the algorithms have tended to be 'black box' and not human-friendly, in that they are not easily understood by a human. With the technological development of artificial intelligence, some applications have developed that access or process others' data in an unacceptable and unethical manner. A few notable, well-reported instances include the Harm Assessment Risk Tool (Oswald & Grace, 2016), the Facebook-Cambridge Analytica case (Carole Cadwalladr & Emma Graham-Harrison, 2018), and COMPAS-Predicting Recidivism Risk (Angwin et al., 2016). Therefore, it is crucial to build machine learning models that are easy for humans to understand in order to reduce potential technological misuse. Furthermore, the European Union's data regulatory restrictions (Li et al., 2019) highlight the urgent need for additional studies in these fields. These motivating factors kept us working on interpretable document classification and clustering methods.

1.3 Research Aim and Objectives

The primary aim of this research is to analyse evolved search query-based document classification and clustering methods. The following three research questions are formulated to cover different perspectives of the research aim.

Research Question 1: How do eSQ-based methods for classification and clustering compare to other algorithms in terms of explainability?

Research Question 2: What is the impact of different feature selection metrics for the eSQ classifiers?

Research Question 3: How do eSQ-based classification and clustering perform for Sinhala documents?

The following research objectives were formulated considering the aim of the research and the research questions to create a list of actionable items. This thesis presents the completed research tasks in the order of the objectives while answering the research questions and achieving the research goal.

Research Objectives:

1. To develop a taxonomy base on human friendliness and organise document classification methods.
2. To analyse the impact of different feature selection metrics on the eSQ document classifiers.
3. To assess the performance of eSQ classifiers for the Sinhala document collections.
4. To evaluate the performance of eSQ clustering methods for the Sinhala document collections.
5. To provide a list of recommendations based on experimental results.

1.4 Methodology

The following research methodology was adopted to address the aim and objectives and the research questions. Three research questions cover the three perspectives of the aim, and the research objectives are highlighting the specific activities carried out.

Selecting a research approach is crucial, and it depends on the nature of the research problem or issue. The main three approaches are quantitative, qualitative and mixed methods (Bryman, 2015; Creswell, 2013). They are not fully discrete approaches or else a study tends to be more qualitative than quantitative or vice versa. Mixed methods research resides in the middle because it incorporates elements of the other two approaches.

Quantitative research primarily relies on numerical data or closed-ended questions to assess variables and relationships between them. It involves rigorous data collection and statistical analysis, making it well-suited for investigations seeking to measure and quantify phenomena. This approach is grounded in a deductive process, where researchers start with an established theory or hypothesis, then proceed to collect data to test the hypothesis, and finally draw conclusions based on the findings. This process allows for hypothesis confirmation or rejection, which may, in turn, lead to the refinement or revision of the underlying theory. The process of deduction:
Theory --> Hypothesis --> Data collection --> Findings --> Hypothesis confirmed/ Reject --> Revise Theory

On the other end of the spectrum lies qualitative research, which revolves around using words, narratives, and open-ended questions to delve deeply into the complexities of a phenomenon. Qualitative research embraces a more inductive approach, whereby researchers gather data from participants or observations and subsequently generate theories or patterns based on the emergent themes. It aims to capture the richness, context, and subjective aspects of human experiences, making it particularly valuable in exploring intricate social and behavioural issues.

Mixed methods research, residing in the middle ground, synthesizes elements from both quantitative and qualitative approaches. This versatile approach acknowledges that some research problems may benefit from a broader perspective, allowing researchers to gain a more comprehensive understanding of the topic at hand. Mixed methods researchers gather and analyze both numerical and narrative data, providing a more holistic view of the research question. By integrating quantitative data for statistical validation and qualitative data for contextual understanding, researchers can leverage the strengths of both approaches to better triangulate their findings.

One of the ways of collecting data in the quantitative approach is experimental methods. This is common in scientific research in computing that requires a complex software solution. For example, the organization of data that is not tabular, or the construction of tools to solve optimization problems. The approach is largely to identify concepts that facilitate solutions to a problem and then evaluate the solutions through the construction of prototype systems (Dodig-Crnkovic, 2002; Okoko et al., 2023).

In this research, we are aiming to achive five objectives. Among them, first objective based on the literature review. It’s outcome based on observations of previous research publications. Achiving outcome of this objective can be classified under qualitative research, But, Next three objectives purely based on the experiment results received, numeriacal numbers, from the changes made to eSQ engine. Last objective devepend on the outcome of previous objectives. Therefore, overall, considering the characteristics of this research, it follows the quantitative approach with a series of experiments than the mixed method of research methodology.

In this research, we are aiming to achieve five objectives. Among them, the first objective is based on the literature review. Its outcome relies on observations from previous research publications. Achieving the outcome of this objective can be classified as qualitative research. However, the next three objectives are purely based on the experimental results received, numerical values derived from the changes made to the eSQ engine. The last objective is achieved based on the outcomes of the previous objectives. Therefore, overall, considering the characteristics of this research, it follows a quantitative approach involving a series of experiments rather than a mixed-method research methodology.

1.5 Key Contributions to Research

This study contributes to the classification and clustering of document collections. We had a particular interest in contributing to human-friendly methods and low-resource languages. The following list highlights specific details:

1. Developed a new taxonomy considering the human friendliness of document classification: We believe that modifiability (ability to fine-tune) of predictive models is important for higher human friendliness, not only interpretability. Considering this idea, we have developed a taxonomy for document classification and evaluated existing work.

2. Analysed the effects of four feature selection metrics on document classification for eSQ classifiers experimentally: We implemented feature selection metrics and integrated them into the eSQ system. Performance analysis for various conditions including different lengths of feature sets has been studied and presented.

3. Enhanced eSQ classifiers for the Sinhalese Language: The Apache Lucene indexing framework supports many human languages but not Sinhalese language.

We have recompiled the Lucene project with basic pre-processing functionalities, including tokenizing, removing stop-words, and stemming for Sinhalese. An enhanced Lucene framework is integrated into the eSQ classifiers, and experimental results have been presented.

4. Extended the eSQ clustering method by integrating new classification models: We have modified the eSQ clustering method by adding classification algorithms for cluster expansion. For Sinhala documents, the performance of eSQ hybrid clustering methods (eSQ-HSW-KNN, eSQ-HSW-KNF, eSQ-HSW-NB) was evaluated.

5. Developed a labelled Sinhala document collection named “SLNG Collection” for machine learning research. Publicly available at https://github.com/psumathipala/SLNGCollection.git

1.6 Overview of the Thesis

Chapter 1 of this thesis provides an introduction to the research. The chapter presents a brief description of the background of the research, the motivation for the research, the aim of the research and research questions, and key contributions to the research.

Chapter 2 provides a comprehensive review of the literature relevant to this research, starting from key stages of text analytics. Principals of the feature selection methods, Apache Lucene, use of genetic algorithms, review on classification and clustering methods, genetic algorithm-based classification methods, genetic algorithm-based document clustering methods, the importance of explainable machine learning, and human friendliness-based taxonomy for classification. Finally, the open research directions and challenges are highlighted.

Chapter 3 presents the details of integrating and analysing feature selection metrics to the eSQ classifiers. From the beginning of the chapter, it explains how eSQ classifiers are used for classification. Next, the architecture of the eSQ engine and step by step process of how it produces search query classifiers are explained. Integration of feature selection metrics and analysis of results are given at the end of the chapter.

Chapter 4 provides the effectiveness of eSQ classifiers for Sinhala documents. This chapter starts by highlighting the limitation of doing text analytic projects for the Sinhala language. Preparation of datasets, improvement made to Apache Lucene, and finally performance analysis of the eSQ classifier for the Sinhala language have been presented in this chapter.

Chapter 5 explains the performance achieved by the eSQ hybrid clustering method for Sinhala documents. It explains the different clustering approaches and then the eSQ clustering approach step by step. The experiment aims and materials used during the experiments are given in the next section. Finally results and discussion about hybrid clustering methods were given.

Chapter 6 provides the Conclusion of the research, including a summary of the research, recommendations, limitations, and future work.

Chapter 2: Background and Literature Review

2.1 Chapter Overview

This chapter begins with a discussion of text analytics research initiatives related to document classification and clustering. We then focus our discussion on related genetic algorithm-based document classification and clustering methods.

Section 2.3 outlines the key stages of a common document classification task. In that section, we discuss how to represent and prepare document collections for further processing, the types of predictive or descriptive models, and what metrics are commonly used when evaluating the results. After that, we start explaining specific details of different stages in the life cycle of the document classification or clustering projects. The feature selection methods are organised into categories in Section 2.4, and the selected set of global feature selection methods are well explained as they are closely related to the study. One of the significant distinctions in the proposed text analytics platform is the use of a full-text search engine called Apache Lucene to store and retrieve text. It is a well-known full-text search engine, and its details are given in Section 2.5. Genetic algorithms (GA) are not that popular for document classification and clustering, but we believe they can make an important contribution in terms of accuracy and interpretability. Therefore, the general execution procedure of a genetic algorithm and related details are given in Section 2.6. We have used a set of popular classification and clustering methods to compare the results of our study. So, in Section 2.7, we briefly explain a few selected algorithms from different approaches that have been used when comparing the experimental results. Sections 2.8 and 2.9 are reserved for discussing existing genetic algorithm-based document classification and clustering methods, as they are closely related to the experiments carried out. Section 2.10 is dedicated to discussing the significance of explainable machine learning, as we hold a particular interest in explainable predictive models. In Section 2.11, details of the taxonomy for human-friendly text classification are provided. Subsequent sections, namely 2.12, 2.13, and 2.14, are utilized to categorize existing methods using the developed taxonomy. Lastly, we outline several open research areas and challenges related to genetic algorithm-based text analytics methods in a section presented at the end of the chapter.

2.2 Text Analytics: Overview

Text analytics is the process of extracting hidden useful insights from the text (Ittoo et al., 2016). The term is roughly synonymous with text mining. It involves the automated extraction of new, previously unknown, and useful information from different text repositories. Text repositories may include tweets, reviews, emails, blogs, websites, articles, books, etc. Some of the frequently used computer tasks in text analytics include document classification, document clustering, entity extraction, sentiment analysis, and text summarization.

Because of the importance of receiving important hidden information from massive volumes of datasets that are already stored or being received through day-to-day operations in many systems, text analytics-related applications have become popular in many industries. It's difficult to think of an industry or field that doesn't use text analytics.

The problem of document classification is defined as follows: Consider a set of documents D = d_1..., d_n, where each document is labelled with one of *i* different discrete categories indexed by *i*= (*1*,...,*c*). Split the collection of D documents into training and testing sets. The training dataset is used to construct a classification model that relates the features in the underlying documents to one of the pre-labelled categories. For a given test instance for which the category is unknown, the trained model is used to predict a category label for this instance (Aggarwal & Zhai, 2012; Giuntini et al., 2023).

The document clustering problem is defined as that of finding groups of similar documents in the document collection. The similarity between the documents is measured through the use of a similarity function.

The focus of this study is to investigate possible improvements related to document classification and document clustering methods. The next section discusses the important stages in a document classification process, some of which are also involved in document clustering. In the following sections and Chapter 5, specific details of document clustering are discussed.

2.3 Key Stages of Document Classification

A broadly common set of computing operations is used in both document classification and clustering methods while performing two different types of text analytical tasks for a given dataset. In this section, key stages of document classification are outlined first, and common operations and some differences in document clustering are given at the end of the section.

The process of document classification involves three key stages. But, it encompasses multiple subtasks within each stage. They are document representation, classifier construction, and model evaluation (Khan et al., 2010; Kowsari et al., 2019). Stage I: prepares document collections for understanding and processing by model-building algorithms. Stage II: Building predictive models from a training dataset. Stage III: evaluate the effectiveness of the built models using performance measures. Figure 2.1 shows a holistic view of the three stages that are involved in the process of document classification.

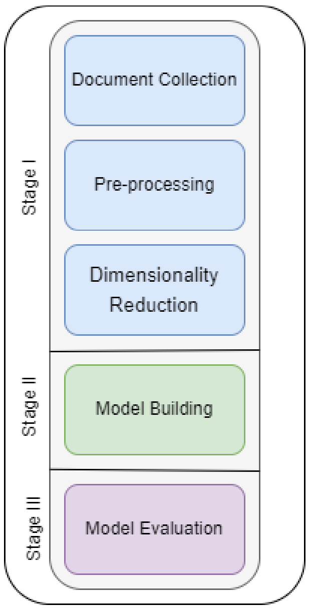


Figure 2-1: Process of document classification

2.3.1 Stage I: Document Representation

For document classification, algorithms need documents to be represented in a way that enables the induction of the classifier. There are multiple ways to represent text documents. Vector models (vector space model, word embedding), N-gram models, graph models (lexical graphs, semantic graphs), and topic models are some of them (El-Kassas et al., 2021).

There are two main considerations when selecting a method for document representation. One of them is the capability of representing the semantics of the document content. Currently, most of the methods are capable of understanding only very little of the meaning of human language (Turney & Pantel, 2010). The high dimensionality of text is another concern since it makes document classification more complicated. Among the existing work, a vector space model (VSM) with basic data pre-processing and dimensional reduction methods was heavily used. Also, recently developed word embedding methods are becoming more popular with artificial neural networks and deep learning. Word embedding methods are relatively rich in handling the semantics of human languages (Kowsari et al., 2019).

Data pre-processing is considered the initial step in document classification. This involves capturing, cleaning, and smoothing the features of a text and organising them to support the process of computing. As the initial pre-processing task, three pathways have been employed in published papers. They are bag-of-word (BoW), bag-of-phrase (BoP), and instance selection (Tsai et al., 2014).

Among these, making a “bag of words” (BoW) is the most popular and widely accepted method (Jindal et al., 2015). In this approach, commonly used treatments are breaking the text into tokens (in most cases, words), removing stop-words, and stemming. These steps are used to remove irrelevant and noisy data before creating a BoWs.

Bag-of-Phrases (BoP) method is semantically richer but computationally expensive in comparison to the BoW method. The other method is to choose an instance or use a sample set from a document collection. However, this method has been used rarely in the pre-processing phase of document classification (Tsai et al., 2014).

Employing Dimensional Reduction (DR) methods is also important to improve the efficiency and accuracy of document classification (Fragoso et al., 2016; Gonçalves et al., 2015; Uğuz, 2011). There are two ways of conducting DR (i.) feature selection and (ii.) feature extraction (Cai et al., 2018; Fragoso et al., 2016; Tang et al., 2016).

Feature Selection (FS) aims to select the best subset of features with the highest predictive power in text classification (Parlak & Uysal, 2023). The ability to separate each of the categories depends on the features selected and is measured through feature evaluation metrics. Broadly, there are three types of FS methods. They are filters, wrappers, and embedded methods (Bashir et al., 2022). The filter methods are independent of the classification process. The wrapper method uses another classification method to rank the features of a dataset. If feature selection is a part of the classification algorithm itself, it is called an embedded method. In section 2.4, feature selection methods are described in more detail.

Feature Extraction (FE) is also called feature transformation algorithms. This aims at extracting features by projecting the original high-dimensional data into a lower-dimensional space through algebraic transformations. A few of the popularly used FE methods are Principal Component Analysis (PCA), Latent Semantic Analysis (LSA) and Linear Discriminant Analysis (LDA)(Velliangiri et al., 2019).

Document representation aims at producing a data structure that represents the document collection before it is used for classifier construction. In a vector space model, a document is transformed into a vector, which contains term features. Further, term weighing methods and normalisation techniques are used to smooth the term vectors in some experiment setups (Turney & Pantel, 2010).

With the recent popularity of neural network-based classifiers, neural word embedding has emerged as a viable alternative for document representation. This approach represents words discretely and symbolically. which not only improves the accuracy of the document classification but also has the added advantage of improving semantic and syntactic similarities between words (Levy & Goldberg, 2014).

As shown in the overview provided above, there are a variety of approaches and methods to prepare documents to be used by model-building algorithms. In this research, feature selection methods are studied and tested against the improved GA base document classification method and more details are given in Chapter 3.

2.3.2 Stage II: Classifier Construction Methods

Classification is a supervised learning method. As a result, supervised learning algorithms utilised already pre-processed and prepared training datasets. A few of the popular classifier construction algorithms are decision tree (DT), k-nearest neighbour (kNN), support vector machine (SVM), neural networks (NN), Bayesian classifier, rule-based methods, and genetic algorithm-based methods. In practice, there are no perfect classifiers since each performs well under certain conditions. Furthermore, there may be instances where two humans disagree on the same classification. So, it is very challenging to find a way of building a model that is capable of classifying a given data point without creating a conflict. Another important aspect of classification is the explainability of the prediction. At the moment, the majority of highly accurate classification methods are of the black-box variety. It has become a major obstacle for certain applications due to the limitations in monitoring and fine-tuning them (Barros et al., 2014). In section 2.7, model-building algorithms are discussed in detail, and section 2.11 discusses the explainability aspect of the predictive models.

2.3.3 Stage III: Model Evaluation

There are two main methods available to evaluate research outcomes either analytically or experimentally. Due to the inherently subjective nature of text analytics, it is non-formalizable. Therefore, experimental-based methods are more appropriate to evaluate document classifiers (Abdullah-Al-Kafi Md. et al., 2022). In document classification, experiments are designed to compute accuracy using two popular validation methods. They are hold-out (train/test) and k-fold cross-validation methods. The hold-out method divides a data set into a training subset and a testing subset. The training data is used to train the classification model, and the same model is tested on independent testing data to evaluate the performance of the model (Yu, 2008). The k-fold cross-validation algorithm divides a dataset into k folds and repeats a classification experiment k times. Each time, one-fold of data is used as the test set, and the classifier is trained on the remaining (k-1) folds. The classification performance is averaged across k runs.

Precision, recall, and the F-measure are used as measures of performance in both validation approaches. The following example shows how to derive accuracy measures from a confusion matrix for a two-class classification problem. Figure 2-2 depicts a confusion matrix for two classes.

		Actual values	
		Positive	Negative
Predicted values	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Figure 2-2: Confusion matrix for two classes

True Positive (TP): Refers to the number of predictions where the classifier correctly predicts the positive class as positive.
True Negative (TN): Refers to the number of predictions where the classifier correctly predicts the negative class as negative.
False Positive (FP): Refers to the number of predictions where the classifier incorrectly predicts the negative class as positive.

False Negative (FN): Refers to the number of predictions where the classifier incorrectly predicts the positive class as negative.

The confusion matrix of a two-class problem categorises predicted results into four types as TP, TN, FP, and FN. Considering this as a basis, the precision and recall measures are defined as follows:

$$Precision = \frac{|\mathrm{TP}|}{|\mathrm{TP}| \quad + \quad |\mathrm{FP}|} \tag{1}$$

Equation (1), that is, precision, is the proportion of truly positive documents in the category that the classifier deems positive.

$$Recall = \frac{|\mathrm{TP}|}{|\mathrm{TP}| \quad + \quad |\mathrm{FN}|} \tag{2}$$

Equation (2), that is, recall, is the fraction of the number of true positive documents that the classifier returns from the number of actual documents of the category in the corpus (Grandini et al., 2020; Polychronopoulos et al., 2014).

Precision and recall are not effective measures of classification performance alone. For example, a recall of 100% can be obtained by trivially labelling all documents in the corpus as positive. Therefore, the F1 measure is commonly used for determining the effectiveness of classification as it has the ability to give equal weight to precision and recall. The F1 measure computes values between 0 and 1. It is the harmonic mean of precision (*p*) and recall (*r*) for a binary classification problem as determined by Equation (3).

$$F1\ measure = \ 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

$$F1\ measure = \ \frac{2 \times p \times r}{p + r} \tag{3}$$

It can also be expressed using TP, FP, and FN, as shown in Equation (4).

$$F1\ measure = \frac{|\mathrm{TP}|}{|\mathrm{TP}| \quad + \frac{1}{2} \quad |\mathrm{FP}\ +\ \mathrm{FN}|} \tag{4}$$

When the F1 measure extends to multiclass classification problems, three types of averaging methods have been used. They are micro-average F1, macro-average F1, and weighted-average F1 measures (Grandini et al., 2020; Phoungphol et al., 2012; Powers, 2011).

In micro-averaging, all classification decisions in the data set are entirely considered without class discrimination. If the classes in a collection are biased, large classes would dominate small ones. The micro-F1 measure can be computed using Equation 5 with redefined precision (*P*) and recall (*R*), which are presented below.

$$P = \frac{\sum_{i=1}^{c} TP_i}{\sum_{i=1}^{c}(TP_i \quad + \quad FP_i)}$$

$$R = \frac{\sum_{i=1}^{c} TP_i}{\sum_{i=1}^{c}(TP_i \quad + \quad FN_i)}$$

$$Micro\ F1 \ = \frac{2 \times P \times R}{P + R} \tag{5}$$

where a pair of (*P*, *R*) corresponds to micro precision and micro recall values respectively and *i* (*i*=1,…,c) is the number of classes, overall classification decisions are made within the entire dataset, not individual classes.

In macro-averaging, however, the F1 measure is computed for each class in the dataset, and the average for overall classes are obtained. In this way, equal weight is assigned to each class without regard to class distributions. The computation of macro-F1 can be formulated as in Equation 4,

$$F1_i \ = \frac{2 \times P_i \times R_i}{P_i + R_i}$$

$$Macro\ F1 = \frac{\sum_{i=1}^{C} F1_i}{C} \tag{6}$$

where a pair of (P_i, R_i) corresponds to the precision and recall values of class *i* (*i*=1,…,c), respectively (Lan et al., 2009; Uysal, 2016).

Weighted F1 is computed by taking the mean of all per-class F1 scores while considering each class's support. Support refers to the number of actual instances of the class in the dataset. Equation 7 presents the formula to compute weighted F1.

$$Weighted\ F1 = \sum_{i=1}^{C} F1_i \times w_i \tag{7}$$

where $w_i = \frac{n_i}{N}$, N is the total number of instances or documents in the dataset and n_i is the number of instances or documents in i^{th} category.

In general, if a dataset is imbalanced and all classes are equally important, the macro average F1 measure is a good option since it considers each class equally. But, for an imbalanced dataset and wish to allocate higher weight to classes with more instances, the weighted average F1 measure is recommended. In weighted averaging, the contribution of each class to the F1 average is weighted according to its size. The micro average F1 measure is appropriate for datasets that are balanced and want a readily comprehensible measure for overall performance, irrespective of class (Mandl et al., 2019; Tahir et al., 2012).

In our studies, we have utilised macro average F1 since category-level performance is important to us in addition to the overall performance of the classifier. Also, some of the employed datasets are imbalanced.

2.4 Principles of the Feature Selection Metrics

There are several advantages of using feature selection methods in text analytics. These methods reduce the number of dimensions in the dataset, making the training faster and improving classification or clustering performance by removing noisy features. The selection of features also aids in avoiding overfitting in model construction.

Broadly, there are three types of FS approaches. They are filters, wrappers, and embedded methods (Bashir et al., 2022). Figure 2-3 shows the categorization of FS methods.

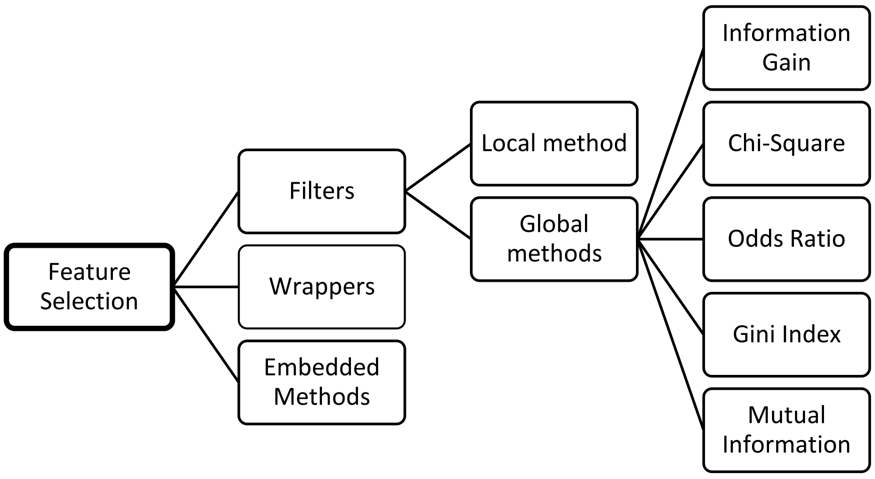


Figure 2-3: Hierarchical View of FS Methods

The filter methods remove irrelevant features from the feature set prior to the application of the classifier construction algorithm (Cunningham, 2008). Therefore, filter methods are independent from the classifier constructors and computationally less demanding. Filter methods are further categorised as local and global methods. The FS method is either global or local depending on how it assigns scores. Local methods assigned multiple class-based scores to each feature, whereas global methods assigned a single score to each feature. As a result, it calculates the local feature selection score by converting multiple local scores depending on the globalisation policy (Uysal, 2016). For example, Information gain, Chi-square, Information Gain, Gini Index, Odds ratio are global FS methods (Uğuz, 2011; Uysal, 2016). The following scenario from text classification further explains the filter-based method.

Consider a scenario where you are tasked with classifying news articles into categories such as “Politics”, “Sports” and “Technology”. To enhance the classification model’s accuracy and interpretability, you apply a filter-based feature selection technique. You compute the chi-squared statistic for each word’s frequency in relation to each category. This statistic measures the independence between word occurrence and category distribution. Consequently, words like “election”, “president” and “legislation” might exhibit high chi-squared values for the “Politics” category, whereas terms like “score”, “team” and “game” could yield high values for the “Sports” category. Subsequently, you retain words with the highest chi-squared values for each category, resulting in a more discriminative set of features for accurate document classification.

Wrapper methods select a set of important features using machine learning algorithms. These algorithms estimate the predictive power of a feature based on accuracy and select it from the feature space. The following scenario further explains the wrapper-based method.

Imagine you are working on sentiment analysis of customer reviews, aiming to categorize them as “Positive”, “Neutral” or “Negative”. Employing a wrapper-based feature selection approach, you initiate with an exhaustive set of features comprising words and n-grams. You then train a classification model, such as a Naive Bayes classifier, using these features. Following model training, you assess the importance scores or coefficients attributed to each feature within the classifier. Features contributing minimally to classification performance, such as common stop-words or irrelevant n-grams, are subsequently removed. Through iterative cycles of model training and feature elimination, the process ultimately yields a feature subset that optimally supports sentiment classification.

In embedded methods, feature selection (FS) is an inherent component of the classification technique, as seen in decision tree induction algorithms (Cunningham, 2008). To illustrate, let’s consider the following scenario. The task involves classifying scientific research papers into subject areas such as “Biology”, “Physics” and “Computer Science”. In this scenario, an embedded feature selection technique can be effectively employed. Opting for a Random Forest classifier is logical due to its intrinsic capability to assess feature importance. During the training process, the Random Forest algorithm evaluates the influence of each word or term on classification performance. Words closely related to biological concepts, such as “gene”, “cell” and “DNA” may receive higher importance scores when categorizing papers under the “Biology” category. Similarly, terms like “equation”, “particle” and “energy” might gain elevated importance within the “Physics” category. By delving into the significance of features within the classifier’s context, valuable insights are gained regarding the pivotal terms essential for precise document classification.

Wrapper FS methods use an extra prediction algorithm to select features. It turns feature selection into a black box (Guyon & Elisseeff, 2003) and has an impact on the classifiers’ overall explainability. Embedded methods are not suitable for eSQ due to design constraints. Therefore, wrapper and Embedded FS methods are omitted from this study. Due to the flexibility of switching between various FS methods and the architectural support for integrating with eSQ, we used filter FS approaches for this investigation. In addition, local methods are omitted from the filter method since combining multiple scores to create a unique score for a feature makes them less interpretable. Finally, the selected global FS methods for the study and several previous studies (Cai et al., 2018; Uysal, 2016) confirmed that global feature selection methods performed well in high-dimensional search spaces such as text classification. They have developed by taking theories from multiple disciplines and successfully applying them to text analytics.

In text analytics, terms in the text are treated as features of documents, and they have been used for document classification. In this analysis, terms in the texts are equivalent to words. Therefore, we have used features, terms, and words with the same meaning across the entire work.

Some of the frequently used global FS methods are given in the next few subsections. These methods were integrated into the enhanced classification method in this study, and the results are given in Chapter 3.

2.4.1 Information Gain

Information Gain (IG) is one of the most popular FS methods in the field of machine learning. It measures the information obtained for category prediction by knowing the presence or absence of a feature which is a term or word in a document. IG gives a higher value to a term if it is a good indicator for assigning the document to any class. Let $\{C_i\}_{i=1}^{c}$ denote the set of categories. The IG of term *t* is defined as follows.

$$IG(t) = -\sum_{i=1}^{c} P\,(c_i)\log P(c_i) + P(t)\sum_{i=1}^{c} P\,(c_i|t)\log P(c_i|t) + P(\bar{t})\sum_{i=1}^{c} P\,(c_i|\bar{t})\log P(c_i|\bar{t})$$

Where *c* is the number of classes, $P(c_i)$ is the probability of class c_i, $P(t)$ and $P(\bar{t})$ are the probabilities of the presence and absence of term *t*, $P(c_i|t)$ and $P(c_i|\bar{t})$ are the conditional probabilities of class c_i given the presence and absence of term *t* respectively. This definition is more general than binary classification models and suitable for multiple categories. This measures the effectiveness of a term globally with respect to all categories on average (Cai et al., 2018; Sharma & Jain, 2023).

2.4.2 Chi-Square Statistic

The Chi-square (CHI) test is used to measure the independence of two random variables. This statistical FS method has been applied frequently in document classification problems. It tests the independence of the occurrence of a term and the occurrence of a class. The chi-square value is small if the variables are independent. The following metric measures the relatedness of feature f to class c. For a binary class variable $\{c_+, c_-\}$, considered feature has r possible values, n_{i+} is the number of positive cases with feature value i and μ_{i+} is the expected value for that figure (Parlak & Uysal, 2023). The Chi-square score defines as follows.

$$\chi^2(c,f) = \sum_{i=1}^{r} \left(\frac{(n_{i+} - \mu_{i+})^2}{\mu_{i+}} + \frac{(n_{i-} - \mu_{-})^2}{\mu_{i-}} \right)$$

2.4.3 Odds Ratio

The Odds Ratio (OR) measure is closely associated with information retrieval and is utilized for relevance ranking. This measure assesses the membership and non-membership of its numerator and denominator, respectively. The membership and non-membership scores are normalized by dividing them from each other (Uysal, 2016). Therefore, for a higher score, the numerator must be maximized, and the denominator must be minimized. The OR is defined to be,

$$OR(t, c_i) = log \frac{P(t|c_i)[1 - P(t|\overline{c_i})]}{[1 - P(t|c_i)]P(t|\overline{c_i})}$$

Where *P*(*t* | *c*$_i$) is the probability of term *t* given the presence of class *c*$_i$, and *P*(*t* | ($\overline{c_i}$) is the conditional probability of term *t* given all the classes except *c*$_i$. A smoothing method is applied to avoid division by zero errors i.e. to prevent the numerator becoming zero.

2.4.4 Mutual Information

Mutual Information (MI) is used to measure word association in statistical language modelling. For rare terms will have a higher score than common terms. Therefore, scores are not comparable across terms of widely differing frequency. For term *t* and a category *c*, MI calculate as follows,

$$MI(t, c_i) = log \frac{P(t, c_i)}{P(t) \times P(c_i)}$$

$$MI_{avg}(t, c_i) = \sum_{i=1}^{m} P(c_i) log \frac{P(t, c_i)}{P(t) \times P(c_i)}$$

2.4.5 Gini Text

Gini Text is a variant of a classical Gini Index. Gini Index is a non-purity split method used in decision tree induction algorithms. It is a global FS method for text classification that has been introduced as Gini Text (GT) by Shang et at. (2007). It can be described as,

$$GT(t) = \sum_{i=1}^{m} P(t|c_i)^2 P(c_i|t)^2$$

Where $P(t \mid c_i)$ is the probability of term t given the presence of class c_i, $P(c_i|t)$ is the probability of class c_i given the presence of term t, respectively.

2.5 Apache Lucene: Full-Text Search Engine

Apache Lucene is a software package for text-based information retrieval (Białecki et al., 2012). It is an open-source project that provides Java-based indexing and search technology. It provides a simple but powerful application programming interface that hides the complexity of indexing and searching (Chang, 2023). The fundamental concepts in the Lucene data model are document, field, and index. A Lucene document consists of fields, where each field has a name and unstructured textual content. A Lucene document may contain multiple fields. A Lucene index is a set of documents stored in persistent storage supported by data structures that provide efficient data retrieval. Some experimental results show that Lucene performs far better compared to full-text search engines available with traditional database management systems (Qian & Wang, 2010). Apache Lucene's highly scalable, high-performance indexing architecture, smaller RAM requirements, support for different query types (for example, Boolean, phrase, wildcard, proximity, and range), and multi-language support (Chang, 2023) have enabled the use of full-text search engines in text analytics. Popular enterprise analytics platforms based on Apache Lucene include Solr and Elasticsearch.

Primarily, full-text search engines are required to pre-process their content to create indexes. Therefore, language-specific pre-processing tools are integrated to improve the quality of their functions. Apache Lucene is one of the popular tools that support many human languages around the globe.

The use of Apache Lucene is one of the differences between this research and previous similar research. The ability to pre-process text data, the number of human languages supported, and especially compatibility with the Java programming language are key motivators for this choice.

2.6 Use of Genetic Algorithms

Genetic Algorithms (GA) were invented based on the Darwinian theory of evolution (Espejo et al., 2010; Holland, 1992). Technically, they are mostly applied in the context of expensive optimization problems (Azzouz et al., 2015; Raschip et al., 2015). GA uses an iteratively progressive approach to develop a population to achieve the desired outcome. Such advancements are commonly influenced by biological mechanisms of evolution. Selection, crossover (recombination), and mutation are genetic operators that are applied to individuals to breed the next generation. Fitness functions are used to assess an individual’s strength (Sohail, 2023). When fitness levels are higher for particular individuals, there is a greater likelihood that these individuals will be chosen to participate in the creation of the next generation. As a result, the genetic material of strong individuals will survive until the end of the evolutionary process. Algorithm 1 depicts the fundamental steps of a GA.

Algorithm 1: Outline of the Basic GA.

```
1. Randomly generate the initial population P0.
2. Determine the fitness function f (x) of each individual of the population P0.
3. Repeat
4.         Select the individual (parents) with the best fitness value from the population P0.
5.         Perform the crossover operation C on the parents to create next-generation
           population P1.
6.         Perform mutation M over the population.
7.         Determine the fitness of the population P1.
8. Until the stopping criteria are fulfilled.
```

As this evolution can produce highly optimised solutions, it has been adopted by classification and clustering methods, and some research attempts have developed comparable solutions.
The following subsections briefly explain basic terminologies and genetic operators.

2.6.1 Fundamental Concepts in Genetic Algorithms

Figure 2-4 illustrates the basic terminologies related to the GA and their interconnection. They have briefly explained below.

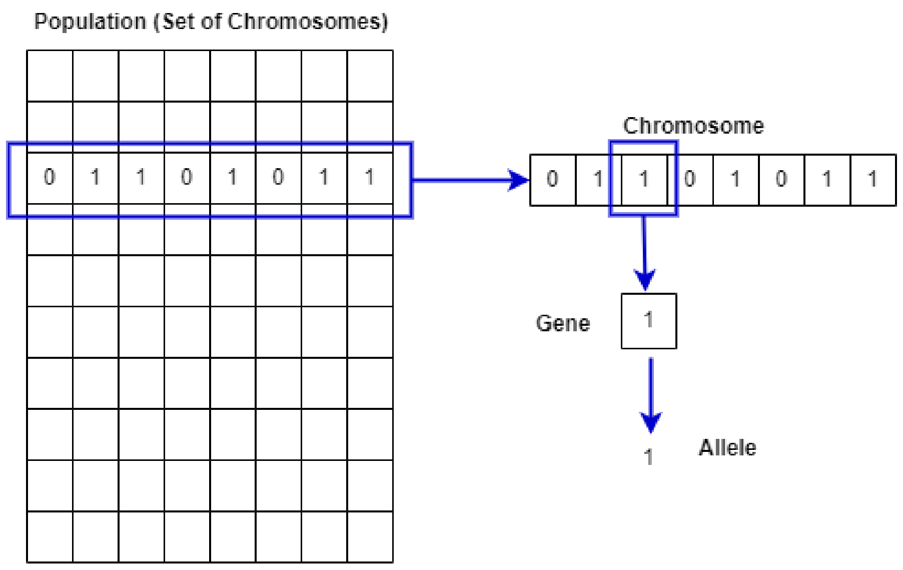


Figure 2-4: Basic Terminologies of GA

Population refers to the subset of all feasible or probable solutions that can address a specific problem in each generation. A chromosome represents one of the solutions in the population for the given problem, and it is formed by a collection of genes. A gene is a constituent element of the chromosome. An allele is the value assigned to a gene within a specific chromosome.

2.6.2 Fitness Function

The fitness function is used to determine the fitness level of an individual within a population. It signifies the ability of an individual to compete with other individuals. Individuals are evaluated at each iteration based on their fitness function.

In a genetic algorithm, the best individuals based on fitness function mate in order to produce offspring that are superior to their parents. Here, genetic operators have a role in modifying the genetic composition of the next generation. Commonly used genetic operators are given in the following subsections.

2.6.3 Selection

After computing the fitness of each member in the population, a selection procedure is employed to choose which individuals will be allowed to reproduce and generate the seed for the next generation. There are several selection techniques, including selection using a roulette wheel, tournament selection, and rank selection. For example, tournament selection selects k individuals at random from the population, then selects the best candidate from among them to be a parent. To choose the next parent, this process is repeated. Figure 2-5 depicts the selection process visually. The fact that this selection method supports even negative fitness values makes it popular in research (Luke, 2017).

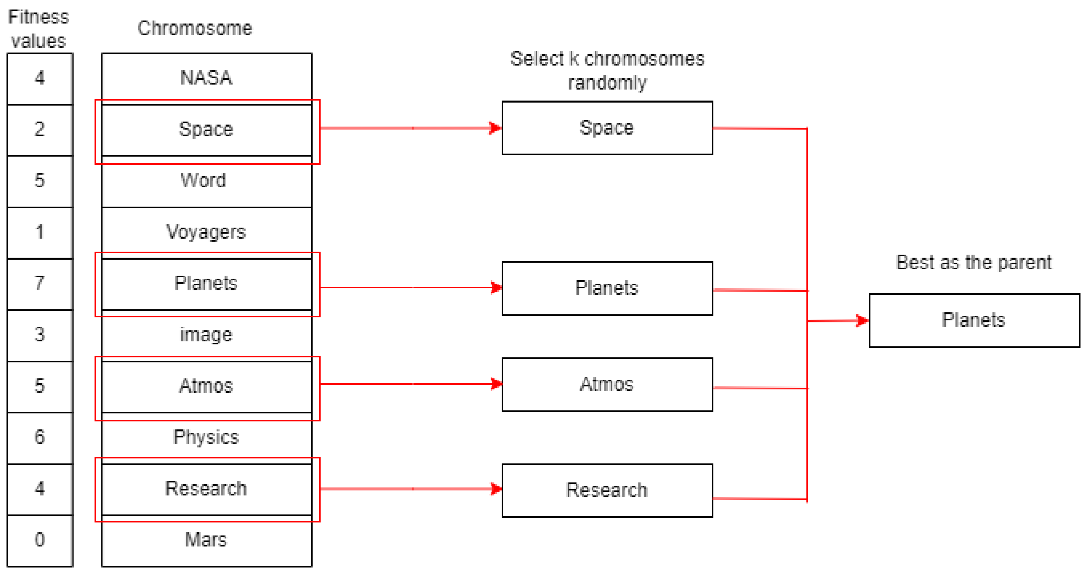


Figure 2-5: Overview of Tournament Selection

2.6.4 Crossover

Reproduction is the method by which a child is created once parent chromosomes are chosen during the selection process. In this phase, genetic algorithms employ two different types of variation operators on the parent population. Crossover and mutation are the two involved operators. In reproduction, the crossover operator plays the most important role in making new individuals. This is done by swapping the genes of two individuals from the pool who were chosen at random. The objective is to produce a large number of strong offspring. Different kinds of crossover operators exist. One-point crossover, multi-point crossover, and uniform crossover are some of them. One-point crossover, for example, picks a random crossover point and swaps the genes of two parents to make new offspring. One point crossover is shown in figure 2-6 below.

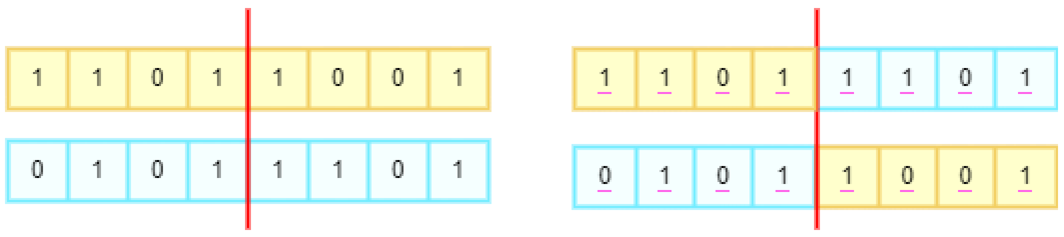


Figure 2-6: One Point Crossover

2.6.5 Mutation

Mutation is a way of introducing diversity to genetic population. It is doing by applying a random tweak to a chromosome to get a new solution. It has been noted that, in contrast to crossover, mutation is necessary for the GA to converge. There are different types of mutation operators available. Some of them are Bit-Flip mutation, Random resetting, Swap mutation, and Inversion mutation. As an example, in Bit-Fit mutation, we select one or more random bits and flip them. The following figure 2-7 shows the Bit-Flip mutation.

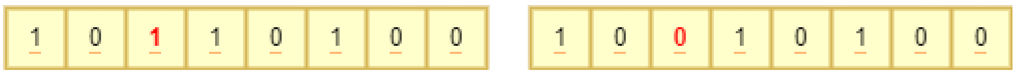


Figure 2-7: Bit-Flip mutation

2.7 Review on Classification and Clustering Methods

2.7.1 Classification Methods

Many different types of classification strategies have been invented by researchers in the past. Also, their origins are very different. This is due to the contributions of experts from various knowledge disciplines. Table 2-1 shows some of the popular classifiers and their categories based on their origin or key characteristics (Aggarwal & Zhai, 2012; Gasparetto et al., 2022).

Table 2-1: Types of Classifiers

Category	**Algorithms**
Tree-based	C4.5, Random Forest (RF)
Rule-based	PART, JRip
Distance-based	k-Nearest Neighbours (kNN)
Function-based	Support Vector Machine (SVM),
Probabilistic	Naïve Bayes (NB), Logistic Regression
Neural based	MultiLLP, LSTM
Genetic Algorithm	Evolved Search Query (eSQ)

More details for some of the popular methods following different approaches have been given below. These algorithms have been used in later chapters to compare results with the improved version of the GA-based classifier.

SVM Classifiers: Support Vector Machine (SVM) is a machine learning method based on statistical learning theory that was developed in the mid-1990s. The SVM classifier is currently one of the most widely used classifiers. The benefits of support vector machines include being effective in high-dimensional spaces, remaining effective when the number of dimensions exceeds the number of samples, using a subset of training points in the decision function (called support vectors), making it memory efficient, and allowing different kernel functions to be specified for the decision function.

SVM is a non-probabilistic binary classifier and does not support multiclass classification natively. But the same principle has been extended to classify multiclass datasets. One-to-one and one-to-rest are two methods for enabling multiclass classification. The one-to-one approach breaks a multiclass classification into one binary classification problem per each pair of classes. But the one-to-rest approach breaks multiclass classification into one binary classification problem per class.

The SVM algorithm creates a model using a pre-labelled training set. It is aiming to find a hyperplane (decision boundary) in a multi-dimensional space (number of dimensions equal to number of features) that distinctly classifies the data points. Then, new data points are mapped into the same space and classified based on which side of the gap they fall.

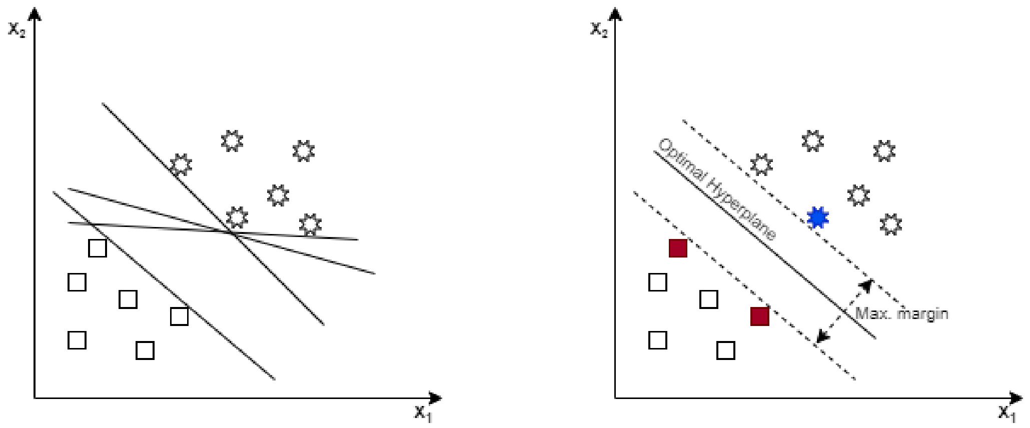


Figure 2-8: Hyperplanes for two datasets

For two classes of a dataset, there are many possible hyperplanes that could be chosen as illustrated in Figure 2-8. The objective is to find a plane that has the maximum distance between data points in both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence. Support vectors are data points that are closer to the hyperplane and have an impact on the hyperplane’s position and orientation. By utilising these support vectors, it maximises the classifier’s margin. Results from experiments in (Joachims, 1998) demonstrate that SVM outperforms other well-known algorithms in document classification. SVM has therefore been used in the study to compare the outcomes of the experiments.

Neural Network Classifiers: Neural networks (NN) are used in a wide variety of domains for the purposes of classification, and they can be classified into different types of networks. The perceptron is the oldest and simplest form of neural network with a single neuron. A commonly used feedforward neural network or multilayer perceptron (MLP) consists of an input layer, a hidden layer, and an output layer. It is widely used in natural language processing. In the context of text data, words or terms with related details are considered for creating input feature vectors. An input vector of document features is transformed into output by layers of units (neurons) that make up a neural network. Each unit receives an input, processes it through a function (often nonlinear), and then transfers the result to the following layer. In general, networks are characterised as being feed-forward, meaning that there is no feedback to the previous layer and each unit feeds its output to all the units on the layer next to it. Signals passing from one unit to another are given weightings, and it is these weightings that are adjusted throughout the training phase to adapt a neural network to the specific classification problem at hand.

Naive Bayes classifiers: The Bayes theorem in statistics has been used to construct the Naive Bayes (NB) classifier. One of the key presumptions of its methodology is the independence of features or words in documents. This assumption makes the computation more efficient. It is assumed that the conditional probability assigned to a word is independent of the conditional probabilities assigned to other words in the same category. The NB algorithm computes the joint probabilities of words and categories and then estimates the probabilities of categories for a given document. Based on the probabilities, it assigns a class label to a new document.

Decision Tree-based Classifiers: A decision tree (DT) is constructed using the different properties of individual documents or collections of documents. These properties are identified using various methods (e.g., information gain) and create hierarchical partitions of the underlying data space, which is called a decision tree. The hierarchical partitioning of the data space is intended to provide class partitions that are more skewed in terms of their class distribution. It finds the division to which a specific text instance is most likely to belong and uses that partition for classification. Generating an optimal hierarchy is one of the main challenges in DT induction algorithms.

ID3 is one of the primary algorithms for DT construction, founded by Quinlan (1986). It starts tree induction with the original training dataset (S). On each iteration, the algorithm determines the properties of the data space by iterating through all the unused attributes of the dataset S, using the entropy H(S) or information gain IG(S) of each attribute. The attribute with the lowest entropy or highest information gain value is then chosen. Subsets of the data are then created by dividing or partitioning the set S according to the chosen attribute. The algorithm keeps repeating itself on each subset of data while considering only attributes that have never been chosen previously.

The ID3 algorithm was developed further, leading to the C4.5 algorithm, which has gained enormous popularity in the machine learning, data mining, and natural language processing fields. The C4.5 algorithm is the same as J48 implemented in the Java-based Weka machine learning platform, which is employed in several experiments in this study and whose findings are presented in the following chapters.

Random Forests (RF) is an ensemble learning method used in document classification. From the training data, it generates a large number of decision trees, and then it combines the predictions of each one. The class chosen by the majority of decision trees is the output of the random forest. Generally, random forests outperform the decision trees, but they are considered a “black box” model for commercial applications.

Rule-based Classifiers: Rule-based classifiers are determined by the word patterns that are most likely to be related to the different classes. Decision criteria are made up of sequence conditions linked by ORs, also known as Disjunctive Normal Form (DNF) rules, that denote the presence or absence of terms in the testing document set, while the clause head denotes the category. There are two types of rule-based classifiers, and (Koklu et al., 2015) emphasise the distinction between these two. They are called direct and indirect rule generative methods depending on whether rules are generated from the classifier as an output or through an addition process.

Genetic Algorithm-based classifiers: Genetic Algorithm-based (GA) methods follow an iteratively progressing approach to develop a population towards achieving the desired end. Such developments are often inspired by biological mechanisms of evolution. The genetic operators; selection, crossover (recombination) and mutation are applied to the individuals to breed the next generations. Fitness functions are used in order to measure the strength of an individual. When the fitness level is higher for a particular individual, there is a higher probability that the individual will be selected to take part in creating the next generation. Thus, the genetic material of strong individuals will survive throughout the evolutionary process until the end. GA methods' unique ability to find an optimal or near-optimal solution from a population of solutions has been used to develop a few document classifiers with extremely high human friendliness (Hirsch, 2010).

2.7.2 Clustering Methods

Clustering is an unsupervised learning technique used when segmenting datasets based on their similarities. Document clustering is a main branch under clustering, and it is important in many real-world application developments. This section describes a few widely used clustering algorithms that were used for result comparison in chapter 6.

k-means Clustering: In Lloyd (1982), the authors proposed a local search solution that is still very widely used. Usually referred to as k-means, Lloyd's algorithm begins with k arbitrary centres, typically chosen randomly from the data points. Each point is then assigned to the nearest centre, and each centre is recomputed as the centre of mass of all points assigned to it. These two steps are repeated until the process stabilizes. Algorithm 2 illustrates typical k-means clustering.

Algorithm 2: k-means

```
1.  Select k random points as the initial centroids.
2.  Repeat
3.      Form k clusters by assigning all points to the closest centroid.
4.      Re-compute the centroid of each cluster.
5.  Until The centroid does not change
```

In Arthur and Vassilvitskii (2007), authors presented enhanced version of the popular k-means algorithm called k-means++. It uses a randomised seeding technique, which is a specific way of selecting initial centres in step 1 in algorithm 2. The rest of the (2 to 5) steps of the algorithm remain as in standard simple k-means.

Farthest first is another initialization method used in the k-means algorithm. To find optimal k centres when initialising k-means, the farthest first traversal method has been used, which is described by Dasgupta and Long (2005).

Expectation-Maximization Clustering: Each cluster is modelled mathematically as a parametric probability distribution. So entire data set is a mixture of these distributions. Therefore, it is possible to cluster the dataset using a finite mixture density model of *k* probability distributions and identify clusters. But the problem is estimating the parameters of probability distributions to fit the data set. In Han et al. (2011) , authors presents the Expectation-Maximization (EM) algorithm can find best fit parameters using an iterative refinement algorithm. Algorithm 3 shows the EM clustering method.

```
Algorithm 3: Expectation-Maximization
1. Make an initial parameter vector: This involves randomly selecting k objects to
   represent the cluster centres and making guesses for the additional parameters.
2. Repeat
3.     Expectation step: For each database record x, compute the membership
       probability of x in each cluster h = 1…k.
4.     Maximization step: Update mixture model parameter (probability weight).
5. Until satisfy the stopping criteria
```

Hierarchical Clustering: For the Agglomerative hierarchical clustering, from the beginning, each data point is a cluster (Han et al., 2011). Next, the two nearest are joined repeatedly till they form a single cluster. To measure the distance between two clusters, several distance measures or linkage types have been implemented. The MEAN link type outperformed the others in terms of document clustering. Algorithm 4 depicts agglomerative clustering.

```
Algorithm 4: Agglomerative Clustering
1. Compute the proximity matrix
2. Let each data point be a cluster
3. Repeat
4.     Merge the two closest clusters
5.     Update the proximity matrix
6. Until only a single cluster remains
```

2.8 Genetic Algorithm-based Classification Methods

Genetic algorithms are heavily used in optimization problems. Some researchers have used this capability to improve supervised or unsupervised learning algorithms in the past. One of the common approaches observed is that GA is applied to find optimal features or optimal parameters to fine-tune the overall classification output. In Gaber et al. (2022) authors have used a GA program to find optimal parameters in their process of classification. Kumar & Gavrilova (2021) presented a GA-based algorithm to reduce the feature set and increase the efficacy of the features extracted. Also, Bidi & Elberrichi (2017) provides details of a feature selection method using a genetic algorithm. In Hong et al. (2015) authors have proposed a GA-based method to find the best features set for document classification and clustering.

Chauhan et al. (2021) propose a document classification method combining GA, and SVM classifiers using an ensemble learning approach. Authors, Khaleel et al. (2016) has been developed an automatic document classification system based on the genetic algorithm. The genetic algorithm classifier generates a predefined number of optimized classification rules using the training documents.

The Olex developed by Rullo et al. (2007), is a novel method for the automatic construction of rule-based text classifiers. Olex relies on an optimization algorithm whereby a set of (both positive and negative) discriminating terms is generated for the category being learned. Such terms are then used to construct a classifier. The Olex-GA presented by Pietramala et al. (2008) used a genetic algorithm for the induction of rule-based text classifiers of the form "classify document *d* under category *c* if $t_1 \in d$ *or ... or* $t_n \in d$ and *not* $(t_{n+1} \in d$ *or ... or* $t_{n+m} \in d)$ holds", where each t_i is a term. Olex-GA relies on several individual rule representations per category and uses the F-measure as the fitness function. Rullo et al. (2009), authors have presented their results of an improved version of Olex-GA in this paper. The Olex system also provides classifiers that are accurate, compact, and comprehensible. The following section has explained the eSQ engine which is developed using GA based method.

2.8.1 Evolution of eSQ-based methods

In this subsection, we will explain the GA-based classification method that we plan to improve during this research. The development of this new GA-based classification method, recently named eSQ classifiers, began in early 2000. It has been experimentally tested over time and has shown significant improvements. This method was initially developed by Hirsch et al. (2005), who published the results of the primary design for evolving text classification rules using a genetic program. The program was designed to create compact classification rules based on combinations of N-Grams (character strings). Some key features of the genetic program (GP) are as follows:

- The basic building block of the rule was N-gram (sequence of N characters).
- Rules generated by the GP program are evaluated as true or false for a specific document.
- Boolean functions (EXISTS, AND, OR, and NOT) are used in the GP functions.
- A classification rule must be evolved for each category, denoted as "c." Fitness is then determined for GPs that produce classification rules that are true for training documents in category c, but not true for documents outside category c. Thus, the documents in the training set represent the fitness cases.

According to the literature, this approach has some similarities to the Olex-GA developed by Pietramala et al. (2008). However, while the Olex-GA considers both positive and negative terms in classification rules, the Hirsch approach only uses positive terms.

This GP program was further improved by Hirsch et al. (2007), who described a slightly different method for generating accurate, compact, and human-understandable text classifiers. In this new version, text datasets were indexed using Apache Lucene, and Genetic Algorithms (GA) were used to construct search query-based classifiers. The GA produces queries that effectively classify documents into specific categories when evaluated against a set of training documents.

The next major improvement made to the GA system, published in Hirsch et al. (2010), involved experimenting with the SpanFirst queries supported by the Lucene framework. In a SpanFirst query, a maximum distance or span is specified, along with a sequence of terms. The search engine then looks for documents or passages where those terms occur within the specified span, with the first term appearing first in the sequence. With this change, the new version of the system generates evolved search queries that are more compact and human-readable compared to the queries generated from the N-gram-based method. The approach was evaluated using standard test sets Reuters-21578 and Ohsumed, and it was compared against several classification algorithms. The results showed that the performance was comparable to popular classification algorithms.

We also discovered that this method is not only human-readable but also modifiable by end-users. This feature is particularly useful in certain applications, especially when end-users want to observe the effects of classification results with slight changes to the rules. In such situations, search query-based methods are highly valuable. In our literature review, most other classification methods do not support modifying classification criteria and observing the outcomes of classifiers again. This motivated us to further study and investigate the possibility of improving search query-based methods. Furthermore, we have observed that there are potential avenues for improvement in the Hirsch approach by integrating feature selection methods and exploring its applicability to other human languages. The literature review has also highlighted the possibility of extending this approach for cluster analysis, opening up new directions for research and development.

2.9 Genetic Algorithm-based Document Clustering Methods

Document clustering is an unsupervised learning technique employed to organize a large collection of documents into clusters or subsets. Each cluster contains similar documents, while dissimilar documents are placed in different clusters. Among nature-inspired optimization algorithms, Genetic Algorithms have been used extensively for document clustering in the past (L. Abualigah et al., 2020; Wei et al., 2009). In the context of document clustering, GA algorithms have been utilized to determine cluster centres and the optimal set of features for dimensionality reduction. However, GA algorithms often face the issue of converging to local solutions (local minima) instead of a global solution. Consequently, the challenge in GA approaches lies in attaining a global optimum solution and mitigating local minima. The list of previous studies below highlights various strategies aimed at addressing this concern and presents several approaches to document clustering using GA-based methods.

In Nooraeni et al. (2021), a k-prototype algorithm was improved using GA capabilities to cluster mixed data. The authors have identified two main weaknesses of this algorithm which are finding the cluster centre accurately for categorical attributes when using the mode and secondly preventing the algorithm from stopping at a local optimum solution. They have proposed a method to overcome the second problem. So, the proposed method is to implement a GA to search for the global optimum solution.

In Mustafi et al. (2022), the authors have proposed an algorithm to improve the separation of clusters using the concept of nearest neighbours separation. They also proposed a parameterized fitness function which can be tuned based on inter and intra-cluster distances of clusters. In addition, they developed a GA algorithm for document clustering and compared it to a well-known k-means algorithm.

In Akter et al. (2013), the authors proposed a method to get rid of the local minima. The authors had partitioned the dataset into groups and the GA-based document clustering algorithm was applied to the individual partitions separately. They then applied a second GA algorithm to the outcome received from previous executions. They were able to minimize local minima issues in this approach. Subsequently, they have extended their work in Akter et al. (2017, 2021). These papers attempted to divide cluster centroids using crossover operations in the GA algorithm which helps to introduce different variations to the population and further minimize the issue of local minima.

In Abualigah et al. (2016), the authors proposed a GA-based method to find a subset of informative features from a document collection and perform clustering. It is named FSGATC. Experiments have been conducted using text datasets and they have compared the results with the classical k-means algorithm. In their next paper Abualigah et al. (2017), presented results received comparing three different feature selection algorithms with a feature weighting schema and dynamic dimension reduction for document clustering.

In Shi (2018), authors have proposed a method to classify data by combining the genetic algorithm and the k-means algorithm. They were aiming to improve the effectiveness of the k-means algorithm by introducing a method to find initial cluster centres. For this, they have used a sorted neighbourhood method to pre-process the data which allows for the detection of duplicate records so they can be eliminated. They then created a method that combined k-means and GA. The k-means algorithm was used for clustering, and GA was used to optimise the clustering processes by removing unwanted features.

In Pizzuti et al. (2016), authors present a GA-based technique to form clusters. Their GA algorithm applies the k-means principle for dividing data points into groups if they have high similarity. An experiment using this method with four different fitness functions on multiple datasets is then compared with k-means performances.

In Ding et al. (2016), the authors have proposed a clustering method enhancing the fuzzy c-means (FCM) algorithm used in pattern recognition. A kernel-based fuzzy c-means (KFCM) clustering algorithm is presented to address the issues with the FCM clustering algorithm. It is based on GA optimization, which combines the enhanced genetic algorithm and the kernel technique (GAKFCM). In this technique, the initial clustering centre is first optimized using the enhanced adaptive genetic algorithm, and then the KFCM algorithm is used to direct categorization in order to enhance the clustering performance of the FCM algorithm.

In Zeebaree et al. (2017) and Sheikh et al. (2008), authors have reviewed papers published about the GA-based clustering approach in the past. It is evident that GA has been studied heavily and used for clustering problems in many problem domains. However, it is noted that it is difficult to compare new methods with previous GA-based methods due to the unavailability of source codes or libraries and performance has been presented using very different datasets.

2.10 Importance of Explainable Machine Learning

2.10.1 Section Overview

The effectiveness of a predictive model is measured not only by its performance or predictive power but also by how well its decision-making model is explainable. Therefore, this section is devoted to delving deeper into explainable machine learning. The section begins with three scenarios demonstrating potential misuse in the development of machine learning technologies. This highlights the low level of explainability of the predictive models as a major limitation in those systems. Due to the consequences of this, authorities revised data protection rules in response to the negative impact of predictive models on society. We briefly explain recent changes to the data protection rules meant to safeguard users.

2.10.2 Criticisms Against the Analytic-based Applications

This subsection presents three real-world scenarios that demonstrate the genuine necessity for developing explainable machine-learning models.

Scenario 1: The Durham Constabulary Harm Assessment Risk Tool (HART)

The use of algorithmic decision-making tools is popular in some problem domains but rarely used in the policing context, especially in the UK. In a recent study based on the openness of information, it was found that just 14% of UK police forces used computational or algorithmic data analysis or decision-making in relation to intelligence analysis (Oswald & Grace, 2016). According to Oswald et al. (2016), there are now three primary uses for intelligence analysis in the context of police. (i) Macro-level predictive policing that includes strategic planning, prioritisation, and forecasting; (ii) Linking and evaluating operational intelligence, which may include, for example, crime reduction activities; and (iii) Individual-specific decision-making or risk evaluations. One particular example was presented by Oswald et al. (2018), who proposed an algorithmic risk-assessment tool (i.e. (iii) above) known as the 'Harm Assessment Risk Tool' (or 'HART'). It is based on the premise that the application of algorithmic tools in the context of police might result in a better outcome in terms of public safety, legality, and cost.

Together with Durham Constabulary, statistical scientists from the University of Cambridge created the instrument. It has been designed to assist custodial officers in estimating the likelihood of future offending and to make arrestees predicted to pose a moderate risk eligible for the Checkpoint programme of the police force. Checkpoint is an intervention presently being evaluated by the Constabulary. It is an "out-of-court disposition" (a method of dealing with an offence that does not need court prosecution) intended to reduce future criminal behaviour.

The HART model has more than 4.2 million decision points, all of which are extremely reliant on the ones that came before them in the tree structure. This information could be made freely available to the public, but understanding them would require a substantial amount of time and effort. It is becoming increasingly challenging to explain to non-computer scientists and non-statisticians how a machine learning forecasting model reaches its conclusions, and the potential for misunderstanding and even purposeful distortion is high. Consequently, it is unlikely that the general public can comprehend machine learning models.

Scenario 2: Facebook - Cambridge Analytica Scandal:

Cambridge Analytica (CA) Ltd was a British political consulting firm. They used data mining, data brokerage, and data analysis with strategic communication during the electoral processes (‘Cambridge Analytica’, 2022). For this, CA has developed an extremely powerful software solution to predict and influence the voters’ choice at the election. And it’s found that the US presidential election campaign in 2016 and the election held seeking UK voters’ views about the EU referendum in 2016 are two main cases where CA played a major role (Carole Cadwalladr & Emma Graham-Harrison, 2018).

CA has harvested millions of user accounts from Facebook in order to construct models of individuals and their inner demons. This information was utilised to influence political campaigns. The data was obtained using the “This Is Your Digital Life” app, whose users consented to have their data taken for academic purposes when taking a personality test. However, the software also collected information on the test-takers’ Facebook acquaintances, resulting in the collection of tens of millions of data points. Facebook's “platform policy” restricted the acquisition of friends’ data to enhance the app user experience and prohibited its sale or usage for advertising.

Together, the algorithm and database became an influential political instrument. It enabled a campaign to identify potential swing voters and design more persuasive communications. The eventual outcome of the training set is the development of a “gold standard” for interpreting personality from Facebook profile data. They aim to develop a database of 2 million “matched” profiles that are recognisable and linked to election registers across 11 states, but there is an opportunity for expansion.

A current study into the use of data analytics for political purposes is examining how political parties and campaigns, data analytics corporations, and social media platforms in the United Kingdom utilise and analyse the personal information of voters to micro-target them.

Scenario 3: COMPAS Model for Predicting Recidivism Risk:

Risk assessment tools are becoming more and more popular within the criminal justice system. In some places in the United States, these tools are even used for sentencing. For it to be commonplace to use such tools, it is crucial to make sure that they are free from discriminatory biases and unethical practices. COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) is a support tool used for predicting recidivism risk or else risk that a criminal defendant will re-offend. However, a team from ProPublica investigated the COMPAS and found that it is biased against black defendants (Angwin et al., 2016).

The risk scoring scheme used by COMPAS has a scale from 1 to 10, with 10 being the highest risk. Additionally, based on risk scores of 1 to 4, 5 to 7, and 8 to 10, it has classified defendants as low risk, medium risk, and high risk, respectively. The system takes 137 factors including age, gender and criminal history of defendants as the input to make the prediction of recidivism. But race was not a direct input feature of this predictive model.

For the experiment, the investigation team has taken more than 10,000 criminal defendants in Broward County, Florida. They have compared predicted recidivism rates of COMPAS with the rates that occurred over a two-year period. The Data were collected from defendants using a questionnaire and fed into the COMPAS system for predictions. The tool was developed to predict multiple risk scores including “Risk of Recidivism” and “Risk of Violent Recidivism”.

They compared actual recidivism rates after two years with the predicted rate by the COMPAS tool and found that accuracy is 61 per cent for risk of recidivism, but only 20 per cent for violent recidivism. When the tool forecasts who will re-offend, the accuracy of black and white defendants is roughly the same, but it misclassifies the black and white defendants differently after a two-year follow-up period. Black defendants were frequently predicted to be at a higher risk than they actually were. Their analysis found that black defendants who did not recidivate over two years were nearly twice as likely to be misclassified as higher risk compared to their white counterparts. And they have made more analyses that showed further system bias towards black defendants.

This investigation team’s in-depth analysis of the outcome reveals that obviously there is discrimination in COMPAS but the technical details of why this happened is absent. This is because of the black-box nature of both the model-building process and the use of the resultant model for predictive purposes.

General Review of Three Case Studies:

The three case studies considered above: The Durham Constabulary Harm Assessment Risk Tool, Facebook - Cambridge Analytica Scandal and COMPAS Model for Predicting Recidivism Risk, confirm that there are challenges to using predictive models in real-world applications. They must be developed further to meet the expectations of society. Some research efforts have even emphasized these issues. In Mittelstadt et al. (2016), authors have highlighted the following ethical concerns raised in society due to these models includes: (1) inconclusive evidence leading to unjustified actions; (2) inscrutable evidence leading to opacity; (3) misguided evidence leading to bias; (4) unfair outcomes leading to discrimination; and (5) transformative effects leading to challenges for autonomy and informational privacy.

However, the root cause of these all-unethical incidences is due to the black box nature of these predictive models. It is inscrutable since it is a black box. This can lead to other problems in the public sector when these tools are used. Therefore, it requires a combination of approaches to prevent this. Obviously, a technical solution to minimize the opacity of these predictive models is required. In addition, independent audits and context-specific regulatory frameworks should be in place to overcome some requirements of society in using such methods. The next section, briefly explains some of the initiatives taken to minimize the resistance against the predictive models.

2.10.3 Recent Changes in Data Protection Regulations

General Data Protection Regulation (GDPR)[1], which is the toughest privacy and security law in the world. It was drafted by the European Union (EU). Any organisation that targets or gathers data about individuals in the EU is subject to its requirements. This regulation has been in action since May 25, 2018, as data protection legislation that aims to make Europe ready for the digital era. It introduces rules for processing, storing, and managing the data of people who are within the European Union. This legislation only protects the privacy of EU citizens' data. But its impact is beyond the EU.

The European Convention on Human Rights, which was drafted in 1950, is where the GDPR's history begins, and it includes the right to privacy. The EU's development of the GDPR was based on the principle that "everyone has the right to respect for his private and family life, his home, and his correspondence." The EU realised the need for data protection once the Internet was invented. So, in 1995, it passed the European Data Protection Directive, which established minimum standards for data privacy and security and served as the foundation for implementing laws in each of the member countries. Facebook formed its operations in 2006 and created a new paradigm for social media. A Google user filed a claim in 2011 because it had

[1] https://gdpr.eu/tag/gdpr/

scanned her emails. With this incident, the European Union's data protection authority declared two months later that the 1995 directive needed to be updated and demanded “a comprehensive approach to personal data protection.” As a result, GDPR was introduced in 2018.

In Article 5.1-2 of the GDPR, there are seven data protection and accountability principles[2] to adhere to when processing data. In brief, they are as follows:

i. Lawfulness, fairness and transparency — Processing must be lawful, fair, and transparent to the data subject.
ii. Purpose limitation — Process data for the legitimate purposes specified explicitly to the data subject when it was collected.
iii. Data minimization — Collect and process only as much data as absolutely necessary for the purposes specified.
iv. Accuracy — Keep personal data accurate and up to date.
v. Storage limitation — Only store personally identifying data for as long as necessary for the specified purpose.
vi. Integrity and confidentiality — Processing must be done in such a way as to ensure appropriate security, integrity, and confidentiality.
vii. Accountability — The data controller is responsible for being able to demonstrate GDPR compliance with all of these principles.

GDPR is mainly to protect consumers in the EU. As a result, the EU has tightened requirements for data controllers and processors. And also, organisations should offer the following privacy rights to EU residents:

i. The right to be informed
ii. The right of access
iii. The right to rectification
iv. The right to erasure
v. The right to restrict processing
vi. The right to data portability
vii. The right to object
viii. Rights in relation to automated decision-making and profiling.

With these new GDPR conditions, companies have to rethink how to meet these requirements and implement them. It takes lots of manpower and resources to update technological platforms, update privacy policies, and change marketing and other practices. More importantly, this has emerged as a significant barrier to the advancement of artificial intelligence and machine learning (AIML). Because highly accurate predictive models are not explainable at present. Therefore, GDPR will affect

[2] https://gdpr.eu/what-is-gdpr/

the development of intelligent applications by increasing their costs and limiting their features. The three main challenges faced by the computing industry are as follows:

1. It is a technically challenging problem to explain to the general public the functionality of a complex decision-making system.
2. Sharing details of decision-making criteria may reveal some trade secrets and
3. Violate the rights and freedoms of others since predictive models have been developed using others' data

Due to the wide popularity of artificial intelligence and machine learning (AIML) related applications in the modern digital era, facilitating the growth of the field has become a necessity. Therefore, current regulations are taken into consideration for discussion among researchers and government policymakers (Lord Clement-Jones, 2020; Science and Technology Committee, 2016).

Since GDPR has a big impact on AIML applications, we are investigating the human friendliness of classification methods in the following sections. To make it more organised, a taxonomy for popular classification methods has been developed in the following section based on their human friendliness.

2.11 Human Friendliness based Taxonomy for Classification

With the popularity of machine learning research, accuracy and efficiency dominated when designing and developing classification methods. However, it has been discovered that some applications require an equally important additional characteristic called human friendliness. We define "human friendliness" as the ability of end users to interpret and modify the classification decision-making criteria. The following taxonomy, depicted in Figure 2-9, was created based on the human friendliness of classifiers.

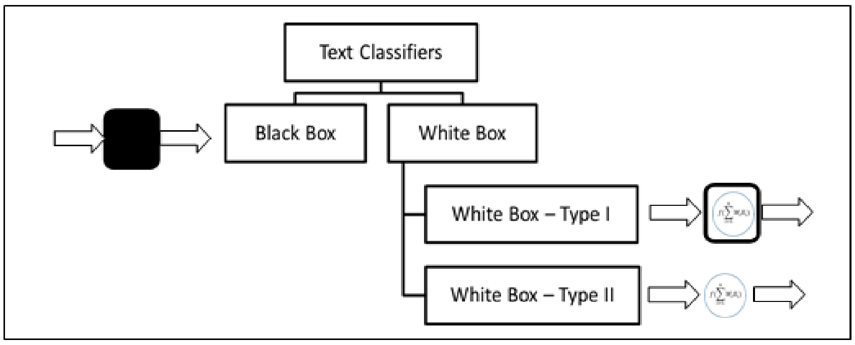


Figure 2-9: Taxonomy for Classification

The two principal branches of this taxonomy are “Black-box” and “White-box”. Additionally, Type I and Type II categories are included in the white-box classifiers. These make use of the following definitions: If the mechanism of the classification is understandable to end users, the classifiers are human-interpretable. And if the classification method is adjustable based on users’ views, it is modifiable.

2.11.1 Black-box Type

End users cannot understand the decision-making criteria if it is a black-box type classifier, and it cannot be modified based on their preferences. i.e., they are classifiers that are neither human-interpretable nor modifiable. As a result, when using black-box classification methods, end users are blind to how the classification was performed and find it difficult to fine-tune the model. Support Vector Machine (SVM) and Artificial Neural Network (ANN) based models are good examples of black-box type classifiers.

2.11.2 White-box Type

White-box classification models are human interpretable classifiers. So, it has a higher level of human friendliness compared to the black-box type. Classifiers in this category can be further divided into two groups:
Type I classifiers can be interpreted by humans but are difficult to fine-tune or modify.
Type II classifiers can be interpreted by humans and modified as needed by domain experts/end users.

For example, end users can derive human interpretable rules from a decision tree, but there is a limitation in how to incorporate end-user knowledge or feedback and reconstruct the tree to produce an optimal decision tree. As a result, a decision tree is classified as a white-box type I classifier. Search query-based classifiers and rule-based classifiers are good examples of white-box type II classifiers (see section 2.14 for more details). Rules represent a specific category and have the capability of categorising data into pre-labelled groups. It is both human-interpretable and modifiable, allowing domain experts’ knowledge to be easily applied. As a result, white-box Type II classifiers have the highest level of human friendliness in the new taxonomy. This taxonomy organises classifiers in increasing order (left to right) of human friendliness as follows:

Black-box Type → White-box Type I → White-box Type II

$\xrightarrow{\text{increasing order of human friendliness}}$

The taxonomy developed above was used in the following sections to review and organise the classification methods based on their human friendliness.

2.12 Black-box Type Classifiers

Adopting various methodologies, several popular classification techniques have been created in the last few decades. These have their roots in various fields of knowledge. Therefore, the mechanism used has its own unique characteristics. This section reviews the human-friendliness of commonly used classifiers.

SVM Classifiers: SVM classifiers are one of the most robust predictive methods used in machine learning. Details of this technique were covered in Section 2.7.1. In brief, SVM uses linear or non-linear functions to partition the high-dimensional data space for different classes or categories. In this method, the main challenge is identifying optimal boundaries between categories. The founder, Joachim (1998), states that “it is not necessary to have humans involved in parameter tuning as there is a theoretically motivated parameter tuning setup in place”. This automatic parameter tuning hides internal behaviour from end users. It stands to reason that SVM predictive models are not interpretable by end users. So, it is classified as a block-box type and also one of the lowest levels of human-friendly classifiers.

Neural Network Classifiers: Artificial Neural Networks (ANN) are popular classification techniques used in many software solutions. These models are inspired by the biological neural networks of animal brains. In ANNs, connectivity between the input layer and output layer is created through hidden layers. Due to the complex nature of connectivity, it makes ANN more complicated to understand. The activation function decides the output of each node for the set of inputs, which makes ANN a non-linear classifier. Also, input values pass through many layers of multiplication with weights. For a single prediction, there can be lots of mathematical operations based on the selected architecture of the neural network. So, it is hard to humans can follow the exact mapping from input to prediction. This complex nature of ANN-based classification techniques hides the criteria of decision-making from users. Therefore, ANN methods are opaque and not human-interpretable. As a result, it is also a black-box type classifier with the lowest level of human friendliness.

Ensemble Classifiers: Ensemble models in machine learning aggregate the predictions produced by different models to increase overall performance. They combine models in quite diverse ways. It ranges from simple approaches such as majority voting, averaging, and weighted averaging to more complex techniques like stacking and blending, bootstrap sampling and bagging, and boosting. The popular Random Forest classifier, for example, is an ensemble model that employs bagging. XGBoost and AdaBoost, on the other hand, employ boosting approaches to merge different models. Because of the complexity developed when multiple models are integrated, ensemble learning becomes a black-box type. A decision tree, for example, is a white-box type I in our taxonomy, changing to a black box when it becomes a random forest (hundreds of decision trees combine using the bagging technique). Ensemble models are capable of producing very accurate prediction models, but they suffer from low human friendliness due to the lack of interpretability and modifiability.

Our research indicates that a significant number of attempts have been made in the past to transform black-box models into a higher level of human-friendly ones. In section 2.13, we describe some of these attempts.

2.13 White-box Type I Classifiers

Decision Tree-based Classifiers: The decision tree (DT) is a hierarchical view of the training set. Class labels or categories are mapped to leaf nodes in the tree. Parent nodes split instances for child nodes based on impurity measures. Information gain and the Gini index are commonly used in decision tree induction algorithms. When all instances belong to a child node, it stops splitting and makes it a leaf node; otherwise, nodes are split recursively.

The decision tree structure itself provides a natural visualization of a predictive model. Therefore, it is transparent to end users. This is one of the main differences compared to most of the other classification methods. To find categories, begin at the root of the tree and traverse to the child nodes until you reach the leaf node. A leaf node is a category or class label, and all parent nodes are features of the training set, and they have been influenced to distinguish data objects depending on a specific criterion from other categories. As a result, it is possible to calculate how much each feature contributed to a specific classification result. Decision trees not only have natural visualization, but it is also possible to determine the contribution of each feature to prediction. Decision trees have not only transparency but also a model that is highly interpretable by humans. But still, this model is difficult to fine-tune by end users and lacks easy modifiability.

Naïve Bayes (NB) is a well-studied classifier in probabilistic machine-learning research. The method used in the NB classifier is that the joint probabilities of features and categories are used to compute the probabilities of categories for a given instance. It bases this on the assumption of feature independence. That is, the conditional probability of a feature for a given category is assumed to be independent from the conditional probabilities of other features given in that category.

Because of the independence assumption of features, the naive Bayes classification method can compute how much each feature contributes to class prediction. As a result, NB classifiers can easily be made interpretable for end users.

Nearest Neighbour Classifiers: The k-Nearest Neighbour (kNN) classification method finds nearby (neighbouring) objects within the training set and assigns class labels. For a new object, the distance of k neighbouring objects is measured, and the label of the "majority class" is assigned. With kNN, distance or similarity measures such as Euclidean distance or cosine values are commonly used (Khan et al., 2010).

To build the classifiers, kNN employs an instance-based learning method. It differs significantly from other commonly used algorithms. The interpretation of a kNN classifier is dependent on its ability to interpret its neighbouring data objects. When neighbouring data objects have only a few features all of which are interpretable, kNN classifications are easily interpretable. i.e., even if features are interpretable, if there are too many of them, the classifier may not be.

2.13.1 White-box Classifiers: Type I versus Type II

There are some applications that really need human friendliness. So, white-box type classifiers are used in such applications that require a higher level of human friendliness. Before trusting the predictions of the system, some end users are curious about how it works and what will happen if they make some changes. This is especially true when users are well-versed in the subject matter. We can find many such examples in the health sector.

Most medical experts would like to know what the system's prediction is and how to adjust the features of the model to see the effect. We see this as important for several reasons.

1. Predictive models are not perfect, so this combines expert knowledge and produces better results.
2. Knowledge missed from training sets can be added back to the classifiers with minor changes.
3. Experts can find better ways to handle the problem by studying the outcomes for different feature values.
4. It creates trust among all parties.

The main distinction of Type II compared to Type I is the ability to modify the classifiers. It enables fine-tuning of the classifiers by end users or domain experts.

The following two examples show rule-based and search query-based classifiers. They are interpretable and easily modifiable. The first example illustrates a text classification rule produced by the popular CONSTRUE proposed by Hayes et al. (1990) rule-based document classifier. It classifies documents in the "wheat" category of the Reuters dataset (see section 3.6 for dataset details).

if ((wheat & farm) **or**
(wheat & commodity) **or**
(bushels & export) **or**
(wheat & tonnes) **or**
(wheat & winter & ¬ soft))
then
WHEAT **else** ¬ WHEAT

The second example is given below, an evolved search query for the acquisitions category of the Reuters dataset, developed by Hirsch et al. (2010) for the GA-SFQ system. This research focused on improving this method.

(buy 10) (company 11) (bid 13) (offer 15)

In GA-SFQ, terms and their positions are taken into account when constructing these types of search query-based classifiers. In the example, retrieve documents where the word “buy” occurred within the first 10 words of a document OR the word “company” occurred within the first 11 words and so on.

Classification rules and search queries are highly interpretable and also words of the classifiers can be amended by end users easily and applied again. These classifiers belong to White box- Type II and have the highest level of human friendliness compared to the Black box type classifiers and White box - Type I classifiers. The following section 2.14 reviews more white box type II classifiers.

2.14 White-box Type II Classifiers

White-box type II classifiers include rule-based methods and search query-based classifiers. These are not only interpretable by humans, but they also have the flexibility of changing the rules or search queries based on domain expert opinions. This will be useful in some real-world applications especially when the decisions are very crucial, decision makers tend to input their opinions and observe the variations of outcome. We now move on to discuss these rule-based methods and related approaches below.

2.14.1 Rule-based Classifiers

Rule-based classifiers are straightforward IF-THEN statements that include a condition (also known as an antecedent) and a prediction (also known as a consequent). IF-THEN rules follow the following general format:

IF (*condition*) THEN (*prediction*)

When multiple conditions are required for the prediction, they connect using AND operators. The example extracted from the medical diagnostic system developed by Letham et al. (2015) is given in figure 2-10.

if *hemiplegia* **then** *stroke risk* 59.0% (53.4% - 64.6%)
else if *cerebrovascular disorder* **then** *stroke risk* 44.7% (41.2% - 48.3%)
else if *hypovolaemia* **and** *chest pain* **then** *stroke risk* 14.6% (11.6% - 17.9%)
else if *transient ischaemic attack* **then** *stroke risk* 29.9% (24.0% - 36.2%)
else if *age_70* **then** *stroke risk* 4.5% (3.6% - 5.5%)
else *stroke risk* 9.0% (8.0% - 10.0%)

Figure 2-10: Rule from stroke prediction system

Rule-based classifiers have two important characteristics: rules may not be mutually exclusive, or rules may not be exhaustive.

If rules are not mutually exclusive, the same object can be retrieved from multiple rules resulting in contradictory predictions based on different rules. In order to address this issue, decision list and decision set strategies have been introduced. The order of the rules matters in decision lists, but in decision sets the order of rules are not important. Decision sets naturally produce different solutions to overlapping problems.

If rules are not exhaustive, some objects may not be retrieved from any of the rules. To avoid this problem, objects are assigned to a default category if not any of the rules yield a prediction for an object. By adding the default category, decision list or decision sets become exhaustive.

Decision lists are very similar to how humans comprehend something. That is, it is simple for humans to decide whether something is true or false based on the order of the conditions. So, decision lists are simple to understand, and if someone believes that the order of conditions or conditions should be changed, it is simple to do so. As a result, the decision lists have a very high level of human friendliness. However, decision sets have some limitations in terms of interpretability. In decision sets strategy, we must consider the predictive power of different rules and how they contribute to the final results. In order to implement this, therefore, different strategies can be used, and it will affect the interpretability.

We have seen that rule-based approaches produce highly human-friendly classifiers. Black box type and even some classifiers with lower levels of human friendliness generate rules for prediction. Consequently, all rule-based classifiers can be broadly defined as; direct or indirect rule-based classifiers.

Direct methods generate classification rules directly from the underlying classifier construction algorithm. But indirect methods use the black box method for classification model building and rule generation, which is carried out using a secondary algorithm in two different stages. This is an indirect way of achieving a higher level of human friendliness for black-box type and white-box type I classifiers. Popular direct rule-based classifiers are discussed in Section 2.14.2 and indirect methods in Section 2.14.3.

2.14.2 Rule-based direct methods

The IREP rule learning algorithm proposed by Fürnkranz et al. (1994) is one of the base algorithms for rule-based classifiers and it has been improved later for better results. The RIPPERk presented by Cohen (1995) is one of the successors of the IREP rule learning algorithm and authors have compared its results with C4.5 rules. As per their experiments, RIPPERk is very comparative with respect to error rates but much more efficient on large datasets and rule sets are very friendly. In Sasaki et al. (1998), authors have presented that is capable of automatically categorizing web documents to enable effective retrieval of web information. Based on the rule learning algorithm RIPPER, they have proposed an efficient method for hierarchical document categorization. In Vasile et al. (2006), authors describe TRIPPER – it is a rule induction algorithm, and that is another extended version of RIPPER. TRIPPER uses background knowledge in the form of taxonomies. And taxonomies are incorporated in the rule-growth phase, followed by the rule-pruning phase. Their experiments show that the rules generated by TRIPPER are generally more accurate and more concise compared to RIPPER.

The paper published in Koklu et al. (2015) describes a human-interpretable medical scoring system. They produce decision lists in the form of a series of if...then... statements. Those if…then… statements group a high-dimensional feature space into a series of simple, interpretable decision statements. The authors have introduced a generative model called Bayesian Rule Lists that yields a posterior distribution over possible decision lists. This is an alternative to the CHADS2 score, actively used in clinical practice for estimating the risk of stroke in patients.

2.14.3 Rule-based indirect methods

In Fung et al. (2005), the authors describe an algorithm that generates non-overlapping classification rules from linear support vector machines. This algorithm was designed as a constrained-based optimization problem, and it extracts classification rules iteratively. These rules can easily be understandable to humans, unlike support vector machines.

In Hu & Li (2005), the authors present a method of deriving an accurate rule set using association rule mining. Commonly used rule-based classifiers are preferred to be small i.e. for there to be a small rule set rather than a large rule set. But small rule sets are sensitive to the missing values in unseen test data. This paper presents a classifier that is less sensitive to the missing values in unseen test data.

In the research work published by Villena Román et al. (2011), the authors outline a new hybrid approach to text classification. They have combined a kNN with a rule-based system. The kNN classifier was used in building the base model for a given labelled dataset whilst a rule-base expert system was used to improve the accuracy carefully handling false positives and false negatives. This system can easily be fine-tuned by humans by adding or removing classification rules.

Few research efforts have given the above evidence that the combination of black-box or white-box type I models with rule-generation techniques produces more user-friendly classification models.

2.14.4 Search query-based classifiers

The Olex et al. (2007), is also a method of creating rule-based classifiers. It was developed using an optimization algorithm. Both positive and negative terms generated by an optimization algorithm are used in constructing classification rules. In Pietramala et al. (2008) presents an extended version of Olex. It is a genetic algorithm, called Olex-GA, for the induction of rule-based text classifiers of the form "classify document *d* under category *c* if $t_1 \in d$ or ... or $t_n \in d$ and not ($t_{n+1} \in d$ or ... or $t_{n+m} \in d$) holds", where each t_i is a term. Olex-GA relies on efficient several individual rule representations per category and uses the F-measure as the fitness function. Results of the improved version have been presented in Rullo et al. (2009). The Olex system also provides classifiers that are accurate, compact, and comprehensible.

In Hirsch et al. (2007) a Genetic Algorithm (GA) is described which is capable of producing accurate compact and human interpretable text classifiers. Document collections are indexed using Apache Lucene and a GA is used to construct Lucene search queries. Evolved search queries are binary classifiers. The fitness function helps produce effective classifiers for a particular category when evaluated against a set of training documents.

2.14.5 Review findings

One of the key findings of this review is that rule-based classifiers have native explainability and the ability to be fine-tuned easily. This has prompted other black box models to add additional rule generators to their base models. This research main intention was to improve GA-based search query classifiers. They are slightly different from the rule-based classifiers, but search queries are highly interpretable and modifiable, like the rule-based classifiers. Also, GA-based models are capable of finding optimal solutions for complex problems. GA-based models perform well when the dataset is multidimensional. Therefore, this research attempts to focus on improving GA-based search query classifiers, and the methods used and received results are presented in the respective chapters.

2.15 Challenges and possible research directions

The benefits of machine learning have come under scrutiny in recent times, primarily due to the opaque nature of widely used methods like artificial neural network-based models. These models are adept at constructing remarkably accurate predictions, but their decision-making lacks transparency, making them non-intuitive and challenging to explain from a human perspective. As a result, there is an ongoing need to devise methodologies for document classification and clustering that maintain a high level of accuracy while also being interpretable. This is an imperative and pertinent research challenge in the present scenario.

Document classification and clustering are two different types of analysis that follow different types of strategies to develop algorithms. But common techniques have been used in the initial stage (see section 2.3.1) for pre-processing and dimensionality reduction in both types of text analytics projects. Dimensionality reduction in text analytics has a very important role compared to other types of data due to the higher number of dimensions in documents and the low accuracy it causes. But it is unknown for search query classifiers are the same or not due to the inherent nature of finding optimal solutions by GA systems. So, it is important to investigate the performance of search query classifiers against different feature selection techniques by analysing experimental results. In Chapter 3, we present experiments conducted and an analysis of the results received.

During the last two decades, text analytics related tools and techniques have been developed for many languages. But there are isolated languages around the world that do not have enough library support to conduct text analytics due to a lack of research in those languages. Supporting such low-resource languages is an important requirement to connect and share knowledge around the world. The Sinhalese language is one of the low-resource languages, and fundamental tools have yet to be developed. Apache Lucene is a text search engine library that comes with a lot of features. With real-time text indexing and low hardware requirements, it is extremely scalable. Originally written in Java, Apache Lucene has now been ported to a variety of computer languages, including Python. It also supports over 40 different human languages across the world. One of the reasons for its popularity is this. But unfortunately, Lucene does not support many LRLs including the Sinhalese language. We have identified this as one of the research gaps in our studies, and in Chapter 4 we present some of the contributions made during this research.

In the last ten years, machine-learning research has grown by leaps and bounds, and many new machine-learning strategies have been developed. According to the literature, one popular strategy for improving model accuracy is combining two models and then synthesising the results into a single outcome, also known as hybrid models. For document clustering, hybrid models are a less studied area for GA-based methods. This motivated us to see the possibility of improving accuracy as worth investigating. The space in Chapter 5 is used to describe hybrid clustering methods developed for a GA-based evolving search query engine.

One of the major limitations identified in doing text analytic research, especially with GA-based methods, is finding similar algorithms developed using GA to compare results with proposed solutions. which is a major barrier to comparing results. Also, preparing datasets and the basic tools required for pre-processing in Sinhala is very time-consuming and challenging.

2.16 Chapter Summary

There has been a significant amount of study done on document categorization and clustering in the past using various methods. Background information on text analytics, including steps of document classification, has been provided from the beginning of the chapter. The next section provides an explanation of widely used feature selection techniques and how they are used in text analytics. The following two sections explain the Apache Lucene full-text search engine as well as the operation of a GA-based algorithm because they are both essential to the enhanced GA-based solutions that are suggested in later chapters.

In Section 2.7, we have reviewed commonly used document classification and clustering methods. They will be used to compare the results of some experiments. GA-based classification and clustering methods have been studied in sections 2.8 and 2.9 respectively since the improved solution is closely related to them. An important discovery made when analysing popular classification and clustering techniques was that most of these techniques are of the black box type. However, several attempts to make them interpretable were found, and in fact, a few of them are by nature. As a consequence of our investigation, we discovered that some applications benefit from the option to modify classification criteria and not enough to make them interpretable.

We were unable to find any previously published studies that employed experimental testing to classify or cluster Sinhala documents using GA-based approaches. Consequently, this study identified Sinhala document classification and clustering as prospective research areas in text analytics. GA-based search query classifiers were chosen for the experiments due to their potential to improve accuracy and explainability, as well as the availability of the underlying base model. However, we discovered in our research that this GA engine does not enable switching feature selection metrics and has not been previously researched. This was investigated further, and the search query classifiers were improved using a few FS metrics. Details are provided in Chapter 3. The thesis’s chapters 4 and 5 are intended to test GA-based search queries for Sinhala document classification and clustering, respectively.

Chapter 3: Integrating and Analysing Feature Selection Metrics

3.1 Chapter Overview

The goal of this chapter is to analyse how well the eSQ classifiers perform when utilising different feature selection strategies. From the beginning of the chapter, we introduce the eSQ classifiers and provide an overview of the eSQ platform. Next, we have given the method of producing eSQ classifiers using a GA engine. We have integrated four feature selection metrics and produced results for a few benchmarking datasets. Finally, analyse the experiment’s outcomes and report them at the end of the chapter.

3.2 Introduction to eSQ Classifiers

The paper published by Hirsch et al. (2005) introduced the initial evolving search query-based classifiers, later refined by Hirsch (2007). It has also been used for research activities leading to publications several times in the last few years. The evolution of eSQ has been presented in section 2.8.1. This system was written in Java. The genetic algorithm has been developed using the ECJ evolutionary computation framework. ECJ library was designed for large, heavyweight experimental needs in mind, and it includes tools that enable many popular evolutionary computing algorithms and conventions, with a focus on genetic programming. ECJ is open-source software that comes with a BSD-style academic license.

The overall classifier construction process of the eSQ system is depicted in Figure 3-1. From the beginning of the process, basic data pre-processing steps are employed, such as converting text into lower case, tokenizing, and removing stop-words. Next, the most important step in pre-processing, the extraction of important features for each category from the collection of documents, is carried out. These features feed into the GA engine. After many evolutions, the GA engine produces search queries for each category, as shown in Figure 3-1. These search queries are a set of words selected from the training test and connected using the OR operator. They are readable and can be changed easily before use for classifications.

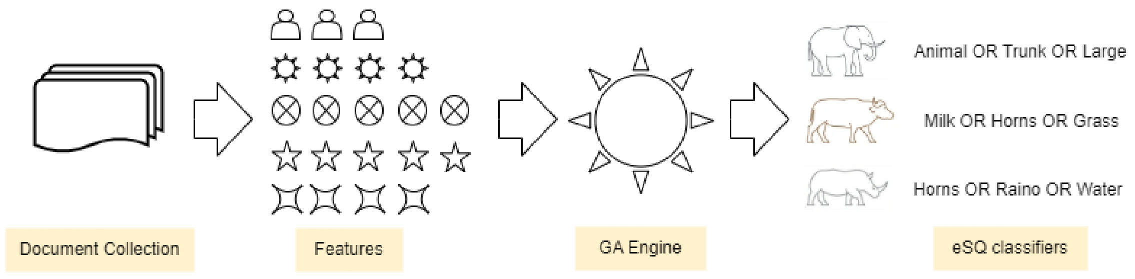


Figure 3-1: Overall process of eSQ classifier construction

In past experiments, eSQ has produced comparable results for document classification compared to widely accepted methods. It also has the advantage of being human-interpretable, which some highly accurate classifiers do not have. However, this has not been tested against different types of feature selection metrics and languages other than English. Therefore, the goal of this chapter is to apply multiple feature selection metrics to the eSQ engine and observe its behaviour. The eSQ performances for the Sinhala language are presented in the next chapter.

Feature selection was given little attention in the previous system, and it was restricted to a F-measure, which has been implemented in the *getF1TermQueryList* method. The GA Engine in the eSQ system has developed to evolve using the given feature set (the output of *getF1TermQueryList*) and produced eSQ classifiers. These classifiers are just a few words connected using logical operators, and they are highly interpretable by humans. The design of the eSQ system is capable of finding the best few words to retrieve similar documents by refusing dissimilar documents or documents from other categories.

We have developed several different feature selection metrics and integrated them into the eSQ system by updating the method of taking important word lists from the system. By doing so, we were able to input different word lists or feature sets into the GA engine to produce highly accurate classifiers. In the following section, we present the architectural overview of eSQ classification systems.

3.3 Architectural Overview: eSQ Classification System

The high-level architectural view of the eSQ classification system is shown in Figure 3-2. The data loader module is used for accessing and managing document collections. Tokenization and stop-words removal modules are used for basic data pre-processing. The processed dataset is directed to the index builder. It will prepare documents to index in the Apache Lucene text search engine, which is a widely used full-text indexing software framework, that was used to save them. The index reader is used for retrieving documents from the Apache Lucene engine. The feature selection and ranking methods created are available in the important term selection module. It will construct crucial terms required for the GA system to begin based on the configurations, and it will also be the input of the GA system. After many evolutions, constructed classifiers can be accessed using the eSQ classifiers component.

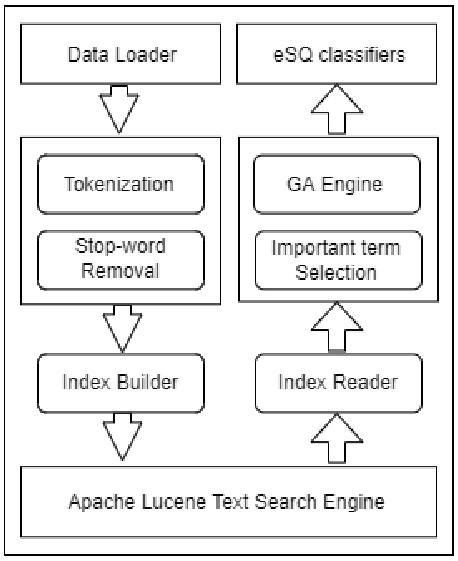


Figure 3-2: High-level Architecture of eSQ Classification System

The following set of GA parameters were used for the GA engine in all of the experiments, as shown in Table 3-1. To increase diversity in the GA population, subpopulations (the island model) are used. There is only limited communication (immigration/emigration) between subpopulations. It transmitted three individuals between the two subpopulations every 20 generations.

Table 3-1: Parameters of GA Engine

Parameter	**Value**
Population	1024
Generations	300
Selection type	Tournament
Tournament size	5
Termination	Max generations
Mutation probability	0.1
Reproduction probability	0.1
Crossover probability	0.8
Elitism	No

Subpopulations	2 (exchange 3 individuals every 20 generations)
Chromosome length	variable
Genome size	8
Engine	ECJ 21: http://cs.gmu.edu/~ eclab/projects/ecj/

The following section presents step-by-step the design used for building the search queries using the GA engine.

3.4 The method used in the eSQ Classification System

The following steps go over some of the design elements of the eSQ classification systems.

Step 1 – Data Pre-processing and Indexing

Data preparation is the initial step in the system. First, all the text is placed in lower case and stop-words, which are very common words throughout the corpus and have a detrimental impact on accuracy, are deleted once each text is converted to a set of terms (words) using a tokenizer. Following that, for each dataset, a Lucene inverted index is built, and each document is labelled (via Lucene fields) according to its category and test or training status. So, the training set and testing set are different in the model evaluation process.

Step 2 – Create an Important-Term list

The overall number of unique terms in a document collection can be fairly large. If each term were offered as a potential feature for a GA system, the search space would grow prohibitively large. Therefore, it applies feature selection metrics to reduce the number of dimensions or features. It is called the “Important-Term list” which is an ordered list of possibly relevant terms for each category in the dataset.

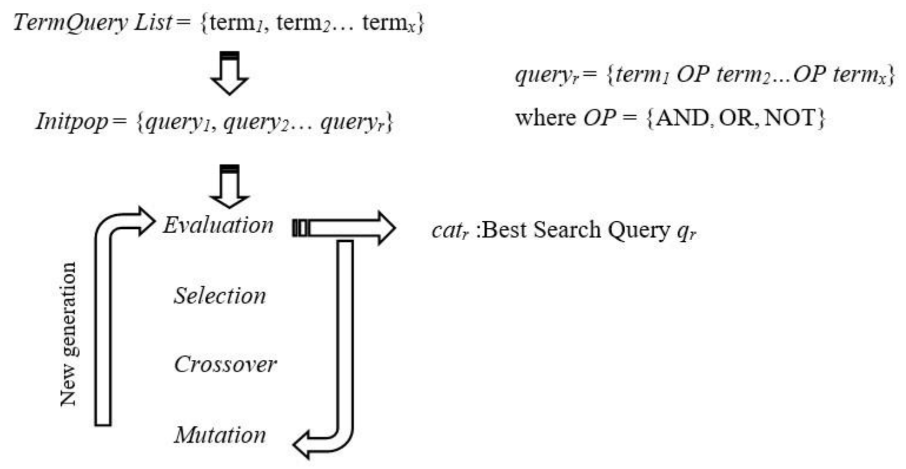
To provide a good set of evolved queries, the best list of terms (words) or best starting feature set for classifier creation is critical. So, for a category, each term discovered in the training data is graded based on its efficacy as a single-term classifier for that category. For example, if it locates the term “oil” in the training data for a specific category, it builds a query based on that single term that returns all documents that contain the term (word) “oil”. It assigns a value to the term (F1 score) based on the number of positive and negative documents retrieved from the training data by the single-term query. It can then generate an Important-Term list of length *n* for each category by simply sorting the terms by their F1 score and choosing the top *n* items.

For this investigation, multiple feature selection metrics were developed by following the strategies given in Section 2.4. These feature selection metrics produce a score for each term using positive and negative document counts. It's known as the Important Terms List. This modification to eSQ enables switching between different feature selection methods and different eSQ classifiers. In addition, it is capable of changing the length *n* of the Important-Term list.

Step 3 – Building Evolved Search Queries (eSQ) Classifiers

A GA has been developed to evolve seven different types of queries for classification purposes. However, this experiment is limited to OR queries since it has been proven that OR queries have performed well in previous studies (Hirsch & Brunsdon, 2018). A full description of the indexing system and query syntax is given at the official Lucene site[3] together with the Java source code and other useful information concerning Lucene.

To build eSQ classifiers, the best feature set collected using a chosen feature selection metric from the training set or Important-Terms list generated for one category at a time becomes the input of the GA system. From the Important-Term list, the randomly generated initial population of solutions is evolving to find optimal classifiers. The F-measure is used in calculating the fitness value and it evaluates the candidate solutions. The steps of the GA system are depicted in Figure 3-3 as well.



TermQuery List = {term$_1$, term$_2$… term$_x$}
query$_r$ = {*term$_1$ OP term$_2$…OP term$_x$*}
where *OP* = {AND, OR, NOT}
Initpop = {*query$_1$, query$_2$… query$_r$*}
Evaluation
cat$_r$:Best Search Query q_r
Selection
Crossover
Mutation
New generation

Figure 3-3: Overview of GA Engine

[3] http://lucene.apache.org/

The following example gives more details of eSQ classifier construction. This has been explained using the crude category of the R10 dataset and the GA process. Also, note that the example is based on OR-type queries since previous results have shown that highly effective classifiers can be evolved using OR queries. Table 3-2 displays the first eight words from the Important-Terms list.

Table 3-2: R10 Crude category F1 Important Term List

0	oil
1	crude
2	barrels
3	opec
4	petroleum
5	energy
6	barrel
7	production

When the OR operator is applied to two or more words, it is simply required that one of the words occurs in a document for that document to be returned. As mentioned in step 2, for each category it constructs an Important-Terms list of *n* terms which are likely to be useful for classifier query construction. A four-word query could be constructed from a randomly generated GA such as:

6	1	4	0

Figure 3-4: GA Randomly Generated Query

The eSQ system creates a Lucene Boolean query and adds the term “barrels” from the list (in Lucene syntax using BooleanClause.Occur.SHOULD) and then repeats the process for the words “crude,” “petroleum,” and “oil” as determined by the Figure 3-4. The query will then return documents that contain any of the four selected words and the GA engine calculate the F1 fitness as determined by the number of relevant and irrelevant documents returned. This method of classifier construction is achieved as one of the primary goals of developing classifiers easily interpreted by a human. The next section discusses further integrating different types of FS methods into the eSQ classification system.

3.5 Integration of Feature Selection Metrics

Feature selection is a vital step for improving the accuracy and efficiency of text analytics. F1 measure is the only FS method used in the existing eSQ engine. In this experiment, another popular three FS methods have been integrated and tested against English corpora. Selected methods belong to the category of filter feature selection metrics. Filter methods are independent of classifier construction or model-building methods. So, it is easy to port between different FS methods. This section mainly focusses on providing more details about integrating Chi-Square (CHI), Information Gain (IG), and Odds Ratio (OR) to the eSQ classifier. Details of these FS methods are given in section 2.4.

To integrate new FS methods or prepare Important-Term lists, developing a technique to implement using Apache Lucene's boolean queries was important. In this implementation, development was carried out based on the methods presented by Forman (2003). Table 3-3 depicts the confusion matrix and notations used for formulating feature selection methods.

Table 3-3: Confusion Matrix for FS Methods

		Actual	
		Yes	No
System prediction	Yes	*tp*	*fp*
	No	*fn*	*tn*

Interpretation of the notations used as follows,

tp: true positive = number of positive cases containing the word

fp: false positive = number of negative cases containing the word

fn: false negative = number of positive cases doesn't contain the word

tn: true negative = number of negative cases doesn't contain the word

pos: no of positive cases = *tp* + *fn*

neg: no of negative cases = *fp* + *tn*

fpr: sample false positive rate = *fp*/*neg*

tpr: sample true positive rate = *tp*/*pos*

recall = *tpr*

precision = *tp* / (*tp* + *fp*)

$P_{pos} = pos/\ all$

$P_{neg} = neg / all$

$P_{word} = (tp + fp) / all$

$P_{\cancel{word}} = 1 - P_{word}$

The following equations 1, 2, 3, and 4 define the F1 measure, IG, CHI and OR respectively.

$$F\ measure\ (F1) = \frac{2 \times recall \times precision}{(recall + precision)} = \frac{2tp}{pos + tp + fp} \quad (1)$$

$$Information\ Gain\ (IG) = e(pos, neg) - [P_{word} e(tp, fp) + P_{\cancel{word}} e(fn, tn)] \quad (2)$$
$$where\ e(x, y) = -\frac{x}{x+y} log_2 \frac{x}{x+y} - \frac{y}{x+y} log_2 \frac{y}{x+y}$$

$$Chi - Squared\ (CHI) = t\big(tp, (tp + fp)P_{pos}\big) + t\big(fn, (fn + tn)P_{pos}\big) +$$
$$t\big(fp, (tp + fp)P_{neg}\big) + t\big(tn, (fn + tn)P_{neg}\big) \quad (3)$$
$$where\ t(count, expect) = (count - expect)^2 / expect$$

$$Odds\ Ratio\ (OR) = \frac{tpr(1 - fpr)}{(1 - tpr)fpr} = \frac{tp \times tn}{fp \times fn} \quad (4)$$

All of the above FS methods are implemented and integrated into the eSQ engine. The methodology involves testing it against well-known textual datasets, with comprehensive experimental details provided in the subsequent section.

3.6 Datasets – Document Collections

Reuters-21578 collection – The news articles in the Reuters-21578 collection were gathered from the Reuters newswire in 1987. Personnel from Reuters Ltd. and Carnegie Group, Inc. compiled the documents and categorised them.

In 1990, Reuters and CGI made the documents available to the University of Massachusetts at Amherst's Information Retrieval Laboratory for research reasons. David D. Lewis and Stephen Harding at the Information Retrieval Laboratory formatted the documents and produced the supporting data files.

David D. Lewis and Peter Shoemaker of the University of Chicago's Centre for Information and Language Studies completed further formatting and data file development in 1991 and 1992. In January 1993, this version of the data was made publicly available through FTP as "Reuters-22173, Distribution 1.0." From 1993 to 1996, Distribution 1.0 was hosted on a set of FTP sites operated by the Computer Science Department at the University of Massachusetts at Amherst's Centre for Intelligent Information Retrieval.

A group of text categorization experts explored how published results on Reuters-22173 should be made more comparable across experiments at the ACM SIGIR'96 conference in August 1996. It was decided that a new version of the collection should be produced with less ambiguous formatting and including documentation carefully spelling out standard methods of using the collection. The opportunity would also be used to correct a variety of typographical and other errors in the categorization and formatting of the collection.

From September to November 1996, Steve Finch and David D. Lewis cleaned up the collection, depending primarily on Finch's SGML-tagged version of the collection from a previous study. The removal of 595 documents that were identical copies of other documents in the collection was one effect of the collection's re-examination. As a result, the new collection is known as the Reuters-21578 collection or "Reuters-21578, Distribution 1.0." It contains only 21,578 documents (Dua & Graff, 2017).

We have used the top 10 (R10), which is a subset Reuters-21578 collection and is comprised of 9,980 news stories. The R10 is one of the most widely used text datasets in text classification research. An in-depth discussion of the Reuters dataset is given in (Debole & Sebastiani, 2005).

20 Newsgroups[4] – The 20 Newsgroups data set contains roughly 20,000 newsgroup documents that have been partitioned (almost) evenly among 20 newsgroups. Ken Lang was the one who first gathered it. The collection of 20 newsgroups has become a prominent data set for machine learning experiments in text applications, such as text classification and text clustering.

The information is divided into 20 newsgroups, each of which corresponds to a distinct topic. Some newsgroups are closely connected (for example, comp.sys.ibm.pc.hardware / comp.sys.mac.hardware), while others are completely unrelated (for example, misc.forsale / soc.religion.christian).

[4] http://qwone.com/~jason/20Newsgroups/

Here is a list of the 20 newsgroups:

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Three different versions of the data set are given below. The first ("19997") is the original version, which has not been altered. The second ("bydate") set is sorted by date into training (60%) and test (40%) sets, excludes cross-posts (duplicates), and excludes newsgroup-identifying headers (Xref, Newsgroups, Path, Followup-To, Date). The third ("18828") has only the "From" and "Subject" headers and excludes cross-posts.

The "bydate" version was used in this study because cross-experiment comparison is easier (no randomness in train/test set selection), newsgroup-identifying information has been removed, and it is more realistic because the train and test sets are separated in time.

3.7 Analysis of Experimental Results

3.7.1 Objectives of the analysis

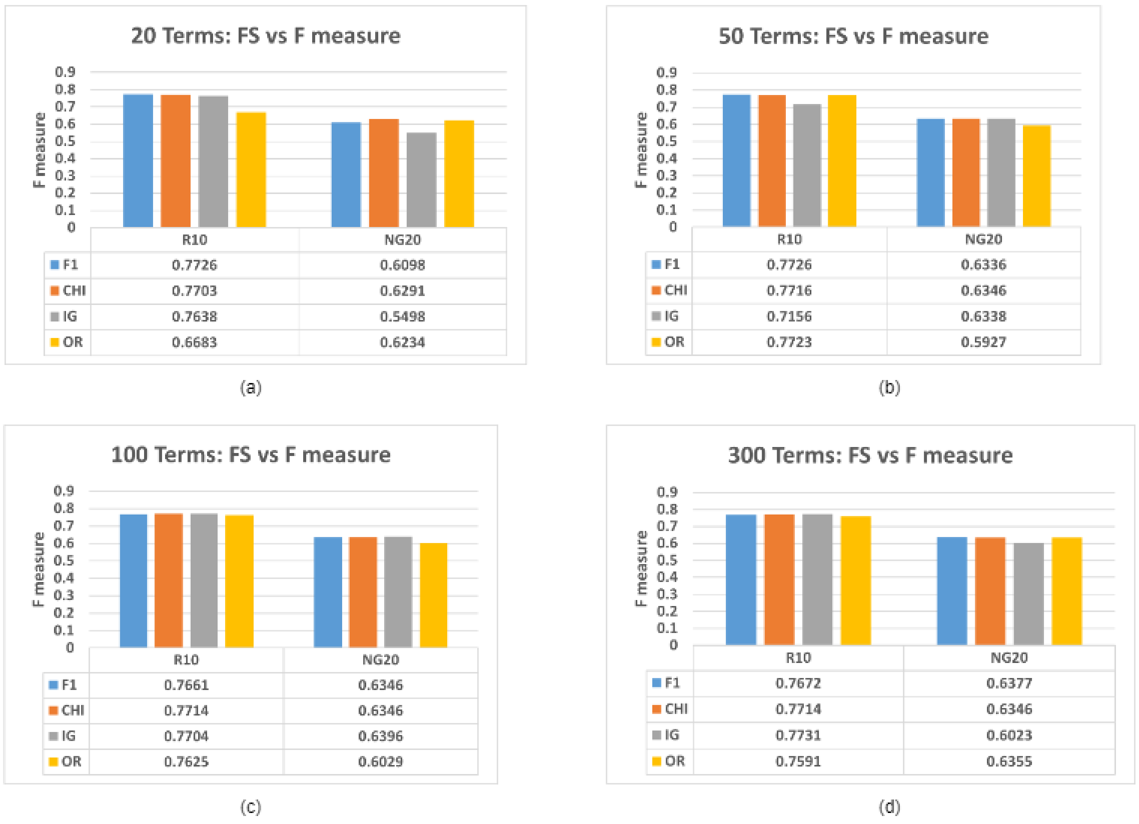
With the following objectives, we conducted several exploratory analyses:

a. To explore the effect of feature selection metrics on the performance of eSQ Classifiers.
b. To examine the impact of important-term list lengths (feature set sizes) on the performance of eSQ Classifiers.
c. Conduct exploratory analysis to identify common traits across FS methods.

3.7.2 Effect of feature selection metrics

We conducted a set of experiments to explore the effect of feature selection metrics on the performance (F measure) of eSQ classifiers. An analysis of the results is given in this section. Document classification was carried out using the eSQ system for two benchmarking datasets. Also, it has been tested for various lengths of important-term lists (feature set sizes), which is the number of terms fed into the GA engine to generate a random population. The important-term lists of 20, 50, 200, and 300 were tested for both the R10 and NG20 datasets. The summary of the results received is presented in Figure 3-5.

Figure 3-5 (a) shows the F measure received for important-term list length equal to 20 for multiple FS methods that include F1, CHI, IG, and OR. The F1-measure presented in the graphs has been arranged in the order of the R10 and then NG20 datasets. The same experiments were conducted for important-term list lengths equal to 50, 100, and 300, and the F1-measure was measured. The results received are presented in Figures 3-5 (b), 3-5 (c), and 3-5 (d), respectively.



20 Terms: FS vs F measure

	R10	NG20
🟦 F1	0.7726	0.6098
🟧 CHI	0.7703	0.6291
⬜ IG	0.7638	0.5498
🟨 OR	0.6683	0.6234

50 Terms: FS vs F measure

	R10	NG20
🟦 F1	0.7726	0.6336
🟧 CHI	0.7716	0.6346
⬜ IG	0.7156	0.6338
🟨 OR	0.7723	0.5927

100 Terms: FS vs F measure

	R10	NG20
🟦 F1	0.7661	0.6346
🟧 CHI	0.7714	0.6346
⬜ IG	0.7704	0.6396
🟨 OR	0.7625	0.6029

300 Terms: FS vs F measure

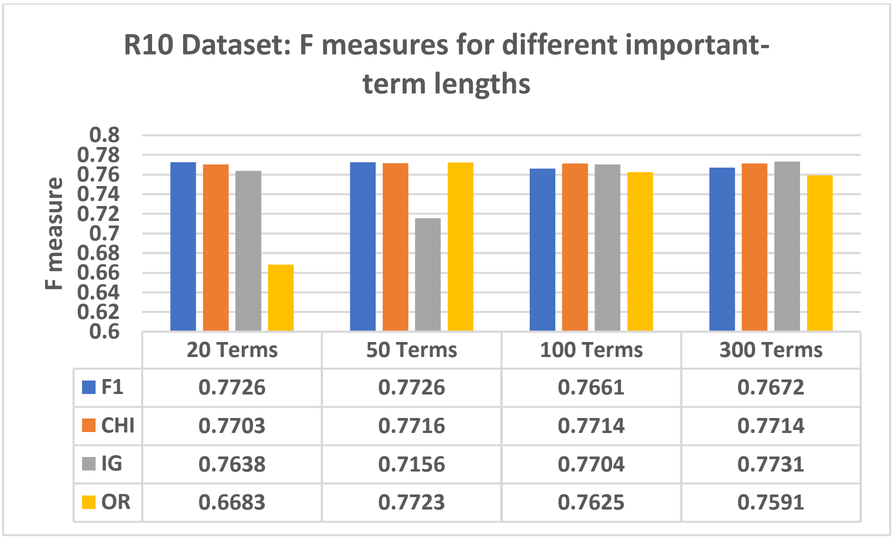
	R10	NG20
🟦 F1	0.7672	0.6377
🟧 CHI	0.7714	0.6346
⬜ IG	0.7731	0.6023
🟨 OR	0.7591	0.6355

Figure 3-5: Overall F1-measure for different FS methods

These figures demonstrate that the 20 Newsgroups (NG20) dataset consistently produces less accurate results than the Reuters (R10) dataset, regardless of the FS method and length of the import-term list chosen. That is performance is greatly dependent on the document collection employed. CHI or F1 FS methods have received the comparable highest F1-measure, with the exception of Figure 3-5 (d), where IG has received the best F1-measure value.

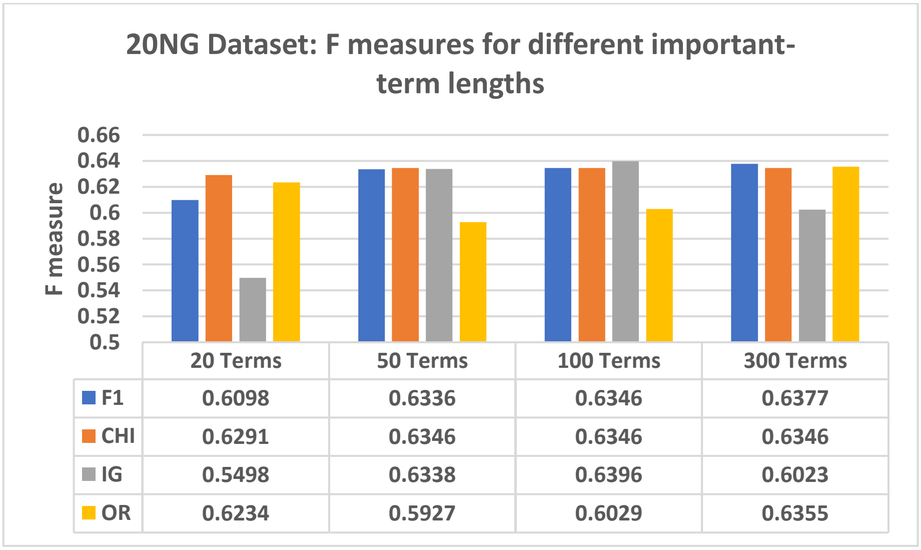
3.7.3 Impact of important-term list lengths

The second purpose of this study was to examine the influence of important-term list lengths (feature set sizes) on eSQ classifier performance. Consequently, the F1-measure of document classification for varying lengths of important-term lists for different feature selection methods was measured. Considered lengths for lists of important-term are 20, 50, 100, and 300 terms. The system was executed with all FS methods (i.e. F1, CHI, IG, and OR) integrated with the eSQ system. The F measure obtained for the R10 and NG20 datasets is depicted in Figures 3-6 and 3-7, respectively.



	20 Terms	50 Terms	100 Terms	300 Terms
🟦 F1	0.7726	0.7726	0.7661	0.7672
🟧 CHI	0.7703	0.7716	0.7714	0.7714
⬜ IG	0.7638	0.7156	0.7704	0.7731
🟨 OR	0.6683	0.7723	0.7625	0.7591

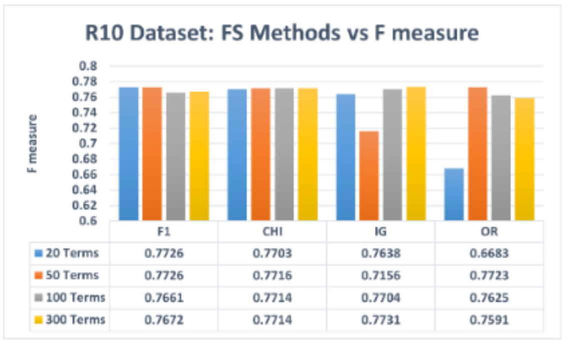
Figure 3-6: R10: FS Methods vs F1-measure



	20 Terms	50 Terms	100 Terms	300 Terms
🟦 F1	0.6098	0.6336	0.6346	0.6377
🟧 CHI	0.6291	0.6346	0.6346	0.6346
⬜ IG	0.5498	0.6338	0.6396	0.6023
🟨 OR	0.6234	0.5927	0.6029	0.6355

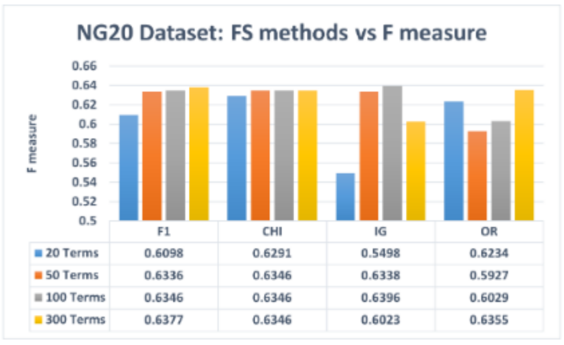
Figure 3-7: NG20: FS methods vs F1-measure

The CHI and F1 methods are less sensitive to the length of important-term lists and perform well for shorter lengths, as seen in Figures 3-6 and 3-7. However, IG and OR behave differently for varying lengths of important-term lists. It is further confirmed in Figure 3-8 (a) and Figure 3-8 (b).



	F1	CHI	IG	OR
🟦 20 Terms	0.7726	0.7703	0.7638	0.6683
🟧 50 Terms	0.7726	0.7716	0.7156	0.7723
⬜ 100 Terms	0.7661	0.7714	0.7704	0.7625
🟨 300 Terms	0.7672	0.7714	0.7731	0.7591

(a)



	F1	CHI	IG	OR
🟦 20 Terms	0.6098	0.6291	0.5498	0.6234
🟧 50 Terms	0.6336	0.6346	0.6338	0.5927
⬜ 100 Terms	0.6346	0.6346	0.6396	0.6029
🟨 300 Terms	0.6377	0.6346	0.6023	0.6355

(b)

Figure 3-8: FS methods vs. F1-measure

In conclusion, CHI and F1 feature selection methods with shorter lengths of important-term lists give more accurate results than IG and OR when utilising the eSQ classifiers. Both perform well at length size 50, while CHI also performs reasonably well at length size 20. IG and OR are very sensitive to the length of the important-term list.

3.7.4 Performance across the categories for FS methods

The following Figure 3-9 and Figure 3-10 show the F1-measure of each category for different FS methods. This experiment was carried out for R10 and NG20 datasets and 20 terms in the important-term list. For the R10 dataset, CHI, F1 and IG F1-measures are very close in every category but the Odds Ratio (OR) presents less F1-measure in many categories.

For the NG20 dataset, OR performed poorly. But for the other FS methods, category-wise F1 measures are very close, and CHI outperforms in many categories.

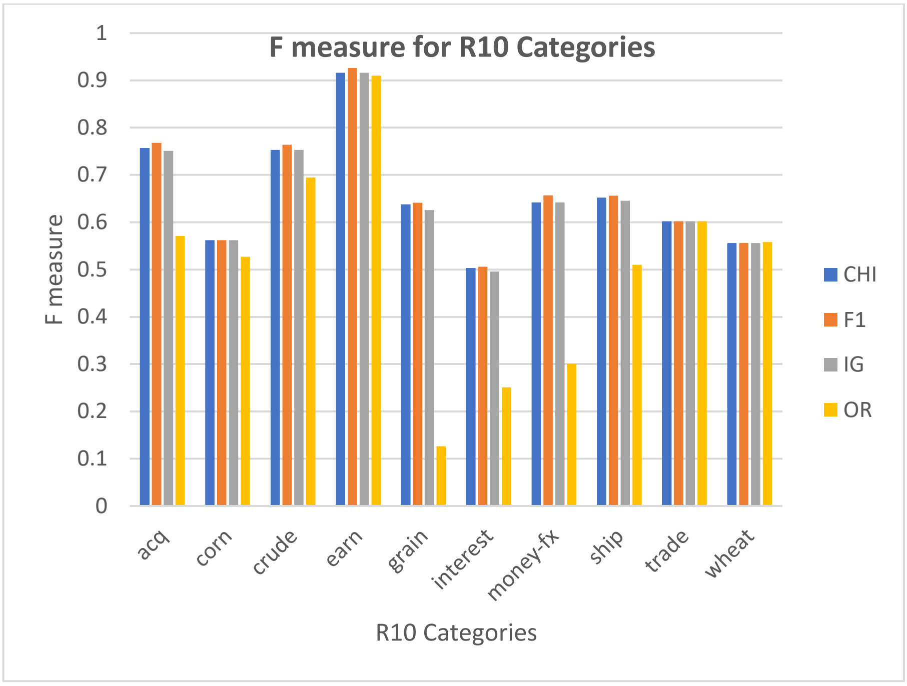


Figure 3-9:R10- Category wise F1-measure

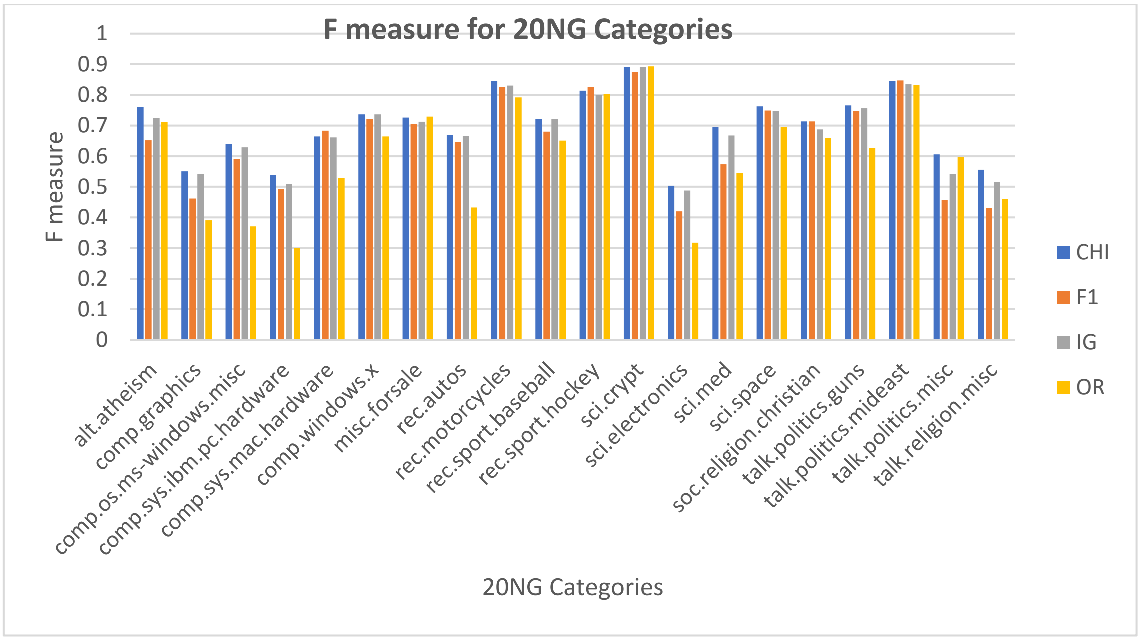


Figure 3-10: NG20 Category-wise F measure

3.7.5 Analysis of important term lists for R10-Crude and NG20-Space categories.

R10-Crude Category: The following Figure 3-11, a four-way Ven diagram illustrates common term distribution across the four feature selection methods employed on the Crude category in the R10 dataset. Actual terms selected from these feature selection methods are given the Table 3-4. Figure 3-11 shows that CHI and IG have only one unique term while OR has 13 unique terms. And also, F1 has 5 unique terms and F1 has produced eSQ classifiers with the highest performance.

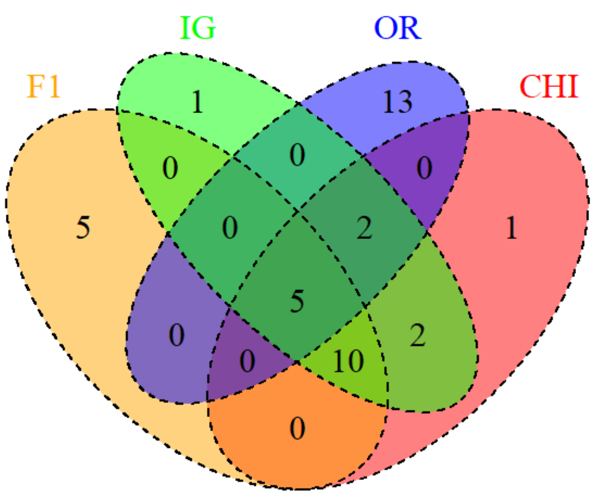


Figure 3-11: Four-way Venn diagram for Important terms - Crude Category

Table 3-4: Important terms - Crude Category

	F1	CHI	IG	OR
1	oil	oil	oil	oil
2	crude	crude	crude	crudes
3	barrels	barrels	barrels	barrels
4	opec	opec	opec	opec
5	bpd	bpd	bpd	bpd
6	petroleum	petroleum	petroleum	fernando
7	barrel	barrel	barrel	pumping
8	energy	energy	energy	benchmark
9	prices	prices	prices	arrived
10	day	day	day	mobil
11	production	production	production	sweet
12	gas	gas	gas	ali
13	exploration	exploration	exploration	brent
14	output	output	output	suharto
15	natural	natural	natural	earthquake
16	minister	gasoline	gasoline	gasoline
17	industry	ecuador	ecuador	ecuador
18	price	quota	quota	sabah
19	texas	refinery	refinery	amoco
20	state	bbl	lt	liftings

NG20-Space category: Figure 3-12, the four-way Ven diagram, shows important terms distribution across the feature selection methods. This was done for the Space category of the NG20 dataset. Table 3-5 presents terms selected by each of the FS methods from the space category. CHI has only one unique term, which gives the highest performance for eSQ classifiers. OR has 13 unique terms and it gives lowers performance for this category.

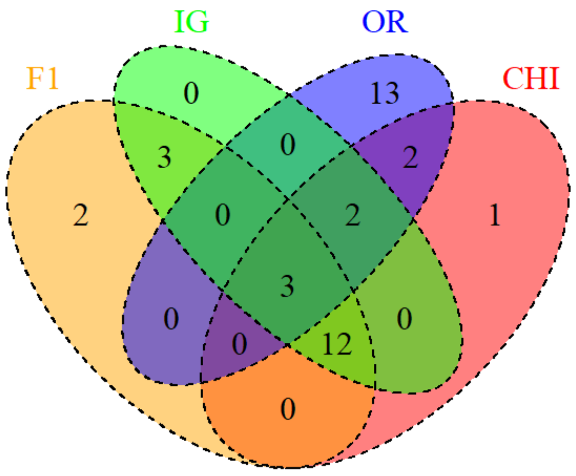


Figure 3-12: Four-way Venn diagram for Important terms - Space Category

Table 3-5: Important terms - Space Category

	F1	CHI	IG	OR
1	spacecraft	spacecraft	spacecraft	spacecraft
2	orbit	orbit	orbit	orbit
3	prb	prb	prb	prb
4	nasa	nasa	nasa	venus
5	launch	launch	launch	kelvin.jpl.nasa.gov
6	space	space	space	booster
7	shuttle	shuttle	shuttle	manned
8	moon	moon	moon	mccall
9	lunar	lunar	lunar	astronaut
10	earth	earth	earth	shafer
11	access.digex.com	access.digex.com	access.digex.com	caterpillar
12	henry	henry	henry	telos
13	flight	flight	flight	dcx
14	zoo.toronto.edu	zoo.toronto.edu	zoo.toronto.edu	sci.astro
15	spencer	spencer	spencer	jupiter
16	access.digex.net	mars	access.digex.net	centaur
17	solar	baalke	solar	baalke
18	pat	sci.space	pat	sci.space
19	nsmca	aurora.alaska.edu	aurora.alaska.edu	aurora.alaska.edu
20	cost	orbital	orbital	Orbital

3.8 Chapter Summary

This chapter presents a performance comparison of four different feature selection metrics integrated into the eSQ classification system. The effects of different feature selection methods on overall performance, the effects of different lengths of important term lists, and category-level analyses of the important terms chosen by FS methods have all been discussed. CHI squired and F1 methods are relatively performing well in small lengths of the important-term list.

Chapter 4: Effectiveness of eSQ Classifiers for Sinhala Documents

4.1 Chapter Overview

This chapter discusses the contribution to Sinhala document classification. Sinhala, being a low-resource language, presents challenges in conducting research on text analytics due to the absence of essential pre-processing tools. Therefore, to address this issue, several crucial pre-processing tools have been developed or enhanced and integrated into the Apache Lucene framework. Subsequently, we evaluated the feasibility of utilizing eSQ for the classification of Sinhala documents, and the findings are presented in this chapter.

This chapter is structured as follows: Section 4.2 discusses the limitations of Text Analytics for Sinhala, highlighting the challenges faced when analysing this specific language. In Section 4.3, the barriers to using eSQ for Sinhala documents are reviewed, examining the obstacles that need to be addressed. Section 4.4 outlines the preparation of datasets, providing an overview of the necessary steps taken to ensure accurate and reliable analysis. Section 4.5 focuses on the improvements made to Apache Lucene, a powerful tool used in the analysis of language content. The discussion in this section explores the enhancements made to enhance its performance. In Section 4.6, the experimental results obtained from the research are presented, offering insights into the effectiveness and potential of the implemented techniques. Finally, Section 4.7 presents a summary of the chapter, summarizing the key findings and contributions discussed throughout.

4.2 Background and Limitations in Text Analytics: Sinhalese Language

Document analysis is a mature research field that has continued to grow over the past few decades. It remains an active field of research and development not only because of the complex nature of building computing tools and techniques and the ever-evolving nature of technology but also because of inherent problems due to the variety of languages spread all over the world. For example, languages that are isolated (whether geographically or in their evolution) have their own unique features. Therefore, language and computing experts from these communities need to fine-tune any technological tools, which are often developed with more widely spoken languages in mind. However, due to a lack of experts, some languages do not have even the basic tools to analyse documents. In terms of modern technologies, this makes some communities disconnected from others in the world as they do not have access to recent technological NLP tools. This chapter aims to present the development of such tools for the Sinhalese language (Sinhala), which at present is one of the under-resourced and underrepresented languages in the world in terms of available technological tools.

Sinhala, which belongs to the Indo-Aryan language family, is the native language of the Sinhalese people who comprise the largest ethnic group in Sri Lanka. Although English is used in Sri Lanka for communicating internationally, Sri Lankans predominantly use Sinhala in both formal and informal activities. Further, it is one of the three official languages in the country alongside Tamil and English. Most of the official government documents are written in Sinhala. As a low-resource language, many government officers face a range of practical problems in the modern digital world to processing Sinhala documents.

The Sinhala alphabet has two versions. Pure Sinhalese (ශුද්ධ සිංහල) alphabet is the core set of letters, and mix Sinhalese (මිශ්‍ර සිංහල) alphabet is the extended version. The Mix Sinhalese alphabet consists of 54 letters, 12 of which are vowel letters. Most of the latter are circular shapes, and straight lines are hard to find. Each consonant has an inherent vowel that may be altered using the various vowel signs or eliminated. The letters of this alphabet are written left to right. Sinhala has a unique set of symbols and a numbering system that is rarely used. Additionally, it features certain punctuation that is specific to Sinhala, such as ෴ (kunddaliya). The Sinhalese language has evolved over the thousands of years and it has unique grammatical rules compared to other languages. Therefore, it needs particular consideration throughout the development of computing tools for language analytics.

Numerous innovative applications leveraging the fusion of data and computational capabilities have emerged, bringing substantial advantages to society. Notably, advanced computing techniques, both supervised and unsupervised, are employed to analyze vast datasets, employing powerful processors and scalable computing frameworks. These methodologies efficiently extract a variety of valuable insights from the input data. Regrettably, due to limited resources and computing tools, Sinhala documents remain largely unprocessed in numerous institutions across the country.

At present, supervised and unsupervised learning algorithms are facing a new set of challenges and are being interrogated more closely. Human interpretability and transparency are missing in such ‘black box’ algorithms and this has severely affected the level of trust in and accountability of such algorithms. Computing professionals should be able to provide valid justifications for the actions and decisions that have been taken. If this is not the case, then such methods are less effective because their results are not trusted.

In our research, we have placed special emphasis on developing human-interpretable solutions while contributing to Sinhala text analytics. Therefore, our focus is on enhancing Evolved Search Queries (eSQ) to improve human interpretability, transparency, and explainability, specifically for supporting Sinhala text analytics. However, the application of eSQ in the context of Sinhala faces certain challenges due to it being a low-resource language. Primarily, obtaining readily available datasets for research purposes proved to be extremely difficult. Additionally, NLP toolkits do not offer language-specific preprocessing methods such as tokenization, stop-word elimination, and stemming for Sinhala. Furthermore, full-text search engines like Apache Lucene lack support for Sinhala. In section 4.3.2, we have provided detailed information regarding the language support of Apache Lucene. Therefore, the objective of this chapter is to explore various options for effectively applying eSQ in Sinhala document classification, considering these limitations.

In summary, to overcome some of the limitations, we carried out the following activities: web scraping to produce required datasets, development of a tokenizer for Sinhala, generation of a Sinhala stop-word list using tf-idf, and development of a Sinhala stemming method. Also, we integrated the above with an Apache Lucene full-text search engine to create an evolved search query builder using a Genetic Algorithm (GA).

Continuing our study, we proceeded to conduct experiments aiming to compare, evaluate, and observe the enhancements made to the Lucene framework when applied to our Sinhala dataset. (i.) We have compared the performances of popular classifiers against our evolved search classifiers. This analysis allows us to determine how well our evolved search classifier performs in comparison to established classifiers commonly used in the field. (ii.) Additionally, we have compared the performances of newly developed tools with existing tools of the Lucene frameworks. This evaluation helps us understand the improvements achieved or limitations through the development of our tools. (iii.) Furthermore, we have evaluated the level of interpretability or explainability in Sinhala document classification. This assessment focuses on understanding how well our classification models can provide meaningful explanations and insights into their decision-making process. By examining the interpretability of our models, we can ensure transparency and facilitate users' understanding and validation of the classification results in Sinhala documents.

The subsequent section delves into more specific details regarding the current status of the Sinhala language in the analytics field, as well as the support provided by Apache Lucene as a full-text search engine.

4.3 Barriers for using eSQ in the Sinhalese Language

4.3.1 Low Resource Language

The tools of Natural Language Processing (NLP) underwent a substantial transformation in the 1990s, moving away from rule-based approaches and toward statistical techniques. The vast majority of today's NLP research focuses on only 20 of the world's 7000 languages, leaving the vast majority of languages understudied. Low resource languages (LRLs) are a term used to describe these languages. Less studied, resource-limited, less computerised, less privileged, less often taught, or low density are some of the terms used to describe LRLs (Magueresse et al., 2020).

LRLs are important for a variety of reasons. Around 2000 LRLs are found in Africa and India, which have a population of more than 2.5 billion people. The development of technologies for these languages brings up significant commercial opportunities. Supporting a language using NLP tools can also help it avoid extinction, expand, and make the knowledge contained in original works accessible to everyone.

Stop-words, tokenizers, stemmers, and prepared datasets are some of the most important basic NLP tools for developing many languages-specific applications. For eSQ integration, the unavailability of these tools in the Lucene framework was one of the main barriers. This chapter presents an overview of the efforts made to integrate Sinhala language support into Lucene and evaluates its performance using eSQ.

4.3.2 Apache Lucene support for Sinhala

Apache Lucene is a robust text search engine library renowned for its advanced capabilities in indexing and searching text. With real-time text indexing and low hardware requirements, it is extremely scalable. Originally written in Java, Apache Lucene has now been ported to a variety of computer languages, including Python. It also supports over 40 different human languages across the world. One of the reasons for its popularity is this. But unfortunately, Lucene does not support many LRLs including the Sinhalese language.

The org.apache.lucene.analysis package is used for text analysis in Apache Lucene. Which provides an API and code for converting text into indexable or searchable tokens via Analysers and related classes. Lucene accepts only plain text input for indexing and search library. As a result, applications that base their search capabilities on Lucene may support documents in a variety of formats, including HTML, XML, PDF, and Word. Lucene is unconcerned about the parsing of these and other document formats, and it is the application's responsibility to use an appropriate parser to convert the original format into plain text before passing that plain text to Lucene.

Plain text passed to Lucene for indexing goes through a process generally called tokenization. Tokenization is the process of dividing input text into small indexing elements known as tokens. The manner in which input text is divided into tokens has a significant impact on how people will be able to search for that text.

In some cases, simply breaking the input text into tokens is insufficient; a more in-depth analysis may be required. Lucene supports both pre and post tokenization analysis. Stripping HTML markup and transforming or removing text matching arbitrary patterns or sets of fixed strings are examples of pre-tokenization analysis. Multiple post-tokenization steps can be taken, such as stop-word removal, stemming, and synonym expansion.

But, there is no Sinhala analyser or associated classes in Apache Lucene. It was a significant roadblock in the development of Sinhala text-based applications. So, this chapter lays forth some of the steps that were taken to overcome this impediment. In section 4.4 presents preparation of datasets, and in section 4.5, we present our specific contributions to various post-tokenization steps and their seamless integration into the

Lucene framework through the addition of a Sinhala analyzer. This integration facilitates the testing of the eSQ engine using Sinhala document collections.

4.4 Datasets

4.4.1 Creating SLNG Datasets

Obtaining prepared datasets for the Sinhalese language poses a challenge when conducting text analytics research and developments. To address this issue, two primary document collections have been created since benchmark datasets for Sinhalese are not readily available. These collections consist of publicly accessible news articles gathered from the web.

The first dataset, SLNG_rands, consists of 19,845 documents collected randomly. Which is publicly available for research activities[5]. In our study, we utilized this dataset for stop-word generation. Since it lacks categorized documents required for classification purposes, it holds potential for unsupervised learning and other research in natural language processing (NLP) including clustering, topic modelling, or language modelling, due to its diverse collection of randomly obtained documents. It can serve as a valuable resource for exploratory analysis, data preprocessing, and feature engineering in NLP research.

One particular application of SLNG_rands in our research was stop-word generation. Stop-words are commonly occurring words in a language that are often removed during text processing to improve the efficiency and relevance of NLP tasks. By analyzing the frequency and distribution of words in SLNG_rands, we were able to identify and extract a comprehensive set of stop-words specifically tailored for the Sinhala language. More details are given in section 4.5.

The second dataset, the SLNG collection comprises pre-labelled news articles. In the process of creating the SLNG Collection, publicly available news articles were gathered through web scraping. Additionally, a small dataset published by Ekanayake et al. (2018) and a dataset published by Lakmali and Haddela (2018)were incorporated into the collection. The SLNG collection consists of five different datasets referred to as SLNG3, SLNG4, SLNG5, SLNG6, and SLNG7.

The SLNG3 dataset comprises news articles from three sports categories: cricket, football, and rugby. These categories exhibit a higher degree of overlapping terms,

[5] https://github.com/psumathipala/SLNGCollection

indicating similarities in their content. Additionally, each category within SLNG3 contains an equal number of documents, resulting in a balanced dataset.

To enhance the diversity and expand the number of categories in our dataset, we created four additional datasets: SLNG4, SLNG5, SLNG6, and SLNG7. These datasets include documents from the entertainment, politics, crime, and religion categories, respectively in addition to the SLNG3 category. The purpose behind gradually introducing these categories was to increase the variety of documents in the collections and investigate the performance of different classifiers. The specific details of these datasets, including the dataset name, category name, number of documents and number of categories, can be found in Table 4.1, providing a clear overview of their composition.

Table 4-1: The Structure of Datasets

Dataset name	No. of categories	No. of documents	Category names [Category size]
SLNG3	3	2550	cricket [850] / football [850] / rugby [850]
SLNG4	4	4050	cricket [850] / football [850] / rugby [850] / entertainment [1500]
SLNG5	5	4250	cricket [850] / football [850] / rugby [850] / entertainment [1500] / politics [200]
SLNG6	6	4450	cricket [850] / football [850] / rugby [850] / entertainment [1500] / politics [200] / crime [200]
SLNG7	7	4650	cricket [850] / football [850] / rugby [850] / entertainment [1500] / politics [200] / crime [200] / religion [200]

The introduction of this document collection, it makes possible to utilize the Sinahalese language for various text analytics purposes, such as constructing classifiers. Pre-labelled datasets play a crucial role in training machine learning models and assessing their performance. Since benchmark datasets specifically tailored for Sinhalese are lacking, the news articles in these collections serve as valuable resources for developing and testing text analytics models. Furthermore, we would like to emphasize that both datasets are freely and publicly available for future research purposes. We believe that the development of Sinhala datasets will play a crucial role in fostering significant growth in the field of Sinhala text analytics in future.

4.4.2 Data Conversion

The SLNG collection was utilized in experiments with two types of data formats. One format was used for the Lucene index framework, while the other was used for the Weka ML platform.

Our objective was to enhance the Lucene framework to enable indexing of the Sinhalese language by introducing a dedicated analyser specifically designed for Sinhala. Therefore, it was necessary to assess the performance of the proposed new Lucene Sinhala analyser using the Sinhala dataset. So, we created Lucene indexes using the Lucene standard analyser and proposed a new Sinhala analyser to compare performances. Hence, preparing the SLNG collection for indexing using Lucene was an important task.

Lucene was originally designed to index plain text datasets. Thus, we were able to utilize Sinhalese datasets, consisting of documents from various news categories (thousands of files), for our experiments with minor modifications to the scraped dataset. The articles collected through web scraping and other sources had different character encodings. Prior to using this dataset in our experiments or creating Lucene indexes, we converted the documents into Unicode Transformation Format - 8 bits (UTF-8) encoding. This conversion was performed using the following PowerShell script:

```
Get-ChildItem -Filter *.txt |
Foreach-Object {
Get-Content .$_ | Set-Content -Encoding Default ($_.BaseName + '_utf8.txt')
}
```

After converting the SLNG collection into UTF-8 encoding, we developed a Java program to create Lucene indexes for the SLNG collection. Indexes created from both the new Sinhala analyser and the Lucene standard analyser were used for experiments and produced results for performance comparison.

As we utilized some of the ML algorithms supported by the Weka platform for result comparison of eSQ versus others, we also converted the SLNG collection into the Weka data format using the conversion class provided by the Weka API itself.

The following section describes the changes and developments that were done to enhance the Lucene framework.

4.5 Improvements made for Apache Lucene

4.5.1 Overview of New Lucene Sinhala Analyser

The language analyser is a core component of any document processing system. Processing documents in the Sinhalese language was a challenge due to the unavailability of the basic data pre-processing tools required. As a part of this research project, we have designed and developed a language analyser for the Sinhalese language. This consists of a Sinhala tokenizer, stop-word list and stemmer and these are fully compatible with the Lucene framework.

4.5.2 Sinhala Tokenizer

Tokenization is the separating and (possibly) breaking into small units of a string of input characters. The resulting tokens (terms) are then passed on to other language processing tools. A tokenizer forms the initial step and creates a starting point for other data pre-processing operations.

Tokenization is highly language dependent. For example, tokenizers developed for English cannot be used as is for Chinese or Arabic since the languages are inherently different in many ways. Therefore, it is useful to have language-specific tokenizers. Rule-, statistical-, fuzzy-, lexical- and feature-based techniques are often employed when designing a tokenizer. Our Sinhala tokenizer was developed using a rule-based technique and in developing it we considered languages that are similar in terms of tokenisation. It has two main components: (i.) punctuation-based tokenization and (ii.) dependent word tokenization.

The Sinhalese language has 15 punctuation symbols and some of these are unique to the language. For example, the ෴ symbol is used to show the end of a sentence or a paragraph in old documents. Furthermore, the meaning of some Sinhala punctuation marks differs depending on the context in which it is used. Therefore, in the Sinhala tokenizer, language-specific patterns have been identified and appropriate rules have been applied to produce the token set. The dependent word tokenization component is aimed at identifying words that differ in meaning when they are together rather than separate. However, finding dependent words is a computationally high-cost operation. The details of the development of the tokenizer and our experiments are published in Senanayake et al. (2019).

4.5.3 Sinhala Stop-words List

Stop-word removal is a basic pre-processing step in NLP. It filters out redundant words that hold little information and have low or no semantic meaning for the given text (Raulji & Saini, 2017). “Is”, “are”, “in”, “for”, and “that” are some examples of stop-words in English. Removal of stop-words helps us to decrease the size of the corpus and increases the efficiency and performance of NLP tasks (Saini et al., 2016).

In Fox (1990), the author is one of the main contributors to finding stop-words and his method has created a standard list for English consisting of 421 words. In the literature, we found that stop-word lists have been generated for Arabic (El-Khair, 2006) and regional languages such as Sanskrit (Raulji & Saini, 2017), Punjabi (Puri et al., 2013), Gujarati (Patel, 2014) and many more languages. We also found some attempts for Sinhala which used a rather small group of documents (Gunasekara & Haddela, 2018; Lakmali & Haddela, 2018).

In this research, we generated a stop-word list using our SLNG_rands dataset which contains 19845 randomly collected news articles. Our stop-word list was produced using tf-idf ranking and consists of 210 Sinhala words. We further tested it using several classifiers and the results have been published in Jayaweera et al. (2019). This list is called inside the Lucene Sinhala analyser that we have developed.

4.5.4 Sinhala Light Stemmer

Stemming converts the original word into its root format, which is called its stem. The stemming process plays a prominent role in NLP because it makes applications more efficient and effective. There are five types of words in Sinhala. Each of these words is formed by combining one or more morphemes with the base form. Sinhala morphemes are divided into four main types known as bases, suffixes, prefixes and infixes. We have developed a set of rules that reduce words back to their base form. However, initial experiments found that the stemming algorithm over truncates and this is damaging to effectiveness. Also, in the Sinhala language, if a prefix is removed from a word then it gives the opposite meaning of the word. Again, this badly affected the effectiveness of classification. Experiment results and more details about the algorithm have been published in Kariyawasam et al. (2019). Therefore, the Sinhala ‘Light Stemmer’ that we have integrated with Lucene does not consider the prefix removal of words during stemming.

4.6 Performance Analysis of eSQ for the Sinhalese Language

4.6.1 Experimental objectives

We conducted a series of experiments with the following three objectives:

a. To investigate whether the performance of our eSQ classifier when classifying Sinhalese documents deviates significantly from other popular classifiers.

b. To compare classification performance between our Lucene Sinhala Analyser and the Lucene Standard Analyser when classifying Sinhalese documents using eSQ.

c. To Investigate the human readability of the eSQ classifiers produced by our Sinhala analyser and the Lucence Standard analyser.

4.6.2 eSQ performance in Sinhala document classification

The results in Table 4-2 show the Average Macro F score for the 9 popular classifiers for the five datasets detailed in Table 4-1, with the best results displayed in bold. The average Macro F score has been computed after executing each classifier 10 times on each dataset.

Table 4-2: Summary of Average Macro F

	eSQ	C4.5	RF	PART	JRip	kNN	SVM	NB	DeepL
SLNG3	0.9441	0.9405	0.9443	0.9303	**0.9452**	0.8451	0.9420	0.8897	0.9361
SLNG4	0.9369	0.9525	0.9518	0.9457	0.9554	0.7229	**0.9589**	0.9176	0.9404
SLNG5	0.9315	0.9547	0.9491	0.9460	0.9550	0.7041	**0.9607**	0.9056	0.9424
SLNG6	0.9044	0.9472	0.9422	0.9375	0.9430	0.6820	**0.9596**	0.9020	0.9223
SLNG7	0.9038	0.9388	0.9426	0.9304	0.9304	0.6544	**0.9590**	0.9079	0.9245

Figure 4-1 uses boxplots to show the variability of the Macro F values for each classifier. It shows that most of the classifiers have comparable results, except the kNN classifier, which is considerably worse. For this reason, we omit the kNN classifier from further analyses. This also confirms that our eSQ classifiers are among the top classification techniques when classifying Sinhalese documents.

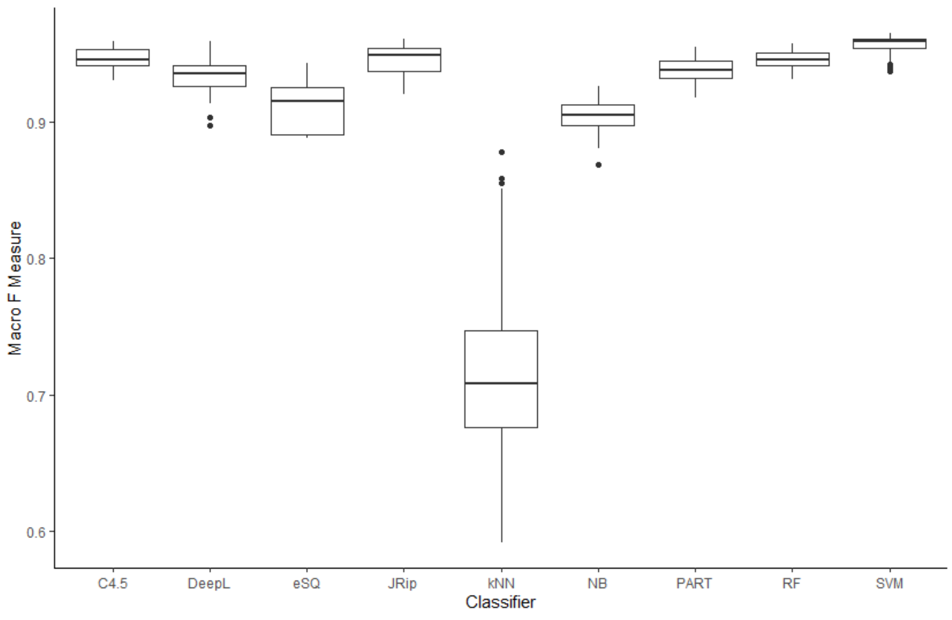


Figure 4-1: Variability in Macro F

Before conducting a detailed analysis using ANOVA, we transformed Macro F scores into logit Macro F values. This ensures that our target variable has the whole real line as its range of possible values, rather than just values in the interval [0,1].

```
                           Df   Sum Sq    Mean Sq    F value     Pr(>F)
Algorithm                   7     5956      850.8     304.15   <0.00001    ***
Dataset                     4      932      233.0      83.31   <0.00001    ***
Algorithm:Dataset          28     1255       44.8      16.02   <0.00001    ***
Residuals                 360     1007        2.8
```

Figure 4-2**:** Summary of ANOVA Test

Figure 4-2 shows the ANOVA test output and it confirms that there is a significant interaction between the algorithm used and the dataset being considered, so that different algorithms are better with different datasets.

Figure 4-3 shows the relationship between classifiers and logit F for each dataset. This shows that our eSQ classifier performs well when a dataset has more overlapping categories than when it has a number of diverse categories.

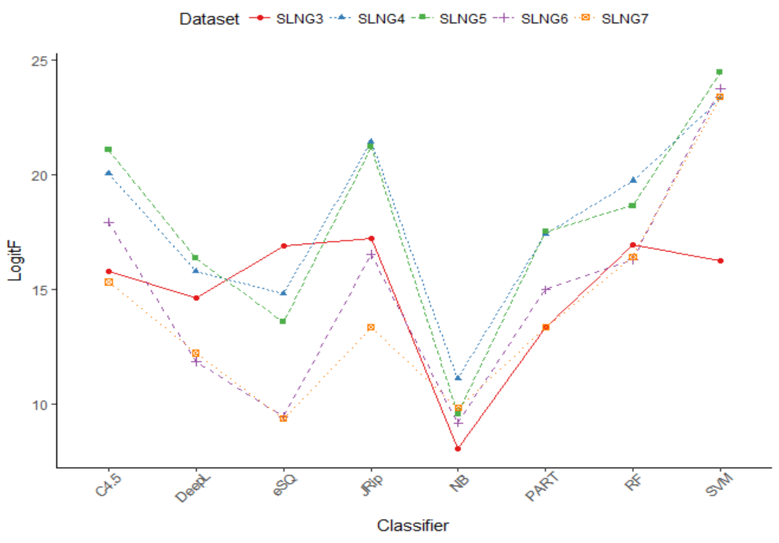


Figure 4-3: Interaction between Classifiers, Datasets and LogitF

4.6.3 Performance of proposed Sinhala Analyser

Experiments were conducted using both the Standard Lucene Analyser and our recently developed Sinhala Analyser. Table 4-3 shows the performance observed during Sinhalese document classification using our eSQ classifier.

Table 4-3: Macro F for the Lucene and Sinhala Analyser

	SLNG3	SLNG4	SLNG5	SLNG6	SLNG7
Lucene Analyser	0.9437	0.9498	0.9297	0.8995	0.8948
Sinhala Analyser	0.9186	0.9088	0.9080	0.8915	0.8854

As can be seen from our experimental results, the overall performance of the Sinhala analyser is slightly less compared with that of the Standard Lucene Analyser. However, we found that for some categories, the Sinhala analyser performed better than the Standard Lucene Analyser despite the fact that its overall performance was slightly lower. These category-level details are presented in Figure 4-4.

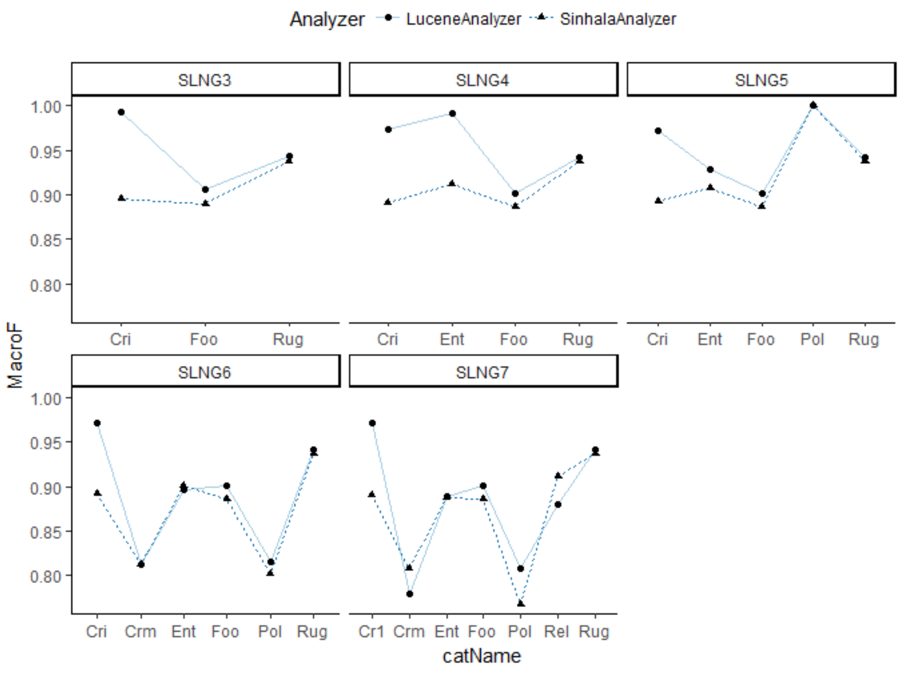


Figure 4-4: Performance for Category Level

Some identified reasons for the low performances of Sinhala Lucene’s analyser are as follows. The integrated stemming algorithm in the analyser requires significant improvement. The current implementation applies certain rules to incorrect terms, resulting in meaningless words or terms. The tokenization method used in our analyser is capable of identifying special punctuation symbols used in the Sinhalese language. However, these symbols are rare in recent web-based news articles and are predominantly used in formal Sinhalese documents. As a result, the benefits of the tokenization method do not manifest in the results due to the dataset used. However, enhancing the Lucene framework to support the Sinhalese language is a significant improvement. enhancing the Lucene framework to support the Sinhalese language is a significant improvement. It enables others to easily enhance the framework through future research and observe results with less effort. On the other hand, the standard Lucene analyser only supports the English language. It utilizes a stop-word list, tokeniser, stemming algorithms, and other algorithms finely tuned for English. Hence, it is crucial to develop a language-specific analyser. Therefore, we believe that this initial step towards developing a language-specific analyser is an important contribution to research.

4.6.4 Interpretability of eSQ Classifiers

The eSQ classifiers are highly human interpretable. Ultimately, they are merely a small number of words put together. Table 4-4 shows the eSQ classifiers produced for the SLNG3 dataset using both the Standard Lucene Analyser and our new Sinhala analyser.

Table 4-4: Sinhala eSQ classifiers

Cat Name	Analyser	F score	eSQ classifier
Rugby	Lucene Analyser	0.952	උත්සාහක(try) හැව්ලොක්ස් (Havelock) දිනුමකින්(win) රග්බි (rugby)
	Sinhala Analyser	0.952	දිනුම(win) රග්බි (rugby) හැව්ලොක්ස් (Havelock)
Football	Lucene Analyser	0.906	පාපන්දු (football) ගෝලය (goal) සම්මේලනයේ (association)
	Sinhala Analyser	0.893	පාපන්දු (football) සම්මේලන (association)
Cricket	Lucene Analyser	0.973	ඉනිම (innings) ක්‍රිකට් (cricket) දැවී (out) කඩුලු (wicket) ඉනිමේ (innings)
	Sinhala Analyser	0.910	පිත්(bat) ඉනිම (innings) කඩුල් (wicket) නොදැවෙ (not out) ඉනිම් (innings) දැවෙ (out)

For the Rugby and Football categories, our Sinhala analyser has produced more compact queries without losing much performance. However, the opposite is true in the cricket category, where the query is both longer and less accurate.

4.7 Chapter Summary

Sinhala document classification is not a well-studied subdomain of text analytics, despite the fact that this field is well matured for some languages. Our experiment has shown that eSQ is a good text classifier and produces comparable results to other popular methods while having the added advantage of human interpretability. As a part of this study, we have created a new Sinhala analyser for the Lucene full-text search engine. Results confirm that our new analyser performs better for some categories than the standard analyser does. It is also capable of producing more compact search queries. However, we note that the Sinhala stemmer integrated into our new analyser should be further improved to improve the analyser as a whole.

Chapter 5: Performance of eSQ Hybrid Clustering for Sinhala Documents

5.1 Chapter Overview

This chapter aims to cover the eSQ based clustering methods and performance received for Sinhala document collections. The chapter starts by introducing different clustering strategies, measurements, and some popular algorithms. Section 5.3, presents the method followed in eSQ clustering methods. There are two main approaches that have been taken for clustering. They are single-word based and multi-word based clustering approaches. Among them, the single-word clustering method has been studied extensively. It has been improved with hybrid clustering methods that combine three classification methods. eSQ clustering base model output was directed to kNN, kNF or NB classifiers and expanded cluster size while improving clustering performances. Results received for Sinhala document collection are presented in Section 5.5. The performance and interpretability of the multi-word clustering algorithm have been discussed at the end of the chapter.

5.2 Introduction

Document clustering is one of the main branches in text analytics. It is a popular computing technique used in many applications where text data is used. So, it is heavily employed everywhere without any boundaries. This is extremely useful for organizing, summarizing, and visualising large amounts of digital documents. Clustering methods divide a collection of documents into subgroups or clusters by looking for similarities. Therefore, documents within the same cluster should be similar, whereas documents in different clusters should be dissimilar.

Finding the similarity (closeness) or dissimilarity (distance) of an object to another object varies based on the nature of the object. For example, checking the similarity between two documents is different from doing the same for two images. However, it is mainly because of the method followed to represent the objects for computing purposes, not the way measures the similarity. Therefore, identifying features of the objects and representing them digitally is an important task before performing clustering. In most cases, this digitally represented object is referred to as a data point, whereas others are referred to as instances, records, samples, objects, cases, and so on. The vector space model is one of the popular methods used very frequently in text analytics. This model represents documents as vectors, and it helps to perform mathematical operations on text data and process them easily. So, checking the similarity between two documents becomes a measuring distance between two vectors in the vector space model in text analytics.

Few similarity measures have been used frequently to measure the distance between data points. Euclidean distance, Manhattan distance, Cosine similarity, Hamming similarity, and Minkowski distance are among them. They are different from each other based on the way they compute the distance. The following equations show the Euclidean distance and Manhattan distance respectively.

Euclidean distance: $d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$

Manhattan distance: $d(p,q) = \sum_{i=1}^{n} |q_i - p_i|$

Clustering methods have been developed using various approaches to support different types of real-world clustering problems. Similarity or dissimilarity (distance) measures are integral parts of clustering algorithm designs. Regarding the characteristics of quantitative data, distance is recommended to identify the link between the data. When working with qualitative data aspects, similarity is desired. Some of the types of clustering include the following:

Centroids-based Clustering (Partitioning methods): This type of clustering starts with a predefined number of clusters *(k).* In different algorithms, they used different methods to find the initial *k*-number of centroids. For example, the popular k-means algorithm follows a partitioning-based approach, and it uses random, farthest first, k-means++, and canopy as initialization methods. After defining the initial centroids, it creates clusters by assigning the closest points to the centroid. For the next iteration, it computes centroids again and assigns new data points. This procedure is followed until the centroid does not change or data points do not move between clusters. k-means and k-medoids are centroid-based clustering algorithms. One of the common properties of partitional methods is that every object must belong to a single cluster and does not allow overlapping. For document clustering, every document must belong to a single cluster and is not allowed to be in multiple clusters.

Connectivity-based Clustering (Hierarchical clustering): It is very common to organise things in hierarchies, and they are also very easy to interpret. Every object is connected to its neighbours, but the degree of relationship (proximity distance) varies. Based on this phenomenon, hierarchical clustering has been developed. Top-down and bottom-up are the two main approaches in hierarchical clustering. In a top-down or divisive approach, the whole dataset is considered a single large cluster from the beginning, and it breaks into subsets based on similarity repeatedly. In a bottom-up or agglomerative approach, every data point is a single cluster from the beginning, and it merges the closest clusters repeatedly. There are multiple ways of computing the distance between two clusters using the distance method. Computing the closest distance between two clusters (single link), the maximum distance between two clusters (complete linkage), and the average distance between two clusters are some of them. In addition, it is possible to develop a problem-specific objective function to measure the distance between clusters. As a result, multiple answers to the same issue might be produced depending on how the algorithm measures the distance between two clusters. One of the main advantages of hierarchical clustering is to start the clustering process without knowing the number of clusters. Later, it is possible to decide a required number of clusters (*k*) by looking at the dendrogram. The number of document categories may thus be decided later. Overlapping documents across clusters is another characteristic of the techniques. It occurred with the hierarchy’s upper levels (ancestors). Figure 5-1, for instance, displays the hierarchy of sports documents developed to group publications about sports. One of the articles included in the local cricket match category is undoubtedly a subset of the local tournaments cluster. BIRCH, CURE are two popular hierarchical algorithms.

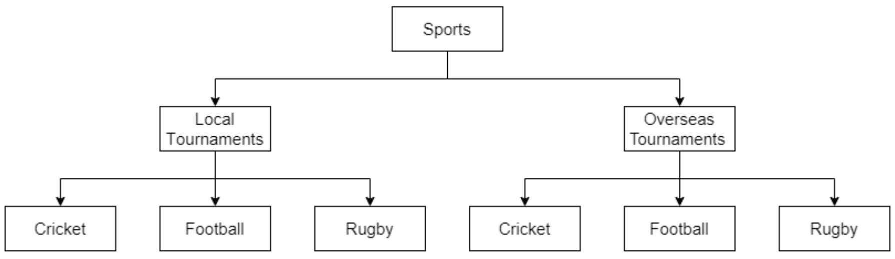


Figure 5-1- Hierarchy of Sports Documents

Density-based Clustering (Model-based methods): The density-based clustering approach prioritises density over distance. Data points are organised into regions with high concentrations of data points that are surrounded by low concentration regions. A maximum set of connected data points are grouped into the clusters in this approach. At first, it counts the number of data points for a given radius. Based on the predefined minimal data point concentration, it determines core points and non-core (marginal) points. Sequentially, it connects core data points and forms clusters by locating high-density regions. This method is capable of finding clusters that are not globular in shape compared to other methods. DBSCAN, OPTICS, Mean-shift are density-based clustering methods.

There are a few other alternative clustering approaches found in the literature. However, compared to these three, they are not as popular. Other methods include constraint-based clustering, fuzzy clustering, and distribution-based clustering (Xu & Tian, 2015). For algorithm specific details see Section 2.7.2.

In the following Section 5.3, the proposed search query based clustering (eSQ Clustering) method is explained, along with the methods and materials used.

5.3 Method used in eSQ Clustering

5.3.1 eSQ Clustering

The evolved Search Query (eSQ) clustering approach uses GA. It is critical to determine the number of clusters required early in the clustering process and to set this in the GA. If the number of clusters is equal to k, it generates k search queries, one for each cluster. The key steps of the eSQ clustering method are demonstrated below using the 20Newsgroup dataset (NG5).

Step 1: Data Pre-processing

From the beginning of the pre-processing, the text is converted into lowercase for English language and tokenising, removing stop-words, and stemming is carried out as necessary. It helps to improve the clustering performance. Creating Lucene indexes is performed next, and every document is assigned to a pre-set category using a Lucene field. But GA does not have access to these categories. It is used only for cluster evaluations at the end.

Step 2: Creating wordlist

In this step, it creates a list of important words or feature set for the clustering process. GA engine uses them for query building. To create the important word list, the TF*IDF (Term Frequency * Inverse Document Frequency) score for each word in the documents is calculated. TF is the number of occurrences of a word in a document and IDF is the inverse of the number of documents in which the word occurs. TF-IDF is used in text analytics often as a term weighting method. In this study, it is used to find an important list of words or feature set that has the most power to separate one from another document.

After the words have been ranked according to their total TF*IDF score, the top 100 terms are chosen to be used in the construction of queries. Before the beginning of the evolution, it only needs to do this step once for each index; after that, the list is considered to be finalised. The integer index is merely a representation of the position of each word inside the TF*IDF ordering. The length of the list is just eight items in the illustration that is provided in Table 5-1.

Table 5-1: Example Important Wordlist

0	1	2	3	4	5	6	7
space	nasa	god	orbit	hockey	file	sale	game

Step 3: Determine *k*

Determine the number of required categories (k). For this clustering method, it is essential to determine the number of clusters from the beginning. In other words, we need to decide the value for k or the required number of categories.

Step 4: Create generation 0

Table 5-2 displays a representative chromosome from the population of generation 0. The chromosomes are represented as integers ranging from 0 to 100, corresponding to the word list’s maximum size.

Table 5-2: Creating single-word search queries

	SQ0	SQ1	SQ2	SQ3	SQ4
Chromosome	0	4	5	1	7
Query Word	space	hockey	file	nasa	game

The number of chromosomes in the population is determined by the parameters of the genetic algorithm (GA). In this study, the number of chromosomes is set to 512, as shown in Table 5.3.

In the following example, each gene within the chromosome represents a single-word search query (SQ). Each search query identifies a cluster consisting of a collection of documents that contain the specified word. Let’s consider the SQ2 chromosome, which includes the integer 5, representing the word “file” from the word list provided in Table 5-1. Consequently, the initial population for SQ2 is comprised of documents that match the search query “file”. This means that every document within this specific category from the document collection contains the word “file”.

Step 5: Fire each query in the set.

In this specific example, five distinct search queries are constructed for the NG5 dataset, representing five different categories within the given chromosome. Each search query contained in the individual in the population is executed, and subsequently, the individual’s fitness value is determined (refer to section 5.3.2 for details on the fitness function). The fitness value is computed by counting the number of unique documents returned by each search query within the individual. This process should be repeated for every individual in the population. The resulting unique hit counts serve as fitness functions.

Step 6: Apply genetic operators to create a new generation.

eSQ uses the following set of parameters in experiments. They are summarized in Table 6-3, and brief explanations have been given below.

Table 5-3: GA Parameters for eSQ Clustering

Parameter	**Value**
Selection Type	Tournament
Tournament size	5
Subpopulations	4
Population Size	512
Generations	300
Crossover Probability	0.8

Mutation Probability	0.1
Elitism	Best 2 individuals

The GA program utilizes the Tournament selection method with a tournament size of five. In each tournament, a subset of five individuals is randomly selected from the population. The fitness of each individual in the tournament is evaluated, and only the fittest individual, with the highest fitness value among the five, is selected as the winner. This winner then proceeds to the next generation or mating pool, while the remaining individuals in the tournament are not chosen for reproduction.

The subpopulation structure consists of four separate subpopulations. Each subpopulation operates independently and contains its own set of individuals. This approach allows for parallel processing and encourages the exploration of diverse regions within the solution space. By maintaining separate subpopulations, the algorithm increases the chances of finding optimal or near-optimal solutions and avoids premature convergence to suboptimal regions. However, in some cases, limited communication between subpopulations is useful to introduce a level of cross-pollination. This migration can help share beneficial genetic material and promote exploration across different subpopulations. We use an island model with 4 subpopulations to increase diversity and exchange 3 individuals every 50 generations.

The population size is set to 512, determining the total number of individuals in the entire population. A larger population size enhances the diversity and exploration capabilities of the algorithm. With more individuals, the algorithm can explore a broader range of potential solutions, increasing the likelihood of finding better solutions.

The GA program runs for a total of 300 generations, representing the number of iterations or evolutionary cycles the algorithm undergoes. In each generation, individuals are selected through tournament selection, undergo crossover and mutation operations, and are evaluated for their fitness. Through successive generations, the algorithm aims to improve the fitness of the population and converge towards optimal solutions.

The crossover probability is set at 0.8, indicating an 80% chance of crossover occurring between selected individuals during reproduction. Crossover involves exchanging genetic information between two parent individuals to create offspring. This process promotes exploration and the combination of beneficial genetic material, potentially leading to the discovery of better solutions.

The mutation probability is defined as 0.1, representing a 10% chance of genetic mutations occurring in individuals during reproduction. Mutation introduces random changes to the genetic material, allowing for additional exploration and the possibility of escaping local optima. It facilitates the discovery of novel solutions that may not be present in the initial population.

The elitism strategy is implemented by preserving the best two individuals from each generation. These top-performing individuals, determined by their fitness values, are directly carried over to the next generation without any modifications. By retaining the best solutions, the algorithm ensures that valuable genetic material is not lost and maintains a level of consistency throughout the evolutionary process, aiding in convergence towards better solutions.

Step 7: Repeat steps 5-6 for 300 generations as the termination criteria, and select the individual with the highest fitness. For each new population, fire search queries and calculate the fitness level. Use the offspring of the selected individuals to create the next generation. Repeat these steps for 300 generations before terminating the evolution.

Step 8: Cluster evaluation

Following the completion of the runs, the F1-measure was utilised to assess the performance of the clusters. When calculating the F1-measure, we used the original category labels. We performed the experiment 11 times since GA comprises many random components. Further, we analysed the F1-measure values of 11 runs and found no significant variability between different runs. Figure 5-2 depicts the results, showing that there is no significant variability.

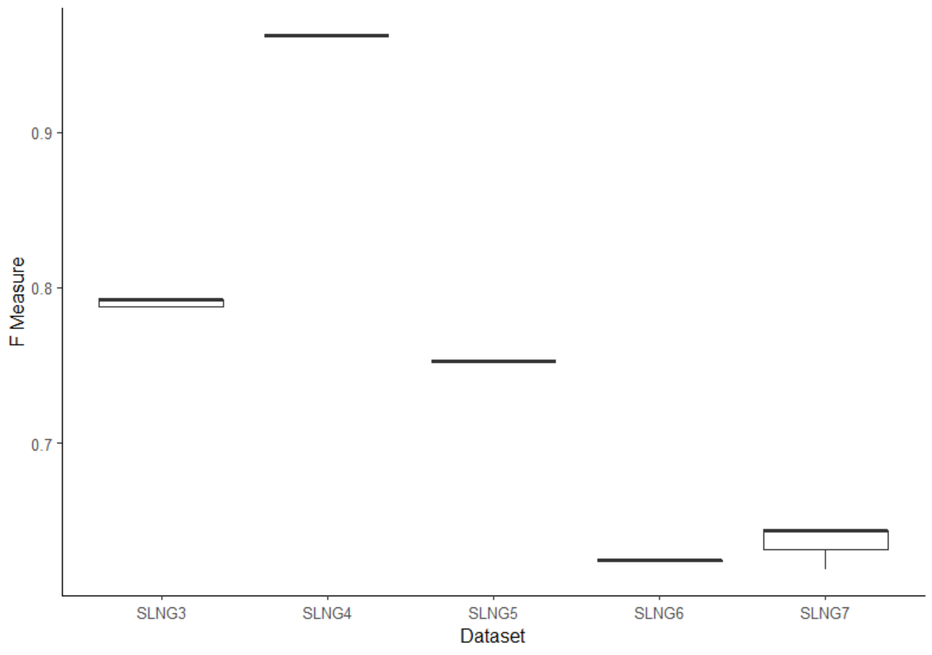


Figure 5-2: F1-measure for 11 runs

5.3.2 Fitness Calculation

Document clustering is to return collections of documents that are similar to one another but not documents in other clusters. We have developed a fitness function whose purpose is to cluster a document collection. This fitness function has been constructed assuming that the required number of clusters (*k*) is known beforehand.

When measuring fitness from a set of queries created by a chromosome, we define unique documents (*uniqueHits*) as the number of documents returned by precisely one query. The summation of *uniqueHits* of an individual is considered as the fitness function.

5.3.3 eSQ Single-word – Hybrid Clustering

At the conclusion of iterations of eSQ clustering method, the individual that was chosen will generate a set of search queries of a single word, and a cluster will be the collection of documents that include that search query word. On the other hand, it's possible that many of the documents in the collection don’t have any of the query terms at all. It has found that by using a classifier to allocate the unassigned documents to their closest search query determined clusters, it is able to boost the efficacy of our overall operation.

We use the Lucene implementation of the K-Nearest Neighbour (KNN), Naïve Bayes classifier (NB), and k-Nearest Fuzzy classifier (KNF) algorithms to add each unassigned document to its closest cluster. It creates three type of hybrid clustering methods and we have named them as eSQ-HSW-KNN, eSQ-HSW-KNF, and eSQ-HSW-NB.
eSQ-HSW-KNN stand for eSQ Hybrid Single-word *k*-Nearest Neighbour
eSQ-HSW-NB stand for eSQ Hybrid Single-word Naïve Bayes classifier
eSQ-HSW-KNF stand for eSQ Hybrid Single-word k-Nearest Fuzzy classifier

5.3.4 eSQ Multi-word Clustering

eSQ clustering changed for multi-word queries by extending the length of the genome. For example, doubling the length of the genome allows for two-word queries and taking the modulus of *k* to determine which query each gene relates to. A word can only be added once to a set of queries so if the genome specifies two or more occurrences of a particular word, only the first occurrence is used. Where a query is made of two or more words they are connected with a logical OR (disjunction) such that documents are returned which contain any of the words in the query.

When building a query specified by a chromosome, we have found it useful to add a requirement for queries made of two or more words such that the sets of documents returned by queries made from the individual query terms intersect. Before we add a new term (*newTerm*) to a query already containing a term (*rootTerm*), we must first check that the intersect requirement is met by calculating the following:

andCount:	count of documents containing the *newTerm* AND the *rootTerm*
newTermCount:	count of documents containing the *newTerm*
intersectRatio:	*andCount*/*newTermCount*

This method has the advantage of making the first word in a query more likely to be a good cluster label. We have experimented with various values for the minimum *intersectRatio* and have found 0.6 to be a suitable value. If *intersect Ratio* >= 0.6 the word is added to the query otherwise nothing is added. In other words, before we add a new word to a query, we check that at least 60% the documents which contain the new word also contain the root word.

The following example generating 3 search queries (SQ0, SQ1, SQ2). The chromosome details are given in Table 5-4.

Table 5-4: chromosome to determine k and create 3 search queries (SQ).

Representation	**K**	**SQ0**	**SQ1**	**SQ2**	**SQ0**	**SQ1**	**SQ2**	**SQ0**	**SQ1**	**SQ2**
Gene Index	0	1	2	3	4	5	6	7	8	9
Chromosome	3	0	7	2	3	3	5	1	4	2

In this case the chromosome specifies a *k* value of 3 meaning that 3 clusters will be created. By referring to the Table 5-1 important word list, the following queries shown in Table 5-5 will be created.

Table 5-5: Creating multi-word queries

	Gene	**Specified Words**	**Final Query**	**Comment**
SQ0	0,3,1	*space, orbit, nasa*	*space* OR *orbit* OR *nasa*	*orbit* and *nasa* both have a high intersect value with space
SQ1	7,3,4	*game, orbit, hockey*	*game* OR *hockey*	*orbit* does not meet the intersect requirement for the root word *game*
SQ2	2,5,2	*god, file, god*	*god*	*file* does not meet the intersect requirement for the root word *god*. Repeated word is ignored.

The experiments were carried out in Section 5.4, and the findings were described in Section 5.5.

5.4 Experiment aims and materials

5.4.1 Experimental objectives

Experiments were conducted with the following objectives:

a. To verify experimentally if the performances of the three proposed hybrid clustering methods deviate significantly from one another.
b. To compare performance improvement of eSQ-HSW over eSQ
c. To compare performances of eSQ-HSW and eSQ-MW methods.
d. To compare Performances of eSQ-based methods with other popular methods for Sinhala and English document clustering.
e. To observe the human readability of the search queries produced by eSQ-based clustering methods.

5.4.2 Datasets

With a few modifications, we used the SLNG Collection described in Section 4.4 for this set of experiments. Only publicly accessible Sinhala news stories are included in this collection, which was created via web scraping. More details about the Sinhala document collections are given in the Table 5-6.

Table 5-6: The Structure of SLNG Collection

Dataset name	No. of categories	No. of documents	Category names [Category size]
SLNG3	3	2550	cricket [850] / football [850] / rugby [850]
SLNG4-2	4	1600	sports [400] / health [400] / business [400] / entertainment [400]
SLNG5-2	5	800	sports [200] / crime [200] / politics [200] / religion [200]
SLNG6	6	4450	cricket [850] / football [850] / rugby [850] / entertainment [1500] / politics [200] / crime [200]
SLNG7	7	4650	cricket [850] / football [850] / rugby [850] / entertainment [1500] / politics [200] / crime [200] / religion [200]

In addition to SLNG, several benchmarking English document sets were used in the experiments, and details about them is previously given in Section 3.6.

5.5 Results and Discussion

5.5.1 Evaluation of hybrid clustering methods

The following Table 5-7 presents summary of results received by executing eSQ-HSW-KNN, eSQ-HSW-KNF, and eSQ-HSW-NB clustering methods. These results received from SLNG collection (Sinhala news document collection) given in Table 5-6. From the initial observations, we found that accuracies highly depend on datasets and performances slightly varies based on the Hybrid clustering method used.

Table 5-7: Mean and Standard Deviation of three hybrid methods

Dataset	eSQ-based Clustering Methods	F1-measure	
		mean	std
SLNG3	eSQ-HSW-KNF	0.7900	0.0020
SLNG3	eSQ-HSW-KNN	0.7990	0.0007
SLNG3	eSQ-HSW-NB	0.8020	0.0038
SLNG4-2	eSQ-HSW-KNF	0.5380	0.0251
SLNG4-2	eSQ-HSW-KNN	0.5430	0.0275
SLNG4-2	eSQ-HSW-NB	0.5580	0.0215

SLNG5-2	eSQ-HSW-KNF	0.5490	0.0051
SLNG5-2	eSQ-HSW-KNN	0.5490	0.0016
SLNG5-2	eSQ-HSW-NB	0.5320	0.0132
SLNG6	eSQ-HSW-KNF	0.6250	0.0005
SLNG6	eSQ-HSW-KNN	0.6240	0.0010
SLNG6	eSQ-HSW-NB	0.6190	0.0015
SLNG7	eSQ-HSW-KNF	0.6370	0.0117
SLNG7	eSQ-HSW-KNN	0.6390	0.0092
SLNG7	eSQ-HSW-NB	0.5520	0.0746

We plotted standard error bars for three clustering algorithms against five datasets from the SLNG Collection, as shown in Figure 5-3. It also demonstrates that there are significant variances in F measure for some datasets. Three clustering techniques performed well on the SLNG3 dataset, with very little difference in F measure. We have news stories from three sports categories in SLNG3, as well as several overlapping terms between them. However, because GA-based methods are effective at identifying optimum solutions in such settings, eSQ Clustering approaches provide very good accuracy (F1 measure). Categories of other datasets are substantially diverse, which may be one of the reasons for their poor performance. We can also observe the SLNG7 and SLNG4-2 datasets, and the performance varies between runs during the studies.

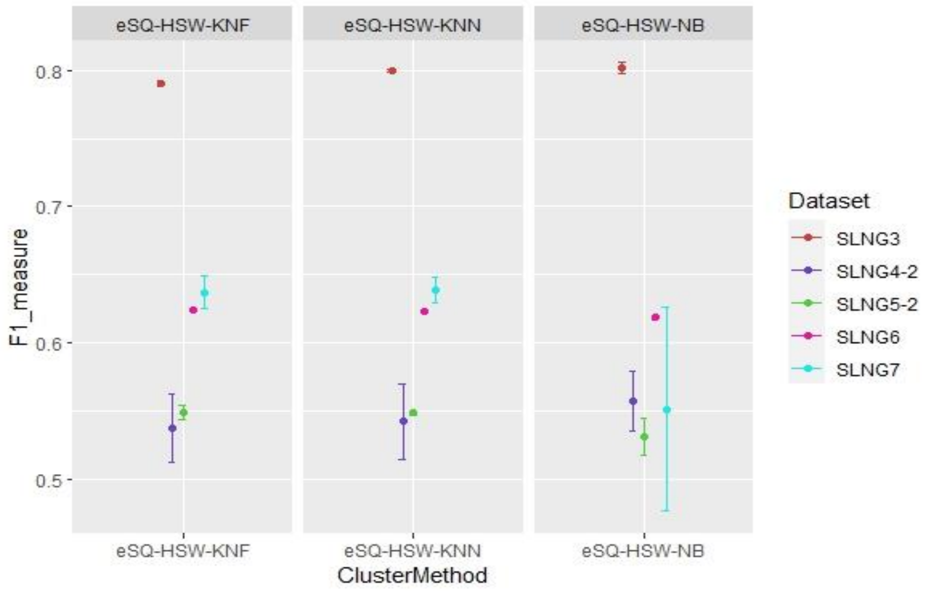
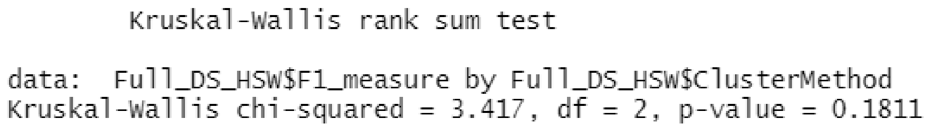


Figure 5-3: Standard error bars

To determine whether these three methods has significantly different performances or not, we have conducted formal statistical analysis. Since we have lower number of observations (11) for a group and measured values are not normally distributed, nonparametric tests are more appropriate. Kruskal-Wallis rank test is useful to check the significant of the results of more than two independent systems. We assess the following hypothesis using the Kruskal-Wallis test. Null Hypothesis and alternative hypothesis as follow,

H_0: There is no significant difference in performance between the three hybrid methods.
H_a: There is a significant difference in performance between the three hybrid methods.

The Figure 5-4 show the output of Kruskal-Wills test.

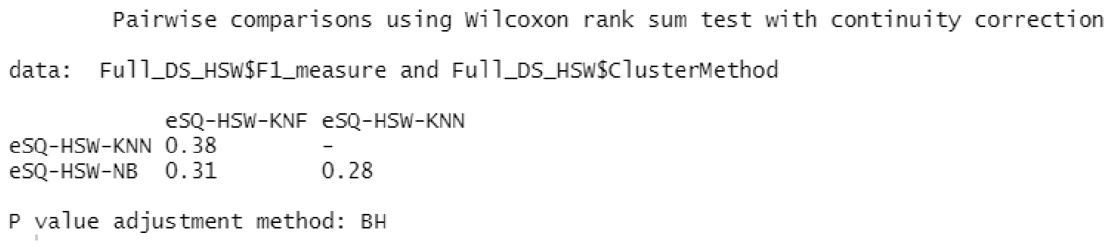


```
                Kruskal-Wallis rank sum test

data:  Full_DS_HSW$F1_measure by Full_DS_HSW$ClusterMethod
Kruskal-Wallis chi-squared = 3.417, df = 2, p-value = 0.1811
```

Figure 5-4: Kruskal-Wills test results for Hybrid clustering

As shown in the Figure 5-4, p-value of Kruskal-Wallis test is greater than 0.05, that is H_0 is accepted and there are no significant differences between medians of different groups of results produces by three hybrid clustering methods.

We applied the Wilcoxon test to see pairwise comparisons. The Figure 5-5 illustrate the results received from the Wilcoxon test. There isn't a single combination with a p-value less than 0.05. That is, there is no significant difference between any two of the eSQ clustering methods for the given dataset.



```
        Pairwise comparisons using Wilcoxon rank sum test with continuity correction

data:  Full_DS_HSW$F1_measure and Full_DS_HSW$ClusterMethod

            eSQ-HSW-KNF eSQ-HSW-KNN
eSQ-HSW-KNN 0.38        -
eSQ-HSW-NB  0.31        0.28

P value adjustment method: BH
```

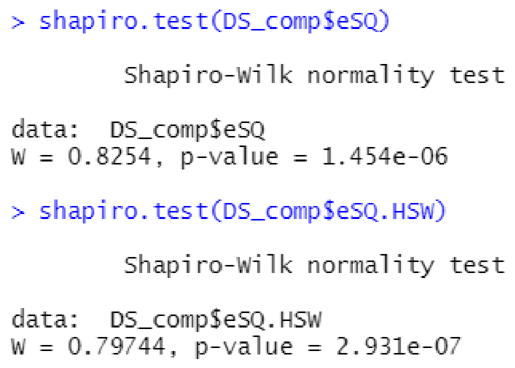
Figure 5-5: Wilcoxon test results for Hybrid methods

From the series of statistical tests, it confirmed that there are no significant deviations between three hybrid clustering methods called eSQ-HSW-KNN, eSQ-HSW-KNF, and eSQ-HSW-NB.

5.5.2 Comparison between eSQ vs eSQ-HSW

eSQ clustering method is single word query returns the documents of a particular category. We expanded eSQ clusters using different classifiers supported by Lucene framework. We observed that eSQ Hybrid Single-word clustering method outperformed compared to eSQ clustering method. In this section, we confirm the same using statistical methods.

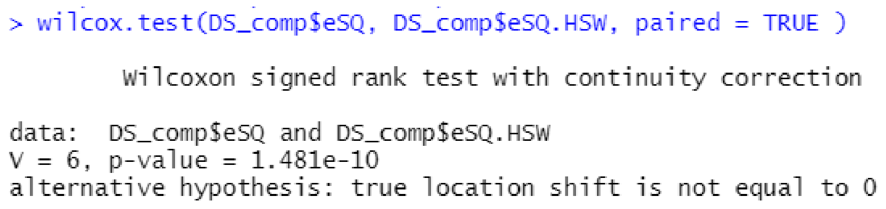
First, we checked the normality of the results using Shepiro-Wilk test. The Figure 5-6 shows that p-values received for both eSQ and eSQ-HSW F1 measure values are less than 0.05. Therefore, results are not normally distributed, and parametric test are not suitable.



```
> shapiro.test(DS_comp$eSQ)

        Shapiro-Wilk normality test

data:  DS_comp$eSQ
W = 0.8254, p-value = 1.454e-06

> shapiro.test(DS_comp$eSQ.HSW)

        Shapiro-Wilk normality test

data:  DS_comp$eSQ.HSW
W = 0.79744, p-value = 2.931e-07
```

Figure 5-6: Normality test for eSQ vs Hybrid method

We have applied Wilcox test to assess the significance of the performance levels of two method. We compared results as a paired test since we want to compare the results of systems where their performances before develop hybrid method and after develop hybrid method.



```
> wilcox.test(DS_comp$eSQ, DS_comp$eSQ.HSW, paired = TRUE )

        Wilcoxon signed rank test with continuity correction

data:  DS_comp$eSQ and DS_comp$eSQ.HSW
V = 6, p-value = 1.481e-10
alternative hypothesis: true location shift is not equal to 0
```

Figure 5-7: Wilcoxon Test for eSQ vs eSQ-HSW results

The Figure 5-7 depicts the results of Wilcoxon test. Its p-value is lesser than 0.05. Therefore, it confirmed that there is a significant performance difference between these two methods. The following Figure 5-8, boxplots are drowned for the same dataset. It is visually show that the median values of two methods are clearly deferent.

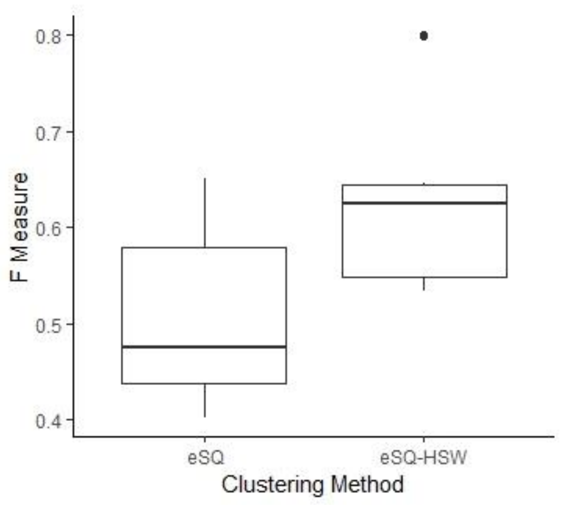


Figure 5-8: Boxplot for eSQ and eSQ-HSW F measure

5.5.3 Comparison between eSQ-MW vs eSQ-HSW

The eSQ-MW is a variant of eSQ based clustering method. It produces multi word search queries to perform clustering. These words connected from OR operator. The following Figure 5-9 shows F measure comparison visually. Except for SLNG3 dataset, other datasets F measures are comparable for Sinhalese document collection. This result motivates to integrate classifiers to expand muti-word clustering.

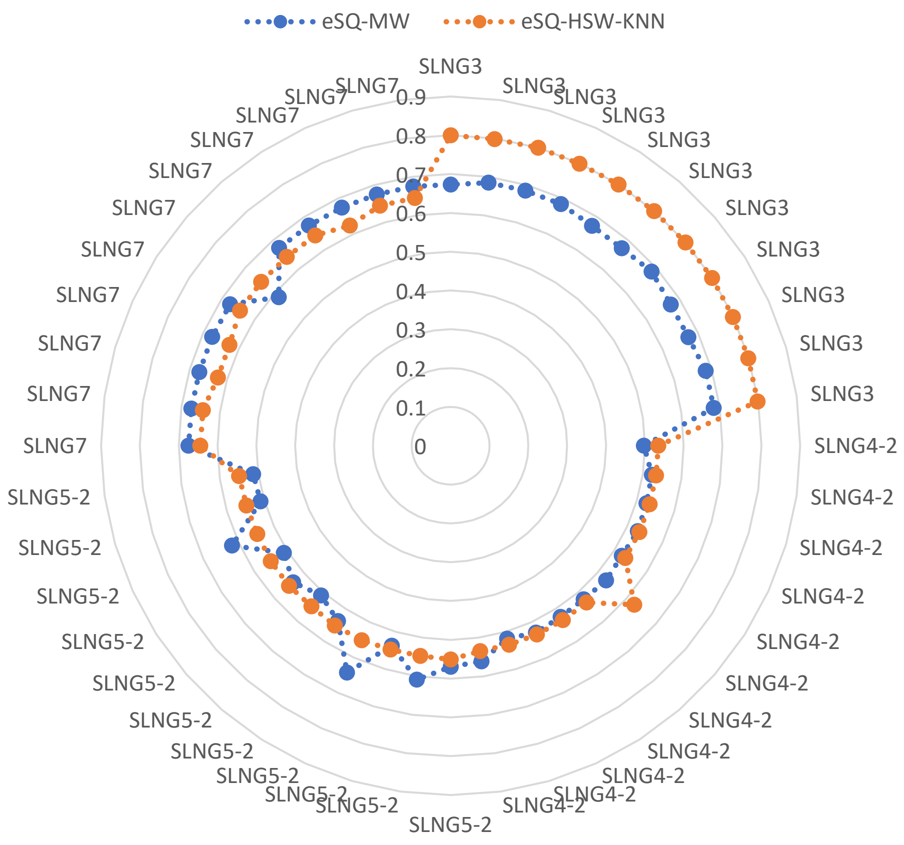


Figure 5-9: Comparison of Multi-word vs Single-word Clustering

5.5.4 Performance of eSQ based methods with others for English Documents

We have presented F measures received for benchmarking English document collections in the Table 5-8. It includes eSQ based clustering methods. In the Table 5-9 shows F measure improvement from eSQ to eSQ hybrid model for English document clustering. These results presented in Hirsch et al. (2021).

Table 5-8: F1 comparison with other popular methods

Collection	eSQ	eSQ-HSW-KNN	farthest first	kmeans++	kmeans	EM	HieraClus
CLASSIC4	0.554	0.822	0.780	0.742	0.750	**0.916**	0.347
CRISIS3	0.743	**0.862**	0.760	0.748	0.711	0.776	0.677
NG3	0.671	0.915	0.588	0.926	**0.935**	0.916	0.866
NG5	0.685	**0.895**	0.729	0.607	0.645	0.670	0.596
NG6	0.637	**0.895**	0.818	0.725	0.641	0.599	0.628
R4	0.784	**0.913**	0.758	0.763	0.771	0.693	0.687
R5	0.887	**0.941**	0.658	0.679	0.596	0.717	0.775
R6	0.619	**0.672**	0.651	0.660	0.653	0.602	0.606

Table 5-9: F measure comparison between eSQ and eSQ-HSW-KNN

Collection	eSQ		eSQ-HSW-KNN	
	F1 average	std	F1 average	std
CLASSIC4	0.554	0.000	0.822	0.000
CRISIS3	0.743	<0.001	0.862	0.000
NG3	0.671	0.000	0.915	<0.001
NG5	0.685	0.000	0.895	0.000
NG6	0.637	<0.001	0.895	0.007
R4	0.784	0.000	0.913	<0.001
R5	0.887	0.000	0.941	0.004
R6	0.619	<0.001	0.672	<0.001

These results further confirmed that eSQ based clustering methods are comparable with other popular methods.

5.5.5 eSQ based clustering verses other popular methods against Sinhala document

The Figure 5-10 illustrate the F measures received for different clustering methods including the eSQ based methods. These results received for SLNG Sinhala news collection. These results show that eSQ based clustering methods comparable with other popular methods for Sinhalese datasets.

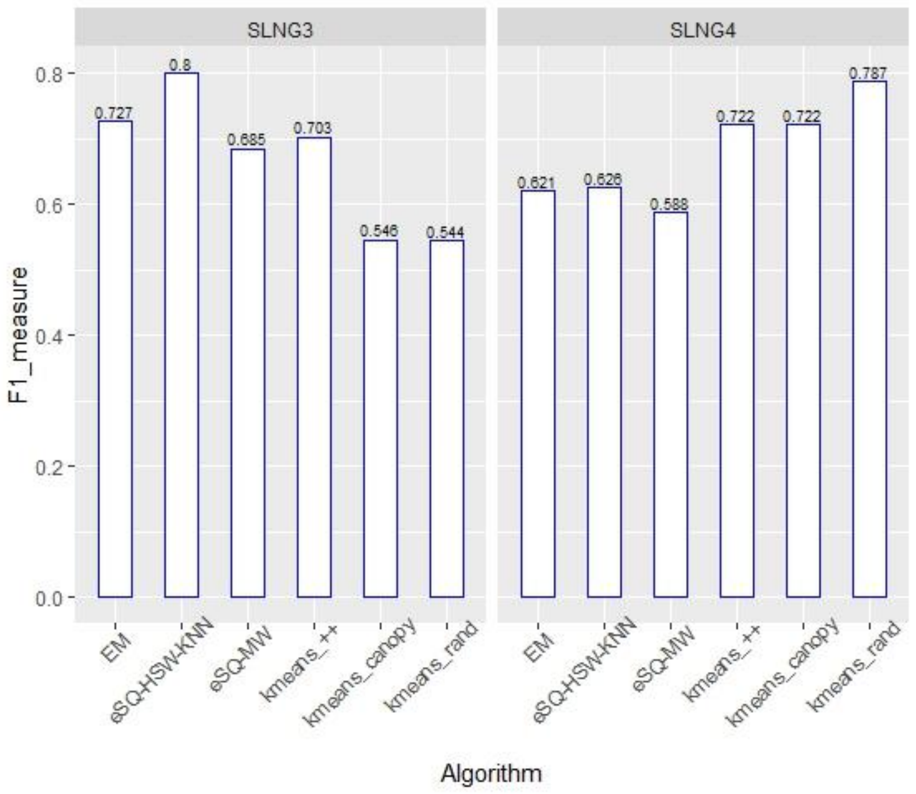


Figure 5-10: F measures of clustering methods for Sinhala documents

5.5.6 Explainability of eSQ based Clustering

From two clustering methods, eSQ Single-word hybrid and eSQ Multi-word, eSQ Multi-word method is more human interpretable method than hybrid method. Even if the F measure is comparable to other popular clustering strategies, our search query-based method is obviously the most human friendly. For example, Table 5-10 and Table 5-11 show queries generated by for Crisis3 and NG5 datasets. Returning documents from search queries confirmed that they contain any of these words.

Table 5-10: Crisis 3 Query Set

Category	**Search Query**	**F1**	
Boston bombing	boston tragedy heart explosions fbi marathon suspect	0.76	0.78
Colorado wildfires	colorado fire fires wildfire homes waldocanyonfire	0.83	
Queensland floods	bigwet brisbane coast river qpsmedia bundaberg	0.74	

Table 5-11: NG5 Query Set

Category	**Search Query**	**F1**	
soc.religion.christian	god jesus christians	0.85	0.8
comp.os.ms-windows.misc	windows file	0.81	
rec.sport.hockey	nhl hockey team players	0.84	
sci.space	space orbit nasa	0.81	
misc.forsale	sale	0.71	

The following tables, Table 5-12 and Table 5-13, contains queries generated from eSQ Multi word clustering for Sinhalese language. It also produces human interpretable set of words like English search queries.

Table 5-12: SLNG3 Query set

Evolved Search Queries from eSQ-MW			
Category	Query	F1	
Football	මහතා සභාපති කටයුතු මම සම්මේලනයේ පිළිබඳව	0.48	0.68
Cricket	කඩුලු දැවී පන්දුවක්	0.88	
Rugby	උත්සාහක මිනිත්තු දඬුවම් දඬුවම් හැව්ලොක්ස්	0.68	

Table 5-13: SLNG4 Query set

Evolved Search Queries from eSQ-MW		F1	
Category	Query		
Football	පාපන්දු සම්මේලනයේ	0.88	0.79
Cricket	ක්‍රිකට් ඉනිම කඩුලු දැවී	0.96	

Rugby	උත්සාහක හැව්ලොක්ස් රග්බි	0.94	
Entertainment	ඇය	0.33	

Tables 5-14 and 5-15 below display search queries created by the eSQ Single word hybrid clustering algorithm. These search queries are appropriate for cluster labelling, but not for clustering. Because clustering is an outcome of two-step process. First clusters formed using eSQ method and then cluster expansion careered out using a classifier. Some of the documents close to search queries are assigned by classifier it is not visible to end users. Therefore, eSQ Single-word clustering method is not fully interpretable to end users.

Table 5-14: SLNG 3 Single word query set

	Evolved Search Queries from eSQ-HSW		
	Category	Query	F1
Dataset: SLNG3	Football	මහතා	0.80
	Cricket	කඩුලු	
	Rugby	උත්සාහක	

Table 5-15: SLNG4 Single word query set

	Evolved Search Queries from eSQ-HSW		
	Category	Query	F1
Dataset: SLNG4	Football	පාපන්දු	0.95
	Cricket	ක්‍රිකට්	
	Rugby	රග්බි	
	Entertainment	වන	

Search queries found by eSQ GA engine are having power of distinguishing documents into the categories from the document collection. The fitness test of GA guarantees that these words, which are not included in any of the other categories, are included in this category. Also, intersect requirement gives extra importance to the first word in each query. We found that in many queries, the original label or a variant of it was picked as the first word. This is important human interpretability point of view and creates trust among users about the systems as an additional advantage.

.

5.6 Chapter Summary

We presented different clustering strategies and related concepts. The eSQ clustering method is explained using a step-by-step process. In this chapter, two main eSQ clustering methods are experimentally tested for Sinhala documents. Single-word hybrid clustering is one of them, and this includes the eSQ-HSW-KNN, eSQ-HSW-KNF, and eSQ-HSW-KNB methods. The experiment results were statistically analysed, and a conclusion was recorded. For multi-word clustering, the results received were also studied statistically and presented in the relevant sections. We finally added a discussion on the interpretability of the eSQ clustering method.

Chapter 6: Conclusions and Future Work

6.1 Chapter overview

This chapter is to summarise the entire study and its findings. In section 6.2, we provide an overview of the completed work. In section 6.3, a list of recommendations based on the results and acquired experience will be provided. The section 6.4 discusses the primary contributions to the research, while the section that follows describes the limitations encountered throughout the study. Finally, we would like to suggest a few directions for future research.

6.2 Revisiting the Research

This research was aimed at analysing search query evolution in document classification and clustering. This analysis was carried out to answer three main research questions.

First question, how explainable are eSQ based methods compared to other document classification and clustering methods? We have defined an objective to study existing classification methods and categorise them based on human friendliness. In our taxonomy, one of the unique characteristics considered was the ability to modify the classification methods, as we found that it is useful for certain applications, especially to incorporate the knowledge of domain experts. It could happen for two reasons. One is that the data collected does not include some useful details for predictions. The second one could be that model construction techniques are not capable of detecting such details from a training set. However, it is essential, to incorporate this intelligence back into the model by allowing users to fine-tune it. Our analysis reveals that eSQ classifiers and some rule-based methods have the highest level of human friendliness and capable to modify by end users. Most of the other popular methods are not as easy to modify compared to eSQ and rule-based classifiers. As a workaround, attempts have been made to extract classification rules from base models if the predictive model is a black box.

Second question, what is the impact of different feature selection metrics when producing an important-term list for the eSQ GA engine? The eSQ system's GA engine takes a set of words as an important-term list to produce evolved search queries. These important-term lists are produced by ranking words in a document category based on different feature selection metrics. We analysed four global feature selection metrics on two different benchmarking document datasets. Further, we tested eSQ performance over different lengths of feature sets (important-term lists). Overall, there is no significant impact on performance because the GA engine can find optimal solutions. However, we can see that the odds ratio produces slightly different important-term lists and caused for low classification accuracies when compared to the other three methods. CHI and F1 feature selection methods with shorter lengths of important-term lists give more accurate results than IG and OR when utilising the eSQ classifiers. And both perform well at length size 50, while CHI also performs reasonably well at length size 20. IG and OR are very sensitive to length of important-term list.

Third research question, how do eSQ based classification and clustering perform for Sinhala documents? Two objectives of the research design have been covered by this question. In general, we examined the eSQ system's performance for document classification and clustering methods.

Specific objectives and conclusions drawn based on the analysis of document classification are listed below.

- The first objective was to investigate whether the performance of our eSQ classifier when classifying Sinhalese documents deviates significantly from other popular classifiers. It is reasonable to report that eSQ classifiers perform well for Sinhala documents and within top classifiers.
- The second objective was to compare classification performance between our Lucene Sinhala Analyser and the Lucene Standard Analyser when classifying Sinhalese document using eSQ. The overall performance of the Sinhala analyser is slightly less compared with that of the Standard Lucene Analyser. However, we found that for some categories, the Sinhala analyser performed better than the Standard Lucene Analyser despite the fact that its overall performance is slightly lower.
- The third objective was to investigate the human readability of the eSQ classifiers produced by our Sinhala analyser and the Lucence Standard analyser. Experimental results do not show any damage to the readability of the search query classifiers due to the Sinhala analyser. However, we notice that it required some improvements to the stemming methods used. It has a negative impact on the Sinhala analyser's performance.

Specific objectives and conclusions drawn based on the analysis of document clustering are listed below.

- A study to see if the performance of three proposed hybrid clustering methods differed significantly from one another concluded with no significant differences.
- The results of an analysis comparing the F measure improvement of eSQ-HSW and eSQ for Sinhala document clustering indicate that cluster expansion employing classifiers has significantly enhanced performance.
- The study’s final purpose was to assess the human readability of search queries generated using eSQ-based clustering algorithms. With the use of classifiers for cluster expansion, the hybrid clustering approach becomes increasingly complex and hides expansion details from users. Therefore, we cannot state that the eSQ hybrid clustering approach is entirely interpretable. It requires more development to be fully interpretable by end users.

6.3 Recommendations

We would like to do the following recommendations based on the observations and findings of the experiments.

- The eSQ classifiers are among the top solutions for document classifications if the explainability is important. eSQ capable of producing best results when the document categories are very closely related. For example, to separate football, rugby and cricket documents from a single document repository.

- eSQ-HSW (Hybrid Single Word) clustering is the best performing clustering method from eSQ base clustering methods. It does not have significant performance variation between developed cluster expansion methods (i.e., KNN, KNF and NB). eSQ clustering methods also performed well compared to other methods when the clusters are having many overlapping documents or documents from closely related areas.

- Interpreting clusters in unsupervised learning is a manual process. But eSQ clustering queries are highly human interpretable and single word queries, it can be used for automatic cluster labelling without human involvement. This is a very unique feature of eSQ clustering.

- From CHI, F1, IG and OR four feature selection methods, CHI and F1 performs well for small feature set lists whilst IG and OR is sensitive to length of the features set.

- From four feature selection metrics, F1 performs slightly better than CHI for all the categories of the R10 dataset. But for CHI produces relatively better results for most of the categories. Therefore, we recommend to replace default F1 feature selection method from CHI for eSQ classifiers.

6.4 Contribution to Research

Classification and clustering of documents have been studied for many years, but they continue to be one of the most active study fields. In this work, we have made the following significant contributions to the field of study:

1. Developed a new taxonomy considering the human friendliness of document classification. It is important for higher human friendliness, not only interpretability. Existing studies have not given much attention to the modifiability (ability to fine-tune) of predictive models for end users' requirements. Considering this idea, we have developed a taxonomy for document classification and evaluated it against the popular classifiers.

2. We have extended the capabilities of eSQ classifiers by implementing feature selection metrics and integrating them into the eSQ system. Furthermore, we analysed the effects of four feature selection metrics on document classification for eSQ classifiers experimentally.

3. Enhanced eSQ classifiers for Sinhala: The Apache Lucene indexing framework supports many human languages but not Sinhala. We have recompiled the Lucene project with basic pre-processing functionalities, including tokenizing, removing stop-words, and stemming for Sinhala. An enhanced Lucence framework is

integrated into the eSQ classification system to conduct experiments with Sinhala documents.

4. We have extended the eSQ clustering method by integrating new classification models for cluster expansion. For Sinhala documents, the performance of eSQ hybrid clustering methods (eSQ-HSW-KNN, eSQ-HSW-KNF, eSQ-HSW-NB) was evaluated.

5. Developed a labelled Sinhala document collection named “SLNG Collection” for machine learning research. SLNGCollection is publicly available at https://github.com/psumathipala/SLNGCollection.git

6.5 Limitations

This study was conducted with some constraints. Some of them were obvious from the start of the research, but others were discovered only afterwards. The eSQ system was originally developed using the Java and Groovy programming languages. And the text indexing framework Lucene is also developed using Java. So, we have to stick with these Java and Groovy programming languages for making changes. But, later, it became one of the barriers, as Java doesn’t support much for machine learning compared to Python. One of the challenges we faced was finding a library that was capable of explaining the predictive models.

Working with Sinhalese documents became another trouble since it was hard to find well-established pre-processing tools like stop-word lists, tokenizers, and stemmers. And also, we were not able to find suitable datasets to conduct experiments. As a result, we had to find a way to develop the dataset ourselves and use untested materials for our experiments. It affected drawing strong conclusions since no previous work had been done on the same datasets.

6.6 Future work

With the experience of doing this study, we would like to recommend a few possible research directions to improve document classification and clustering related research.

Extend the explainability of eSQ classification and clustering methods using recently introduced XAI frameworks. For example, the What-if Tool developed by Google's PAIR team is capable of visualising data points that influence a particular prediction and has a counterfactual facility. In the What-If Tool, counterfactuals are datapoints that are most similar to a selected datapoint but are classified differently by a model. This feature does not modify the classifier, but it is very useful to monitor the behaviour of neighbouring features before making a crucial decision. By integrating such tools for eSQ, we can easily leverage the power of eSQ methods for better results since they are flexible to modify.

At the moment, it is difficult to find a standard scale for assessing the explainability of a predictive model. It is a limitation for comparing different explainable strategies. Also, with the recent changes to GDPR, it is important to compare decision-making models to ensure they satisfy the requirements, and this gives a proper direction for computing professionals. So, we think that it is a timely and important measurement to be defined.

There are a couple of measures that we can use to measure the quality of clusters. We have used precision, recall, and F measures in this study. But V-measure (Rosenberg & Hirschberg, 2007) is becoming popular to measure the quality of the output of unsupervised learning. which is computed by taking the average of homogeneity and completeness. A perfect homogeneous clustering is one where each cluster has samples belonging to the same class label. A perfect complete clustering is one in which all samples from the same class are grouped together in the same cluster. So, we would like to suggest the v-measure to measure cluster quality in addition to the F measure in future research.

The eSQ-HSW clustering (hybrid method) is not human interpretable like eSQ classifiers. It happened due to the cluster expansion using a classifier. Another research direction is to develop a surrogate model for the eSQ clustering method.

LIME, SHAP, What-If Tool, DeepLIFT, AIX360, Skater, Activation Atlases, and iml - R package are recently introduced explainability tools. We were unable to find a comparative analysis of these tools. Therefore, it's a useful research direction.

6.7 Chapter Summary

This chapter provides a high-level overview of the entire research project. A list of recommendations has been given based on the experience and findings of the study in the next section. Key contributions to research have been summarised before the limitations section. Finally, a few research ideas for future research have been provided.

References

Abdullah-Al-Kafi Md., Tasnova, I. J., Wadud Islam Md., & Banshal Sumit Kumar. (2022). Performances of Different Approaches for Fake News Classification: An Analytical Study. In S. K. and P. K. K. and V. A. and V. P. Woungang Isaac and Dhurandher (Ed.), *Advanced Network Technologies and Intelligent Computing* (pp. 700–714). Springer International Publishing.

Abualigah, L., Gandomi, A. H., Elaziz, M. A., Hussien, A. G., Khasawneh, A. M., Alshinwan, M., & Houssein, E. H. (2020). Nature-inspired optimization algorithms for text document clustering—a comprehensive analysis. *Algorithms*, *13*(12), 345.

Abualigah, L. M., Khader, A. T., & Al-Betar, M. A. (2016). Unsupervised feature selection technique based on genetic algorithm for improving the text clustering. *2016 7th International Conference on Computer Science and Information Technology (CSIT)*, 1–6.

Abualigah, L. M., Khader, A. T., Al-Betar, M. A., & Alomari, O. A. (2017). Text feature selection with a robust weight scheme and dynamic dimension reduction to text document clustering. *Expert Systems with Applications*, *84*, 24–36.

Aggarwal, C. C., & Zhai, C. (2012). A survey of text classification algorithms. In *Mining text data* (pp. 163–222). Springer.

Akter, R., & Chung, Y. (2013). An Evolutionary Approach for Document Clustering. *IERI Procedia*, *4*, 370–375.

Akter, R., & Chung, Y. (2017). An improved evolutionary approach for document clustering. *Proceedings of the 2017 Research in Adaptive and Convergent Systems, RACS 2017*, *2017-January*.

Akter, R., & Chung, Y. (2021). An Improved Genetic Algorithm for Document Clustering on the Cloud. In *Research Anthology on Multi-Industry Uses of Genetic Programming and Algorithms*.

Angwin, J., Larson, J., Mattu, S., & Kirchner, L. (2016). Machine bias. In *Ethics of Data and Analytics* (pp. 254–264). Auerbach Publications.

Arthur, D., & Vassilvitskii, S. (2007). k-means : The advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 1027–1035.

Azzouz, R., Bechikh, S., & Ben Said, L. (2015). Multi-objective optimization with dynamic constraints and objectives: new challenges for evolutionary algorithms.

Proceedings of the 2015 on Genetic and Evolutionary Computation Conference, 615–622.

Barros, R. C., Basgalupp, M. P., Freitas, A., & de Carvalho, A. C. (2014). Evolutionary design of decision-tree algorithms tailored to microarray gene expression data sets. *Evolutionary Computation, IEEE Transactions On*, *18*(6), 873–892.

Bashir, S., Khattak, I. U., Khan, A., Khan, F. H., Gani, A., & Shiraz, M. (2022). A novel feature selection method for classification of medical data using filters, wrappers, and embedded approaches. *Complexity*, *2022*.

Białecki, A., Muir, R., Ingersoll, G., & Imagination, L. (2012). Apache lucene 4. *SIGIR 2012 Workshop on Open Source Information Retrieval*, 17.

Bidi, N., & Elberrichi, Z. (2017). Feature selection for text classification using genetic algorithms. *Proceedings of 2016 8th International Conference on Modelling, Identification and Control, ICMIC 2016*.

Bryman, A. (2015). *Social research methods*. Oxford university press.

Cai, J., Luo, J., Wang, S., & Yang, S. (2018). Feature selection in machine learning: A new perspective. *Neurocomputing*, *300*, 70–79.

Cambridge Analytica. (2022). In *Wikimedia*. Wikimedia Foundation, Inc.

Carole Cadwalladr, & Emma Graham-Harrison. (2018, May 18). 50 million Facebook profiles harvested for Cambridge Analytica in major data breach. *Guardian News & Media Limited*.

Chang, X. (2023). The Analysis of Open Source Search Engines. *Highlights in Science, Engineering and Technology*, *32*, 32–42.

Chauhan, A., Agarwal, A., & Sulthana, R. (2021). Genetic Algorithm and Ensemble Learning Aided Text Classification using Support Vector Machines. *International Journal of Advanced Computer Science and Applications*, *12*(8).

Cohen, W. W. (1995). Fast effective rule induction. *Proceedings of the Twelfth International Conference on Machine Learning*, 115–123.

Creswell, J. W. (2013). *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications.

Cunningham, P. (2008). Dimension reduction. *Machine Learning Techniques for Multimedia*, 91–112.

Dasgupta, S., & Long, P. M. (2005). Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, *70*(4), 555–569.

Debole, F., & Sebastiani, F. (2005). An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, *56*(6), 584–596.

Ding, Y., & Fu, X. (2016). Kernel-based fuzzy c-means clustering algorithm based on genetic algorithm. *Neurocomputing*, *188*, 233–238.

Dodig-Crnkovic, G. (2002). Scientific methods in computer science. *Proceedings of the Conference for the Promotion of Research in IT at New Universities and at University Colleges in Sweden, Skövde, Suecia*, 126–130.

Dua, D., & Graff, C. (2017). *UCI Machine Learning Repository*.

Ekanayaka, R. K. S. K., Lorensuhewa, S. A. S., & Kalyani, M. A. L. (2018). Sinhala news analysis using text mining and machine learning. *5th Ruhuna Int. Science and Technology Conference*.

El-Kassas, W. S., Salama, C. R., Rafea, A. A., & Mohamed, H. K. (2021). Automatic text summarization: A comprehensive survey. In *Expert Systems with Applications* (Vol. 165).

El-Khair, I. A. (2006). Effects of stop words elimination for Arabic information retrieval: a comparative study. *International Journal of Computing & Information Sciences*, *4*(3), 119–133.

Espejo, P. G., Ventura, S., & Herrera, F. (2010). A Survey on the Application of Genetic Programming to Classification. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions On*, *40*(2), 121–144.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, *3*(Mar), 1289–1305.

Fox, C. (1990). A stop list for general text. *ACM SIGIR Forum*, *24*(1–2).

Fragoso, R. C. P., Pinheiro, R. H. W., & Cavalcanti, G. D. C. (2016). Class-dependent feature selection algorithm for text categorization. *Neural Networks (IJCNN), 2016 International Joint Conference On*, 3508–3515.

Fung, G., Sandilya, S., & Rao, R. B. (2005). Rule extraction from linear support vector machines. *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, 32–40.

Fürnkranz, J., & Widmer, G. (1994). Incremental reduced error pruning. *Proceedings of the 11th International Conference on Machine Learning (ML-94)*, 70–77.

Gaber, A., Youness, H. A., Hamdy, A., Abdelaal, H. M., & Hassan, A. M. (2022). Automatic Classification of Fatty Liver Disease Based on Supervised Learning and Genetic Algorithm. *Applied Sciences*, *12*(1).

Gasparetto, A., Marcuzzo, M., Zangari, A., & Albarelli, A. (2022). A Survey on Text Classification Algorithms: From Text to Predictions. *Information*, *13*(2), 83.

Giuntini, R., Holik, F., Park, D. K., Freytes, H., Blank, C., & Sergioli, G. (2023). Quantum-inspired algorithm for direct multi-class classification. *Applied Soft Computing*, *134*, 109956.

Gonçalves, E. C., Plastino, A., & Freitas, A. A. (2015). Simpler is Better: a Novel Genetic Algorithm to Induce Compact Multi-label Chain Classifiers. *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*, 559–566.

Grandini, M., Bagli, E., & Visani, G. (2020). *Metrics for Multi-Class Classification: an Overview*.

Gunasekara, S. V. S., & Haddela, P. S. (2018). Context aware stopwords for Sinhala Text classification. *2018 National Information Technology Conference, NITC 2018*.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, *3*(Mar), 1157–1182.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hayes, P. J., Andersen, P. M., Nirenburg, I. B., & Schmandt, L. M. (1990). Tcs: a shell for content-based text categorization. *Artificial Intelligence Applications, 1990., Sixth Conference On*, 320–326.

Hirsch, L. (2010). Evolved Apache Lucene SpanFirst queries are good text classifiers. In *Evolutionary Computation (CEC), 2010 IEEE Congress on* (pp. 1–8).

Hirsch, L., & Brunsdon, T. (2018). A comparison of Lucene search queries evolved as text classifiers. *Applied Artificial Intelligence*, *32*(7–8), 768–784.

Hirsch, L., Hirsch, R., & Saeedi, M. (2007). Evolving Lucene search queries for text classification. In *GECCO ’07* (pp. 1604–1611).

Hirsch, L., Nuovo, A. di, & Haddela, P. (2021). *Document Clustering with Evolved Single Word Search Queries*.

Hirsch, L., Saeedi, M., & Hirsch, R. (2005). Evolving text classification rules with genetic programming. *Applied Artificial Intelligence*, *19*(7), 659–676.

Holland, J. H. (1992). Genetic algorithms. *Scientific American*, *267*(1), 66–73.

Hong, S.-S., Lee, W., & Han, M.-M. (2015). The Feature Selection Method based on Genetic Algorithm for Efficient of Text Clustering and Text Classification. *International Journal of Advances in Soft Computing & Its Applications*, *7*(1).

Hu, H., & Li, J. (2005). Using association rules to make rule-based classifiers robust. *Proceedings of the 16th Australasian Database Conference-Volume 39*, 47–54.

Ittoo, A., van den Bosch, A., & others. (2016). Text analytics in industry: Challenges, desiderata and trends. *Computers in Industry*, *78*, 96–107.

Jayaweera, A. A. V. A., Senanayake, Y. N., & Haddela, P. S. (2019). Dynamic Stopword Removal for Sinhala Language. *2019 National Information Technology Conference, NITC 2019*.

Jindal, R., Malhotra, R., & Jain, A. (2015). Techniques for text classification: Literature review and current trends. *Webology*, *12*(2), 1.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Machine Learning: ECML-98*, 137–142.

Kariyawasam, K. T. P. M., Senanayake, S. Y., & Haddela, P. S. (2019). A Rule Based Stemmer for Sinhala Language. *2019 IEEE 14th International Conference on Industrial and Information Systems: Engineering for Innovations for Industry 4.0, ICIIS 2019 - Proceedings*.

Khaleel, M. I., Hmeidi, I. I., & Najadat, H. M. (2016). An automatic text classification system based on genetic algorithm. *ACM International Conference Proceeding Series*.

Khan, A., Baharudin, B., Lee, L., & Khan, K. (2010). A Review of Machine Learning Algorithms for Text- Documents Classification. *Journal of Advances in Information Technology*, *1*(1), 4.

Koklu, M., Kahramanli, H., & Allahverdi, N. (2015). Applications of Rule Based Classification Techniques for Thoracic Surgery. *Managing Intellectual Capital and Innovation for Sustainable and Inclusive Society: Managing Intellectual Capital and Innovation; Proceedings of the MakeLearn and TIIM Joint International Conference 2015*, 1991–1998.

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150.

Kumar, K. N. Pavan., & Gavrilova, M. L. (2021). Latent Personality Traits Assessment From Social Network Activity Using Contextual Language Embedding. *IEEE Transactions on Computational Social Systems*.

Lakmali, K. B. N., & Haddela, P. S. (2018). Effectiveness of rule-based classifiers in Sinhala text categorization. *2017 National Information Technology Conference, NITC 2017*, *2017-September*.

Lan, M., Tan, C. L., Su, J., & Lu, Y. (2009). Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(4), 721–735.

Letham, B., Rudin, C., McCormick, T. H., & Madigan, D. (2015). Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, *9*(3), 1350–1371.

Levy, O., & Goldberg, Y. (2014). Dependency-based word embeddings. *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference*, *2*.

Li, H., Yu, L., & He, W. (2019). The Impact of GDPR on Global Technology Development. *Journal of Global Information Technology Management*, *22*(1), 1–6.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, *28*(2), 129–137.

Lord Clement-Jones. (2020). *Predictive and Decision-Making Algorithms in Public Policy*.

Luke, S. (2017). ECJ Then and Now. *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 1223–1230.

Magueresse, A., Carles, V., & Heetderks, E. (2020). Low-resource Languages: A Review of Past Work and Future Challenges. *CoRR*.

Mandl, T., Modha, S., Majumder, P., Patel, D., Dave, M., Mandlia, C., & Patel, A. (2019). Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. *Proceedings of the 11th Forum for Information Retrieval Evaluation*, 14–17.

Mittelstadt, B. D., Allo, P., Taddeo, M., Wachter, S., & Floridi, L. (2016). The ethics of algorithms: Mapping the debate. *Big Data & Society*, *3*(2).

Mustafi, D., Mustafi, A., & Sahoo, G. (2022). A novel approach to text clustering using genetic algorithm based on the nearest neighbour heuristic. *International Journal of Computers and Applications*, *44*(3).

Nooraeni, R., Arsa, M. I., & Projo, N. W. K. (2021). Fuzzy centroid and genetic algorithms: solutions for numeric and categorical mixed data clustering. *Procedia Computer Science*, *179*, 677–684.

Okoko, J. M., Tunison, S., & Walker, K. D. (2023). *Varieties of qualitative research methods: Selected contextual perspectives*. Springer Nature.

Oswald, M., & Grace, J. (2016). Intelligence, policing and the use of algorithmic analysis: a freedom of information-based study. *Information Rights, Policy & Practice*, *1*(1).

Oswald, M., Grace, J., Urwin, S., & Barnes, G. C. (2018). Algorithmic risk assessment policing models: Lessons from the Durham HART model and 'experimental' proportionality. *Information and Communications Technology Law*, *27*(2), 223–250.

Parlak, B., & Uysal, A. K. (2023). A novel filter feature selection method for text classification: Extensive Feature Selector. *Journal of Information Science*, *49*(1), 59–78.

Patel, P. H. (2014). Pre-Processing Phase of Text Summarization Based on Gujarati Language Gender and Number Identification: Rule-Based Approach View project. In *International Journal of Innovative Research in Computer Science & Technology (IJIRCST)* (Issue 2).

Phoungphol, P., Zhang, Y., & Zhao, Y. (2012). Robust multiclass classification for learning from imbalanced biomedical data. *Tsinghua Science and Technology*, *17*(6), 619–628.

Pietramala, A., Policicchio, V. L., Rullo, P., & Sidhu, I. (2008). A genetic algorithm for text classification rule induction. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 188–203.

Pizzuti, C., & Procopio, N. (2016). A k-means based genetic algorithm for data clustering. *International Joint Conference SOCO'16-CISIS'16-ICEUTE'16*, 211–222.

Polychronopoulos, V., Pendar, N., & Jeffery, S. R. (2014). QuIET: A text classification technique using automatically generated span queries. *Semantic Computing (ICSC), 2014 IEEE International Conference On*, 52–59.

Powers, D. M. (2011). *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation*.

Pugliese, R., Regondi, S., & Marini, R. (2021). Machine learning-based approach: global trends, research directions, and regulatory standpoints. *Data Science and Management*, *4*, 19–29.

Puri, R., Bedi, R. P. S., Goyal, V., & Supervisor, R. (2013). Automated Stopwords Identification in Punjabi Documents. *Research Cell: An International Journal of Engineering Sciences*, *8*.

Qian, L., & Wang, L. (2010). An evaluation of Lucene for keywords search in large-scale short text storage. *Computer Design and Applications (ICCDA), 2010 International Conference On*, *2*, V2-209.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

Raschip, M., Croitoru, C., & Stoffel, K. (2015). Guiding Evolutionary Search with Association Rules for Solving Weighted CSPs. *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*, 481–488.

Raulji, J. K., & Saini, J. R. (2017). Generating Stopword List for Sanskrit Language. *7th International Advance Computing Conference (IACC)*.

Rosenberg, A., & Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 410–420.

Rullo, P., Cumbo, C., & Policicchio, V. L. (2007). Learning rules with negation for text categorization. *Proceedings of the 2007 ACM Symposium on Applied Computing*, 409–416.

Rullo, P., Policicchio, V. L., Cumbo, C., & Iiritano, S. (2009). Olex: effective rule learning for text categorization. *IEEE Transactions on Knowledge and Data Engineering*, *21*(8), 1118–1132.

Saini, J. R., Raulji, J. K., & Ambedkar, B. (2016). Stop-Word Removal Algorithm and its Implementation for Sanskrit Language. *Article in Int. Journal of Computer Applications*, *150*(2), 975–8887. https://doi.org/10.5120/ijca2016911462

Sasaki, M., & Kita, K. (1998). Rule-based text categorization using hierarchical categories. *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference On*, *3*, 2827–2830.

Science and Technology Committee. (2016). *Robotics and artificial intelligence*.

Senanayake, S. Y., Kariyawasam, K. T. P. M., & Haddela, P. S. (2019). Enhanced Tokenizer for Sinhala Language. *2019 National Information Technology Conference, NITC 2019*.

Shang, W., Huang, H., Zhu, H., Lin, Y., Qu, Y., & Wang, Z. (2007). A novel feature selection algorithm for text categorization. *Expert Systems with Applications, 33*(1), 1–5.

Sharma, S., & Jain, A. (2023). Hybrid ensemble learning with feature selection for sentiment classification in social media. In *Research Anthology on Applying Social Networking Strategies to Classrooms and Libraries* (pp. 1183–1203). IGI Global.

Sheikh, R. H., Raghuwanshi, M. M., & Jaiswal, A. N. (2008). Genetic algorithm based clustering: a survey. *2008 First International Conference on Emerging Trends in Engineering and Technology*, 314–319.

Shi, H., & Xu, M. (2018). A data classification method using genetic algorithm and K-means algorithm with optimizing initial cluster center. *2018 IEEE International Conference on Computer and Communication Engineering Technology (CCET)*, 224–228.

Sohail, A. (2023). Genetic Algorithms in the Fields of Artificial Intelligence and Data Sciences. *Annals of Data Science*, *10*(4), 1007–1018.

Tahir, M. A., Kittler, J., & Yan, F. (2012). Inverse random under sampling for class imbalance problem and its application to multi-label classification. *Pattern Recognition*, *45*(10), 3738–3750.

Tang, B., Kay, S., & He, H. (2016). *Toward optimal feature selection in naive Bayes for text categorization*.

Tsai, C.-F., Chen, Z.-Y., & Ke, S.-W. (2014). Evolutionary instance selection for text classification. *Journal of Systems and Software*, *90*, 104–113.

Turney, P. D., & Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, *37*(1), 141–188.

Uğuz, H. (2011). A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. *Knowledge-Based Systems*, *24*(7), 1024–1032.

Uysal, A. K. (2016). An improved global feature selection scheme for text classification. *Expert Systems with Applications*, *43*, 82–92.

Vasile, F., Silvescu, A., Kang, D.-K., & Honavar, V. (2006). TRIPPER: Rule learning using taxonomies. *Advances in Knowledge Discovery and Data Mining*, 55–59.

Velliangiri, S., Alagumuthukrishnan, S., & others. (2019). A review of dimensionality reduction techniques for efficient computation. *Procedia Computer Science*, *165*, 104–111.

Villena Román, J., Collada Pérez, S., Lana Serrano, S., & González Cristóbal, J. C. (2011). *Hybrid approach combining machine learning and a rule-based expert system for text categorization*.

Wei, J. X., Liu, H., Sun, Y. H., & Su, X. N. (2009). Application of genetic algorithm in document clustering. *Proceedings - 2009 International Conference on Information Technology and Computer Science, ITCS 2009*, *1*.

Xu, D., & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, *2*(2), 165–193.

Yu, B. (2008). An evaluation of text classification methods for literary study. *Literary and Linguistic Computing*, *23*(3), 327–343.

Zeebaree, D. Q., Haron, H., Abdulazeez, A. M., & Zeebaree, S. R. (2017). Combination of K-means clustering with Genetic Algorithm: A review. *International Journal of Applied Engineering Research*, *12*(24), 14238–14245.

Zhang, Z., Jasaitis, T., Freeman, R., Alfrjani, R., & Funk, A. (2023). Mining Healthcare Procurement Data Using Text Mining and Natural Language Processing–Reflection From An Industrial Project. *ArXiv Preprint ArXiv:2301.03458*.

Appendices

Appendix A – Experimental Results for FS Metrics

Appendix A.1

System: *eSQ Classifiers*

Dataset: *R10 Dataset*

Feature set size: *Top 300 terms*

Measurement: *F measure*

Category	*F1*	*CHI*	*OR*	*IG*
acq	0.782	0.781	0.774	0.782
corn	0.562	0.562	0.562	0.562
crude	0.764	0.762	0.762	0.759
earn	0.936	0.935	0.936	0.935
grain	0.642	0.642	0.617	0.642
interest	0.509	0.509	0.481	0.509
money-fx	0.658	0.665	0.561	0.665
ship	0.655	0.659	0.657	0.655
trade	0.602	0.639	0.604	0.603
wheat	0.556	0.556	0.56	0.556
Avg F_1	0.7672	0.7714	0.7591	0.7731

Appendix A.2

System: *eSQ Classifiers*

Dataset: NG20 *Dataset*

Feature set size: *Top 300 terms*

Measurement: *F measure*

Category	*F1*	*CHI*	*IG*	*OR*
alt.atheism	0.76	0.763	0.763	0.763
comp.graphics	0.544	0.55	0.55	0.415
comp.os.ms-windows.misc	0.629	0.639	0.643	0.395
comp.sys.ibm.pc.hardware	0.526	0.539	0.539	0.537
comp.sys.mac.hardware	0.687	0.664	0.664	0.618
comp.windows.x	0.736	0.736	0.736	0.736

misc.forsale	0.716	0.726	0.724	0.731
rec.autos	0.66	0.668	0.669	0.51
rec.motorcycles	0.837	0.845	0.845	0.845
rec.sport.baseball	0.723	0.722	0.722	0.655
rec.sport.hockey	0.869	0.817	0.816	0.816
sci.crypt	0.899	0.893	0.893	0.893
sci.electronics	0.472	0.503	0.503	0.464
sci.med	0.71	0.698	0.698	0.655
sci.space	0.765	0.763	0.763	0.747
soc.religion.christian	0.726	0.713	0.716	0.674
talk.politics.guns	0.766	0.766	0.766	0.766
talk.politics.mideast	0.851	0.847	0.847	0.846
talk.politics.misc	0.564	0.606	0.606	0.608
talk.religion.misc	0.554	0.556	0.553	0.492
Avg F_1	0.6377	0.6346	0.6355	0.6023

Appendix A.3

System: *eSQ Classifiers*

Dataset: *R10 Dataset*

Feature set size: *Top 100 terms*

Measurement: *F measure*

Category	*F1*	*CHI*	*OR*	*IG*
acq	0.782	0.781	0.774	0.782
corn	0.562	0.562	0.562	0.562
crude	0.764	0.762	0.755	0.759
earn	0.936	0.935	0.936	0.935
grain	0.642	0.642	0.617	0.642
interest	0.506	0.509	0.481	*0.526*
money-fx	0.658	0.665	0.561	0.665
ship	0.652	0.659	0.657	0.659
trade	0.602	*0.639*	0.604	0.603
wheat	0.556	0.556	0.56	0.556
Avg F_1	0.7661	0.7714	0.7625	0.7704

Appendix A.4

System: *eSQ Classifiers* Dataset: *NG20 Dataset* Feature set size: *Top 100 terms* Measurement: *F measure*				
Category	*F1*	*CHI*	*OR*	*IG*
alt.atheism	0.76	0.763	0.763	0.763
comp.graphics	0.544	0.55	0.415	0.55
comp.os.ms-windows.misc	0.629	0.639	0.377	0.643
comp.sys.ibm.pc.hardware	0.526	0.539	0.537	0.539
comp.sys.mac.hardware	0.687	0.664	0.618	0.664
comp.windows.x	0.736	0.736	0.736	0.736
misc.forsale	0.712	0.726	0.731	0.724
rec.autos	0.66	0.668	0.51	0.669
rec.motorcycles	0.837	0.845	0.845	0.845
rec.sport.baseball	0.723	0.722	0.655	0.722
rec.sport.hockey	0.869	0.817	0.816	0.816
sci.crypt	0.899	0.893	0.893	0.893
sci.electronics	0.472	0.503	0.464	0.503
sci.med	0.71	0.698	0.655	0.698
sci.space	0.765	0.763	0.747	0.763
soc.religion.christian	0.726	0.716	0.674	0.716
talk.politics.guns	0.766	0.766	0.766	0.766
talk.politics.mideast	0.857	0.847	0.846	0.847
talk.politics.misc	0.564	0.606	0.608	0.606
talk.religion.misc	0.554	0.553	0.492	0.556
Avg F_1	0.6346	0.6346	0.6029	0.6396

Appendix A.5

System: *eSQ Classifiers* Dataset: *R10 Dataset* Feature set size: *Top 50 terms* Measurement: *F measure*				
Category	*F1*	*CHI*	*OR*	*IG*
acq	0.815	0.782	0.633	0.782
corn	0.544	0.562	0.535	0.562
crude	0.708	0.756	0.748	0.756
earn	0.936	0.935	0.928	0.929
grain	0.661	0.642	0.607	0.641
interest	0.547	0.509	0.286	0.506
money-fx	0.641	0.664	0.48	0.663
ship	0.624	0.656	0.632	0.656
trade	0.586	0.602	0.602	0.602
wheat	0.549	0.556	0.56	0.556
Avg F_1	0.7726	0.7716	0.7156	0.7723

Appendix A.6

System: *eSQ Classifiers* Dataset: *NG20 Dataset* Feature set size: *Top 50 terms* Measurement: *F measure*				
Category	*F1*	*CHI*	*OR*	*IG*
alt.atheism	0.746	0.763	0.755	0.763
comp.graphics	0.507	0.55	0.393	0.55
comp.os.ms-windows.misc	0.599	0.639	0.377	0.637
comp.sys.ibm.pc.hardware	0.515	0.539	0.48	0.539
comp.sys.mac.hardware	0.687	0.664	0.559	0.664
comp.windows.x	0.731	0.736	0.731	0.736
misc.forsale	0.712	0.726	0.731	0.718
rec.autos	0.647	0.668	0.462	0.668

rec.motorcycles	0.83	0.845	0.845	0.845
rec.sport.baseball	0.723	0.722	0.652	0.722
rec.sport.hockey	0.864	0.817	0.817	0.814
sci.crypt	0.899	0.893	0.893	0.891
sci.electronics	0.465	0.503	0.388	0.499
sci.med	0.69	0.698	0.644	0.69
sci.space	0.75	0.763	0.727	0.763
soc.religion.christian	0.713	0.716	0.663	0.712
talk.politics.guns	0.759	0.766	0.669	0.764
talk.politics.mideast	0.85	0.847	0.845	0.845
talk.politics.misc	0.501	0.606	0.605	0.603
talk.religion.misc	0.489	0.553	0.492	0.552
Avg F$_1$	0.6336	0.6346	0.5927	0.6338

Appendix A.7

System: *eSQ Classifiers*

Dataset: *R10 Dataset*

Feature set size: *Top 20 terms*

Measurement: *F measure*

Category	*F1*	*CHI*	*OR*	*IG*
acq	0.768	0.757	0.571	0.751
corn	0.562	0.562	0.527	0.562
crude	0.764	0.753	0.695	0.753
earn	0.926	0.916	0.91	0.916
grain	0.641	0.638	0.126	0.626
interest	0.506	0.503	0.251	0.496
money-fx	0.657	0.642	0.3	0.642
ship	0.656	0.652	0.51	0.645
trade	0.602	0.602	0.602	0.602
wheat	0.556	0.556	0.558	0.556
Avg F$_1$	0.7726	0.7703	0.6683	0.7638

Appendix A.8

System: *eSQ Classifiers* Dataset: *NG20 Dataset* Feature set size: *Top 20 terms* Measurement: *F measure*				
Category	*F1*	*CHI*	*OR*	*IG*
alt.atheism	0.652	0.76	0.711	0.724
comp.graphics	0.462	0.55	0.391	0.541
comp.os.ms-windows.misc	0.59	0.639	0.371	0.629
comp.sys.ibm.pc.hardware	0.493	0.539	0.3	0.51
comp.sys.mac.hardware	0.683	0.664	0.529	0.661
comp.windows.x	0.722	0.736	0.664	0.736
misc.forsale	0.705	0.726	0.729	0.712
rec.autos	0.647	0.669	0.433	0.665
rec.motorcycles	0.826	0.845	0.792	0.83
rec.sport.baseball	0.68	0.722	0.651	0.722
rec.sport.hockey	0.826	0.814	0.802	0.799
sci.crypt	0.874	0.891	0.893	0.891
sci.electronics	0.42	0.503	0.318	0.488
sci.med	0.574	0.696	0.545	0.667
sci.space	0.749	0.763	0.696	0.747
soc.religion.christian	0.713	0.713	0.659	0.687
talk.politics.guns	0.747	0.766	0.627	0.756
talk.politics.mideast	0.847	0.845	0.832	0.835
talk.politics.misc	0.458	0.606	0.597	0.541
talk.religion.misc	0.43	0.556	0.46	0.515
Avg F_1	0.6098	0.6291	0.5498	0.6234

Appendix B - eSQ-MW clustering method dataset for 11 Jobs

Data received from eSQ-MW Clustering Method – Sinhala Language

Job No	Dataset	F measure	Precision	Recall
0	SLNG3	0.6735	0.7287	0.6326
1	SLNG3	0.6848	0.7320	0.6463
2	SLNG3	0.6848	0.7320	0.6463
3	SLNG3	0.6845	0.7314	0.6463
4	SLNG3	0.6735	0.7287	0.6326
5	SLNG3	0.6735	0.7361	0.6263
6	SLNG3	0.6848	0.7320	0.6463
7	SLNG3	0.6734	0.7282	0.6329
8	SLNG3	0.6731	0.7301	0.6302
9	SLNG3	0.6848	0.7320	0.6463
10	SLNG3	0.6848	0.7320	0.6463
0	SLNG4-2	0.4973	0.5453	0.4944
1	SLNG4-2	0.5237	0.5603	0.5106
2	SLNG4-2	0.5251	0.5578	0.5181
3	SLNG4-2	0.5301	0.5989	0.4969
4	SLNG4-2	0.5242	0.5691	0.5000
5	SLNG4-2	0.5295	0.5973	0.4975
6	SLNG4-2	0.5237	0.5603	0.5106
7	SLNG4-2	0.5237	0.5603	0.5106
8	SLNG4-2	0.5295	0.5973	0.4975
9	SLNG4-2	0.5179	0.5497	0.5125
10	SLNG4-2	0.5615	0.6092	0.5363
0	SLNG5-2	0.5695	0.6420	0.5750
1	SLNG5-2	0.6090	0.6382	0.6140
2	SLNG5-2	0.5368	0.6282	0.5480
3	SLNG5-2	0.6423	0.6795	0.6290
4	SLNG5-2	0.5368	0.6282	0.5480
5	SLNG5-2	0.5098	0.5942	0.5200
6	SLNG5-2	0.5368	0.6282	0.5480
7	SLNG5-2	0.5098	0.5942	0.5200
8	SLNG5-2	0.6185	0.6476	0.6220
9	SLNG5-2	0.5098	0.5942	0.5200
10	SLNG5-2	0.5132	0.5961	0.5280
0	SLNG6	0.6936	0.8315	0.7026
1	SLNG6	0.6935	0.8315	0.7024
2	SLNG6	0.6936	0.8315	0.7026
3	SLNG6	0.6935	0.8315	0.7024
4	SLNG6	0.6935	0.8315	0.7024

5	SLNG6	0.6936	0.8315	0.7026
6	SLNG6	0.6936	0.8315	0.7026
7	SLNG6	0.6936	0.8315	0.7026
8	SLNG6	0.6935	0.8315	0.7024
9	SLNG6	0.6935	0.8315	0.7024
10	SLNG6	0.6935	0.8315	0.7024
0	SLNG7	0.6746	0.7967	0.7084
1	SLNG7	0.6746	0.7967	0.7084
2	SLNG7	0.6746	0.7967	0.7084
3	SLNG7	0.6746	0.7967	0.7084
4	SLNG7	0.6746	0.7967	0.7084
5	SLNG7	0.5853	0.6981	0.6227
6	SLNG7	0.6747	0.7968	0.7084
7	SLNG7	0.6746	0.7967	0.7084
8	SLNG7	0.6746	0.7967	0.7084
9	SLNG7	0.6746	0.7968	0.7082
10	SLNG7	0.6747	0.7968	0.7084

Appendix C - eSQ-HSW clustering method dataset for 11 Jobs

Data received from eSQ-HSW Clustering Method – Sinhala Language

Dataset	Job No	ClassifyMethod	F1 Measure	Precision	Recall
SLNG3	0	KNN	0.8001	0.7929	0.8074
SLNG3	0	NB	0.8050	0.7871	0.8239
SLNG3	0	KNF	0.7917	0.7826	0.8010
SLNG3	1	KNN	0.7987	0.7910	0.8066
SLNG3	1	NB	0.7978	0.7773	0.8195
SLNG3	1	KNF	0.7878	0.7771	0.7988
SLNG3	2	KNN	0.8001	0.7929	0.8074
SLNG3	2	NB	0.8050	0.7871	0.8239
SLNG3	2	KNF	0.7917	0.7826	0.8010
SLNG3	3	KNN	0.7987	0.7910	0.8066
SLNG3	3	NB	0.7978	0.7773	0.8195
SLNG3	3	KNF	0.7878	0.7771	0.7988
SLNG3	4	KNN	0.8001	0.7929	0.8074
SLNG3	4	NB	0.8050	0.7871	0.8239
SLNG3	4	KNF	0.7917	0.7826	0.8010
SLNG3	5	KNN	0.8001	0.7929	0.8074
SLNG3	5	NB	0.8050	0.7871	0.8239
SLNG3	5	KNF	0.7917	0.7826	0.8010
SLNG3	6	KNN	0.8001	0.7929	0.8074
SLNG3	6	NB	0.8050	0.7871	0.8239
SLNG3	6	KNF	0.7917	0.7826	0.8010
SLNG3	7	KNN	0.8001	0.7929	0.8074
SLNG3	7	NB	0.8050	0.7871	0.8239
SLNG3	7	KNF	0.7917	0.7826	0.8010
SLNG3	8	KNN	0.7987	0.7910	0.8066
SLNG3	8	NB	0.7978	0.7773	0.8195
SLNG3	8	KNF	0.7878	0.7771	0.7988
SLNG3	9	KNN	0.7987	0.7910	0.8066
SLNG3	9	NB	0.7978	0.7773	0.8195
SLNG3	9	KNF	0.7878	0.7771	0.7988
SLNG3	10	KNN	0.7987	0.7910	0.8066
SLNG3	10	NB	0.7978	0.7773	0.8195
SLNG3	10	KNF	0.7878	0.7771	0.7988
SLNG4-2	0	KNN	0.5344	0.6000	0.4818
SLNG4-2	0	NB	0.5512	0.6188	0.4969
SLNG4-2	0	KNF	0.5300	0.5925	0.4795
SLNG4-2	1	KNN	0.5344	0.6000	0.4818
SLNG4-2	1	NB	0.5512	0.6188	0.4969
SLNG4-2	1	KNF	0.5300	0.5925	0.4795

SLNG4-2	2	KNN	0.5344	0.6000	0.4818
SLNG4-2	2	NB	0.5512	0.6188	0.4969
SLNG4-2	2	KNF	0.5300	0.5925	0.4795
SLNG4-2	3	KNN	0.5344	0.6000	0.4818
SLNG4-2	3	NB	0.5512	0.6188	0.4969
SLNG4-2	3	KNF	0.5300	0.5925	0.4795
SLNG4-2	4	KNN	0.5344	0.6000	0.4818
SLNG4-2	4	NB	0.5512	0.6188	0.4969
SLNG4-2	4	KNF	0.5300	0.5925	0.4795
SLNG4-2	5	KNN	0.6256	0.6225	0.6287
SLNG4-2	5	NB	0.6225	0.6250	0.6200
SLNG4-2	5	KNF	0.6134	0.6100	0.6169
SLNG4-2	6	KNN	0.5344	0.6000	0.4818
SLNG4-2	6	NB	0.5512	0.6188	0.4969
SLNG4-2	6	KNF	0.5300	0.5925	0.4795
SLNG4-2	7	KNN	0.5344	0.6000	0.4818
SLNG4-2	7	NB	0.5512	0.6188	0.4969
SLNG4-2	7	KNF	0.5300	0.5925	0.4795
SLNG4-2	8	KNN	0.5344	0.6000	0.4818
SLNG4-2	8	NB	0.5512	0.6188	0.4969
SLNG4-2	8	KNF	0.5300	0.5925	0.4795
SLNG4-2	9	KNN	0.5344	0.6000	0.4818
SLNG4-2	9	NB	0.5512	0.6188	0.4969
SLNG4-2	9	KNF	0.5300	0.5925	0.4795
SLNG4-2	10	KNN	0.5344	0.6000	0.4818
SLNG4-2	10	NB	0.5512	0.6188	0.4969
SLNG4-2	10	KNF	0.5300	0.5925	0.4795
SLNG5-2	0	KNN	0.5506	0.6350	0.4860
SLNG5-2	0	NB	0.5200	0.6058	0.4555
SLNG5-2	0	KNF	0.5448	0.6225	0.4844
SLNG5-2	1	KNN	0.5475	0.6383	0.4793
SLNG5-2	1	NB	0.5453	0.6267	0.4827
SLNG5-2	1	KNF	0.5546	0.6342	0.4927
SLNG5-2	2	KNN	0.5475	0.6383	0.4793
SLNG5-2	2	NB	0.5453	0.6267	0.4827
SLNG5-2	2	KNF	0.5546	0.6342	0.4927
SLNG5-2	3	KNN	0.5506	0.6350	0.4860
SLNG5-2	3	NB	0.5200	0.6058	0.4555
SLNG5-2	3	KNF	0.5448	0.6225	0.4844
SLNG5-2	4	KNN	0.5506	0.6350	0.4860
SLNG5-2	4	NB	0.5200	0.6058	0.4555
SLNG5-2	4	KNF	0.5448	0.6225	0.4844
SLNG5-2	5	KNN	0.5475	0.6383	0.4793
SLNG5-2	5	NB	0.5453	0.6267	0.4827
SLNG5-2	5	KNF	0.5546	0.6342	0.4927

SLNG5-2	6	KNN	0.5506	0.6350	0.4860
SLNG5-2	6	NB	0.5200	0.6058	0.4555
SLNG5-2	6	KNF	0.5448	0.6225	0.4844
SLNG5-2	7	KNN	0.5506	0.6350	0.4860
SLNG5-2	7	NB	0.5200	0.6058	0.4555
SLNG5-2	7	KNF	0.5448	0.6225	0.4844
SLNG5-2	8	KNN	0.5475	0.6383	0.4793
SLNG5-2	8	NB	0.5453	0.6267	0.4827
SLNG5-2	8	KNF	0.5546	0.6342	0.4927
SLNG5-2	9	KNN	0.5475	0.6383	0.4793
SLNG5-2	9	NB	0.5453	0.6267	0.4827
SLNG5-2	9	KNF	0.5546	0.6342	0.4927
SLNG5-2	10	KNN	0.5506	0.6350	0.4860
SLNG5-2	10	NB	0.5200	0.6058	0.4555
SLNG5-2	10	KNF	0.5448	0.6225	0.4844
SLNG6	0	KNN	0.6244	0.6418	0.6080
SLNG6	0	NB	0.6199	0.6378	0.6029
SLNG6	0	KNF	0.6249	0.6421	0.6086
SLNG6	1	KNN	0.6244	0.6418	0.6080
SLNG6	1	NB	0.6199	0.6378	0.6029
SLNG6	1	KNF	0.6249	0.6421	0.6086
SLNG6	2	KNN	0.6244	0.6418	0.6080
SLNG6	2	NB	0.6199	0.6378	0.6029
SLNG6	2	KNF	0.6249	0.6421	0.6086
SLNG6	3	KNN	0.6225	0.6405	0.6055
SLNG6	3	NB	0.6170	0.6356	0.5994
SLNG6	3	KNF	0.6239	0.6414	0.6073
SLNG6	4	KNN	0.6244	0.6418	0.6080
SLNG6	4	NB	0.6199	0.6378	0.6029
SLNG6	4	KNF	0.6249	0.6421	0.6086
SLNG6	5	KNN	0.6244	0.6418	0.6080
SLNG6	5	NB	0.6199	0.6378	0.6029
SLNG6	5	KNF	0.6249	0.6421	0.6086
SLNG6	6	KNN	0.6225	0.6405	0.6055
SLNG6	6	NB	0.6170	0.6356	0.5994
SLNG6	6	KNF	0.6239	0.6414	0.6073
SLNG6	7	KNN	0.6225	0.6405	0.6055
SLNG6	7	NB	0.6170	0.6356	0.5994
SLNG6	7	KNF	0.6239	0.6414	0.6073
SLNG6	8	KNN	0.6244	0.6418	0.6080
SLNG6	8	NB	0.6199	0.6378	0.6029
SLNG6	8	KNF	0.6249	0.6421	0.6086
SLNG6	9	KNN	0.6225	0.6405	0.6055
SLNG6	9	NB	0.6170	0.6356	0.5994
SLNG6	9	KNF	0.6239	0.6414	0.6073

SLNG6	10	KNN	0.6244	0.6418	0.6080
SLNG6	10	NB	0.6199	0.6378	0.6029
SLNG6	10	KNF	0.6249	0.6421	0.6086
SLNG7	0	KNN	0.6442	0.6386	0.6498
SLNG7	0	NB	0.5953	0.6363	0.5592
SLNG7	0	KNF	0.6439	0.6364	0.6516
SLNG7	1	KNN	0.6442	0.6386	0.6498
SLNG7	1	NB	0.5953	0.6363	0.5592
SLNG7	1	KNF	0.6439	0.6364	0.6516
SLNG7	2	KNN	0.6243	0.6755	0.5803
SLNG7	2	NB	0.4352	0.3419	0.5986
SLNG7	2	KNF	0.6187	0.6650	0.5785
SLNG7	3	KNN	0.6257	0.6767	0.5819
SLNG7	3	NB	0.4363	0.3433	0.5984
SLNG7	3	KNF	0.6197	0.6656	0.5797
SLNG7	4	KNN	0.6442	0.6386	0.6498
SLNG7	4	NB	0.5953	0.6363	0.5592
SLNG7	4	KNF	0.6439	0.6364	0.6516
SLNG7	5	KNN	0.6450	0.6390	0.6512
SLNG7	5	NB	0.5953	0.6359	0.5596
SLNG7	5	KNF	0.6442	0.6363	0.6521
SLNG7	6	KNN	0.6442	0.6386	0.6498
SLNG7	6	NB	0.5953	0.6363	0.5592
SLNG7	6	KNF	0.6439	0.6364	0.6516
SLNG7	7	KNN	0.6442	0.6386	0.6498
SLNG7	7	NB	0.5953	0.6363	0.5592
SLNG7	7	KNF	0.6439	0.6364	0.6516
SLNG7	8	KNN	0.6243	0.6755	0.5803
SLNG7	8	NB	0.4352	0.3419	0.5986
SLNG7	8	KNF	0.6187	0.6650	0.5785
SLNG7	9	KNN	0.6450	0.6390	0.6512
SLNG7	9	NB	0.5953	0.6359	0.5596
SLNG7	9	KNF	0.6442	0.6363	0.6521
SLNG7	10	KNN	0.6450	0.6390	0.6512
SLNG7	10	NB	0.5953	0.6359	0.5596
SLNG7	10	KNF	0.6442	0.6363	0.6521

Appendix D - eSQ-MW clustering method: SLNG3

Evolved Search Queries from eSQ-MW for every 11 jobs			
Dataset	**SLNG3**	Job No	**1**
Category	Query		
Football	මහතා සම්බන්ධයෙන් මම කටයුතු පිළිබඳව සම්මේලනයේ අන්තර්ජාතික		
Cricket	දැවී නොදැවී පන්දුවක්		
Rugby	උත්සාහක මිනිත්තු දඬුවම් හැව්ලොක්ස් දඬුවම්		
Dataset	**SLNG3**	Job No	**2**
Category	Query		
Football	මහතා සභාපති කටයුතු මම සම්මේලනයේ පිළිබඳව		
Cricket	කඩුලු දැවී පන්දුවක්		
Rugby	උත්සාහක මිනිත්තු දඬුවම් දඬුවම් හැව්ලොක්ස්		
Dataset	**SLNG3**	Job No	**3**
Category	Query		
Football	මහතා සම්මේලනයේ කටයුතු සභාපති මම පිළිබඳව		
Cricket	දැවී පන්දුවක් කඩුලු		
Rugby	උත්සාහක මිනිත්තු දඬුවම් හැව්ලොක්ස් දඬුවම්		
Dataset	**SLNG3**	Job No	**4**
Category	Query		
Football	මහතා පිළිබඳව මම සම්මේලනයේ කමිටුවේ කටයුතු සභාපති		
Cricket	කඩුලු දැවී පන්දුවක්		
Rugby	උත්සාහක දඬුවම් දඬුවම් හැව්ලොක්ස් මිනිත්තු		
Dataset	**SLNG3**	Job No	**5**
Category	Query		
Football	මහතා කටයුතු සම්බන්ධයෙන් මම අන්තර්ජාතික සම්මේලනයේ පිළිබඳව		
Cricket	දැවී නොදැවී පන්දුවක්		
Rugby	උත්සාහක දඬුවම් මිනිත්තු හැව්ලොක්ස් දඬුවම්		
Dataset	**SLNG3**	Job No	**6**
Category	Query		

Football	මහතා සම්මේලනයේ මම කටයුතු සම්බන්ධයෙන් සභාපති පිළිබඳව		
Cricket	නොදැවී දැවී පන්දුවක්		
Rugby	උත්සාහක හැව්ලොක්ස් මිනිත්තු දඬුවම්		
Dataset	**SLNG3**	Job No	**7**
Category	Query		
Football	මහතා පිළිබඳව කටයුතු මම සම්මේලනයේ සභාපති		
Cricket	කඩුලු පන්දුවක් දැවී		
Rugby	උත්සාහක දඬුවම් දඬුවම් මිනිත්තු හැව්ලොක්ස්		
Dataset	**SLNG3**	Job No	**8**
Category	Query		
Football	මහතා සම්බන්ධයෙන් සම්මේලනයේ මම අන්තර්ජාතික පිළිබඳව කටයුතු		
Cricket	නොදැවී දැවී පන්දුවක්		
Rugby	උත්සාහක දඬුවම් හැව්ලොක්ස් දිනුමක් දඬුවම් මිනිත්තු		
Dataset	**SLNG3**	Job No	**9**
Category	Query		
Football	මහතා පිළිබඳව කටයුතු සම්බන්ධයෙන් මම සභාපති සම්මේලනයේ		
Cricket	දැවී නොදැවී පන්දුවක්		
Rugby	උත්සාහක හැව්ලොක්ස් දඬුවම් මිනිත්තු දඬුවම්		
Dataset	**SLNG3**	Job No	**10**
Category	Query		
Football	මහතා සභාපති කටයුතු පිළිබඳව සම්මේලනයේ මම		
Cricket	දැවී පන්දුවක් කඩුලු		
Rugby	උත්සාහක දඬුවම් හැව්ලොක්ස් මිනිත්තු දඬුවම්		
Dataset	**SLNG3**	Job No	**11**
Category	Query		
Football	මහතා මම පිළිබඳව සභාපති සම්මේලනයේ කටයුතු		
Cricket	කඩුලු දැවී පන්දුවක්		
Rugby	උත්සාහක මිනිත්තු දඬුවම් හැව්ලොක්ස් දඬුවම්		