University of Eastern Finland
Philosophical Faculty
School of Humanities
MDP in Linguistic Data Sciences
Language Technology in Linguistics and Translation Studies

Aryan Yekrangi

Leveraging simple features and machine learning approaches for assessing the CEFR level of English texts

under the supervision of
Professor Kimmo Kettunen

MA Thesis
April 2022

ITÄ-SUOMEN YLIOPISTO – UNIVERSITY OF EASTERN FINLAND

Tiedekunta – Faculty Philosophical Faculty	**Osasto – School** School of Humanities

Tekijät – Author
Aryan Yekrangi

Työn nimi – Title
Leveraging simple features and machine learning approaches for assessing the CEFR level of English texts

Pääaine – Main subject	**Työn laji – Level**		**Päivämäärä – Date**	**Sivumäärä – Number of pages**
MDP in Linguistic Data Sciences	Pro gradu -tutkielma Sivuainetutkielma Kandidaatin tutkielma Aineopintojen tutkielma	**x** 	12.04.2022	87 pages + References (7) + Appendices (3)

Tiivistelmä – Abstract

This study set out to determine the ability of simple textual features for assessing the CEFR level (difficulty level) of English texts. Simple features were defined as easily implementable features that require little time to compute. A feature extraction tool was created to calculate 19 simple features for 331 texts from the Cambridge English Readability Dataset. Using feature selection algorithms, six meaningful feature sets were developed and the hyperparameters of five machine learning models were adjusted.

Finally two experiments were conducted. Experiment 1 tested the performance of 30 classifiers, the combination of five machine learning approaches and six feature sets, using 1000 folds of training and testing. This narrowed the number of classifiers analyzed in Experiment 2 from 30 to nine. Experiment 2 used k-fold cross validation to test the performance of the nine chosen classifiers. Additionally, we analyzed the extent to which the classifiers mislabel texts, and identified problems of the classifiers overfitting the training data,

Firstly, the performance of simple features as a whole was evaluated. In our experiments, simple features showed significant predictive ability over the CEFR levels of English texts. Our classifiers showed a mean accuracy of 57.6 and the best classifier reached an accuracy of 73.6. This performance persisted in Experiment 2 when conducting k-fold cross validation, even when using precision and recall as measures of evaluation. However, our classifiers performed relatively poorly on C-level texts. We hypothesize that this may be an inherent limitation of simple features.

Secondly, the performance of different feature combinations was explored. Out of the six feature sets developed for this study, feature set 2 (with 13 features) and feature set 3 (with three features) performed well with a mean accuracy of 68.6 and 63.3, respectively. These two feature sets also showed no signs of overfitting the training data. Other feature sets either contained significantly more features, showed relatively low performance, or showed signs of overfitting the training data. Our experiments show that while it is possible to significantly reduce the number of features in the feature set and achieve satisfactory results, removing the total text length feature results in a significant loss of performance.

Finally, the performance of different machine learning approaches was tested. Out of the five classification approaches explored, the support vector machine with the RBF kernel showed the highest performance. Additionally, while linear regression classifiers showed low accuracy scores, they exceeded other classifiers in approximate prediction, which was measured through one-off accuracy. The linear regression model trained on feature set 5 performed at 97.2 one-off accuracy with only 12 features, excluding the total text length feature.

The combination of our research questions in conjunction with a wide range of methodologies used in this study enabled us to identify the strengths and weaknesses of simple features. This allowed us to lay a solid foundation for research in the field of CEFR level assessment using limited time and computational resources.

Avainsanat – Keywords
Readability assessment, text classification, machine learning, CEFR

Table of Contents

List of abbreviations

Acc.	accuracy
AEG	automated essay grading
ARA	automated readability assessment
CEFR	common European framework of reference for languages
FS	feature set
KNN	k-nearest neighbors (machine learning method)
L2	second language (non-native)
LR	linear regression (machine learning method)
ML	machine learning
NLP	natural language processing (research field)
Prec.	precision
RQ	research question
SVM	support vector machine (machine learning method)
SVML	support vector machine with linear kernel (machine learning method)
SVMP	support vector machine with polynomial kernel (machine learning method)
SVMR	support vector machine with RBF kernel (machine learning method)

List of abbreviated features

Abbreviated features are written in italic capital letters to distinguish them from other kinds of abbreviations.

ABV	average band value
AJCV	average CEFR-J value
APPS	average pronouns per sentence
ARI	automated readability index
ASL	average sentence length
ASL.AVPS	the interaction term between average sentence length and average verbs per sentence
ATTR	adjusted type token ratio
AVPS	average verbs per sentence
AWL	average word length
BPERA	B per A ratio (the ratio of B-level words to A-level words)
CLI	Coleman-Liau index
DCRS	Dale-Chall readability score
FKG	Flesch-Kincaid grade level
FRE	Flesch reading ease
JCPP	percentage of words not in the CEFR-J word list (CEFR-J prime percentage)
LEN	total text length
TTR	type token ratio

1 Introduction

Assessing the readability of English texts has been important ever since educators have tried to assign reading materials to students of different grades (Dale & Chall, 1949: 19). As English is further establishing its position as a global language, the assessment of text readability for speakers of English as a second language (L2) is gaining considerably more emphasis, especially with the emergence of the Common European Framework of Reference for Languages, known as CEFR (Council of Europe, 2001). CEFR consists of six proficiency levels and has allowed for the standardization of teaching and learning materials across different languages.

Assessing the CEFR level of texts serves as a tool to link suitable texts to L2 readers with different levels of language proficiency. Matching the correct texts to the reader is not only important in language education (Arias, 2007: 132), but essential for ensuring equality in language exams. Countries, such as the UK (Home Office, 2021) and Canada (Government of Canada, 2021), require residents to pass language tests in order to apply for naturalization. Texts in these language tests need to be chosen carefully, especially in regard to their difficulty, to assure fairness for all participants. Furthermore, governments should ensure that important announcements are comprehensible to an extensive portion of residents, with different degrees of education, including non-native speakers. Texts that are too difficult may result in citizens missing essential information. Risks of difficulties in reading and understanding texts delivered by the Dutch government are acknowledged by Velleman and van der Geest (2014: 351).

While it is possible to make use of professionals to assess the difficulty of texts, this can be expensive and time consuming. Automated text readability assessment is able to solve and assist in the above mentioned problems while keeping costs low. This is because the same tool can be used on any number of texts after it has been developed. This makes the field of automated readability assessment very valuable.

1.1 Readability and readability assessment

Dale and Chall (1949: 23) have defined readability as *the sum of all elements in textual material that affect a reader's understanding, reading speed, and level of interest in the material*. Dale and Chall (1949: 20) summarize the findings of Gray and Leary (1935) and conclude three main factors of texts that affect readability. These factors are the format and organization, its content, and the style of expression. According to Dale and Chall, format and organization primarily relate to typography, which is generally evaluated through the choice of font, spacing between the lines, width of margins, etc. Content, on the other hand, is mainly concerned with the topic of discussion and how engaged the readers are in the act of reading. Finally, the style of expression refers to linguistic aspects of the text such as choice of vocabulary and use of grammar.

The third factor, style of expression, falls under the field of linguistics. Questions such as *"What kind of vocabulary, sentence structure and other expressionless elements best suit the abilities of particular groups of readers?"*, as posed by Dale and Chall, (1949: 22), are the main focus for this branch of readability assessment. By focusing on this aspect of readability, we transform the problem of readability into one of comprehensibility without considering the content of the text. This may be referred to as text difficulty or complexity, which is the focus of this study.

Estimating the difficulty of a text can be referred to as readability assessment, and if this process is automated through a machine, it may be referred to as automated readability assessment. In order to assess the difficulty of a text, we need to consider specific aspects of the document, commonly referred to as features. For instance, we may analyze the length of a reading passage. If a text is very short, it is likely to be suitable for beginners, but if it is long, it might be more suitable for advanced students. We can also look at more complex aspects of the text such as the length of sentences, difficulty of vocabulary, and grammatical structures. Analyzing such features allows the machine to predict the difficulty of a text.

In order to reliably predict the difficulty of texts through a machine, analyzing more than one feature is required. A combination of two or more predefined features is referred to as a feature set. Developing cohesive feature sets is essential for automated readability assessment, as not all combinations of features function cooperatively. According to (Brownlee, 2020: 112), this is especially true when the feature set includes features that are not relevant to the target variable (CEFR levels in our case). This brings us to the current research gap.

1.2 Research gap

As of yet, no standard feature set has been delimited for the task of automated readability assessment. With recent advancements in computing power, long and repetitive calculations have become a trivial matter. This in turn has encouraged many researchers to simply test as many feature combinations as possible and see which work best. While this approach yields satisfactory results in many scenarios, there are situations where time and computational resources are limited. In such cases it may be desirable to trade off performance for calculation speed, which can be done through the use of simple features in training machine learning models. To the best of my knowledge, the term ‘simple feature’ has not been previously defined in the field of automated readability assessment. This is because the majority of research has focused on improving the performance of classifiers, and very little attention has been given to matters such as time and computational costs.

When starting this project, two main limitations had to be considered. The first limitation was a lack of significant experience by the author of this thesis. Of course, a large amount of linguistic, mathematical, and programming knowledge had to be acquired throughout conducting this MA thesis, however these were within a specific limit. The second limitation was that all experiments had to be conducted on a personal computer, which severely limited the use of features that have long calculation time. This forced us to take computational cost and time very seriously to allow for the thesis to be completed successfully and in time.

Thus, we had to introduce our own definition of simple features which fits the aims of this study. In this study, simple features have been defined as easily implementable features that require little time to compute. After studying and analyzing different types of features, shallow features, also referred to as textual features, emerged as the best match for our criteria of simple features. According to Feng (2010: 77), shallow features are features whose predictive ability is limited to superficial text properties, but are computationally less expensive. Feng (2010: 77) gives total document length, average word and sentence length, as well as readability formulas as examples for shallow features.

With term simple features defined, this study aims to test the predictive ability of simple features. More specifically, it aims to create and test minimal feature sets, composed of simple features only, that can adequately predict the CEFR level of English texts.

1.3 Research questions

This study aims to develop and compare various feature sets consisting of simple features only. The performance of the feature sets is compared by training and testing classifiers using different machine learning (ML) approaches. The ML approaches chosen for this study are k-nearest neighbors (KNN), linear regression (LR), and support vector machines (SVMs). These three classification approaches were chosen based on their success in previous research, as well as their ease of implementation. Details on these machine learning approaches are provided in Section 3.3, however a simplified introduction to each ML approach is given here to allow for a better understanding of our research questions and hypotheses.

K-nearest neighbors is a classification approach that classifies an observation by comparing a new data point to all other observations in the data set. The most similar data points determine the class of the new observations. Linear regression, on the other hand, attempts to write a response variable (CEFR levels in our case) as a linear function of other explanatory variables (features). Once a model is created, the values of the features can be simply plugged into a formula to calculate the response variable. Finally, support vector machines

function by separating an N-dimensional space into regions (one region per class). If a new observation falls in the region of a specific class, it is predicted as belonging to that class. Furthermore, SVMs have special properties, known as kernels, that allow for non-linear mapping of features to classes. In this thesis, three kinds of SVM models with different kernels are explored. These are SVM using the linear kernel (SVML), SVM using the polynomial kernel (SVMP), and SVM using the RBF kernel (SVMR).

This creates a total of five classification approaches to explore in this study. Additionally, two experiments are designed to uncover the performance of the classifiers, a classifier being a specific combination of feature set and classification approach. The first experiment is conducted using 1000 folds of training and testing 30 classifiers. The second experiment is conducted using three repetitions of k-fold cross validation. This is done to answer the following research questions (RQ):

> **Research Question 1:** *How powerful are simple features for predicting the CEFR level of English texts?* The answer to this question will heavily depend on the performance of the classifiers. Higher accuracy and precision rates indicate that using only simple features is a reliable alternative to computationally expensive features. Low performance, on the other hand, will inevitably prove that simple features on their own are not sufficient for accurately predicting the CEFR level of English texts.

> **Research Question 2:** *What minimal combination of simple features produces the best results for predicting the CEFR level of English texts?* Reducing the number of features in the feature sets is a central goal of this study, as our objective is to define a minimal viable feature set to reduce computational costs. Additionally, some of the developed feature sets exclude the total text length (*LEN*) feature. Succeeding in building classifiers that do not use the *LEN* feature will expand the application of the classifier to potentially classify non-complete texts (partial texts). Petersen and Ostendorf (2009: 92) also refrain from using the total text length feature for a similar reason in their study. Thus, this research question is answered by comparing feature sets across two dimensions, namely through feature reduction (reducing the number of features) and the removal of the total text length feature.

Research Question 3: *Which classification approaches perform best for predicting the CEFR level of English texts when only using simple features?* Answering this question requires finding answers to questions such as *How do different machine learning approaches perform when using the same feature set?* and *How does the number of features in the feature set affect models with different machine learning approaches?*

These three questions are answered by analyzing the performance of different classifiers. The combination of these research questions will allow us to find the best performing classifiers for predicting the CEFR level of English texts using minimal time and computational resources. This will eventually lead to creating viable classifiers which can be used in fields such as language education, information retrieval, and communication. In the future, the classifiers can be combined with factors such as user interest to produce even more accurate results.

1.4 Expectations and hypotheses

Setting an expectation for the performance of the classifiers is important, because there are always more features and transformation of features to explore, and as shown in Figure 2 (page 16), improving a machine learning model can only be stopped if the researcher is satisfied with the performance of the machine. In order to define an expectation for the performance of our classifiers, previous research in the field of automated readability assessment was closely examined. Unfortunately, a significant portion of research on CEFR does not report the exact performance of their classifiers, and instead stick to a more qualitative analysis of the classifiers. Thus, the scope of previous research was expanded to grade level assessment (GLA) and automated essay grading (AEG), in addition to CEFR level assessment (CLA). These are neighboring research fields that often use the same feature sets and classification approaches, and while they pursue different goals and objectives, they contribute greatly to one another.

Based on various factors such as corpus size, feature type, feature set size, the total number of classes, as well as the time and resources available, it was decided that an approximate accuracy of 50 percent was needed for at least one of the classifiers before conducting any of the final experiments. Additionally, we have formalized our hypotheses for the proposed research questions. Some of the research questions for the thesis were developed and refined while conducting experiments and writing drafts of the thesis, thus they do not have strong hypotheses or expectations.

Hypothesis for RQ1: ***How powerful are simple features for predicting the CEFR level of English texts?***

Based on the performance of classifiers in previous research (summarized in Table 2, page 21), where state of the art classifiers perform around 70 percent by using a large number of simple and complex features, we expect the classifiers to perform around 40-50 percent in regard to their accuracy. Reaching such a level of performance would make simple features a valuable tool for text assessment, however this would prove that simple features alone are not sufficient to reliably predict the CEFR level of English texts, and that they need to be combined with complex features to yield satisfactory results.

A higher performance (50–60 percent) would make simple features good predictors of the CEFR levels of English texts. Achieving a performance lower than 40 percent is not likely to occur, other than for specific feature set and machine learning approach combinations. It is also crucial to note that an accuracy of 40–50 percent is considerably lower than the state-of-the-art performance which is around 70 percent (Hancke 2013, Xia 2019, Forti et al. 2020 to name a few). However, I believe that this level of expectation is justified considering the limited number of features used in this study, as well as the inherent limitations of simple features.

Hypothesis for RQ2: ***What minimal combination of simple features produces the best results for predicting the CEFR level of English texts?***

No predefined feature sets were set out to be explored in this thesis. Instead, using a combination of different methods, including feature selection and descriptive statistics, the most promising and meaningful combinations of features were accumulated into feature sets. Expectations were only set after the feature sets were developed. In general, larger feature sets (containing more features) are expected to perform better, however reduced feature sets consisting of approximately ten features should perform similarly. Moreover, we expect the total text length (*LEN*) feature to be a powerful predictor of CEFR levels, and feature sets including this feature should perform significantly better.

Hypothesis for RQ3: ***Which classification approaches perform best for predicting the CEFR level of English texts when only using simple features?***

Most classification approaches are expected to perform similarly, as previous research (Collins-Thompson, 2014: 111) has indicated that the choice of classification approach has less effect on the performance of classifiers than the choice of features. However, it is likely that SVM models outperform other classification approaches by a small margin. This is because SVMs have been used quite successfully in previous readability assessment tasks by Schwarm & Ostendorf (2005) and Petersen & Ostendorf (2009). Additionally, SVMs have shown to outperform other classification approaches in Xia (2019: 26) and Forti et al. (2020: 7208). SVM with the RBF kernel (SVMR) and SVM with the polynomial kernel (SVMP) should theoretically perform better than SVM with the linear kernel (SVML), as they have the ability to map non-linear relationships (Boswell, 2002: 5). However, tuning the hyperparameters[1] of SVMP and SVMR is somewhat challenging, thus SVML is expected to perform slightly better, as it only requires tuning a single hyperparameter.

I do not believe that linear regression models will have sufficient predictive power over CEFR levels. This is because many assumptions of linear regression (Matloff, 2017: 75) are

1 A hyperparameter, also referred to as a tunable parameter, is a special kind of parameter whose value is defined prior to training a machine learning model. This is in contrast to other parameters whose values are derived through training a model.

violated; firstly, some of the features used for this study are non-linear, which violates the linearity condition of linear regression. Secondly, many of the variables are linked and thus measure the same factor, which violates the multicollinearity condition of linear regression. Finally, even though CEFR levels can be viewed as scale measures (0–5), the CEFR level of texts is still a categorical variable and the distance between the CEFR levels is not homogeneous. Lowie (2013: 27), for instance, mentions that the skill range of the user increases dramatically with an increase in CEFR level. These three factors pose many problems for the LR classifiers, which should limit the performance of these classifiers.

Having defined the goals and expectations of this study, the remainder of this MA thesis is structured as follows: Chapter 2 introduces concepts related to readability, machine learning, and automated readability assessment. The chapter also summarizes previous research in the field of automated readability assessment. Chapter 3 describes the data and methodology used for this study. This consists of the corpus used for this study, the explored features and feature sets, the utilized machine learning approaches, the best performing hyperparameters for each kind of classifier, as well as measures of evaluation. Chapter 4 describes the two experiments conducted for this thesis and attempts to answer our proposed research questions based on the results. Chapter 5 makes a final conclusion about our work and discusses the future path for CEFR level assessment of English texts.

2 Background and relevant research

The following section will discuss the Common European Framework of Reference for languages (CEFR), readability formulas, important concepts in machine learning and automated readability assessment, as well as relevant research in the field of automated readability assessment.

2.1 CEFR proficiency levels

The introduction of CEFR, short for the Common European Framework of Reference for Languages (Council of Europe, 2001), was a turning point for L2 learning, teaching, and assessment. CEFR introduced general can-do descriptors (non-language-specific). This allowed for the standardization of teaching and testing material, including reading materials, as well as the comparison of such material across dozens of languages. While we only focus on assessing the difficulty of English texts, CEFR allows us to utilize and reference CEFR related research for languages such as Dutch, Estonian, and Italian. For this study, the six-level CEFR scale, which ranges from A1 to C2, is used (Council of Europe, 2001: 23). This is most common in CEFR related research, however Uchida and Negishi (2018) have used the CEFR-J scale, which consists of 12 levels, and have achieved satisfactory results nevertheless. The proficiency descriptors for each of the six CEFR levels is given in Table 1 (Council of Europe, 2020: 54).

Table 1: Reading comprehension can-do descriptors by CEFR level

Level	Description
A1	Can understand extremely short, simple texts one phrase at a time, picking up familiar names, words, and basic phrases. Multiple reading sessions are often required.
A2	Can comprehend short, simple texts containing the highest frequency vocabulary, including some borrowed vocabulary items.

B1	Can comprehend short and easy texts on familiar matters of a concrete type, as long as they consist of high frequency every day or work-related language.
B2	Can read with a high degree of independence, adapting style and speed of reading depending on different texts and purposes, and using appropriate reference sources selectively. Has a vast active reading vocabulary, however, may potentially experience difficulties with non-common idioms and expressions.
C1	Can comprehend lengthy, detailed, and complex texts, regardless of whether these relate to their own area of specialty, as long as the reader has the option to read difficult sections multiple times. Can comprehend a wide variety of texts including literary writings, newspaper, and magazine articles. Can also understand specialized academic and professional publications, provided the reader can read difficult sections again and has access to reference tools such as dictionaries.
C2	Can comprehend all types of texts including abstract, literary and non-literary, structurally complex, and highly colloquial writings. Can comprehend a wide range of long and complex texts, and even has the ability to appreciate subtle distinctions of style.

Identifying meaningful minimal feature sets that can discriminate between the different CEFR levels is the main objective of this study. In our study, CEFR levels are mostly viewed as labels similar to any other classification task. However, linear regression views CEFR levels as scale values ranging from 0 to 5 rather than actual classes. CEFR levels are also viewed as scales when analyzing classification error, which measures the extent to which classifiers misclassify texts.

2.2 Automated readability assessment

While no formal definition for automated readability assessment exists, it can be defined as the calculation of the features of a text, along with the classification of the text itself by a

classifier (model), without the need for human intervention after the model has been created. This can be in the form of simple models such as linear regression, all the way to complex models that use artificial neural networks (used in Azpiazu & Pera, 2019; and Martinc, 2021). The next section discusses the earliest form of automated readability assessment, important concepts in machine learning, as well as the steps required for automated readability assessment.

2.2.1 Readability formulas

According to Dale and Chall (1949: 22), the process of matching an index of difficulty, known as labels, to elements in the passage, known as features, gave rise to readability formulas. These formulas are mainly in the form of linear regression models that take input values, typically in the form of simple lexical difficulty and syntactic difficulty features, and output a grade level, which is generally interpreted as the number of years of education needed to understand the text. Thus, readability formulas can be viewed as the earliest form of automated readability assessment. Crossly et al. (2008: 476) cite over 50 readability formulas, which shows the importance of these formulas.

While readability formulas appeared to be a promising tool for assessing text readability, their use in the creation of textbook programs was abandoned in the 1980s. This was the result of new research which revealed the shortcomings of these formulas (Hiebert, 2002: 338). In fact, readability formulas have been criticized for their weaknesses ever since they were first introduced. These weaknesses can be summarized in three main points; firstly, readability formulas do not evaluate conceptual difficulty (Lorge, 1949: 91). In other words, they do not account for the difficulty of the topic discussed, presumed knowledge, or abstractness of concepts. Secondly, readability factors are often measured through proxy variables (Collins-Thompson, 2014: 99) rather than a direct measure of the text characteristics. Proxy variables act as an indirect measure, such as sentence length in tokens being a proxy variable for syntactic complexity, and word length in syllables being a proxy variable for lexical complexity. Finally, the ordering and sequencing of words and sentences

are not taken into consideration for the evaluation (Collins-Thompson, 2014: 5; and Lorge, 1949: 91).

Some attempts were made to remedy the use of proxy variables by finding more direct ways of measuring lexical difficulty. The Dale-Chall readability formula (Dale and Chall, 1948) and the Lexile Framework (Stenner, 1996) are two such examples. This solved some problems with readability formulas, and in fact the Lexile Framework is still in relatively wide use today, both commercially and in the field of readability research.

In spite of the flaws of readability formulas, some research shows that they should not be completely disregarded. In experiments conducted with various feature types, Feng (2010) discovered that even though the Flesch-Kincaid score performed poorly when used as a direct measure of text difficulty, it showed significant discriminative power in regard to text difficulty assessment when it was used as a feature in advanced machine learning methods. Readability formulas are particularly important for our study, as they make up five out of the nineteen features explored in this thesis. Details on these readability formulas are given in Section 3.2.1, however the Flesch-Kincaid grade level is given here as a reference for what readability formulas look like.

$$\text{Flesch-Kincaid grade level} = 0.39 \times \left(\frac{\textit{total words}}{\textit{total sentences}}\right) + 11.8 \times \left(\frac{\textit{total syllables}}{\textit{total words}}\right) - 15.59 \quad (1)$$

As shown in Equation (1), two features are used in the Flesch-Kincaid grade level. These are an average sentence length (*ASL*) feature (calculated by dividing total words by total sentences), and an average word length (*AWL*) feature (calculated by dividing total syllables by total words in a text). The Flesch-Kincaid grade level, in and of itself, is a linear regression classifier that produces values between 1–12 (grade levels) instead of 0–5 (CEFR levels).

2.2.2 Machine learning and automated readability assessment

According to Mohri et al. (2018: 1) machine learning (ML) can be defined as computational methods that utilize human-labeled training data sets to make predictions about a new observation. The goal in machine learning is not to simply memorize the entire data set, but to extract meaningful generalizations that can be applied to all observations. This is achieved through striking a good balance between sample size and complexity (Mohri et al., 2018: 8). Complexity is especially important, as a model too complex may have the ability to map extremely complex relationships between observations. This may allow the classifier to correctly classify all observations in the training data set, however this could lead to worse generalizations. This is demonstrated in Figure 1, which is adapted from Mohri et al. (2018: 8). Such behavior, i.e. studying training data too closely, is referred to as overfitting and almost always leads to worse generalization ability. The model on the right, on the other hand, uses a simple model, which allows for the misclassification of training data. This results in a lower performance score on training data (more misclassified observations within the training data), however it is clear that the knowledge of the model on the right is more generalized, and it should perform better on unseen data. It is also possible for a classifier to fail at finding a line that separates the data into two groups. If the machine fails to make meaningful connections between variables, or distinctions between the classes, the machine is said to be underfitting the training data. Both overfitting and underfitting can result in high error on unseen data.

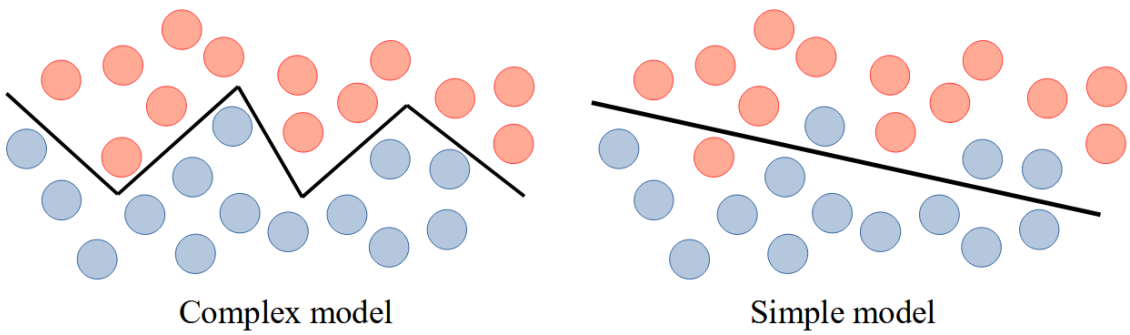

Figure 1: Complex model vs. simple model and their ability to generalize

Text readability assessment is a subfield of text classification, which falls under machine learning. Similar to machine learning, automated readability assessment requires following a series of steps to implement. Collins-Thompson (2014: 102-104) summarizes automated readability prediction into three steps, which are described below (steps 1-3). Steps 4 and 5 are added by the author of this thesis, and are required for understanding and improving the performance of the created classifier. These five steps are illustrated in Figure 2. Standard conventions were used for drawing the flow chart (Dahlgaard et al., 2008: 113).

1. **Collect, label, and split the corpus:** Refers to gathering a collection of labeled texts separated into training and testing data. Training data refers to the data the machine sees and attempts to generalize. Testing data, on the other hand, refers to previously unseen data which is used to validate the performance of the machine.

2. **Define feature set:** Refers to the identification and employment of features (predictors) that will be used to match the text to its label, CEFR level in this case.

3. **Train the machine learning model:** Showing the training data to the machine and asking it to find relations between features and labels. The hyperparameters of the machine learning algorithms are set at this stage. The algorithm, referred to as a classifier, is ready at this point and needs to be tested in regard to its performance.

4. **Test the machine learning model:** Exposing the classifier to unseen data (testing data) and asking it to predict the labels of the data based on the previously defined features and the knowledge the machine has formed of the relationship between features and labels.

5. **Adjust the classifier:** If the accuracy of the classifier is not desirable, the hyperparameters of the classifier can be tweaked and training and testing can happen from scratch. The feature set may also be tweaked before retraining.

These exact steps were followed for this study. However, because the focus of this research lies primarily in exploring the performance of classifiers using simple features for CEFR

level assessment, the six developed feature sets were implemented into the machine and tested one after another rather than changing feature sets if satisfactory results are not achieved. The feature sets were developed using feature selection, which is explained in Chapter 3.

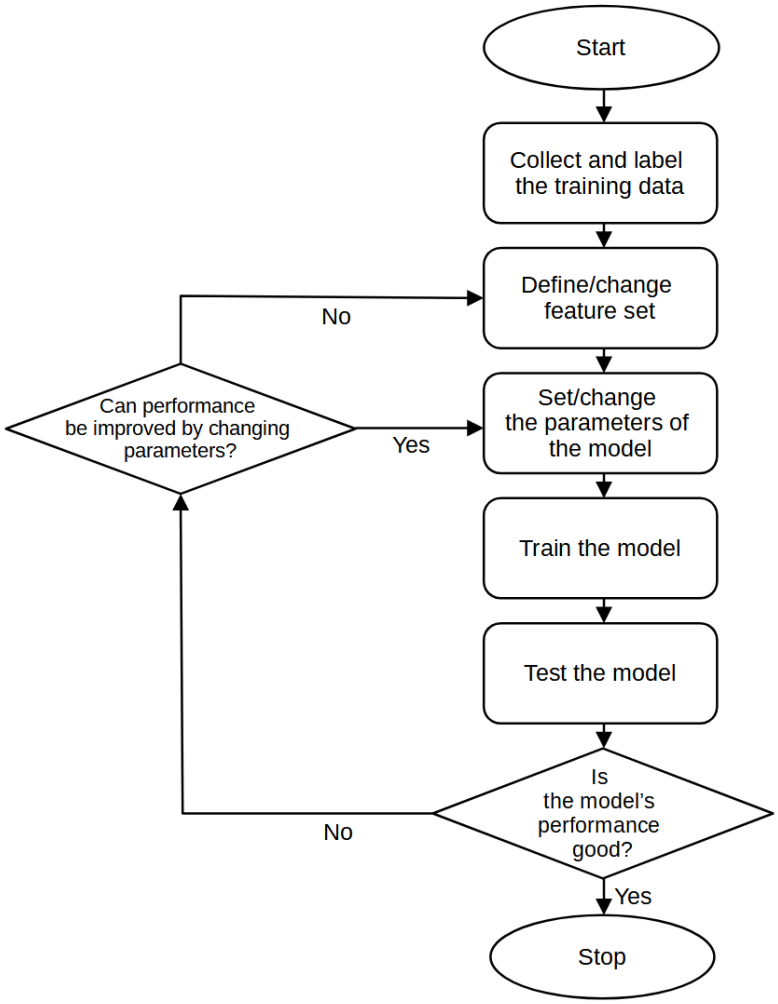

Figure 2: Flow chart for automated readability assessment

2.3 Relevant research

A wide range of research in the field of CEFR level assessment, grade level assessment, and automated essay grading has been studied and referenced for this study. The following section will describe the most relevant research from these fields regarding the choice of features, classification approach, and measures of evaluation.

Readability formulas can be viewed as the earliest form of automated readability assessment. The history, application, and performance of readability formulas has been described in Section 2.2.1. While readability formulas have many useful applications, the type of information they capture does not consider aspects of readability related to the content of the text. This limitation is often linked to their use of proxy variables, as well as their inability to capture the order of words (Collins-Thompson, 2014: 99). Proxy variables are similar to surface or textual features, encompassing features such as average sentence length and average word length. The ability of these features is often described as being limited to superficial text properties (Feng, 2010: 77). Using readability formulas was especially problematic for assessing the difficulty of online texts, as these were significantly different from traditional texts which many readability formulas were tuned to assess. Si and Callan (2001: 574) mention for instance, that the inclusion of text fragments that are not organized into traditional sentences and paragraphs can significantly hinder the performance of readability formulas.

Si and Callan (2001) attempted to remedy the poor performance of surface features by combining surface features with statistical language models. Statistical language modeling is a method of extracting lexical complexity and it works by estimating the probability distribution of linguistic units, generally in the form of words (Rosenfeld, 2000: 1). Si and Callan concluded that using surface features and language models simultaneously yields better results than using any single one separately. Collins-Thompson and Callan (2005) also utilized language models to assess the grade level of online texts. This was based on a special kind of language model[2] which worked by computing the probability of each token belonging to a specific grade level. One model per grade level was constructed. Then, the probability of the given text being generated by each of the 12 models was calculated, and the model with the highest probability was chosen as the prediction for the difficulty of the text.

Meanwhile, other researchers started experimenting with different machine learning approaches for predicting the grade level of texts. Schwarm and Ostendorf (2005) combined

2 Collins-Thompson and Callan's model was a smoothed unigram model which is based on a variation of the multinomial naives Bayes classifier. A unigram model assumes that the probability of a token is independent of the surrounding tokens.

readability formulas with statistical language models and support vector machines to create a more complete model for assessing reading levels of texts. In their study, Schwarm and Ostendorf classify texts ranging from grade 2 to grade 5. They utilize a combination of 25 complex and simple features, and reach an average precision of 55.3 and an average recall of 71.8.

Petersen and Ostendorf (2009) continue this research using the same corpus and feature set, but instead, focus on the contribution of different feature types in the prediction of grade levels. Furthermore, they measure the reliability of human annotators for labeling texts based on difficulty. In their experiments, they find that human annotators often do not agree on the grade level of texts, and even disagreements of over one level are relatively common. Additionally, Petersen and Ostendorf show that their SVM classifier with the RBF kernel can predict labels from the Weekly Reader[3] data set even more reliably than human annotators. To the best of my knowledge, the authors are also the first to introduce the one-off measure of performance, where misclassifications of one level are considered as correctly classified observations. In Petersen and Ostendorf (2009: 96), this is done by comparing systems based on the percentage of texts with labels that are off by more than one grade level. The same measure is referred to as within-1-level-accuracy in Vajjala and Loo (2014). Both of these studies inspired the use of one-off accuracy for this study, which is further explained in Section 3.4.3.

With an increasing amount of research in the field of automated readability assessment, Collins-Thompson (2014) attempted to give a comprehensive background of the literature in the field of automated readability assessment. In addition to summarizing the methodology for automated readability assessment, described in Section 2.2.2, Collins-Thompson concludes seven categories of features used for the task of readability assessment. These are lexico-semantic, morphological, syntax, discourse, higher-level semantics, pragmatics, and user-oriented features. This categorization is based on what the features measure, rather than how easy or difficult the features are to compute or implement.

3 *Weekly Reader* was an educational magazine designed for children which started in 1928. The magazine was discontinued as an independent publication in 2012 and was merged with *Scholastic News* (https://scholasticnews.scholastic.com/).

Many methodologies from readability assessment were also applied to automated essay grading, especially using CEFR. Vajjala and Loo (2014) use Sequential Minimal Optimization (SMO) in their experiments, which is an algorithm used in the training of support vector machines (Platt, 1998). They utilize a variety of features including morphological and Part of Speech (POS) tag density based features to assess Estonian essays graded from A2 – C1 and reach a prediction accuracy of 79%. They also compare CEFR level assessment as classification task and as regression task and conclude that results are slightly better when classification methods are used.

One research that shares many methodological similarities with our study is Forti et al. (2020). In their study, the researchers explored 139 features and reduced them to 54 features, which were used for the assessment of Italian texts based on CEFR. While the published paper only focuses on the SVM classifier, Forti et al. experimented with three different classification models, namely decision trees, random forests, and support vector machines. Additionally, similar to the one-off measure of performance, Forti et al. introduced the macro level accuracy which is achieved by grouping macro CEFR levels, i.e. by combining B1/B2 into the B macro class, and C1/C2 into the C macro class, and evaluating the performance of the classifiers after.

Some studies have also attempted to use simple features similar to this study, however they are not explicitly described as simple features. Velleman and van der Geest (2014) use a combination of six simple features to create an online tool for the assessment of the CEFR level of Dutch texts. This paper is the main inspiration for our study. Unfortunately, Velleman and van der Geest do not report the performance of their classifier in the original paper. The performance of the tool they created is put to the test in Jansen and Boersma (2013), however, this is done in a qualitative manner which does not allow for comparison between their classifier and previous research.

Another study that makes use of simple features only is Uchida and Negishi (2018). In their study, the researchers attempt to utilize textual features to assign CEFR-J levels (12 levels ranging from pre-A1 to C2) to English texts by using linear regression. They use four textual features, namely the automated readability index (*ARI*), average verbs per sentence (*AVPS*),

average of word difficulties (*AvrDiff*), and the ratio of B words to A words (*BPERA*). This study has influenced our choice of features significantly. In fact, all of these four features are used in our study.

There are, however, significant differences between the above mentioned studies and ours. While feature sets used in Velleman & van der Geest (2014) and Uchida & Negishi (2018) fulfill the requirements for simple features, these studies did not specifically aim to reduce calculation time and computational costs for text classification. Furthermore, we compare a wide variety of classification approaches to ensure that the ability of the features and feature sets are expressed to their maximum potential. Some features may not perform well with some classification approaches, but might show strong discriminative power using other methods.

K-nearest neighbors is another machine learning algorithm that has been used in the field of text classification. Wang and Zhao (2012) are one of the first studies to use this algorithm for the task of text classification. In their study, they suggest modifications to the algorithm to make it more suitable for the task of text classification. In another study, Shah et al. (2020) compared the performance of various classification approaches, including KNN, for determining the topical category of texts. Topical categories included business, entertainment, politics, sports, and technology. They conclude that KNN performs only slightly worse than random forests, which show relatively good overall performance.

Finally, as mentioned in Section 1.4, we had to analyze previous research to get a good understanding of the needed corpus size, feature type, and feature set size, range of classes, and the performance of the classifiers. To assist in setting manageable expectations, relevant research that has reported the performance of their classifiers was summarized in Table 2. This has served as a basis for our expectations and hypothesis mentioned in Section 1.4. Classifiers that were reported in PhD and Master theses have not been reported in this literature review, however they were referenced for setting a benchmark for the performance of our classifiers. These are Petersen (2007), Feng (2010), Hancke (2013), and Xia (2019). Referencing these studies has been extremely helpful, as they include more details on their experiments, in addition to reporting the performance of their classifiers.

Table 2: Summary of research on text level assessment

Researcher(s)	Performance	Research	Range	Number of features	Corpus size
Schwarm & Ostendorf (2005)	55.3 (precision) 71.8 (recall)	GLA	GL 2-5	25	2277
Petersen (2007)	70 (F1-score)	GLA	GL 2-5	25	2397
Feng (2010)	74 (precision)	GLA	GL 2-5	267	1433
Hancke (2013)	69.4 (accuracy) 61.1 (hold out)	AEG	CEFR A1-C1	34	1027
Vajjala & Loo (2014)	79 (precision)	AEG	CEFR A2-C1	27	879
Xia (2019)	75.3 (precision)	CLA[4]	CEFR A2-C2	38	4367
Forti et al. (2020)	71.9 (precision) 88.5 (Macro precision)	CLA	CEFR B1-C2	139	692
GL: grade level AEG: automated essay grading		GLA: grade level assessment CLA: CEFR level assessment			

Overall, the literature in automated text readability assessment has covered a wide variety of feature types and machine learning approaches to evaluate the level of texts in the fields of grade level assessment, CEFR level assessment, and automated essay grading. The performance of surface/textual features has also been explored, however no previous research has focused on using simple features to reduce time and computational costs. This is because the majority of research has focused on improving the classifiers as much as possible, which makes our research unique and valuable.

4 In addition to assessing the CEFR level of texts, Xia (2019) also attempted to grade the level of summarized texts. This part of their work is more similar to AEG.

3 Data and methodology

The following section will discuss the material and methods used for this study. Section 3.1 discusses the corpus used in this study. Section 3.2 goes over the features covered in this thesis, as well as the six feature sets that were developed. Section 3.3 discusses the machine learning models and the model hyperparameters. Section 3.4 discusses splitting the data into training and testing sets, k-fold cross validation, as well as relevant evaluation measures used in this study. The two experiments conducted for this thesis will be described in Chapter 4.

3.1 Corpus

Classification, which is a kind of predictive modeling, requires learning and mapping relationships from data (Brownlee, 2020: 10). In Natural Language Processing (NLP), data is generally in the form of a corpus. Park (2014) summarizes a corpus as a collection of spoken or written language material used for linguistic analysis. In the case of CEFR readability assessment, this corpus is a collection of texts which is tagged (or grouped) based on CEFR levels. Originally, a total of 177 CEFR rated texts were manually collected from various online sources. These texts were then preprocessed and used to train and test a variety of classifiers. The performance of these classifiers was recorded and reported in various intermediary reports, however due to various factors including low performance of the classifiers, as well as inability to compare results to previous research, it was decided to retrain all models using a well-accepted corpus, namely the Cambridge English Readability Dataset, also known as the Cambridge English Exams Data (Xia et al., 2016). This corpus was collected by Xia et al. (2016) and consists of 331 texts (in .txt format) used in five different Cambridge English exams, each corresponding to a separate CEFR level ranging from A2 to C2. General information about this data set is shown in Table 3.

Table 3: General information about the Cambridge English Readability Dataset

CEFR level	A2	B1	B2	C1	C2
Source	KET	PET	FCE	CAE	CPE
Number of texts	64	60	71	67	69
Average length	14.75	19.48	38.07	45.76	39.97

Some thought was given to merging our original collected data set with the Cambridge English Readability Dataset, especially because the well accepted data set does not include texts for the A1 CEFR level. However, initial experiments showed that doing so would result in lower performance of the classifiers. However, mixing data from different sources could enable the classifiers to generalize their knowledge to a wider range of text kinds. This issue was not further explored in this study.

Regardless, one clear advantage of changing the data set after initial experimentation is that our choice of features, feature sets and even initialized hyperparameters of the classifiers were supported by data not included in training or testing, which is similar to using a development data set. Development data set refers to data used specifically for tuning the parameters of the classifiers, and is used in Schwarm and Ostendorf (2005). The authors do not provide an explanation for why a development data set is needed, however I believe this is because using such a set can reduce the chance of classifiers overfitting the corpus data. In our case, after changing to the Cambridge English Readability Dataset, the original data set of 177 observations acted as a development data set. Furthermore, switching to the new data set boosted the performance of the classifiers significantly. Some classifiers went from an accuracy of 50% to 70%, which is a significant increase in performance, and shows that the quality of our original data was relatively low. Low performance could also be an indication of inconsistencies labeling the data in the original corpus.

3.2 Features and feature sets

In order to extract features from the text files (files with a .txt extension), a custom feature extraction tool was built. Building the feature extractor has been one of the most time consuming parts of this thesis, however Python packages such as *NLTK* (Bird et al., 2009) and *textstat* (Bansal & Aggarwal, 2021) facilitated the process. The usefulness of each feature was then evaluated and the features were grouped into meaningful feature sets. The feature extractor can be accessed through the link provided in Appendix C.

3.2.1 Features

Various features were explored throughout the development of the project. Some features, such as average tree height, were abandoned relatively early due to their high computational cost. Other features were modified in an attempt to make them more meaningful. In the end, 19 features were extracted which were carefully studied and put into meaningful feature sets for this study. The combination of all 19 features forms Feature set 1 (FS1). For more details on all six feature sets, refer to Section 3.2.3. However, before looking at the features, we need to discuss some important terminology in the field of NLP, namely the definition of 'token', 'type', and 'sentence'. The definitions of token and type have been referenced from Bird et al. (2009: 7-8).

> **Token:** A token is a series of characters delimited by white space, including single space (*' '*), multiple spaces (*' '*), tab (*'\t'*), and new line (*'\n'*). Thus, the sequence *"hi hi hi"* contains three tokens, which are *"hi"*, *"hi"*, and *"hi"*.
>
> **Type:** Type refers to the class of all tokens containing the same character sequence. Thus, the sequence *"hi hi hi"* contains a single type, *"hi"*.
>
> **Sentence:** A sentence is defined as a series of tokens delimited by a period, question mark, or exclamation mark.

Considering the definitions above, the text *Then said the man, "What can be done now? The boy must be christened."* consists of 14 tokens, 12 types, and 2 sentences as shown in Table 4. The counter at the bottom indicates when the counting of token, type, and sentence occurs. For our study, all punctuation marks are removed during tokenization (the process of tokenizing), which makes the measurements by the feature extraction tool more reliable. This is because punctuation marks artificially increase the length of tokens and sentences, which can make these features less meaningful.

Table 4: Example of token, type, and sentence in NLP

	then	*said*	*the*	*man*	*what*	*can*	*be*	*done*	*now*	*the*	*boy*	*must*	*be*	*christened*
Token	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Type	1	2	3	4	5	6	7	8	9		10	11		12
Sentence									1					2

General features: Features that relate to numerous aspects of readability simultaneously, instead of focusing on a single aspect. From the list of 19 features explored in this study, only the total text length (*LEN*) can be classified in the general features category.

1. **Text length (*LEN*):** The total text length (*LEN*) is very simple to calculate and understand, however this feature is relatively complicated. It encompasses a wide variety of features from lexical to syntactic. For instance, longer texts use more varied vocabulary and they inevitably have a lower type token ratio (refer to feature 8). Logically, language students at A1 or A2 level will have a very difficult time reading long texts, even if other linguistic aspects, such as vocabulary and grammar, are relatively simple. Forti et al. (2020) use total text length measured in sentences instead of tokens. The choice to count text length in tokens instead of sentences is due to our original data containing sections such as passage title and author name. Counting text length in tokens would minimize the effect of these sections.

   The total text length is a central feature in our study, and thus three out of six feature sets developed for this study contain the *LEN* feature, and three do not contain the *LEN* feature. This is done to compare the performance of classifiers with and without

the *LEN* feature, which will allow us to determine whether it is possible to train classifiers with high performance that do rely on the total text length feature.

Lexical features: This set of features includes those concerned with the lexical complexity of the text. This is measured through features such as average word length, the type token ratio, as well as a variety of word list features.

2. **Average word length (*AWL*):** Average word length is calculated by counting the number of characters in each token, and taking their average across the entire text. This is technically average token length, however initial experiments with our data showed that the difference between average word length and average token length is minimal. *AWL* has been used in various readability formulas, which are linear regression classifiers. The automated readability index (*ARI*, feature 13) uses *AWL* in characters, while the Flesch-Kincaid grade level (*FKG*, feature 16) uses *AWL* in syllables. The *AWL* feature has also been directly utilized as a feature in Schwarm & Petersen (2005: 526), Vajjala & Loo (2014: 118), Velleman & van der Geest (2014: 354), and Forti et al. (2020: 7208).

   Average band value (*ABV*): The average band value is a feature calculated from the 20,000 most frequent words in the COCA Academic texts (Davies & Gardner, 2013). The word list is split into bands representing word frequency. Each band represents a range of 500 words, thus band 1 encompasses the 1-500 most frequent words, band 2 encompasses the 501-1000 most common words, etc. The band of each word is extracted from the COCA Academic texts word list, and the mean value for the entire text is calculated. In order to compare tokens to the word list, tokens were first converted to dictionary forms (stemmed and lemmatized), and the dictionary form was compared to the word list. Initial experiments showed that using band values directly yielded poor results, thus bands larger than 5 were changed to 5 in order to make the results more meaningful. To the best of my knowledge, the COCA Academic texts word list has not been used for the task of readability assessment previous to this study.

3. **Average band value – max (*ABVMAX*):** Max indicates that where there are multiple instances of a word (e.g. 'like' as a verb and 'like' as a preposition), the largest band value is taken.

4. **Average band value – mean (*ABVMEAN*):** Mean indicates that where there are multiple instances of a word with different bands, the mean of the two values is taken.

5. **Average band value – min (*ABVMIN*):** Min indicates that where there are multiple instances of a word, the smallest band value is taken.

6. **Average CEFR-J value (*AJCV*):** This measure of lexical difficulty is calculated by computing the average CEFR value of words based on the CEFR-J word list[5] (2020). To enable computing an average, words belonging to A1, A2, B1, and B2 were converted to 0, 1, 2, and 3 respectively. Similar to the three ABV features, each token was first converted to its dictionary form and only then compared to the word list. This feature was used in Uchida and Negishi (2018: 3), however the original CEFR-J word list has been slightly modified for this study to better suit our needs. The original data set uses American and British English spellings as a single observation in the word list (in the form of *neighbor / neighbour*). These were separated into different observations which facilitated text processing. Secondly, the CEFR levels were converted to scales from 0-3 to allow for the calculation of an average value. Finally, non-essential data, such as the word category (e.g. *travel* or *daily life* tags) were removed to make the word list easier to work with.

7. **B per A ratio (*BPERA*):** The B per A ratio is another measure of lexical difficulty and is calculated based on the CEFR-J word list (2020). More specifically, this feature is calculated by dividing the number of B words (words belonging to B1 and B2 CEFR levels) by A words (words belonging to A1 and A2 CEFR levels). This feature was also introduced and used by Uchida and Negishi (2018).

5 The CEFR-J word list was created through careful analysis of commonly used English textbooks used in China, Korea, and Taiwan (Neigishi et al., 2013: 151).

8. **Percentage of tokens not in CEFR-J (*JCPP* – CEFR-J prime percentage):** *JCPP* is another measure of lexical complexity and is calculated by counting the percentage of tokens which are not in the CEFR-J word list, thus not belonging to the A or B CEFR levels. This feature is also based on the CEFR-J word list (2020), and to the best of my knowledge is unique to this study. While writing the thesis, it was noticed that using the percentage of types not in the CEFR-J word list, instead of tokens, would have yielded better results.

9. **Type token ratio (*TTR*):** A measure of lexical diversity, as well as an approximation of morphological complexity according to Kettunen (2014: 242). The *TTR* is calculated by dividing the number of types by the number of tokens in a text. *TTR* has been used in numerous studies, including Vajjala & Loo (2014: 122), Xia (2019: 43), and Forti et al. (2020: 7209).

Syntactic features: Syntactic features can be defined as features that relate to text attributes at the sentence level, and include average sentence length, verb per sentence, and pronouns per sentence. Some of these syntactic features can also be viewed as grammatical features. Average sentence length (*ASL*), for example is classified as a grammar related feature in Heilman et al. (2007: 460). The average per text is used as a feature in this study.

10. **Average sentence length (*ASL*):** Average sentence length is an indirect measure of syntactic complexity and is calculated by taking the average number of tokens per sentence in the entire text. In general, the longer the average sentence length, the more difficult a text is expected to be. Similar to *AWL*, *ASL* has been used in a number of LR classifiers in the form of readability formulas. These include the automated readability index (*ARI*, feature 13), the Coleman-Liau index (*CLI*, feature 14), the Dale-Chall readability score (*DCRS*, feature 15), the Flesch-Kincaid grade level (*FKG*, feature 16), and the Flesch reading ease (*FRE*, feature 17). The *ASL* feature has also been used directly as a feature in Schwarm & Petersen (2005: 526) and Uchida & Negishi (2018: 3).

11. **Average verbs per sentence (*AVPS*):** This measure of syntactic complexity is calculated by counting the average number of verb tokens per sentence. Verb tokens are tokens with a verb Part of Speech (POS) tag, which were tagged using the *nltk.pos_tag_sents* function (*NLTK*'s recommended POS tagger). Similar to *ASL*, we expect higher average verbs per sentence to directly correlate with higher CEFR levels. The *AVPS* feature has been used in Vajjala & Loo (2014: 118), Velleman & van der Geest (2014: 354), Solovyev et al. (2018: 3054), and Forti et al. (2020: 7208).

12. **Average pronouns per sentence (*APPS*):** This feature was used by Velleman and van der Geest (2014: 354) and is meant to measure additional cognitive load put on readers by having to remember what or who the pronouns refer to, in addition to syntactic complexity. We expect CEFR levels to correlate positively with average pronouns per sentence. This feature has also been used in Solovyev et al. (2018: 3054).

Readability formula features: Readability formulas are one of the earliest forms of automated readability assessment and represent a combination of lexical and syntactic features. Feng (2010) concludes that readability formulas show significant discriminative power in regard to text difficulty assessment when they are used as a feature in advanced machine learning models. In our study, the performance of five different readability formulas has been analyzed. All formulas, with the exception of the automated readability index (*ARI*), were implemented using Python's *textstat* package (Bansal & Aggarwal, 2021).

13. **Automated readability index (*ARI*):** The automated readability index is derived from ratios representing word difficulty and sentence difficulty (Senter, 1968: 1) and is calculated as follows:

$$ARI = (0.5 \times \textit{average sentence length}) + (4.71 \times \textit{average word length}) - 21.43 \quad (2)$$

In the literature reviewed for this study, the *ARI* has only been used in Uchida and Negishi (2018).

14. **Coleman-Liau index (*CLI*):** This readability index falls between 1 and 16 and was originally based on calculating cloze percentages and finding the corresponding grade level (Table 5). The cloze percentage was estimated using the following formula (Coleman, 1975: 283-284):

$$\textit{Estimated cloze}\,\% = 141.8401 - (0.214590 \times L) + (1.079812 \times S) \quad (3)$$

where L is the average number of letters per 100 words and S is the average number of sentences per 100 words.

Table 5: Translations of cloze percentages into grade levels (GL)

GL	Cloze %	GL	Cloze %	GL	Cloze %	GL	Cloze %
1	80.5	5	65.9	9	51.3	13	36.7
2	76.9	6	62.3	10	47.7	14	33.1
3	73.2	7	58.6	11	44	15	29.4
4	69.6	8	55	12	40.4	16	25.8

However, modern implementations of *CLI*, including its implementation in *textstat*, use the following formula instead:

$$CLI = (0.058 \times L) - (0.296 \times S) - 15.8 \quad (4)$$

where L is the average letters per word multiplied by 100, and S is the average sentence per word multiplied by 100. In the literature reviewed for this study, the *CLI* has only been used in Xia (2019: 43).

15. **Dale-Chall readability score (*DCRS*):** Unlike most other readability formulas, the Dale-Chall readability score relies on a word list to distinguish difficult words from easy words, which makes it a more direct measure of lexical complexity. The word list contains 3000 easy words, and words outside of this list are considered difficult. Dale and Chall (1948: 44) define simple words as words that are familiar to at least 80

percent of children in the fourth grade. *DCRS* is calculated as follows (Dale & Chall, 1948: 18):

$$DCRS = 0.1579 \times \left(\frac{difficult\ words}{total\ words} \times 100\right) + 0.0496 \times \left(\frac{words}{sentences}\right) \quad (5)$$

16. **Flesch-Kincaid grade level (*FKG*):** This readability index is one of the most commonly used readability scores and generally falls between 1 and 12. The formula is basically the same as the Flesch Reading Ease (*FRE*, feature 17), with different weights and is calculated as follows:

$$FKG = 0.39 \times \left(\frac{total\ words}{total\ sentences}\right) + 11.8 \times \left(\frac{total\ syllables}{total\ words}\right) - 15.59 \quad (6)$$

The Flesch-Kincaid grade level (*FKG*) has been used in Schwarm & Ostendorf (2005: 526), Feng (2010: 77), Solovyev et al. (2018: 3054), and Xia (2019: 43).

17. **Flesch reading ease (*FRE*):** As mentioned above, this readability measure is basically the same as Flesch-Kincaid grade level (*FKG*) with different weights. Additionally, instead of returning a value between 1 and 12, it returns a value between 0 and 100 for most ordinary prose (Flesch, 1948: 225). The Flesch Reading Ease is calculated as follows:

$$FRE = 206.835 - 1.015 \times \left(\frac{total\ words}{total\ sentences}\right) - 84.6 \times \left(\frac{total\ syllables}{total\ words}\right) \quad (7)$$

In the literature reviewed for this study, the Flesch reading ease has only been used in Solovyev et al. (2018: 3054).

Modified features: Attempts were made to make the already collected features more valuable, which resulted in *ASL.AVPS* and *ATTR*. Modifications to the original features were made with the goal of increasing linearity of the features or accounting for some of their

flaws. These two features were explicitly developed for this study, and to the best of my knowledge have not been used in other studies.

18. **Interaction term between *ASL* and *AVPS* (*ASL.AVPS*):** The interaction term between average sentence length (*ASL*) and average verbs per sentence (*AVPS*) is calculated by simply multiplying the two features together. The hope is to create a more linear and meaningful feature by using both features in combination with one another.

19. **The adjusted type token ratio (*ATTR*):** Koizumi (2012) clearly demonstrates that the *TTR* of texts decreases consistently with an increase in text length. The adjusted type token ratio (*ATTR*) is an attempt to account for this phenomenon by taking the text length into consideration. In this paper, the *ATTR* is calculated as follows:

$$ATTR = \log_e(total\ text\ length) \times TTR \quad (8)$$

The choice of log transforming the text length was only due to practical reasons, but other transformation methods that significantly reduce the effect of the total text length could have been used as well. Vajjala and Loo (2014: 122) make use of the corrected type token ratio (CTTR) feature, however they do not specify how the feature is different from the normal *TTR*.

The final result of the feature extraction is a .csv[6] file that is used for training and testing, similar to the data displayed in Table 6. Due to space limitations, the rows and columns in Table 6 have been switched (i.e. the data frame has been transformed). In the actual data frame, columns represent features and rows represent observations. The entire data set can be accessed using the link provided in Appendix B.

6 CSV, short for comma-separated values, is a file format that uses commas to delimit values in order to store information in an organized manner.

Table 6: General structure of the data set used for training and testing classifiers

Feature / Filename	A2 (1).txt	A2 (10).txt	A2 (11).txt		C2 (9).txt
cefr	1	1	1		5
abvmax	2.49	2.79	2.62		2.65
abvmean	1.97	2.34	2.16		2.28
abvmin	1.51	1.87	1.8		1.91
ajcv	0.18	0.44	0.51		0.51
apps	1.27	1.08	1		1.36
ari	6.46	7.34	4.82		10.07
asl	17.09	15.92	11.2		20.82
asl.avps	46.61	38.47	11.2	values for +327 other texts	70.02
attr	3.51	3.27	2.74		3.61
avps	2.73	2.42	1		3.36
awl	4.11	4.42	4.38		4.48
bpera	0.03	0.15	0.13		0.19
cli	6.32	8.01	7.46		9.06
dcrs	6.01	5.66	5.59		7.01
fkg	5.9	8.5	4.2		9.9
fre	80.92	68.6	85.28		65.05
jcpp	13.83	4.71	15.18		11.14
len	188	191	112		458
ttr	0.67	0.62	0.58		0.59

3.2.2 Feature evaluation

After the features were extracted, their predictive ability was evaluated. Evaluating the predictive ability of features serves two main purposes. Firstly, it allows us to see the feature's ability to predict the CEFR level of English texts. If a feature possessed a low predictive ability, various feature transformation methods were used to improve its predictive ability (e.g. using square, square root, or log transformation on the feature). Both *ASL.AVPS* and *ATTR* are products of attempting to improve the predictive ability of already extracted features. Different versions of *ABV* (*ABVMAX*, *ABVMEAN*, *ABVMIN*) were also developed by trying different modifications and seeing whether the predictive power of the feature

increases or not. Secondly, evaluating the goodness of features can assist in building meaningful feature sets. Our main task in regard to developing feature sets is to reduce the number of features without sacrificing too much performance. This can be viewed as dimensionality reduction with the goal of keeping good features and discarding bad features.

For this study, the quality of the features was evaluated in two ways. The first way was through descriptive statistics and box plot visualization (introduced by McGill et al., 1978). The second way was through utilizing feature selection algorithms. Descriptive statistics and box plot visualization of features was only conducted on our own collected data set. Reconducting feature evaluation with the new data set was avoided to reduce chances of overfitting the training data, as discussed in Section 3.1. Feature selection, on the other hand, was reconducted with the Cambridge English Readability Dataset.

3.2.2.1 Box plot visualization and descriptive statistics

The first way the predictive ability of the features was evaluated is through comparing descriptive statistics and boxplot visualizations for different CEFR levels. Both visualization and descriptive statistics evaluate the goodness of a feature in the same way; a good feature will show clear and preferably linear differences between the different CEFR levels. In other words, features that either consistently increase or decrease with change in CEFR level are considered good features.

Figure 3 shows box plots of the average sentence length (*ASL*) feature across all CEFR levels. A clear linear relationship can be observed, which makes it a meaningful feature. Figure 4 on the other hand shows the performance of the *ABVMEAN* feature which is not linear. This does not make the feature unusable, as some classifiers such as SVMR can utilize non-linear features, but most classifiers will interpret such data as noise, which can negatively affect the performance of the classifiers. The standard deviation of the features was also analyzed by looking at the height of the box plot. A lower standard deviation results in shorter boxes, which indicates that the differences between observations of the same class are small. Lower standard deviation, in conjunction with linear increase or decrease of the

explanatory variable, reduces overlap of values between observations from different CEFR levels, which should result in a more meaningful feature.

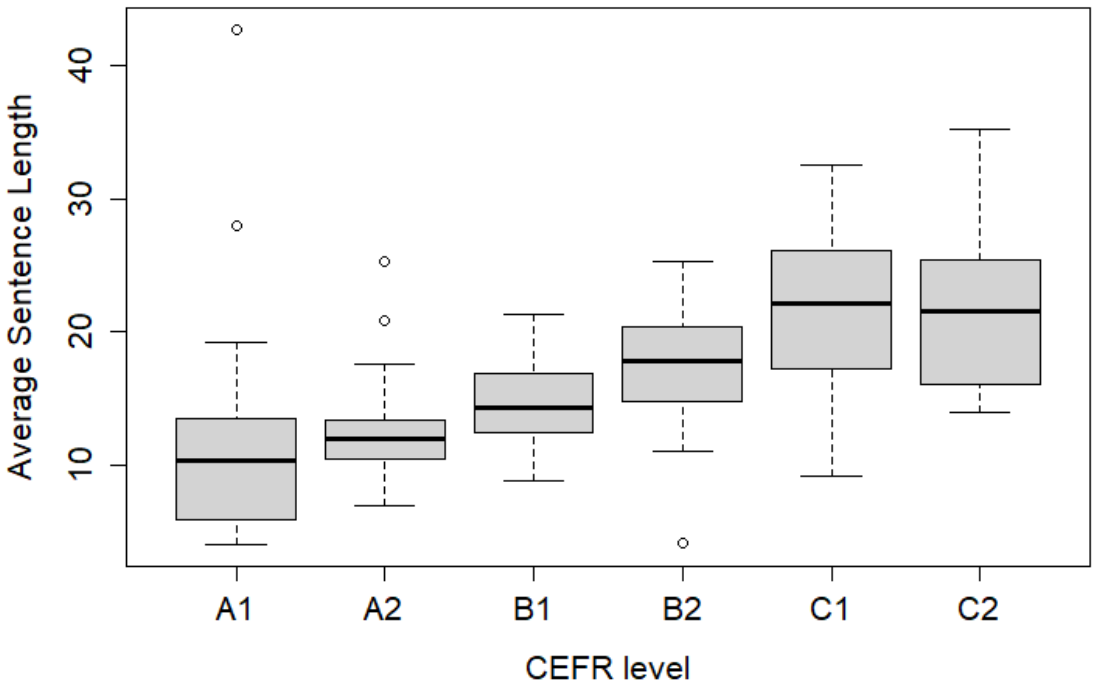

Figure 3: Example of a good feature: average sentence length

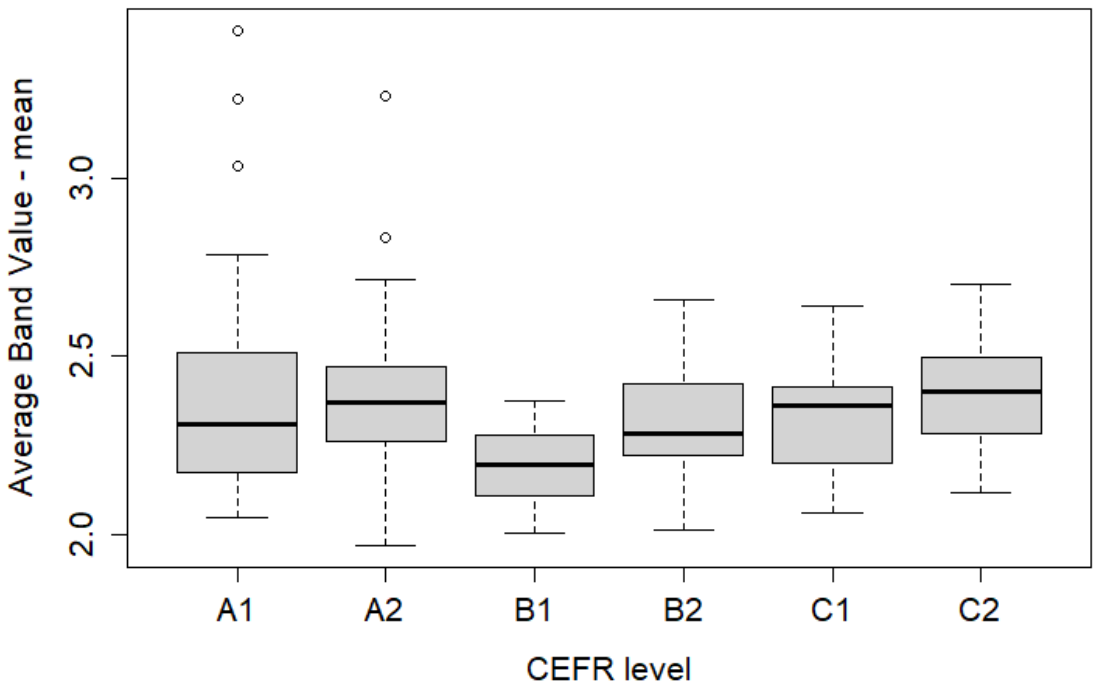

Figure 4: Example of a bad feature: average band value - mean

Because visual interpretation is sometimes unreliable, the mean and standard deviation of each CEFR class was also analyzed separately. This in theory is exactly the same as looking at the box plot visualizations, however it proves to be more reliable, especially in regard to the standard deviation of the classifiers. Similar to using box plot visualization, the goal is to find large differences between the CEFR levels while maintaining low standard deviation for each CEFR level.

3.2.2.2 Feature selection

In addition to box plot visualization and descriptive statistics, we utilized feature selection algorithms to identify the most useful features. Feature selection, similar to dimensionality reduction, aims to reduce the number of input variables (Brownlee, 2020: 113). This is useful because it can significantly reduce computational costs, and in some cases even improve the performance of the classification model (Brownlee, 2020: 111). Improvement in classification accuracy by removing redundant features for SVM classifiers has also been demonstrated by Gabrilovich and Markovitch (2004: 6-7).

There are several ways of applying feature selection to a classifier, depending on the input and output variables and their types (e.g. numerical labels and categorical labels). This study utilizes two different feature selection techniques. These are ANOVA F-test feature selection and correlation score feature selection. Theoretically, results from the ANOVA F-score should be used for classification models (SVM and KNN models), and results from the correlation score feature selection should be used for regression models (LR). However, to allow for a more direct comparison across all classifiers, results from both feature selection methods were combined to develop the same feature sets for all machine learning approaches. Brownlee (2020: 152) warns that due to the stochastic nature of feature selection methods, the results differ every time the algorithm is run. Thus in accordance with his suggestion, feature selection was conducted multiple times to confirm that the results are consistent.

ANOVA F-test feature selection: Because we are dealing with numerical input values and categorical output values (CEFR levels), we can utilize Analysis of Variance (ANOVA) to

identify and remove features that are independent of the target variable. For this task, Scikit-learn's *f_classif* package was used, which is suggested by Brownlee (2020: 140) for situations where we deal with numerical input variables and categorical output variables. The F-score for each feature can be seen in Figure 5. Note that in scikit-learn, higher F-score values translate to lower p-values, and thus generally to more meaningful features.

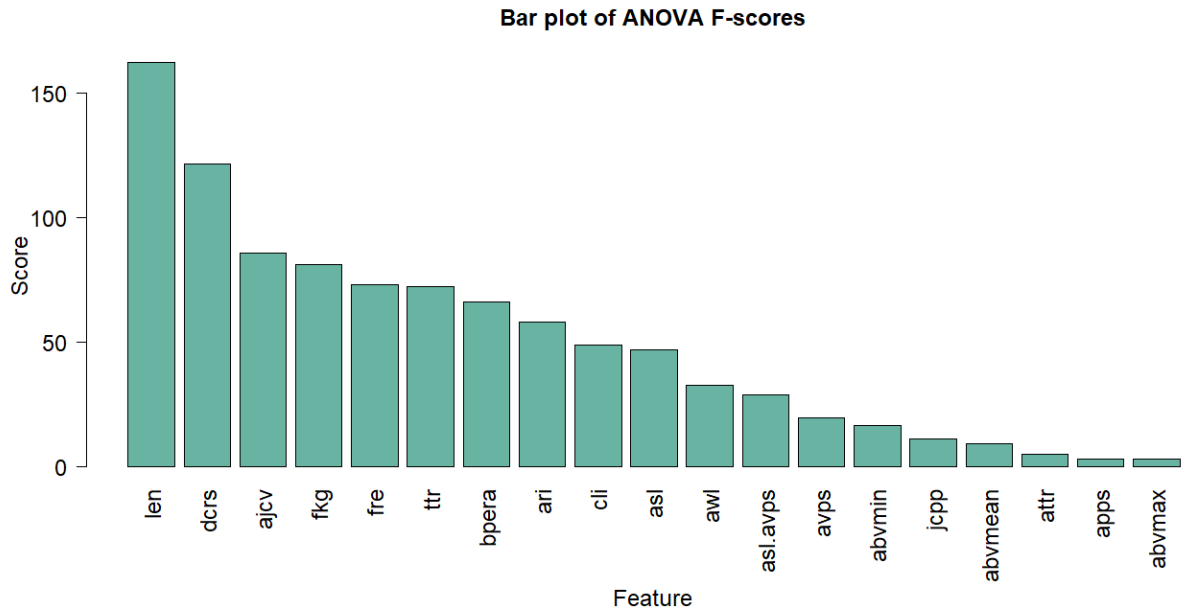

Figure 5: ANOVA F-test feature selection score for each feature

Correlation score feature selection: Correlation is a measure of how two variables change together. If we look at CEFR levels as numerical values rather than categorical ones (ranging from 0 to 5), we can use linear regression and correlation scores to find the relevance of each feature. For this task, Scikit-learn's *f_regression* package was used, which is suggested by Brownlee (2020: 151) for situations where we deal with numerical input variables and numerical output variables. The correlation between two variables is calculated as follows[7]:

$$F_i = \frac{(x_{ij} - \bar{x}_i).(y_j - \bar{y})}{\sigma_{x,j}.\sigma_y} \quad (9)$$

7 Scikit-learn documentation, *f-regression*: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.f_regression.html

where x refers to our features, y refers to our predictions (the classifier's prediction of the CEFR level in our case), σ refers to standard deviation, i refers to the specific feature we wish to analyze, and j refers to the value for a specific observation. The result of Equation (9) is an F-score for each feature which shows how well each feature and CEFR level change together (how well x and y change together). The F-scores can be converted to p-values if needed. The correlation scores in Figure 6 show the contribution of each feature to the prediction of CEFR levels, and similar to ANOVA F-test, higher values correspond to better predictive ability.

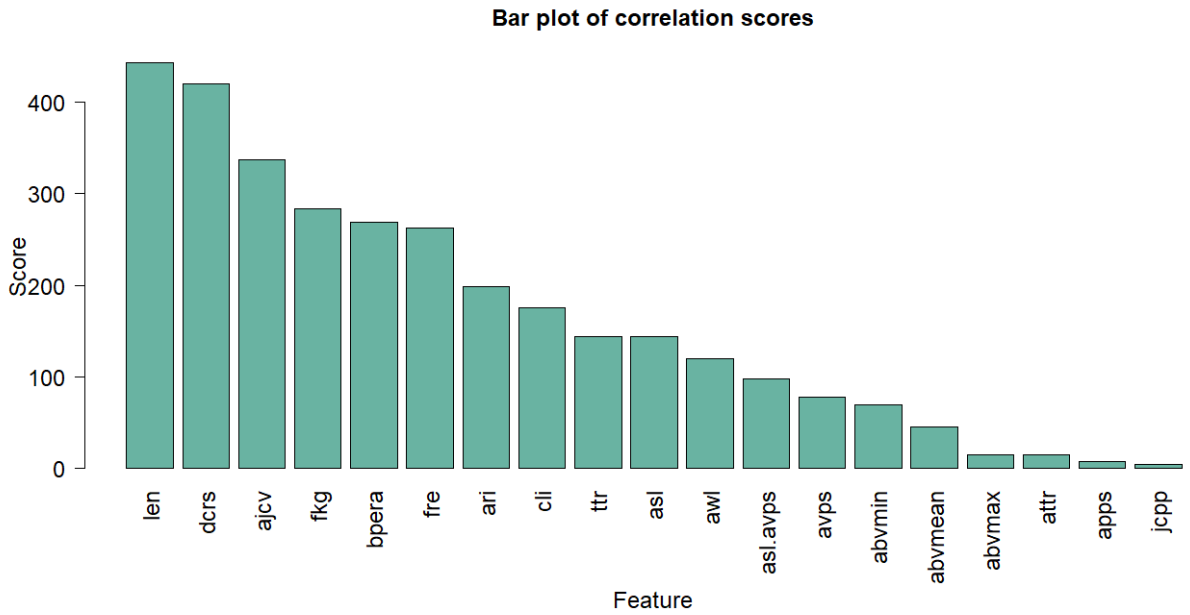

Figure 6: Correlation score feature selection score for each feature

We can see that while the ranking of some features differ for ANOVA F-test and Correlation score feature selection, they direct us to the same conclusion. This further justifies our choice to use the exact same feature sets for both classification and regression tasks. Both feature selection algorithms show that features such as *LEN*, *AJCV*, and *BPERA* are very meaningful, while features such as *TTR*, *ASL*, *AWL*, *ASL.AVPS*, and *AVPS* have sufficient predictive power. Feature selection also shows that many of the new features introduced in this study, namely *JCPP*, *ATTR*, *ABVMAX*, *ABVMEAN*, and *ABVMAX*, perform poorly. The only exception is *ASL.AVPS* which outperforms *AVPS*, but not *ASL*. To our surprise, *APPS* appears to be a relatively weak feature despite being used in previous readability assessment research. The original research (Velleman & van der Geest, 2014) utilized this feature for the Dutch

language, which may indicate that *APPS* is a language specific feature that does not perform well for the English language.

Additionally, our conducted feature selections also support results from Feng (2010) that readability formulas have good predictive ability over readability levels, even for CEFR rated texts. All readability formulas rank relatively high in feature selection, with formulas such as *DCRS*, *FKG*, and *FRE* performing slightly better than *ARI* and *CLI*. The fact that *DCRS* is the best performing readability formula is not surprising, since due to its usage of a word list approach, it is a more direct measure of lexical knowledge. Other readability formulas make use of proxy variables, which are indirect measures of text characteristics.

Feature selection models can be used in *kbest* models (from the Scikit-learn library) to select the k most meaningful features (with the highest F-scores). Instead of using *kbest* models, whose results are difficult to interpret, it was decided to use results from feature selection to settle on meaningful feature sets. This process allowed for a better understanding of the performance of each feature and enabled us to use the same feature set for both classification (KNN and SVM) and regression tasks (LR).

3.2.3 Feature sets

Some previous research, such as Forti et al. (2020: 7210), make use of recursive feature elimination[8] to settle on the best combination of features. However, running these experiments on a personal computer would be very slow, and would require the use of external computers for calculation. One main objective of this thesis is to avoid calculating all possible combinations to reduce computational costs and time. Thus, a more qualitative, yet still data driven, approach was taken and the feature sets were developed by analyzing the feature evaluation measures introduced in Section 3.2.2. In the end, six different feature sets were developed for this study. These feature sets are described below. Table 14 (Appendix A) was created in addition to the description below to allow for better understanding and

8 The recursive feature elimination was developed by Guyon et al. (2002) and attempts to recursively remove the weakest feature in the feature set until a predefined number of features is achieved.

comparison of features in each feature set. The plus sign (+) in Table 14 indicates the presence of a specific feature in the feature set.

1. **Feature set 1 (FS1):** This feature set uses all 19 features measured by our custom feature extractor. These features are *ABVMAX*, *ABVMEAN*, *ABVMIN*, *AJCV*, *APPS*, *ARI*, *ASL*, *ASL.AVPS*, *ATTR*, *AVPS*, *AWL*, *BPERA*, *CLI*, *DCRS*, *FKG*, *FRE*, *JCPP*, *LEN*, and *TTR*. This is the largest feature set, and thus contains the most amount of information. Some classifiers may perform best using this feature set, however because the contribution of some features is minimal, some machine learning approaches (especially LR) should perform better using the reduced feature set (FS2). Overall, it is expected that reducing the size of the feature set should maintain a similar performance.

2. **Feature set 2 (FS2):** This feature set reduces the number of features from 19 to 13 based on the performance of the features in feature selection. Feature set 2 consists of *AJCV*, *ARI*, *ASL*, *ASL.AVPS*, *AVPS*, *AWL*, *BPERA*, *CLI*, *DCRS*, *FKG*, *FRE*, *LEN*, and *TTR*. This feature set is expected to produce results similar to feature set 1, and could even show reduced variance. It is possible to further reduce this feature set by removing overlapping features, such as reducing the number of readability formulas, however this would require training and testing significantly more classifiers to arrive at an optimal feature set.

3. **Feature set 3 (FS3):** This feature set makes use of three non-language-specific features. These are total text length (*LEN*), average word length (*AWL*), and average sentence length (*ASL*). *LEN* serves as a general feature encompassing a number of different text aspects, *AWL* serves as a proxy variable for lexical difficulty, and *ASL* serves as a proxy variable for syntactic and grammatical difficulty. Some thought was given to using *AJCV* instead of *AWL* to measure lexical difficulty. This would have resulted in an increased performance of the classifiers, however by using *AWL* instead, we can get a good idea of the performance of non-language-specific feature sets. While not a direct goal of this MA thesis, extracting results for non-language-

specific features can be useful, especially if I plan to expand this research in the future.

Feature sets 4, 5, and 6 are the same as feature sets 1, 2 and 3 respectively, with the total text length (*LEN*) feature removed. While the *LEN* feature is a very powerful predictor of CEFR levels, including the *LEN* feature limits the functionality of the classifiers to the classification of full length texts. If classifiers show adequate results using Feature sets 4, 5, and 6, they can also be used for the classification of partial and non-complete texts.

4. **Feature set 4 (FS4):** This feature set is identical to feature set 1, with the exception of the *LEN* feature being removed. More specifically, it consists of the following 18 features: *ABVMAX*, *ABVMEAN*, *ABVMIN*, *AJCV*, *APPS*, *ARI*, *ASL*, *ASL.AVPS*, *ATTR*, *AVPS*, *AWL*, *BPERA*, *CLI*, *DCRS*, *FKG*, *FRE*, *JCPP*, and *TTR*. This feature set should perform significantly worse than its counterpart feature set which includes the *LEN* feature (FS1). However, it should slightly outperform its reduced feature set (FS5).

5. **Feature set 5 (FS5):** This feature set is identical to feature set 2, with the exception of the *LEN* feature being removed. More specifically, it consists of the following 12 features: *AJCV*, *ARI*, *ASL*, *ASL.AVPS*, *AVPS*, *AWL*, *BPERA*, *CLI*, *DCRS*, *FKG*, *FRE*, and *TTR*. High performance of this feature set would be very valuable, as it contains a small number of features and excludes the *LEN* feature. However, considering the predictive ability of the *LEN* feature, a massive drop in performance is to be expected when compared to FS2.

6. **Feature set 6 (FS6):** This feature set is identical to feature set 3, with the exception of the *LEN* feature being removed. This feature set serves as an absolute minimum non-language-specific feature set. In fact the original idea for this MA thesis was to build a classifier that uses one simple lexical feature and one simple syntactic feature. However, after studying the literature carefully, it was concluded that such a classifier would perform relatively poorly. If this feature set were to perform moderately well, it could serve in an abundant number of applications, due to its minimal non-language-specific nature. However, in reality this feature set is expected to perform poorly.

In this study, the performance of the feature sets is analyzed in two ways. Firstly, feature reduction is considered in isolation. This is done by comparing the performance of each feature set within the FS1 – FS2 – FS3 group, and the FS4 – FS5 – FS6 group. A drop in performance is expected every time the feature set is reduced. Secondly, the effect of the length feature is evaluated. This is done by analyzing the drop in performance from FS1 to FS4, FS2 to FS5, and FS3 to FS6. Removing the *LEN* feature should result in significantly lower performance of the classifiers.

3.3 Machine learning approaches

According to Collins-Thompson (2014: 12), the task of predicting the readability of a text can be viewed in three different ways. These are readability as a classification task, readability as a regression problem, and readability as a ranking problem. Readability as a ranking problem is relatively rare, however it has been attempted by some studies such as Pitler & Nenkova (2008) and Tanaka-Ishii et al. (2010). Readability as regression and classification, on the other hand, are relatively common. Three different machine learning approaches were explored in this study, namely k-nearest neighbors (KNN), linear regression (LR), and support vector machines (SVM). Among these approaches, LR falls under regression methods, while KNN and SVM fall under classification methods. Other machine learning approaches have also been used previously for readability assessment. These include decision trees and random forests which were explored in Forti et al. (2019) and Milani et al. (2019). The performance of our feature sets using these machine learning approaches was not explored in this study.

3.3.1 Classification approaches

The following section will cover the machine learning approaches explored in this study, general information about these methods, the choice of machine learning approaches, as well as our expectations for their performance. Since these machine learning approaches are used for the task of classification, they can also be referred to as classification approaches.

Understanding how each classification approach works and what adjustable hyperparameters they have is crucial for creating classifiers that perform well. Initial experiments with the data showed that poorly tuned classifiers can significantly underperform, and in some cases were not significantly better than randomly guessing (i.e. an expected value close to 0.2 for five classes). In this study, five classifiers, using three different machine learning approaches, were tested. These machine learning approaches are k-nearest neighbors (KNN), linear regression (LR), and support vector machines (SVM). All of these models were implemented using Python's Scikit-learn library (Pedregosa et al., 2011).

1. **K-nearest neighbors (KNN):** According to Silverman and Jones (1989: 233-234), the concept for k nearest neighbors was introduced by Fix and Hodges in 1951, and the method has been in wide use for classification tasks ever since. KNN is a classification approach that compares a new observation to already existing data, and classifies it based on its similarity to nearby observations (Manning & Schutz, 1999: 604). The k in KNN is a tunable parameter, which gives the k closest (most similar) observations voting power to determine the class of the new observation (Bijalwan et al., 2014: 67). In Scikit-learn, the influence of the nearest neighbors can be implemented in two ways. The first way is for all observations to have the same voting power, regardless of their distance to the new observation. The second way is to base the voting power on the distance to the new observations, so that neighbors closer to the new observation have higher influence. In this study, we use the first method which is the default setting in scikit-learn.

   To determine which k observations are closest to the new observation, and thus have voting power to decide the class of the new observation, a distance function is used to compare the vectors of all known data points to the new one (Bijalwan et al., 2014: 67). This distance is generally measured through the Minkowski distance and is calculated as follows (Merigó & Casanovas, 2011: 124):

$$Minkowski\ Distance = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}} \quad (10)$$

The Minkowski distance is a general distance function that can be modified to calculate different kinds of distances, which is achieved by changing the parameter value for *p*. By selecting *p=1*, we arrive at the Manhattan distance (Lubis et al., 2020: 328), shown in Equation (11), and by selecting *p=2*, we arrive at the Euclidean distance (Merigó & Casanovas, 2011: 124), shown in Equation (12). For this study, *p=2* was selected, which is the default value in the Scikit-learn library[9] and the standard distance according to Khamar (2013: 1918).

$$Manhattan\ Distance = \sum_{i=1}^{n} |x_i - y_i| \quad (11)$$

$$Euclidean\ Distance = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2} \quad (12)$$

KNN inherently supports multi-class classification, making it a suitable machine learning approach for CEFR level assessment. The performance of KNN classifiers has been shown in Wang & Zhao (2012) and Shah et al. (2020). However, according to Guo et al. (2013: 1) there are two main disadvantages of KNN. The first disadvantage is that the performance of the classifier heavily depends on choosing the right value for the hyperparameter *k*. To remedy this problem we will utilize model selection, the process of testing different parameters for a classifier, to arrive at the optimal value for *k*. The second problem with KNN is that it can be somewhat slow due to comparing all data points. However, because the data set for this study is relatively small (331 observations), this will not be a problem.

Experiments with our original corpus of 177 observations showed relatively low performance for KNN classifiers, however our original corpus suffered from two flaws. Firstly, the corpus consisted of only 177 texts, which is relatively small. Secondly, KNN assumes equal distribution of classes in the data set (Tan, 2005: 290), and our original data set did not satisfy this condition. Both of these factors negatively affected the performance of the KNN classifiers. We expect KNN classifiers to

9 Scikit-learn documentation, *neighbors.KNeighborsClassifier*: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

perform better after switching to the Cambridge English Readability Dataset, as the data set does not suffer from the above mentioned shortcomings.

2. **Linear regression (LR):** Linear regression is one of the earliest forms of machine learning and was introduced in 1894 by Sir Francis Galton (Maulud & Abdulazeez, 2020: 140). Linear regression is a statistical technique for mapping the relationship between variables, more specifically between one response variable and one or multiple explanatory variables (Montgomery et al., 2021: 1). The relationship between the explanatory variables and the response variable is represented in the following form (Matloff, 2017: 65-67):

$$y = \beta_0 + \beta_1 x_1 + ... + \beta_m x_m + \varepsilon \quad (13)$$

where y refers to the value of our response variable (CEFR level in our case), m refers to the number of explanatory variables (number of features in our case), and x refers to the value of the explanatory variable (e.g. *ASL* or *AWL* of a text). β refers to the amount of change in the dependent variable for one unit of change in the independent variable, and is determined using linear regression. ε is the error term, which represents the distance between our prediction and the actual value.

Linear regression with one explanatory variable is referred to as simple linear regression, while using more than one explanatory variable is referred to as multiple or multivariate linear regression (Montgomery et al., 2021: 2-4). Since our smallest feature set (FS6) consists of two features, we only utilize multiple linear regression (MLR), however throughout this paper we will refer to our models as simply LR.

Linear regression is not able to directly perform CEFR level assessment, however by transforming the labels into scale values (i.e. by transforming A2–C2 labels into numbers ranging from 1 to 5), we transform the problem of classification into a regression task, which linear regression excels at. Additionally, linear regression will always produce a continuous output rather than a discrete one. To get around this problem, we simply round the classifier's output to the closest whole number. Thus, a

prediction of 2.47 is rounded to 2, which is interpreted as the B1 CEFR level. The same rounding approach was employed by Petersen and Ostendorf (2007: 93) regarding the grade level of texts.

3. **Support vector machines (SVM):** Support Vector Machines were first introduced by Vladimir Vapnik (1995) and have since been used in various NLP and non-NLP classification tasks. SVMs discriminate between classes by defining a hyperplane[10] that separates the data into two classes (Boswell, 2002: 1). This separation is done at a point where maximum distance between the two classes is achieved, which is known as the maximum margin hyperplane. In addition to the maximum margin hyperplane, SVMs allow for misclassification inside the training data, which is known as the soft margin (denoted by *C*). High values for *C* increase the penalty assigned to errors, while low values of *C* decrease this penalty (Ben-Hur & Weston, 2010: 230-231). This in practice means that higher values of *C* correspond to the classifier placing more importance on correctly classifying all the data in the training set (Boswell, 2002: 5).

   SVM does not inherently support multi-class classification, however it can handle such tasks by splitting the task into several bi-class classification problems. Additionally, SVMs appear to be the most successful machine learning approach in grade level assessment and CEFR level assessment. This is supported by the success of classifiers built by Schwarm & Ostendorf (2005), Petersen & Ostendorf (2009), Vajjala & Loo (2014), and Forti et al. (2020).

   One additional benefit of SVMs is that they can utilize non-linear data through the use of kernels. According to Boswell (2002: 5-6), data is often not linearly separable into classes. In such situations, we can utilize kernels to 'preprocess' the data into a higher dimension so that the problem is transformed into finding a simple hyperplane. In this study, in addition to the base SVM, known as linear SVM (SVML), the performance

10 Considering a space with N dimensions, a hyperplane is a subspace with dimensions N-1. Thus, the hyperplane of a two dimensional space is a one dimensional line, and the hyperplane of a three dimensional space is a two dimensional plane. The same concept can be applied to spaces with over three dimensions, however higher dimensions are not visualizable and can only be calculated using linear algebra.

of the polynomial kernel (SVMP) and the RBF kernel (SVMR) was tested. These kernel transformations are explained below.

(a) **SVM with the linear kernel (SVML):** The linear kernel calculates the high-dimensional relationships between observations without transforming the data. SVML has a single tunable parameter, namely the soft margin parameter (*C*). This makes training a SVML quite easy. In fact, initial experiments showed that SVML classifiers outperformed other SVM classifiers due to their simplicity. Section 2.2.2 already explained why simpler models can perform better on unseen data.

(b) **SVM with the polynomial kernel (SVMP):** According to Chang et al. (2010: 1472), the polynomial kernel is extremely popular for many NLP tasks. This makes it a suitable kernel to explore for our study. The polynomial kernel works by computing the high-dimensional relationships between observations and comparing them to each other. The high dimensional relationship between two observations is calculated as follows (Boswell, 2002: 7):

$$K(x_a, x_b) = (x_a . x_b + r)^d \quad (14)$$

where x_a and x_b refer to the observations we want to calculate the high dimensional relationship for, r determines the polynomial's coefficient, and d determines the degree of the polynomial. Both r and d are tunable parameters. According to Ben-Hur et al. (2008: 6), the linear SVM is in fact an SVMP with the degree (d) of 1. For our experiments, we only tuned the parameters d and C, while r was set to zero. This is called a homogeneous polynomial kernel (Shashua, 2008: 37) and is the default setting in the Scikit-learn library.

(c) **SVM with the RBF kernel (SVMR):** The Radial Basis Function (RBF) kernel, also known as the Radial kernel, works by computing the high dimensional relationships between observations and comparing them to each other. The high dimensional relationship between two observations is calculated using the following formula (Prajapati & Patle, 2010: 513):

$$K(x_a, x_b) = e^{-\gamma(X_a - X_b)^2} \quad (15)$$

where e refers to the exponential function, x_a and x_b refer to the observations we want to calculate the high dimensional relationship for, and γ determines the influence of the observations. γ is a tunable parameter, which is determined using cross validation.

The main disadvantage of SVM classifiers is that they require tuning specific hyperparameters, such as C for all SVM models, d and r for the polynomial kernel, and γ for the RBF kernel. This can be relatively time consuming and challenging with limited experience. Additionally, a lack of experience can also cause the classifiers to overfit the data set (Boswell, 2002: 9), which is a risk we encounter by using SVM models.

3.3.2 Tuning the hyperparameters of the classifiers

As mentioned earlier, properly tuning the parameters of the classifiers is essential for training reliable machine learning models. In this study, the hyperparameters of the classifiers were determined using cross validation model selection. This means that different models with differing hyperparameters were trained and tested. The best performing models were then chosen, and their parameters were selected as the final parameter values. Cross validation experiments were conducted using all six feature sets to ensure that the parameters work well with all of our feature sets. I believe that the chosen hyperparameters are relatively close to the ideal values. This claim is supported by our results in Chapter 4 which show close to state-of-the-art performance for the best performing classifiers. Further fine-tuning the parameters of the classifiers is certainly possible and could lead to slightly better results, however the risk of the classifier overfitting the corpus might become even greater in that case. One additional thing should be noted. The hyperparameters of the classifiers were adjusted using their accuracy and one-off accuracy. This is also the reason why Experiment 1 (Section 4.1) focuses on reporting only the exact and one-off accuracy scores.

1. **K-nearest neighbors (KNN):** KNN has a single adjustable hyperparameter, namely *k*. KNN classifiers with *k* ranging from 3 to 21 (with a step of two) were tested. In our experiments, *k=5* performed best and was chosen as the parameter value for *k*.

2. **Linear Regression (LR):** Linear regression does not have any adjustable hyperparameters. The only factor affecting the performance of LR models is the choice of features.

3. **SVM with linear kernel (SVML):** SVML only has a single tunable hyperparameter, namely the soft margin parameter *C,* which all SVM classifiers have. $C=10^n$ with *n* ranging from -5 to 3 (with a step of 1) was tested for the SVML classifiers. Experimenting with higher values of *C* was not possible due to an increase in training and testing session duration as a result of higher values of *C*. In our experiments, *C=1000* performed best and was chosen for the parameter of the SVML classifiers.

4. **SVM with polynomial kernel (SVMP):** The same range of 10^{-5} to 10^3 was explored for the soft margin parameter of this classifier. In addition to the soft margin parameter *C*, SVMP has two other adjustable hyperparameters, namely the degree of the polynomial (*d*) and the polynomial's coefficient (*r*). The polynomial's coefficient *r* was left at its default value (*r=0)*. The degree of the polynomial *d*, was assigned values ranging from 2 to 6 (with a step of 1). According to Boswell (2002: 8), while higher values for *d* allow for easier separation of classes (due to the feature space being implicitly larger), doing so could lead to overfitting, and thus worse generalizations. Additionally, initial experiments with the *d* parameter showed that training and testing models with higher values of *d* was much slower, especially with lower number of features. Feature set 6 which only contained two features even failed to converge. Thus, in contrast to all other SVMP classifiers, which were set to *C=1000* and *d=6*, SVMP-FS6[11] was given the parameters *C=1000* and *d=4* to ensure that the model would converge.

11 SVMP-FS6 refers to the SVMP classifier trained and tested using FS6 (feature set 6). This notation is used throughout this study when discussing specific classifiers, and it indicates a specific combination of classification approach and feature set.

5. **SVM with RBF kernel (SVMR):** Similar to the previous two SVM classifiers, the same range of 10^{-5} to 10^{3} was explored for the soft margin parameter of the SVMR classifier. Different values were also experimented for γ, however setting γ to *'scale'* worked best. In the Scikit-learn library[12], setting γ to *'scale'* will automatically use Equation (16) to calculate the value for γ. Note that γ will be different for every training/testing session due to the way it is calculated.

$$\gamma = \frac{1}{(number\ of\ \ features \times variance\ of\ the\ feature\ set)} \quad (16)$$

In our experiments, *C=1000* worked best for all three SVM classifiers. This value is relatively high and could cause the classifiers to overfit the training data. Additionally, even though the machine works well with unseen data (testing data), all the data comes from a common source. While not necessarily a problem for this thesis, it could prove that the machine has a relatively difficult time generalizing its knowledge to other kinds of texts, such as non-exam texts. The problem of classifying different types of texts is not explored in this study, however the problem of the classifiers overfitting training data is explored in Section 4.2.4.

In addition to adjusting the hyperparameters of the classifiers, an extra step was added to the LR and SVML classifiers. LR and SVML classifiers view CEFR levels as scales, thus it is possible to get prediction values below 0 for texts that are easier than most A1 texts (these could be identified as pre-A1). It is also possible to get prediction values above 5 (these could be viewed as C2+). For these two classification approaches, predictions below 0 were changed to 0 (A1) and predictions above 5 were changed to 5 (C2).

3.4 Splitting the corpus and measures of evaluation

After the data is collected and labeled, the data set needs to be split into training and testing sets. The following section will discuss how the data was split into training and testing sets,

12 Scikit learn documentation on SVMs: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

k-fold cross validation, which is a special method for splitting the data, as well as the measures of evaluation used in our experiments.

3.4.1 Splitting the corpus into training and testing data

Splitting the data into training and testing data can be done in a number of ways. For Experiment 1 (Section 4.1), the data is randomly split into training and testing sub-data sets and test size is set to 0.2. Test size refers to the percentage of data saved as testing data (validation data set), thus a test size of 0.2 means that 20% of the data will be reserved as testing data and is not shown to the machine prior to testing. Saving a portion of the data for testing means that the classifier is forced to label observations it has never seen before.

For Experiment 1, the random splitting of the data is justified because keeping track of its details would not yield much benefits. Furthermore, the high repetition count of 1000 folds will automatically enable comparison across different classifiers due to resulting in a normal distribution spread. Additionally, stratified sampling was used instead of random sampling. Due to the small sample size of 331 observations, randomly splitting the data into training and testing data could result in an imbalanced distribution of data. In stratified sampling, the distribution of each class is kept the same as in the original data set. This applies to both the training and testing set, which can reduce the bias of the model toward any specific class.

A second way is to preemptively split the data into training/testing sets before conducting any experiments. This can allow for easy reproducibility of experiments, however it would limit the experiments to the comparison of a single training and testing session per classifier. To allow for more training and testing sessions, it was decided to split the data right before training using k-fold cross validation for Experiment 2.

3.4.2 K-fold cross validation

K-fold cross validation is a model validation technique for assessing how a trained model will generalize its knowledge to an independent data set. K-fold cross validation is a powerful method for testing the performance of models, especially classification models (Marcot & Hanea, 2021: 1). This is because the same data set can be used to test a classifier multiple times without overlaps of testing data. According to Marcot and Hanea (2021: 2-3), k-fold cross validation is conducted as follows:

The entire data set with N observations is split into k equal sub-data sets, each with a size of $\frac{N}{k}$. Each sub-data set is used as a testing set (validation set), while the remainder of the data set, with the size of $N-\frac{N}{k}$, is used as the training set. Because there are k testing data sets, validation is conducted k folds/times.

This is a very effective way of using the same data set multiple times to get a better understanding of the performance of the models. The process of k-fold cross validation is shown in Figure 7, where in each fold, the orange set is used as the testing (validation) set, while the remainder of the data set (white sets) is used as the training set. Figure 7 is a modified recreation of the figure available in the scikit-learn documentation. Additionally, k-fold cross validation is proven to be an effective tool to detect the extent to which classifiers overfit training data (Hawkins, 2004: 8). As mentioned in Section 2.2.2, overfitting means that the classifier studies the training data too closely, which can hinder it from generalizing its knowledge to unseen data.

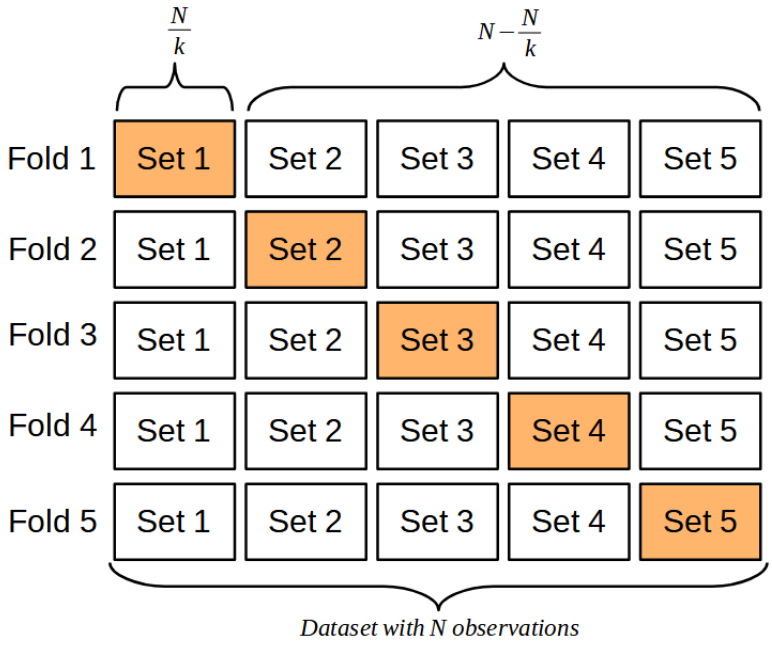

Figure 7: Split of training and testing sets in k-fold cross validation

Error refers to the mean difference in the prediction of the classifier and the actual class of the text. According to Marcot and Hanea (2021: 2), overfitting happens when calibration error rates are low, but cross validation error rates are high. Thus, we first measure the error of the classifier on the training set (calibration error) and the error of the classifier on the testing set (validation error) separately, then proceed to compare the difference in the two types of errors. To measure error for classification tasks, Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are commonly used. According to Chai and Draxler (2014: 1247), both RMSE and MAE are widely used in model evaluation and can show the extent to which our classifiers misclassify the data. Considering n samples of model errors e calculated as (e_i, $i=1, 2, \ldots, n$), the RMSE of the data set is calculated as follows (Chai & Draxler, 2014):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} \quad (17)$$

Using the same notation as above, the MAE of the data set is calculated as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i| \quad (18)$$

While both RMSE and MAE are usable, Chai and Draxler (2014: 1247) argue that RMSE is not a good indicator of the performance of the model and may even provide misleading information about the average error. Even as early as 1982, it has been argued that MAE should generally be preferred to RMSE (Willmott, 1982: 1310). This is because RMSE is more sensitive to extreme values due to the square root function as seen in Equation (17). Thus, we will report the MAE of the classifiers instead of the RMSE for Experiment 2.

Additionally, when using k-fold cross validation, one needs to decide on the value of the parameter *k*, which represents the number of splits. After a series of experiments, Marcot and Hanea (2021: 21) conclude that while *k=10* produces the best results, *k=5* is sufficient in the majority of cases. These two values are the most common values for k, and both were considered for Experiment 2. After considering a number of factors, *k=5* was chosen for this study. This choice is explained in Section 4.2.1.

Furthermore, three repetitions of k-fold cross validation were conducted to assure stability of results. By repeating k-fold cross validation multiple times, we can utilize different splits of data, which can show a clearer picture of the performance of the classifiers. Moreover, to ensure reproducibility of results for Experiment 2, a random state (random seed) is assigned to each split of the data. This is done to ensure a specific split of data is used for training and testing. These seeds are chosen using a random number generator and ensure that our results can be reproduced and checked by other researchers and colleagues. Additionally, similar to Experiment 1, stratified sampling is used instead of normal k-fold cross validation to ensure equal class proportions.

3.4.3 Measures of evaluation

There are various measures that can evaluate the performance of machine learning models. The most commonly used measures are accuracy, precision, recall, and F1-score (Manning &

Schutze, 1999: 268-269). To understand these four measures of performance, we first need to define four terms regarding the type of retrieved information. These are true positive (tp), true negative (tn), false positive (fp), and false negative (fn). Considering a set of information selected from a pool of data which contains a set of target information (i.e. information that we actually want):

tp refers to the data we want and managed to select.
fp refers to the data we do not want, yet managed to select.
tn refers to the data that we do not want and also did not select.
fn refers to the data we want, yet failed to select.

Manning and Schutze, (1999: 268) use a Venn diagram to demonstrate the meaning of these four terms. This simple Venn diagram was recreated for this study (Figure 8).

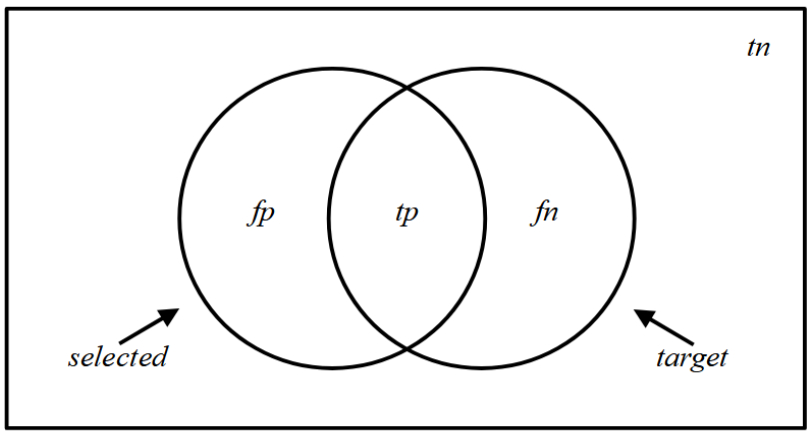

Figure 8: The Venn diagram of tp, fp, tn, and fn

With the help of the above defined terminology, the following section will introduce the measures of evaluation used in this study.

1. **Accuracy:** Accuracy is the ratio of correctly predicted observations to the total number of predictions. Accuracy is calculated as follows:

$$accuracy = \frac{tp}{tp+tn} \quad (19)$$

While accuracy can be very useful, Manning and Schutze (1999: 268) warn that accuracy is not a good measure for problems where the number of non-selected items is extremely large. Despite this warning, we still make use of this measure. Our choice is justified in Section 4.1.1, but in summary this is because using accuracy as a measure is easier to interpret and many of the negative aspects of accuracy have been accounted for in this study.

In addition to accuracy (exact accuracy), this study also reports a one-off accuracy (within-1-level accuracy). In one-off accuracy, misclassifications of one level are regarded as correctly labeled items. Thus, a B1 text labeled as B2 by the machine is considered correctly classified. This modified measure of performance has been used in Petersen and Ostendorf (2009), as well as in Vajjala and Loo (2014).

One-off accuracy can be extremely useful; firstly, it adjusts for the error and inaccuracy of human annotators. Assigning CEFR level to texts is subjective and many evaluators have different definitions and standpoints. This issue has been explored for grade levels by Petersen and Ostendorf (2009: 102). The researchers report that there is a fair amount of disagreement between human annotators for annotating the grade level of texts, even among trained experts. The same problem can be applied to CEFR levels. Secondly, the use of one-off accuracy helps understand the general usefulness of a classifier. An inaccuracy of one level is generally forgivable and can still yield useful results for students and teachers alike. A somewhat similar measure was used by Forti et al. (2020) through the use of macro CEFR levels. This was done by grouping B1/B2 into the B macro level, and C1/C2 into the C macro level.

2. **Precision (prec.):** Precision or confidence is defined as the proportion of relevant information that has been retrieved to the total number of retrieved documents. Precision is calculated as follows:

$$precision = \frac{tp}{tp + fp} \quad (20)$$

According to Powers (2020: 38), precision is the most commonly used measure in machine learning, data mining, and information retrieval.

3. **Recall:** Recall is defined as the proportion of relevant information that has been retrieved to the total number of relevant documents in the database. Recall is calculated as follows:

$$recall = \frac{tp}{tp + fn} \quad (21)$$

According to Powers (2020: 38), recall is widely used in medical scenarios and has even shown some success in computational linguistics, however it is generally neglected or averaged away in many machine learning and computational linguistics applications.

4. **F1-score:** F1-score, introduced by van Rijsbergen (1979: 174), is the weighted average of precision and recall. The most commonly used F1-score measure is calculated as follows:

$$\text{F1-score} = 2 \times \frac{precision \times recall}{precision + recall} \quad (22)$$

The F1-score can serve as a good balance between precision and recall, as it considers both false positives (due to the precision component) and false negatives (due to the recall component).

Chapter 4 discusses two experiments conducted for this thesis. In Experiment 1, only accuracy and one-off accuracy are reported. In contrast, Experiment 2 reports all available measures to allow for better comparison of our classifiers to previous research. Additionally, Experiment 2 will use the *classification_report* function from the Scikit-learn package which in addition to the above mentioned measures, also reports a macro average and a weighted average for the three measures precision, recall, and F1-score separately. According to the

Scikit-learn documentation[13], macro average (macro avg.) refers to the unweighted mean per label (i.e. the proportion of data in each class does not affect the average), while the weighted average (weighted avg.) refers to averaging the support weighted mean per label (i.e. the proportion of data in each class affects the average). Crucially, the macro average F1-score is calculated by taking the mean of the F1-scores across all classes, rather than being calculated from the macro average precision and recall like a normal F1-score.

13 Scikit-learn documentation, *classification_report*: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

4 Experiments and results

Section 3.2.2 discussed feature selection and feature evaluation, while Section 3.3.2 discussed tuning the hyperparameters of the classifiers. These sections are in fact part of our experimental work, however they were included in Chapter 3 to increase the cohesiveness of the thesis. After creating feature sets and tuning the parameters of the classifiers, the next step is to conduct experiments and report the performance of the classifiers. For this study, two experiments were conducted using the Cambridge English Readability Dataset. Experiment 1 is conducted using 1000 folds of training and testing 30 classifiers, which are the combination of five classification approaches and six feature sets. Experiment 2 is conducted using three repetitions of 5-fold cross validation, and compares the best performing classifiers. The setup, goals, expectations, and results of both experiments are explained in the following chapter. Refer to Section 3.3.2 for details on the hyperparameters of the classifiers.

4.1 Experiment 1: 1000-fold training and testing of classifiers

Section 4.1 describes the first out of two experiments that was conducted in this study. Section 4.1.1 describes the experiment setup, and Section 4.1.2 describes the goals and expectations of the experiment. Section 4.1.3 reports the results of the experiment, Section 4.1.4 gives preliminary answers to our research questions, and Section 4.1.5 reports the best performing classifiers in Experiment 1. A similar order of sub-sections is used for Experiment 2 (Section 4.2).

4.1.1 Setup of Experiment 1

To compare the performance of the classifiers, 1000 rounds of training and testing were conducted on each feature set using each of the five machine learning approaches. Stratified sampling was used to ensure equal proportion across all classes for training and testing. A test size of 0.2 (20% testing data) was used to ensure that the results would be similar to

Experiment 2 where sample size equals $\frac{1}{k}$ and k equals 5, also resulting in 20% testing data.

It is important to address the question of why accuracy and one-off accuracy were chosen as performance measures for this experiment. Depending on the functionality of the created tool, different evaluation measures can be used to test a system. Precision and recall, for instance, are pillars of information retrieval (IR), as systems often need to trade off one for the other (Manning & Schutze, 1999: 269). Keeping track of both scores can be essential for IR systems. Of course, it is well accepted that accuracy is not the best measure of evaluation, because it is highly sensitive to error (Manning & Schutze, 1999: 269). However, the choice of using accuracy for Experiment 1 is justified as follows:

Firstly, some scholars such as Shah et al. (2020: 15) believe that accuracy is the most important parameter when implementing a machine learning algorithm on a particular data set. Secondly, accuracy is easier to interpret and allows for a direct comparison to one-off accuracy results, which as justified in Section 3.4.3, is an extremely meaningful measure of performance. Thirdly, since the size of the testing set is relatively small (approximately 66 texts due to the test size of 0.2), differences between accuracy, precision and recall should be minimal. This is because with a small data set, the number of non-selected items is relatively low, and the presence of a large number of non-selected items is the main reason why accuracy performs poorly as a measure (Manning & Schutze, 1999: 269). Additionally, our corpus is relatively balanced, with a similar proportion of observations for each CEFR level (refer to Table 3 for details about the data set). Finally, the results of Experiment 1 will not be reported in isolation and Experiment 2 will complement the accuracy scores by using other measures of evaluation.

All the points mentioned above should negate the downside of using accuracy as a measure of evaluation in Experiment 1. In fact, since this section is written after conducting Experiments 1 and 2, it can be said with certainty that in our particular case, the choice of accuracy was justified. By looking at Tables 11, 12, and 13 (Section 4.2.3), we see that the

average precision, recall, F1-score, and accuracy balance out across classes, which results in minimal difference between the different measures of evaluation.

4.1.2 Goals and expectations of Experiment 1

Both Experiment 1 and 2 try to give answers to our research questions. This section attempts to give expectations on what answers Experiment 1 will provide. In regard to the predictive power of simple features (RQ1), Experiment 1 should provide a relatively satisfactory general answer. This can be extracted by looking at the exact and one-off accuracy scores across all classifiers. High overall accuracy scores should indicate strong predictive power for simple features and low accuracy scores should prove weak predictive power. However, it is possible that specific classifiers significantly underperform on certain classes. Classifiers could, for instance, underperform on the A2 class due to its shorter length, which is a concern brought up by Forti et al. (2020: 7207) as a justification for why A1 and A2 levels were removed from their study. Such issues will be explored in Experiment 2.

With respect to the performance of individual feature sets (RQ2), we can see which feature sets perform best and are more suitable for predicting the CEFR levels of English texts by comparing the mean accuracy scores for the six feature sets. Ideally, we would like feature sets with lower number of features which also exclude the *LEN* feature to perform best. However, a higher number of features should generally result in better performance, and initial experiments have shown that the *LEN* feature is essential for high performance of the classifiers.

In regard to which classification approach performs best using simple features (RQ3), the answer can be extracted by comparing the performance of different classifiers on the same feature sets, as well as by looking at their overall performance across all six feature sets. We can also see whether certain classification approaches struggle with or perform better with specific types of feature sets. LR models could, for instance, perform better after initial dimension reduction (F2 compared to FS1), however they may perform poorly with a limited number of features (i.e. when using FS3 and FS6).

In addition to answering the proposed research questions, Experiment 1 serves as a way to reduce the number of classifiers tested and analyzed in Experiment 2. This is because reporting more measures of performance for all 30 classifiers would be challenging and results would be difficult to interpret.

4.1.3 Results of Experiment 1

For Experiment 1, thirty classifiers were trained and tested 1000 times using six feature sets and five machine learning approaches. The result of the performance of these classifiers over 1000 training/testing iterations has been compiled into Table 7 for accuracy and Table 8 for one-off accuracy. The two best performing classifiers for each feature set are marked in bold font. The full results for Experiment 1, which contains accuracy and one-off accuracy scores for all 1000 training/testing sessions of the 30 classifiers, can be accessed using the link provided in Appendix B.

Table 7: Mean performance of the 30 classifiers in Experiment 1 (exact accuracy)

Feature Set (FS)	FS1	FS2	FS3	FS4	FS5	FS6
Number of features	19	13	3	18	12	2
KNN	**72.4**	**71.4**	**68.7**	43.6	44.0	38.8
LR	62.3	62.9	56.2	51.3	48.0	35.6
SVML	68.6	68.2	61.1	53.0	51.6	39.8
SVMP	70.1	67.9	63.5	**59.4**	**58.6**	**42.6**
SVMR	**73.6**	**72.8**	**66.9**	**56.3**	**56.7**	**40.9**

Table 8: Mean performance of the 30 classifiers in Experiment 1 (one-off accuracy)

Feature Set (FS)	FS1	FS2	FS3	FS4	FS5	FS6
Number of features	19	13	3	18	12	2
KNN	93.1	92.8	92.9	82.0	78.6	72.7
LR	**98.0**	**98.1**	93.4	**97.7**	**97.2**	**84.7**
SVML	**95.2**	**94.7**	**93.8**	**94.1**	**93.0**	**82.2**
SVMP	93.1	92.9	91.6	87.1	87.7	76.2
SVMR	95.0	94.6	**93.7**	86.8	86.5	75.3

Overall, we see relatively good performance for our classifiers, especially those using FS1 and FS2. FS3 and FS5 also perform relatively well, however FS6 performs very poorly. We also observe that SVM models outperform other classification approaches, especially SVM with the RBF kernel (SVMR). LR classifiers, on the other hand, perform significantly worse than other classification approaches. Section 4.1.4 will analyze the results in detail.

Additionally, the standard deviations of the classifiers were recorded and compared. This is the standard deviation within the 1000-fold training and testing, thus it is measured separately for each classifier. The standard deviations of the classifiers ranges from 4.1% to 5.7% for exact accuracy and 1.5% to 4.9% for one-off accuracy. Such values are relatively high and suggest that more data is needed to achieve consistent results. They also show that there are possible outliers in the data set, which affect the results of the experiment when they emerge in the testing data. Because removing outliers from samples with small sizes is somewhat challenging (Van Selst & Jolicoeur, 1994: 632), the problem is best addressed by training and testing the classifiers on a larger data set to either reduce the effect of outliers, or to facilitate the detection and removal of outliers.

Moreover, when creating FS2, we hypothesized that classifiers trained on this feature set would be more stable, and thus should show a smaller standard deviation. The results of the experiment show that this is not the case and no clear difference in the standard deviation can be observed.

4.1.4 Linking results to research questions

In the following section, results from Experiment 1 will be used to give preliminary answers to the research questions of this study. These will be complemented with results from Experiment 2 to give more complete answers to our research questions.

RQ1: How powerful are simple features for predicting the CEFR level of English texts?

Looking at the results in Tables 7 and 8, we see that there is a wide range of performance across the 30 classifiers. The best performing classifier (SVMR-FS1) reaches an accuracy of 73.6, while the worst performing classifier (LR-FS6) goes as low as 35.6. However, if we do not consider the performance of classifiers which use FS6, all other classifiers perform at a mean accuracy of 61.2. Considering the performance of other readability assessment tools (Table 2) and our use of only simple features, this level of performance is relatively respectable.

Furthermore, our best classifiers SVMR-FS1 and KNN-FS1 achieve a performance of 73.6 and 72.4 respectively, which is comparable to the state-of-the-art classifiers. These classifiers achieve this performance by using support vector machines and k-nearest neighbors models, which are completely different machine learning approaches. This shows that simple features can be very powerful predictors of the CEFR level of English texts, and they can perform well using different classification approaches. The performance of simple features will be further explored in Experiment 2.

RQ2: What minimal combination of simple features produces the best results for predicting the CEFR level of English texts?

The mean and standard deviation of the performance of the feature sets across five classification approaches has been recorded in Table 9. This serves as a reference for the performance of each feature set.

Table 9: Performance of the feature sets across five classification approaches

Feature set	FS1	FS2	FS3	FS4	FS5	FS6	Mean
Number of features	19	13	3	18	12	3	-
Mean	69.4	68.6	63.3	52.7	51.8	39.5	57.6
Standard deviation	4.4	3.8	4.9	6.0	6.0	2.6	4.6

Unsurprisingly, FS6 performs poorly with a mean accuracy of 39.5 and is not suitable for the task of CEFR level assessment. FS4 and FS5 perform relatively well with a mean accuracy of 52.7 and 51.8 respectively. These feature sets can be used for CEFR level classification, however their counterparts which include the *LEN* feature (FS1 and FS2) perform significantly better. FS3 performs relatively well and reaches a mean performance of 63.3. This is surprising as FS3 only contains three non-language-specific features, which shows that even a small number of meaningful simple features can have adequate predictive power over the CEFR level of English texts. However, this high performance is primarily due to the *LEN* feature, as FS6 with the same features, excluding the *LEN* feature, performed significantly worse. Finally, FS1 and FS2 perform best, reaching a mean performance of 69.4 and 68.6 respectively.

To get a better idea of the performance of our feature sets, the performance of all 30 classifiers is plotted in Figure 9, which consists of five different connected scatter plots. Blue dots indicate feature sets including the *LEN* feature (FS1, FS2, and FS3 left to right). Red dots indicate the counterpart feature sets that do not include the *LEN* feature (FS4, FS5, and FS6 left to right). Figure 9 allows us to see the effect of feature reduction, by observing change in the x-axis. As a general trend, a reduction in number of features corresponds to a drop in accuracy. Additionally, Figure 9 allows us to analyze the effect of removing the *LEN* feature by comparing the performance of the feature sets along the y-axis. Again, as a general trend, removing the *LEN* feature results in a drop in accuracy, however this drop is more significant than feature reduction.

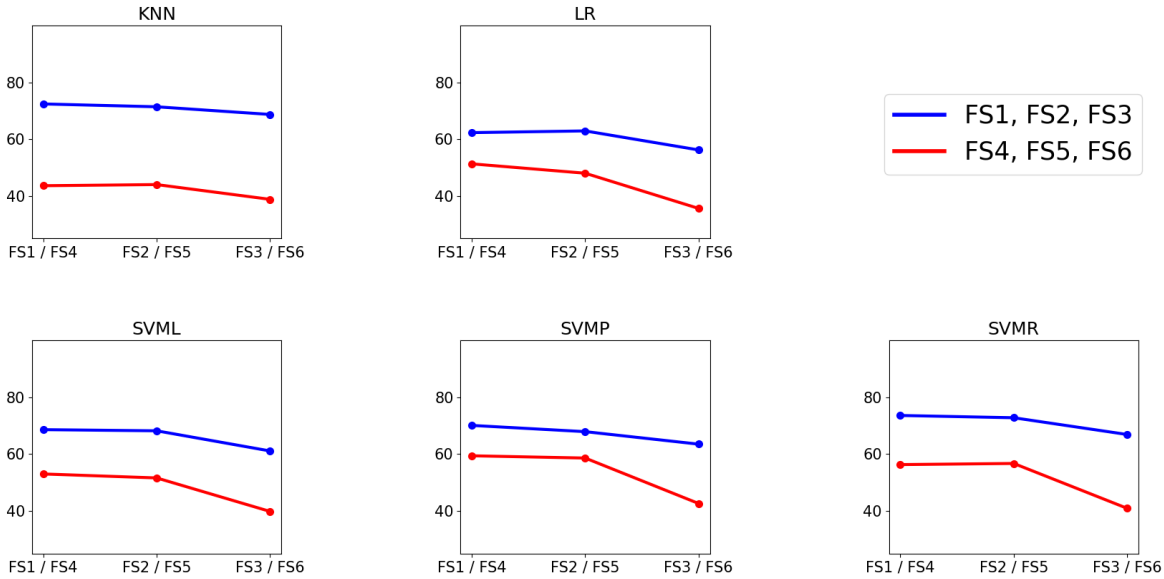

Figure 9: Connected scatter plots for the performance of five classification approaches

While feature reduction (19 to 13 features or 18 to 12 features) has generally resulted in lower performance of the classifiers, we either see an insignificant drop in performance, or in some cases even a slight increase in performance (such as LR-FS2 to LR-FS3, KNN-FS4 to KNN-FS5, and SVMR-FS4 to SVMR-FS5). Considering that the dimensions of the feature sets have been reduced significantly, this small drop in performance is more than acceptable, and the reduced feature sets should be preferred in most cases. The scatter plots also show us the effect of removing the *LEN* feature. The corresponding feature sets are aligned under one another (FS1/FS4, FS2/FS5, and FS3/FS6). It is clear that removing the *LEN* feature results in a significant loss of accuracy for most classifiers (mean loss of 19% accuracy). However, some classification approaches, namely LR and SVMP, are not affected as heavily by the removal of the *LEN* feature (mean loss of 15.5% accuracy for LR and 13.6 for SVMP).

Our goal in this study is to reach a minimal viable feature set, which consists of simple features only. Thus, we actively seek to reduce the number of dimensions in the feature set, and while FS1 and FS4 performed well, they contain a significant number of features (19 and 18 features respectively). Considering that FS2 and FS5 performed similarly with fewer features (13 and 12 features respectively), FS2 and FS5 should be considered the better performing feature sets overall. Additionally, FS3 with only three features performed

relatively well. The performance of these three feature sets (FS2, FS3, and FS5) will be further explored in Experiment 2.

RQ3: Which classification approaches perform best for predicting the CEFR level of English texts when only using simple features?

The mean and standard deviation of the performance of the classification approaches has been recorded in Table 10. By looking at the results, we get a good idea of the performance of each classification approach on six different feature sets. We see that LR performs significantly worse than other classification approaches with a mean accuracy of 52.7. KNN and SVML perform somewhat better with a mean performance of 56.5 and 57.1 respectively. These results are somewhat surprising, since using our initial data set, SVML performed better than other classifiers, and KNN performed even worse than LR. This change in the performance of the classifiers is most likely due to switching to the Cambridge English Readability Dataset as hypothesized in Section 3.3.1. Finally, SVMP and SVMR achieve relatively good results with a mean performance of 60.4 and 61.2 respectively.

Table 10: Performance of the classification approaches across six feature sets

Classification approach	KNN	LR	SVML	SVMP	SVMR	Mean
Mean	56.5	52.7	57.1	60.4	61.2	57.6
Standard deviation	15.9	10.2	11.1	9.8	12.5	11.9

Overall, SVMR performed better than other machine learning approaches and managed to show top 2 performance regardless of feature set when we look at exact accuracy. SVMR also showed a relatively good performance in regard to one-off accuracy, with the exception of feature sets where the *LEN* feature is removed (FS4, FS5, and FS6). The high performance of the SVMR classifier is supported by previous research, where the SVM using the RBF kernel (SVMR) is reported as the best performing classifier (Xia et al. 2019: 26; and Forti et al. 2020). Experiment 1 provides substantial evidence that SVMR classifiers perform well for the task of CEFR level assessment, even when only using simple features.

4.1.5 Performance of specific classifiers

By looking at the performance of the classifiers (Table 7 and Figure 9), we see that SVMR-FS1 performs best with an accuracy of 73.6. This performance is comparable to that of the state-of-the-art classifiers, however the main objective of this study is to define a minimal viable feature set consisting of only simple features. Experiment 1 showed that reduced feature sets (with 12 and 13 features) perform almost as well as complete feature sets (with 18 and 19 features). Thus, SVMR-FS2 is reported as the best performing classifier with an accuracy of 72.8, which is notable considering the use of only 12 simple features.

Furthermore, developing a classifier that can predict partial texts (non-complete texts) can be very useful, and classifiers that exclude the *LEN* feature are expected to perform better for such tasks. Looking at results from Experiment 1, we can see that SVMP performs better than SVMR with the absence of the *LEN* feature. This is clearly illustrated in Table 7 and Figure 9. Thus, the best performing classifier without the *LEN* feature is SVMP-FS5 with an accuracy of 58.6. SVMP-FS4 performed slightly better with an accuracy of 59.4, however this improvement is marginal considering the addition of six features.

Furthermore, investigating the one-off accuracy performance of the classifiers (Table 8) reveals some unexpected results. The SVML and LR classifiers report significantly higher performance when using the one-off accuracy measure. This was especially surprising for the LR classifiers, as LR performed significantly worse when using exact accuracy. LR-FS5 performs at 97.2 one-off accuracy with only 12 features, which is unmatched by any other classifier explored in this study. In addition to using fewer features, this classifier excludes the *LEN* feature, which gives it relatively good potential for classifying partial texts.

Finally, KNN classifiers including the *LEN* feature performed very well and produced results comparable to SVMR. However, KNN classifiers excluding the *LEN* feature performed quite poorly and had the lowest performance of all the classifiers. The one-off accuracy of the KNN classifiers is also relatively low when compared to other machine learning methods. However, KNN classifiers possess one important characteristic that makes them valuable. Tuning the parameter k for KNN is relatively easy and mistake-free, compared to tuning

hyperparameters for SVM models. Boswell (2002: 9) warns that with SVM classifiers, an inexperienced researcher might tune the hyperparameters in such a way that overfits the training data. This risk is even further amplified due to the high value for the parameter *C* for our SVM models (*C=1000*). In contrast, such a risk is generally not acknowledged for KNN, which adds additional value to the KNN-FS1, KNN-FS2, and KNN-FS3 classifiers which showed an accuracy of 72.4, 71.4, and 68.7 respectively. The degree to which classifiers overfit the training data is explored in Experiment 2.

Finally, one main objective for Experiment 1 was to narrow down the number of useful classifiers, so their performance can be analyzed using more sophisticated methods and measures in Experiment 2. Considering the results of Experiment 1, FS2, FS3, and FS5 were chosen for feature sets in Experiment 2. FS1 and FS4 performed well and served as a baseline for comparison, however they contained significantly more features. FS6 performed very poorly and was not considered for further experimentation. Additionally, we narrowed the classification approaches reported in Experiment 2 to KNN, SVMP, and SVMR. SVML classifiers performed worse than expected, and while LR classifiers proved valuable, they only surpassed other classifiers in more accurate approximate predictions (measured through one-off accuracy). Thus, the performance of LR and SVML classifiers is not reported in Experiment 2. The combination of FS2, FS3, FS5 as feature sets and KNN, SVMP, and SVMR as machine learning approaches results in a total of nine classifiers to be analyzed in Experiment 2.

4.2 Experiment 2: Three repetitions of k-fold cross validation

Experiment 1 looked at the performance of 30 classifiers, which were created using a combination of six feature sets and five classification approaches. Experiment 1 revealed that FS2, FS3, and FS5 are the most promising feature sets. It also revealed that KNN, SVMP, and SVMR are the most promising machine learning approaches. This narrowed the number of classifiers to analyze from 30 to nine. The performance of these nine classifiers will be further explored in Experiment 2 using k-fold cross validation.

4.2.1 Setup of Experiment 2

Experiment 2 tests the performance of nine classifiers using k-fold cross validation. More specifically, Experiment 2 is conducted using three repetitions of stratified k-fold cross validation with the parameter *k* set to 5. While *k=10* generally yields more reliable results (Marcot and Hanea, 2021: 21), *k=5* was chosen for this study. This is because results from Experiment 1 showed relatively high variance for the performance of the classifiers, which suggested the presence of outliers in the data set. Considering our small corpus size of 331 observations (*N=331*), the effect of outliers may be extremely high. In k-fold cross validation, the size of the testing set is equal to $\frac{N}{k}$, thus *k=10* would result in 33 observations in each testing data set (approximately 6-7 texts per CEFR level). Choosing *k=5*, on the other hand, results in 66 observations in each validation data set (approximately 13-14 texts per CEFR level). This increase in validation data size should significantly reduce the effect of outliers, therefore *k=5* was preferred for Experiment 2.

To further remedy the problem of high variance in performance of the classifiers, three repetitions of 5-fold cross validation were performed instead of a single round. This is recommended by Brownlee (2020: 31) and will help report more accurate results. Random states were assigned to each repetition of k-fold cross validation to ensure that all classifiers and feature sets use the exact same split of training and testing data. This allows for a fair comparison across all classifiers. Additionally, by using random states, other colleagues and researchers are able to reproduce our results, as long as they use the exact same seed numbers which are 74, 54, and 80. Additionally, the Mean Absolute Error (MAE) of the classifiers is tested and analyzed. By comparing the MAE within the training and testing sessions, we can determine the degree to which classifiers overfit the training data set.

4.2.2 Goals and expectations of Experiment 2

Experiment 2 attempts to remedy the shortcomings of Experiment 1 by using more measures of evaluation, such as precision, recall, and F1-score, which will allow for a more direct

comparison to previous research. Experiment 2 differs from Experiment 1 in a number of ways.

Firstly, we are able to compare classifiers more directly because the exact same data is used for training and testing each classifier through the use of random states. This is true for all folds and all repetitions, which will enable us to give a definitive answer to which classifiers perform best. Secondly, using scikit-learn's *classification_report* in conjunction with k-fold cross validation, we can analyze the performance of the classifiers on individual CEFR levels. This can reveal problems of underperformance on certain classes by the classifiers. In initial experiments with our own collected corpus (mentioned in Section 3.1), many classifiers reported low performance for the C2 class. This is because the C2 class was underrepresented in the original data set (less C2 level texts in the corpus), however such issues may even occur with a balanced corpus. This could be due to the inability of the feature sets to discriminate between specific classes. For instance, Figure 3 (page 35) showed that an increase in *ASL* is expected with increase in CEFR level, however this trend did not persist with an increase in CEFR level from C1 to C2. Finally, we use k-fold cross validation to measure the extent to which our classifiers overfit the training data. This is done by comparing the MAE of the classifiers on training and testing data.

Overall, Experiment 2 should provide a definitive answer to whether simple features have sufficient predictive power over the CEFR level of English texts (RQ1). With respect to the performance of individual feature sets (RQ2), Experiment 1 has already provided a relatively good answer and Experiment 2 will only be concerned with how much each feature set causes overfitting. Finally, in regard to which classification approach performs best using simple features (RQ3), by using scikit-learn's *classification_report* in conjunction with k-fold cross validation, we can determine which classification approach yields the best result and whether any of the classification approaches underperforms on certain CEFR levels.

4.2.3 Results of Experiment 2

The following section will report the performance of the KNN, SVMP, and SVMR classifiers. The results of Experiment 2 show relatively low variance of the performance of the classifiers across different folds and repetitions of k-fold cross validation (mean standard deviation of 0.05 for F1-score). Thus, we only report the mean performance of each classifier across five folds and three repetitions (15 training/testing sessions in total). These results have been summarized in Tables 11, 12, and 13 for KNN, SVMP, and SVMR respectively. The full result of the k-fold cross validation can also be accessed using the link provided in Appendix B.

The difference between precision and recall of the classifiers is relatively small, and these two measures produce similar results. This is not only clear from looking at the performance of the classifiers on each CEFR level, but also by comparing macro average and weighted average scores which are almost identical. I believe this is because our testing data set is relatively small (66 observations), and the classifiers were tuned using accuracy scores rather than precision or recall. Because the precision and recall of the classifiers are relatively close, we will report the F1-Score to represent both of these measures.

The results of Experiment 2 (three repetitions of 5-fold cross validation) are summarized into Tables 11, 12, and 13, which were used to analyze the results for Experiment 2. In these three tables, prec. refers to precision and F1 refers to F1-score. For more information on the different measures of evaluation, refer to Section 3.4.3.

Table 11: Cross validation results for the KNN classifier using FS2, FS3, and FS5

Feature set	Feature set 2			Feature set 3			Feature set 5		
CEFR level	Prec.	Recall	F1	Prec.	Recall	F1	Prec.	Recall	F1
A2	0.87	0.90	0.88	0.88	0.92	0.89	0.68	0.76	0.71
B1	0.79	0.86	0.82	0.83	0.88	0.85	0.39	0.41	0.39
B2	0.71	0.90	0.79	0.66	0.88	0.75	0.32	0.34	0.33
C1	0.62	0.40	0.46	0.51	0.29	0.36	0.40	0.32	0.35
C2	0.64	0.58	0.60	0.60	0.56	0.58	0.40	0.36	0.37
Macro avg.	**0.72**	**0.73**	**0.71**	**0.70**	**0.71**	**0.69**	**0.44**	**0.44**	**0.43**
Weighted avg.	0.72	0.72	0.71	0.69	0.70	0.68	0.44	0.44	0.43
Accuracy	0.72			0.70			0.44		
One-off acc.	0.93			0.93			0.78		

The KNN classification approach reaches a macro average F1-score of 0.71, 0.69, and 0.43 for feature sets 2, 3, and 5 respectively. This performance is almost identical to the accuracy scores reported in Experiment 1 which are 0.71, 0.69, and 0.44. Similar to results in Experiment 1, KNN performs extremely poorly on FS5 which excludes the *LEN* feature. However, it shows similar performance to SVMR using FS2 and FS3 (Table 13).

Table 12: Cross validation results for the SVMP classifier using FS2, FS3, and FS5

Feature set	Feature set 2			Feature set 3			Feature set 5		
CEFR level	Prec.	Recall	F1	Prec.	Recall	F1	Prec.	Recall	F1
A2	0.63	1	0.78	0.64	1	0.78	0.81	0.82	0.8
B1	0.86	0.49	0.62	0.88	0.51	0.64	0.57	0.59	0.57
B2	0.73	0.92	0.8	0.62	0.93	0.74	0.53	0.62	0.56
C1	0.58	0.49	0.51	0.51	0.34	0.39	0.5	0.47	0.48
C2	0.64	0.43	0.5	0.69	0.42	0.51	0.6	0.46	0.51
Macro avg.	**0.69**	**0.67**	**0.64**	**0.67**	**0.64**	**0.61**	**0.6**	**0.59**	**0.58**
Weighted avg.	0.69	0.67	0.64	0.66	0.64	0.61	0.6	0.59	0.58
Accuracy	0.67			0.64			0.59		
One-off acc.	0.93			0.92			0.87		

The SVMP classification approach reaches a macro average F1-score of 0.64, 0.61, and 0.58 for feature sets 2, 3, and 5 respectively. This performance is similar to the accuracy scores reported in Experiment 1 which are 0.68, 0.64, and 0.59 respectively. Overall, the support vector machine classifiers with the polynomial kernel perform extremely well, even on FS5 which contains 12 features and excludes the *LEN* feature.

Table 13: Cross validation results for the SVMR classifier using FS2, FS3, and FS5

Feature set	Feature set 2			Feature set 3			Feature set 5		
CEFR level	Prec.	Recall	F1	Prec.	Recall	F1	Prec.	Recall	F1
A2	0.83	0.87	0.85	0.69	0.9	0.77	0.78	0.81	0.79
B1	0.74	0.78	0.75	0.74	0.58	0.62	0.49	0.52	0.5
B2	0.78	0.89	0.83	0.69	0.9	0.78	0.49	0.62	0.55
C1	0.75	0.44	0.54	0.57	0.27	0.36	0.54	0.44	0.47
C2	0.66	0.73	0.69	0.65	0.66	0.65	0.64	0.47	0.54
Macro avg.	**0.75**	**0.74**	**0.73**	**0.67**	**0.66**	**0.64**	**0.59**	**0.57**	**0.57**
Weighted avg.	0.75	0.74	0.73	0.67	0.66	0.64	0.59	0.57	0.57
Accuracy	0.74			0.66			0.57		
One-off acc.	0.94			0.93			0.86		

The SVMR classification approach reaches a macro average F1-score of 0.73, 0.64, and 0.57 for feature sets 2, 3, and 5 respectively. This performance is again similar to the accuracy scores reported in Experiment 1 which are 0.73, 0.67, and 0.57 respectively. Overall, the support vector machine with the RBF kernel performs extremely well on all three feature sets and can be considered the best classification approach for CEFR level assessment using simple features.

While our results are not directly comparable to previous research due to differences in goals, features, feature sets, and classification approaches, all previous research (Table 2) also focuses on text readability assessment and thus share very large similarities with our research. We can see that our SVMR-FS2 classifier reaches a macro average precision of 0.75, a macro average recall of 0.74, and a macro average F1-score of 0.73 which is comparable to that of the state-of-the-art performance through the use of only 13 simple features.

Additionally, comparing the performance of our classifiers to previous research reveals interesting results. Forti et al. (2020: 7207) exclude A level (A1 and A2) CEFR texts in their study, because according to them, the short length of lower CEFR levels texts could hinder the reliability of the text classification system. However, in our experiments, all three classifiers (SVMP, SVMR, and LR) show very good performance for predicting the level of A2 texts, even using FS5, and show a mean macro average F1-score of 0.8 for the A2 class. This is true for all folds and repetitions of k-fold cross validation.

However, all three classification approaches performed relatively poorly on C1 and C2 texts and only reached a mean macro average F1-score of 0.44 for C1 texts and 0.55 for C2 texts. Since the C1 and C2 classes are not underrepresented in the corpus, as evident by looking at Table 3, and we have no reason to suspect that the texts were labeled inconsistently, such low performance is most likely due to our choice of simple features not being sufficiently powerful for correctly classifying C1 and C2 level texts. For instance, features that rely on the CEFR-J word list (*AJCV* and *BPERA*) are only able to provide information on A and B level words, since the word list does not include C level words. This may have limited the performance of the classifiers on C level texts.

The performance of the classification approaches in Experiment 2 is also compared. Figure 10 uses box plots to compare the performance of the three classification approaches on FS2, FS3, and FS5 over 15 training and testing sessions (three repetitions of 5-fold cross validation). Using box plots to report k-fold cross validation results has been used by Braga-Neto & Dougherty (2004: 378), Moss et al. (2018: 5), and Pantula & Kuppusamy (2020: 10). Figure 10 uses the macro average F1-score as a measure of comparison.

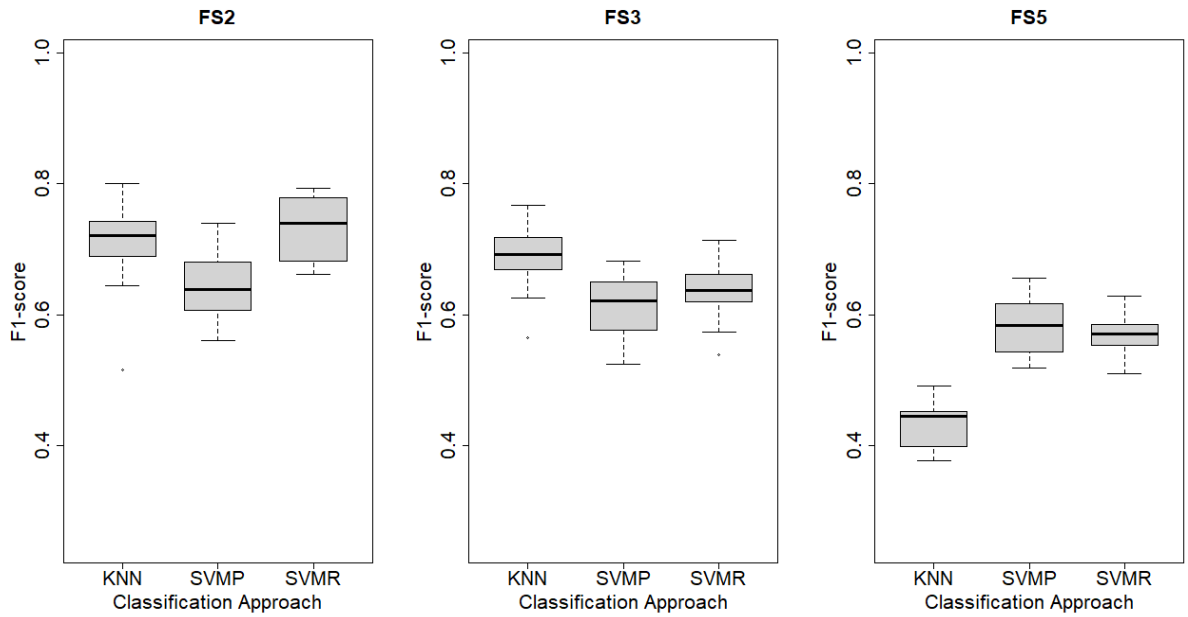

Figure 10: Performance of classifiers using three repetitions of five-fold cross validation

Similar to Experiment 1, each of the classification approaches performs best using one of the feature sets. KNN outperforms SVMR on FS3 by a small margin. SVMP performs best on FS5, however SVMR shows a lower standard deviation. SVMR performs best with FS2, however it shows a higher variance when compared to KNN. Overall, SVMR performs best regardless of the choice of feature set. KNN performs similarly to SVMR when using FS2 and FS3, however it performs extremely poorly with FS5. This is most likely due to FS5 not encompassing enough differences between the classes for the k-nearest neighbors classification (KNN) algorithm.

4.2.4 Evaluating the overfitting of training data by the classifiers

An additional benefit of k-fold cross validation is that it allows for detection of overfitting by our classifiers. This is done by analyzing the Mean Absolute Error (MAE) of the classifiers. Figure 11 shows the MAE of the testing sessions for all nine classifiers. An MAE of 0.0 indicates that the classifier predicts the labels of all observations correctly, and an MAE of 1.0 indicates that on average the classifier misclassifies texts by one level. We can see that classifiers trained on FS2 and FS3 show relatively low MAE values, averaging around 0.4. Because all classifiers show high one-off accuracy scores, this MAE value can be interpreted

as: *on average, one out of two texts is misclassified by one CEFR level*. However, classifiers which use FS5 show a relatively high MAE, especially the KNN classifier. This is also reflected in the performance of the classifiers which was demonstrated in Figure 10. Higher MAE values should directly result in lower performance scores.

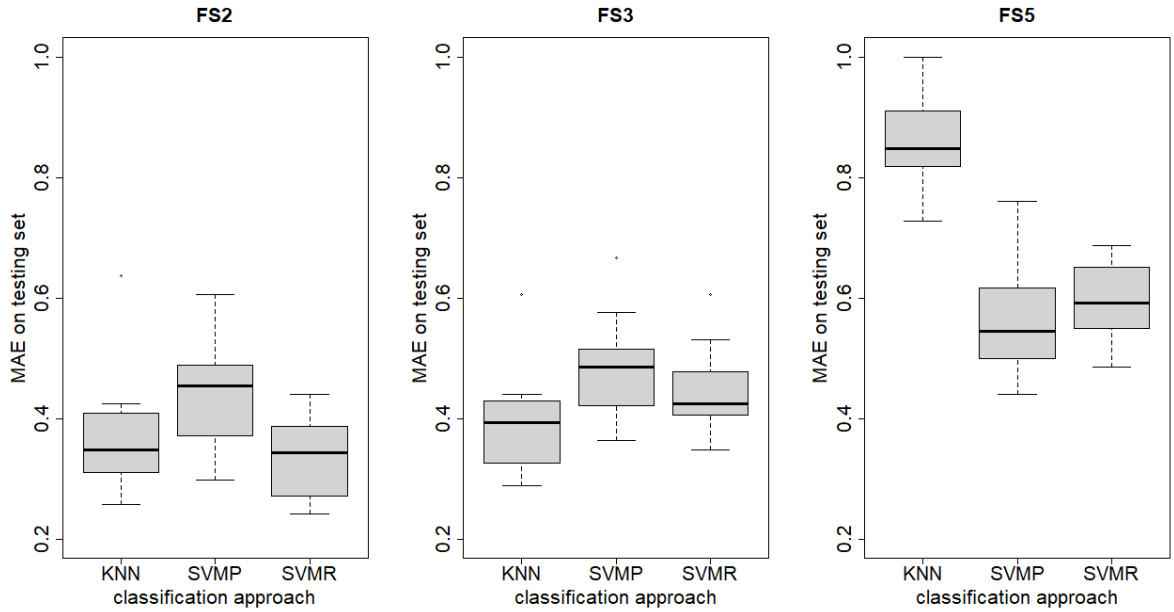

Figure 11: Box plot for the MAE of nine classifiers on testing data

While analyzing the MAE of the classifiers is very valuable, simply looking at the MAE will only tell us how much the classifiers misclassify data and not the reason why misclassification occurs. High error is generally either due to underfitting or overfitting. To understand whether low performance is due to overfitting or underfitting, we can compare the MAE for training and testing data. The difference between MAE_{train} and MAE_{test} is plotted in Figure 12. This difference is calculated by deducting the MAE of the training data from the MAE of the testing data (MAE_{test} - MAE_{train}). If the MAE_{train} and MAE_{test} are both high, then the classifiers are most likely underfitting the training data. In contrast, if MAE_{train} is low and MAE_{test} is high, the classifier is most likely overfitting the training data.

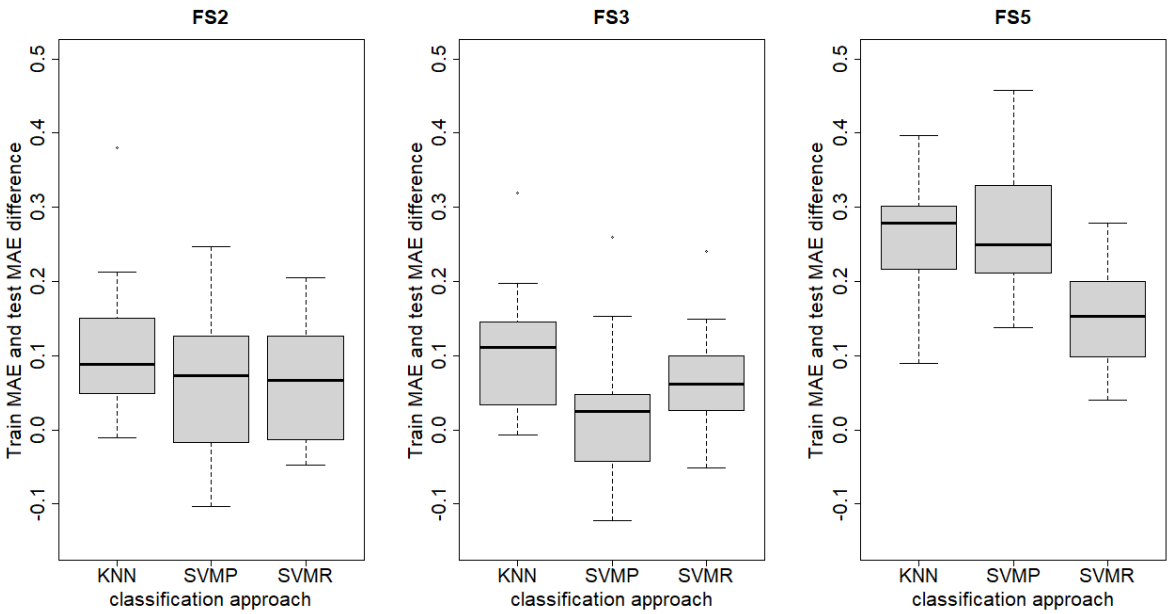

Figure 12: Box plot for the difference between testing and training MAE

Figure 11 showed that the MAE of classifiers trained on FS2 and FS3 is relatively low. Figure 12 shows that the MAE on training and testing data is similar for classifiers trained on FS2 and FS3 (less than 0.1 difference). This provides strong evidence for these six classifiers not overfitting the training data. This claim is further supported by the fact that there are negative values for the MAE differences. Due to the way the difference is calculated, MAE_{test} - MAE_{train} rather than MAE_{train} - MAE_{test}, negative values indicate that our classifier performs equal or better on testing data, which means that the classifiers have the ability to generalize their knowledge to data outside of the training data set. In contrast, FS5 shows a relatively large difference between training and testing MAE. This is true for all three classification approaches, which strongly suggests that these three classifiers are overfitting the training data. Additionally, in contrast to FS2 and FS3, there are no negative differences between training and testing MAE. This is especially problematic because the MAE on testing data is already extremely high for FS5 (Figure 11).

Overall, we observe that classifiers trained on FS2 and FS3 do not show signs of overfitting the training data, however all classifiers trained on FS5 show strong signs of overfitting the training data across different folds. While it is possible to fine-tune the hyperparameters of the classifiers trained on FS5 so they overfit the data less, the high difference in MAE most

likely means that the 12 features in FS5 are not sufficient for accurately predicting the CEFR level of English texts.

4.2.5 Linking results to research questions

In regard to the predictive ability of simple features (RQ1), Experiment 2 confirms that simple features can be powerful predictors of the CEFR levels of English texts. However, our classifiers slightly underperformed on the C1 and C2 CEFR levels, which is most likely due to our specific choice of features not capturing sufficient information for accurately predicting C level texts. Adding features that measure grammatical difficulty or contextual difficulty (i.e. difficulty of the topic discussed) is most likely needed for more accurate predictions of C level texts.

In regard to the best combination of features (RQ2), Experiment 2 shows that FS5 does not capture sufficient information to predict the CEFR level of English texts. However, FS2 and FS3 performed relatively well and showed no signs of overfitting the training data.

Finally, in regard to the performance of classification approaches (RQ3), while KNN and SVMP managed to perform well using specific feature sets, SVMR performed well regardless of the choice of feature set. Experiment 2 confirms results from Experiment 1 that SVMR is the best classification approach for the classification of CEFR rated texts.

4.3 Summary of Experiments and discussion

The first experiment tested the performance of 30 classifiers (combination of six feature sets and five classification approaches) using 1000 folds of training and testing. This narrowed the number of classifiers analyzed in Experiment 2 from 30 to nine. The second experiment used k-fold cross validation to test the performance of the nine selected classifiers. Additionally, Experiment 2 helped measure the mean absolute error (MAE) of the classifiers

and the extent to which the classifiers overfit the training data. Results from Experiments 1 and 2 are used in conjunction to answer three research questions.

RQ1: How powerful are simple features for predicting the CEFR level of English texts?

We hypothesized that simple features would have sufficient predictive power over the CEFR level of English texts, and anticipated that classifiers trained on simple features would perform between 40-50 in regard to their accuracy. Experiment 1 tested the performance of thirty classifiers consisting of different classification approaches and feature sets. These thirty classifiers showed a mean performance of 57.6 and the best classifier reached an accuracy of 73.6. This performance persisted in Experiment 2 when using k-fold cross validation, even when utilizing other measures of evaluation. Such a level of performance is significantly higher than expected, and shows that simple features can be powerful predictors of the CEFR level of English texts.

However, our specific choice of features did not allow for accurate classification of C1 and C2 level texts. We hypothesize that this may be due to the inability of the feature sets to capture text characteristics specific to C level texts, such as topic of discussion and grammatical complexity. Measuring these aspects of texts requires more sophisticated and computationally expensive features, which do not fall under the category of simple features. The inability of simple features to capture such aspects of texts may be an inherent limitation of simple features, which is worth exploring in future studies.

In summary, results from both experiments show that simple features have significant predictive ability over the CEFR level of English texts, even when only using a small number of simple features.

RQ2: What minimal combination of simple features produces the best results for predicting the CEFR level of English texts?

19 meaningful features were explored for this study. These were combined into three feature sets consisting of 19, 13, and 3 features respectively. A counterpart to each feature set was

also created that did not include the *LEN* feature, thus consisting of 18, 12, and 2 features respectively. This resulted in a total of six feature sets to be tested in this study. Creating feature sets in such a way allowed us to analyze the performance of the feature sets along two dimensions, namely through feature selection (i.e. reducing the number of features) and through the removal of the *LEN* feature.

In our experiments, feature sets including the *LEN* feature performed extremely well, even feature set 3 which consisted of only three non-language-specific features. However, removing the *LEN* feature resulted in a significant drop in the performance of the classifiers. Feature set 6 (with 2 features) showed the worst performance and proved inadequate for the task of assessing the CEFR level of English texts. Feature selection, on the other hand, was relatively successful. In our experiments, reduced feature sets, feature set 2 (with 13 features) and feature set 5 (with 12 features), showed performance similar to their full counterparts, feature sets 1 (with 19 features) and feature set 4 (with 18 features) respectively. However, feature set 5 (with 12 features) showed signs of overfitting the training data. Feature set 2 (with 13 features) showed the best performance, achieving a mean performance of 68.6 across five classification approaches.

In summary, feature set 2 with 13 features performed best and is the preferred feature set for assessing the CEFR level of English texts, however feature set 3 showed a relatively high performance considering its use of only three simple non-language-specific features.

RQ3: ***Which classification approaches perform best for predicting the CEFR level of English texts when only using simple features?***

Overall the performance of three different classification approaches was explored in this study. These are k-nearest neighbors (KNN), linear regression (LR), and support vector machines (SVM). LR regression classifiers showed the worst overall performance with a mean accuracy of 52.7 across all six feature sets. The KNN classifier showed very good performance with feature sets including the *LEN* feature (feature sets 1, 2, and 3), however it performed poorly on the counterpart feature sets which exclude the *LEN* feature. SVMs performed better than other classification approaches. This was especially true for SVM

classifiers with the RBF kernel which showed consistently high performance across all feature sets. SVM with the polynomial kernel performed slightly worse when using feature sets including the *LEN* feature, however it performed even better than SVMR with feature sets that excluded the *LEN* feature (feature set 4, 5, and 6). SVM with the linear kernel only outperformed other SVM classifiers in approximate prediction (measured through one-off accuracy), however it showed lower overall performance.

In summary, support vector machines with the RBF kernel performed best for the task of CEFR level assessment. Support vector machines with the polynomial kernel did not perform as well, however they showed better performance when using feature sets which excluded the *LEN* feature.

The results from our experiments show that simple features have significant predictive ability over the CEFR level of English texts. This is in contrast to previous research that utilizes complex and computationally expensive features in addition to simple features, such as Feng (2010) and Forti et al. (2020). We also showed that it is possible to significantly reduce the number of features and maintain good performance of the classifiers. However, in our experiments, removing the *LEN* feature resulted in poor performance of the classifiers, and the classifiers even showed signs of overfitting the training data. Furthermore, Experiment 2 revealed that our classifiers performed poorly for the classification of higher CEFR level texts (C1 and C2 texts). I believe that this is an inherent shortcoming of simple features in assessment of CEFR levels, as capturing aspects such as topic and discussion and grammatical complexity requires more complex features, which are generally more computationally expensive.

Additionally, our study shows that Support Vector Machines (SVMs) perform best for the classification of CEFR texts, which supports previous research in text readability assessment, such as Collins-Thompson (2014: 111), Xia (2019: 26) and Forti et al. (2020: 7208). To the best of our knowledge, our study is the first to acknowledge the great potential of LR models for approximate prediction, which was measured through one-off accuracy. We observed a similar phenomenon for SVM using the linear kernel. I believe this is due to the scale-based approach of these classification approaches, which looks at CEFR levels as values ranging

from 0 to 5. KNN, SVML, and SVMP look at CEFR levels as classes, which prevents them from achieving a higher level of one-off accuracy. This problem is worth further exploring in the future.

Perhaps the biggest shortcoming of this study is that we fail to provide a definitive definition for what constitutes a simple feature. In this study, simple features were defined as easily implementable features that require little time to compute. We even attempted to draw parallel to well-accepted feature types such as shallow and textual features. While this definition was sufficient for this MA thesis, expanding the research on readability assessment using simple features requires a more rigid definition. This can be done through the use of calculation time limits or resources required for implementation. Additionally, the high performance of our classifiers may only apply to exam texts and the classifiers may perform poorly on other kinds of texts. Thus, it would be useful to test the performance of the classifiers on non-exam data sets, similar to our original data set of 177 observations.

5 Conclusion

This MA thesis set out to determine the predictive ability of simple features in identifying the CEFR level of English texts. Simple features were defined as easily implementable features that require little time to compute. We proposed three research questions to expand the research on CEFR level assessment using minimal computational resources:

> ***Research Question 1:*** *How powerful are simple features for predicting the CEFR level of English texts?*
> ***Research Question 2:*** *What minimal combination of simple features produces the best results for predicting the CEFR level of English texts?*
> ***Research Question 3:*** *Which classification approaches perform best for predicting the CEFR level of English texts when only using simple features?*

To answer our proposed research questions, a number of tasks were completed. Firstly, a feature extraction tool was created to convert all 331 texts from the Cambridge English Readability Dataset (Xia et al., 2016) to a data frame. 19 simple features were calculated for each text in the data set. Then, six meaningful feature sets were developed using feature selection algorithms. Additionally, the hyperparameters of five machine learning approaches were adjusted. Finally, two experiments were conducted to provide answers to the research questions. The first experiment tested the performance of 30 classifiers (combination of six feature sets and five classification approaches) using 1000-folds of training and testing. This narrowed the number of classifiers analyzed in Experiment 2 from 30 to nine. The second experiment used k-fold cross validation to test the performance of the nine selected classifiers. Additionally, Experiment 2 calculated the mean absolute error (MAE) of the classifiers, which showed the extent to which the classifiers misclassify data, as well as help identify problems of the classifiers overfitting the training data.

In regard to the predictive ability of simple features (RQ1), results from both experiments showed that simple features have significant predictive ability over the CEFR level of English texts. The 30 classifiers tested in this study showed a mean accuracy of 57.6 and the best classifier reached an accuracy of 73.6. This performance persisted in Experiment 2 using

k-fold cross validation, even when measured through other means of evaluation, such as precision and recall. However, our classifiers performed relatively poorly on C level texts. We hypothesize that this may be an inherent limitation of simple features, as features that can capture characteristics specific to higher CEFR levels, such as topic of discussion and grammatical complexity, inherently require more computational resources. This excludes such features from the category of simple features.

In regard to the best minimal combination of simple features (RQ2), feature set 2, feature set 3, and feature set 5 showed the best performance in Experiment 1. The performance of these three feature sets was further explored in Experiment 2. The second experiment showed that feature set 5 was overfitting the training data. This, in conjunction with an already mediocre accuracy score of 51.8, showed that feature set 5 is not suitable for CEFR level assessment of English texts. Feature sets 2 and feature set 3 did not show any signs of overfitting.

Overall feature set 2, with 13 features, performed best and showed a mean accuracy of 68.6 across five different classification approaches. This feature set should be the preferred feature sets for assessing the CEFR level of English texts using minimal computational resources. Feature set 3 showed an accuracy of 63.3, which is a relatively high performance considering its use of only three simple non-language-specific features. This feature set serves as an absolute minimal feature set for the assessment of CEFR levels. Furthermore, our experiments showed that while it is possible to significantly reduce the number of features in the feature set and maintain good performance, removing the total text length feature resulted in a significant loss of performance.

Finally, in regard to the performance of machine learning approaches (RQ3), both experiments showed that support vector machine classifiers outperform other types of classifiers. More specifically, support vector machines with the RBF kernel performed best for the task of CEFR level assessment. Support vector machines with the polynomial kernel performed slightly worse, however they showed better performance on feature sets that excluded the total text length feature. Our findings support previous research on text readability assessment, such as Collins-Thompson (2014: 111), Xia (2019: 26) and Forti et al.

(2020: 7208), where SVM consistently outperformed other machine learning models for text readability assessment.

Additionally, our study is the first to acknowledge the great potential of linear regression models for approximate prediction, which was measured through one-off accuracy. In our experiments, linear regression models showed the highest performance, when using the one-off measure of accuracy (within-1-level-accuracy). The linear regression model trained on feature set 5 performed at 97.2 one-off accuracy, with only 12 features excluding the total text length feature, which is unmatched by any other classifier explored in this study.

It is also important to acknowledge the shortcomings of this study. The first shortcoming is that we fail to clearly delimit what constitutes a simple feature. In this study, simple features were defined as easily implementable features that require little time to compute. We attempted to draw parallel to well-accepted feature types such as shallow and textual features. While this definition was sufficient for this MA thesis, expanding the research on readability assessment using simple features requires a more rigid definition. This could be done through the use of calculation time limits or resources required for implementation.

The second shortcoming is that we trained and tested the classifiers on data from a single source. This is not a major problem, as the majority of studies referenced for this study also make use of a single corpus for training and testing classifiers. Exceptions include Schwarm & Ostendorf (2005), Petersen & Ostendorf (2009), Feng (2010), and Xia (2019), which utilize texts from various sources. Regardless, it would be beneficial to test the performance of the classifiers on real life data, i.e. on non-exam texts. For instance, it could be useful to test the performance of the classifiers on non-exam texts, similar to our original data set of 177 observations.

In spite of the above mentioned shortcomings, the combination of our research questions in conjunction with the wide range of methodologies used in this study enabled us to identify the strengths and weaknesses of simple features. This has allowed us to lay a solid foundation for future research in the field of CEFR level assessment using simple features.

References

Arias, I. J. 2007. Selecting reading materials wisely. *Letras.* 41. 131-151.

Azpiazu, I. M., & Pera, M. S. 2019. Multiattentive recurrent neural network architecture for multilingual readability assessment. *Transactions of the Association for Computational Linguistics.* 7. 421-436.

Bansal, S. & Aggarwal, C. 2021. *textstat 0.7.2*. <https://pypi.org/project/textstat/>

Ben-Hur, A., & Weston, J. 2010. A user's guide to support vector machines. In *Data mining techniques for the life sciences*. Moscow: Humana Press. 223-239.

Ben-Hur, A., Ong, C. S., Sonnenburg, S., Schölkopf, B., & Rätsch, G. 2008. Support vector machines and kernels for computational biology. *PLoS computational biology*. 4:10. e1000173.

Bijalwan, V., Kumar, V., Kumari, P., & Pascual, J. 2014. KNN based machine learning approach for text and document mining. *International Journal of Database Theory and Application.* 7:1. 61-70.

Bird, S., Klein, E., & Loper, E. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. Sebastobol: O'Reilly.

Boswell, D. 2002. Introduction to support vector machines. *Departement of Computer Science and Engineering University of California San Diego*.

Braga-Neto, U. M., & Dougherty, E. R. 2004. Is cross-validation valid for small-sample microarray classification?. *Bioinformatics*. 20:3. 374-380.

Brownlee, J. 2020. *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*. Vermont, Victoria. Machine Learning Mastery.

Chai, T., & Draxler, R. R. 2014. Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. *Geoscientific model development*. 7:3. 1247-1250.

Chang, Y. W., Hsieh, C. J., Chang, K. W., Ringgaard, M., & Lin, C. J. 2010. Training and testing low-degree polynomial data mappings via linear SVM. *Journal of Machine Learning Research*. 11: 4. 1471-1490.

Coleman, M., & Liau, T. L. 1975. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*. 60:2. 283–284. <https://doi.org/10.1037/h0076540>

Collins-Thompson, K. 2014. Computational assessment of text readability: A survey of current and future research. *ITL-International Journal of Applied Linguistics*. 165:2. 97-135.

Council of Europe. 2020. Common European Framework of Reference for Languages: Learning, teaching, assessment – Companion volume. Strasbourg: Council of Europe Publishing. <www.coe.int/lang-cefr>

Council of Europe. Council for Cultural Co-operation. Education Committee. Modern Languages Division. 2001. *Common European Framework of Reference for Languages: learning, teaching, assessment*. Cambridge: Cambridge University Press.

Crossley, S. A., Greenfield, J., & McNamara, D. S. 2008. Assessing text readability using cognitively based indices. *Tesol Quarterly*. 42:3. 475-493.

Dahlgaard, J. J., Khanji, G. K., & Kristensen, K. 2008. *Fundamentals of total quality management*. Routledge.

Dale, E., & Chall, J. S. 1948. A Formula for Predicting Readability: Instructions. *Educational Research Bulletin.* 27: 2. 37–54. <http://www.jstor.org/stable/1473669>

Dale, E., & Chall, J. S. 1949. The concept of readability. *Elementary English.* 26:1. 19-26.

Feng, L. 2010. Automatic readability assessment. doctoral dissertation. *City University of New York*.

Flesch, R. 1948. A new readability yardstick. *Journal of Applied Psychology*. 32:3. 221–233. <https://doi.org/10.1037/h0057532>

Forti, L., Alfredo, M., Luisa, P., Santarelli, F., Santucci, V., & Spina, S. 2019. Measuring text complexity for Italian as a second language learning purposes. In *14th Workshop on Innovative Use of NLP for Building Educational Applications (held with ACL 2019)*. 360-368. Association for Computational Linguistics.

Forti, L., Grego, G., Filippo, S., Santucci, V., & Spina, S. 2020. MALT-IT2: A New Resource to Measure Text Difficulty in light of CEFR levels for Italian L2 learning. In *12th Language Resources and Evaluation Conference*. 7206-7213. The European Language Resources Association (ELRA).

Gabrilovich, E., & Markovitch, S. 2004. Text categorization with many redundant features: Using aggressive feature selection to make SVMs competitive with C4. 5. In *Proceedings of the twenty-first international conference on Machine learning*. 41-48.

Gardner, D., & Davies, M. 2014. A new academic vocabulary list. *Applied linguistics*. 35:3. 305-327. Retrieved from<https://www.academicvocabulary.info/download.asp> [accessed on 07/06/2021].

Gray, W. S., & Leary, B. E. 1935. *What makes a book readable*. University of Chicago Press.

Government of Canada. 2021. Guide: Application for Canadian Citizenship: Adults – Subsection 5 (1) CIT 0002: <https://www.canada.ca/en/immigration-refugees-citizenship/services/application/application-forms-guides/guide-0002-application-canadian-citizenship-under-subsection-5-1-adults-18-years-older.html> [accessed on 09/10/2021].

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. 2003. KNN model-based approach in classification. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. 986-996. Springer, Berlin, Heidelberg.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. 2002. Gene selection for cancer classification using support vector machines. *Machine learning.* 46*:*1-3. 389–422.

Hancke, J. 2013. Automatic prediction of CEFR proficiency levels based on linguistic features of learner language. Master's thesis. *University of Tübingen*.

Hawkins, D. M. 2004. The problem of overfitting. *Journal of chemical information and computer sciences*. 44:1. 1-12.

Heilman, M., Collins-Thompson, K., Callan, J., & Eskenazi, M. 2007. Combining lexical and grammatical features to improve readability measures for first and second language texts. In *Human language technologies 2007: the conference of the North American chapter of the association for computational linguistics; proceedings of the main conference*. 460-467.

Hiebert, E. H. 2002. Standards, assessment, and text difficulty. *What research has to say about reading instruction*. 3. 337-369.

Home Office. 2021. *Knowledge of language and life in the UK*. Version 27.0. <https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/968974/koll-v27.0ext.pdf> [accessed on 09/10/2021].

Jansen, C., & Boersma, N. 2013. Meten is weten? Over de waarde van de leesbaarheidsvoorspellingen van drie geautomatiseerde Nederlandse meetinstrumenten. [Measuring is knowing? About the value of the readability predictions of three automated Dutch measuring instruments]. *Tijdschrift voor taalbeheersing*. 35:1. 47-62.

Kettunen, Kimmo. 2014. Can Type-Token Ratio be Used to Show Morphological Complexity of Languages?. *Journal of Quantitative Linguistics*. 21. 223-245.

Khamar, K. 2013. Short text classification using kNN based on distance function. *International Journal of Advanced Research in Computer and Communication Engineering*. 2:4. 1916-1919.

Koizumi, R. 2012. Relationships between text length and lexical diversity measures: Can we use short texts of less than 100 tokens. *Vocabulary Learning and Instruction*. 1:1. 60-69.

Lorge, I. 1949. Readability formulae-an evaluation. *Elementary English*. 26:2. 86-95.

Lowie, W. 2013. The CEFR and the dynamics of second language learning: Trends and challenges. *Language Learning in Higher Education*. 2:1. 17-34.

Lubis, A. R., & Lubis, M. 2020. Optimization of distance formula in K-Nearest Neighbor method. *Bulletin of Electrical Engineering and Informatics*. 9:1. 326-338.

Manning, C., & Schutze, H. 1999. Foundations of statistical natural language processing. MIT press.

Marcot, B. G., & Hanea, A. M. 2021. What is an optimal value of k in k-fold cross-validation in discrete Bayesian network analysis?. *Computational Statistics*. 36:3. 2009-2031.

Martinc, M., Pollak, S., & Robnik-Šikonja, M. 2021. Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*. 47:1. 141-179.

Matloff, N. 2017. *Statistical regression and classification: from linear models to machine learning*. Chapman and Hall/CRC.

Maulud, D., & Abdulazeez, A. M. 2020. A review on linear regression comprehensive in machine learning. *Journal of Applied Science and Technology Trends*. 1:4. 140-147.

McGill, R., Tukey, J. W., & Larsen, W. A. 1978. Variations of box plots. *The American Statistician.* 32:1. 12–16.

Merigó, J. M., & Casanovas, M. 2011. A new Minkowski distance based on induced aggregation operators. *International Journal of Computational Intelligence Systems.* 4:2. 123-133.

Milani, A., Spina, S., Santucci, V., Piersanti, L., Simonetti, M., & Biondi, G. 2019. Text classification for Italian proficiency evaluation. In *International Conference on Computational Science and Its Applications*. 830-841. Springer, Cham.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. 2018. *Foundations of machine learning*. MIT press.

Montgomery, D. C., Peck, E. A., & Vining, G. G. 2021. *Introduction to linear regression analysis*. Hoboken, New Jersey: John Wiley & Sons.

Moss, H. B., Leslie, D. S., & Rayson, P. 2018. Using JK fold cross validation to reduce variance when tuning NLP models. *arXiv preprint arXiv:1806.07139*.

Negishi, M., Takada, T., & Tono, Y. 2013. A progress report on the development of the CEFR-J. In *Exploring language frameworks: Proceedings of the ALTE Kraków Conference*. 135-163.

Pantula, M., & Kuppusamy, K. S. 2020. A machine learning-based model to evaluate readability and assess grade level for the web pages. *The Computer Journal*.

Park, K. 2014. Corpora and language assessment: The state of the art. *Language Assessment Quarterly.* 11:1. 27-44.

Petersen S. E. 2007. Natural language processing tools for reading level assessment and text simplification for bilingual education. doctoral Dissertation. *University of Washington, USA*. Advisor(s) Mari Ostendorf. Order Number: AAI3275902.

Petersen, S. E., & Ostendorf, M. 2009. A machine learning approach to reading level assessment. *Computer speech & language.* 23:1. 89-106.

Pitler, E. & Nenkova, A. 2008. Revisiting readability: a unified framework for predicting text quality. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (EMNLP '08). Association for Computational Linguistics, Stroudsburg, PA, USA. 186–195. <http://dl.acm.org/citation.cfm?id=1613715.1613742>

Platt, J. 1998. Sequential minimal optimization: A fast algorithm for training support vector machines. <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-98-14.pdf?fbclid=IwAR1s3TmaWUFynkazhyyrdR-LEJCCgdO_ZfPPoQTQcuJBzosftl2TTbRYp9k>

Powers, D. M. 2020. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*.

Prajapati, G. L., & Patle, A. 2010. On Performing Classification Using SVM with Radial Basis and Polynomial Kernel Functions. *2010 3rd International Conference on Emerging Trends in Engineering and Technology.* 512–515. <https://doi.org/10.1109/ICETET.2010.134>

Schwarm, S. E., & Ostendorf, M. 2005. Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. 523-530.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research.* 12. 2825-2830.

Senter, R. J., & Smith, E. A. 1967. *Automated readability index*. University of Cincinnati, Ohio.

Shah, K., Patel, H., Sanghvi, D., & Shah, M. 2020. A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augmented Human Research*. 5:1. 1-16.

Shashua, A. 2009. Introduction to machine learning: Class notes 67577. *arXiv preprint arXiv:0904.3664*.

Si, L., & Callan, J. 2001. A statistical model for scientific readability. In *Proceedings of the tenth international conference on Information and knowledge management*. 574-576.

Silveman, B. W., Jones, M. C., Fix, E., & Hodges, J. L. 1951. An important contribution to nonparametric discriminant analysis and density estimation. *International Statistical Review*. 57: 3. 233-247.

Solovyev, V., Ivanov, V., & Solnyshkina, M. 2018. Assessment of reading difficulty levels in Russian academic texts: Approaches and metrics. *Journal of intelligent & fuzzy systems*. 34:5. 3049-3058.

Stenner, A. J. 1996. Measuring Reading Comprehension with the Lexile Framework. Presented at *the Fourth North American Conference on Adolescent/Adult Literacy*.

Tan, S. 2005. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*. 28:4. 667-671.

Tanaka-Ishii, K., Tezuka, S., & Terada, H. 2010. Sorting by readability. *Computational Linguistics*. 36:2. 203-227.

The CEFR-J Wordlist Version 1.6. Compiled by Yukio Tono, Tokyo University of Foreign Studies. <http://cefr-j.org/download.html> [accessed on 07/06/2021].

Uchida, S., & Negishi, M. 2018. Assigning CEFR-J levels to English texts based on textual features. In *Proceedings of the 4th Asia Pacific Corpus Linguistics Conference*. 463-468.

Vajjala, S., & Loo, K. 2014. Automatic CEFR level prediction for Estonian learner text. In *Proceedings of the third workshop on NLP for computer-assisted language learning*. 113-127.

Van Rijsbergen, C. J. 1979. *Information Retrieval*. London: Butterworths. Second Edition.

Van Selst, M., & Jolicoeur, P. 1994. A solution to the effect of sample size on outlier elimination. *The Quarterly Journal of Experimental Psychology Section A*. 47:3. 631-650.

Vapnik, V., & Cortes, C. 1995. Support-vector networks. *Machine learning.* 20:3. 273-297.

Velleman, E., & van der Geest, T. 2014. Online test tool to determine the CEFR reading comprehension level of text. *Procedia computer science*. 27. 350-358.

Wang, L., & Zhao, X. 2012. Improved KNN classification algorithms research in text categorization. In *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*. 1848-1852. IEEE.

Willmott, C. J. 1982. Some comments on the evaluation of model performance. *Bulletin of the American Meteorological Society*. 63: 11. 1309-1313.

Xia, M. 2019. Text readability and summarisation for non-native reading comprehension. Doctoral dissertation. *University of Cambridge*.

Xia, M., Kochmar, E., and Briscoe, T. 2016. Text Readability Assessment for Second Language Learners. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications.* 12–22, San Diego, CA. Association for Computational Linguistics.

Appendix A: Feature sets

The following table contains details about the six feature sets developed for this study. This is done to allow for an easy comparison of the feature sets.

Table 14: The six feature sets constructed for this thesis and the features they contain

Feature set	FS1	FS2	FS3	FS4	FS5	FS6
Number of features	19	13	3	18	12	2
ABVMAX	+			+		
ABVMEAN	+			+		
ABVMIN	+			+		
AJCV	+	+		+	+	
APPS	+			+		
ARI	+	+		+	+	
ASL	+	+	+	+	+	+
ASL.AVPS	+	+		+	+	
ATTR	+			+		
AVPS	+	+		+	+	
AWL	+	+	+	+	+	+
BPERA	+	+		+	+	
CLI	+	+		+	+	
DCRS	+	+		+	+	
FKG	+	+		+	+	
FRE	+	+		+	+	
JCPP	+			+		
LEN	+	+	+			
TTR	+	+		+	+	

Appendix B: Full data sets and results

Two data sets were used throughout conducting this study. The first was our own collected data set of 177 observations and the second data set is the Cambridge English Readability Dataset. Both of these data sets are collections of .txt files, which were converted to data frames using our custom feature extraction tool. Our own collected data set was used for feature evaluation and initialization of the classifier hyperparameters. The Cambridge English Readability Dataset was used for training and testing the classifiers reported in Experiments 1 and 2. Both of these data sets can be accessed through the first link below. Additionally, the full experiment results (Experiments 1 and 2) have also been organized into two separate Google Sheets, which can be accessed using links 2 and 3.

1. Data sets used for this study (the Cambridge English Readability Dataset and our original data set of 177 observations):
   https://docs.google.com/spreadsheets/d/1dEGWEsNpW6BTw4igXlFoYjh315oMcqugVQgsSIgxJoI/edit?usp=sharing

2. Experiment 1 results, 1000-fold training and testing of 30 classifiers:
   https://docs.google.com/spreadsheets/d/1RHH9tiw10KRE0rv3qQvVJ5MKw_dDBj7GWMujNcXwssM/edit?usp=sharing

3. Experiment 2 results, three repetitions of k-fold cross validation (nine classifiers):
   https://docs.google.com/spreadsheets/d/1w_TNmuWCIAgOCkmZ40acXNIGiNTfLZZkfSvoaMZcmIg/edit?usp=sharing

Appendix C: Python files

A number of Python programs were written for this Master's thesis. This includes the feature extraction tool (Section 3.2), the feature selection algorithms (Section 3.2.2.2), the programs for Experiment 1 (Section 4.1) and Experiment 2 (Section 4.2). All of these Python files have been compiled into a GitHub Repository to allow for reproducibility of our results. This repository can be accessed under the following link:
https://github.com/AryanYekrangi/_MAthesis