Peter Raatikainen

Automatic detection of developmental dyslexia from eye movement data

Master's Thesis in Information Technology

May 28, 2019

University of Jyväskylä

Department of Mathematical Information Technology

Author: Peter Raatikainen

Contact information: pelaalra@student.jyu.fi

Supervisors: Tommi Kärkkäinen, and Paavo Nieminen

Title: Automatic detection of developmental dyslexia from eye movement data

Työn nimi: Lukemisen erityisvaikeuden automaattinen havaitseminen silmänliikedatasta

Project: Master's Thesis

Study line: Ohjelmistotekniikka

Page count: 47+0

Abstract: Dyslexia is the most common neurological learning disability found worldwide. Though it can seriously hinder individuals' academic success, detecting and treating it early on can drastically reduce its negative effect. Detecting dyslexia reliably and with ease is thus of paramount importance. In this thesis, a method using machine learning and eye movement data to predict if the reader has dyslexia is presented. By using the design science approach, it was possible to obtain new information regarding the data used in addition to a model capable of reliably predicting reading disorders. Support Vector Machine and Random Forest were the methods studied and applied to the data. The best model was obtained by the Support Vector Machine classifier using Random Forest to select the most important features: the general accuracy achieved was 89.8% and the accuracy of detecting dyslexics was 75.9%.

Keywords: Dyslexia, machine learning, eye movement, Support Vector Machine, Random Forest, design science

Suomenkielinen tiivistelmä: Lukemisen erityisvaikeus eli dysleksia on maailmanlaajuisesti yleisin neurologinen oppimisvaikeus. Se voi hoitamattomana merkittävästi haitata yksilön akateemista menestystä. Erityisvaikeuden tunnistaminen ja hoitaminen aikaisessa vaiheessa voi kuitenkin vähentää huomattavasti häiriön aiheuttamia ongelmia. Tässä tutkimuksessa esitetään menetelmä tunnistaa dysleksia koneoppimisen avulla silmänliikedatasta. Hyödyntämällä suunnittelutieteen periaatteita oli mahdollista saada uutta tietoa käytettyyn aineistoon

liittyen sekä luoda koneoppimismalli, joka pystyy luotettavasti tunnistamaan lukemisen erityisvaikeudesta kärsivät henkilöt. Tutkimuksessa käytettiin tukivektorikone- ja satunnaismetsämenetelmiä ennustavien mallien luomiseksi. Parhaan saadun mallin tunnistamisen yleistarkkuus oli 89,8% ja dyslektikkojen tunnistamisen tarkkuus 75,9%.

Avainsanat: Dysleksia, koneoppiminen, silmänliikkeet, tukivektorikone, satunnaismetsä, suunnittelutiede

List of Figures

List of Tables

Contents

1 Introduction

Dyslexia is the most common neurological learning disability (Handler, Fierson, et al. 2011). It causes difficulties in reading, writing and spelling. All these can affect academic success, self-esteem, and social-emotional development. As approximately 10% of the people worldwide are dyslexic, it is a concern of many children and adults around the world. Finding a way to help the lives of a dyslexic would be of great benefit to whole societies. Studies (Snowling and Hulme 2012; Torgesen 2000; Glazzard 2010) have shown that the earlier dyslexia is detected and support is given in teaching, the more its negative effects can be mitigated. Therefore, developing a reliable and objective screening method to diagnose dyslexia at an early age would be of utmost importance.

Using an eye-tracker, it is possible to record the movements of eyes during various activities. Tracking them during reading is especially fruitful in the case of dyslexics, as it has been proven that readers with dyslexia have different eye movements than normal readers (Rayner 1998). Dyslexics display more and longer fixations, shorter saccades, and overall more irregular eye movement (Deans et al. 2010; De Luca et al. 2002; Lefton et al. 1979). Knowledge of this phenomenon serves as a valuable starting point in building a tool to separate normal readers from dyslexics. For this purpose, machine learning provides methods in identifying patterns and making predictions based on them. Combining the known differences between dyslexic and normal eye movements with the feature-based predictions provided by machine learning methods seems a natural combination to be tested.

Applying machine learning in the detection of dyslexia from eye movements is a relatively new approach. Rello and Ballesteros (2015), Lustig (2016) and Benfatto et al. (2016) have studied this method and obtained promising results. All studies applied the Support Vector Machine classifier for separating dyslexics from normal readers. Lustig additionally used Feed-Forward Neural Networks in the classification with good results. All of the studies conclude that predicting the reading ability of individuals from eye movements recorded with eye-tracking can be efficient and reliable.

In this thesis, we applied machine learning in detecting students with dyslexia from a large

eye movement recordings data. The data had been collected for a study regarding Internet reading skills among students with and without learning disabilities. Our goal was to create a software that could reliably predict individuals with dyslexia from the given data. Additionally, we wanted to gain understanding on how to best use the available data to detect dyslexia.

The research method for this thesis was based on the design science methodology. As stated by Holmström, Ketokivi, and Hameri (2009), design science is "research that seeks (i) to explore new solution alternatives to solve problems, (ii) to explain this explorative process, and (iii) to improve the problem-solving process". The goal, in our case, was to understand how to best utilize machine learning methods to detect the students with dyslexia. Our goal was achieved by using iterations, in which a new approach was tried, the results recorded, and a new goal set based on these results. By using these iterations and the designed artifact, we also gained more knowledge and understanding of the problem and the data.

As the most important result of this study, a model based on a Support Vector Machine classifier was created capable of separating dyslexic individuals from normal readers with an accuracy of 89.8%. This model used Random Forest in first selecting the most relevant eye movement features to create a better prediction. Furthermore, the most relevant features found agree with other studies regarding the idiosyncrasies of dyslexics' eye movements. Other valuable results include gaining more knowledge in what features are relevant and the fact we were able to find another usage for the data originally gathered for a different purpose.

This thesis begins with an overview of dyslexia and how it can be detected from eye movements in Chapter 2. In Chapter 3 the data and how it was obtained is discussed. After that, in Chapter 4, there is a brief discussion of the theory behind the chosen machine learning methods. Then, in Chapter 5, the different methods used in this study and the feature extraction from the data is explained. Chapter 6 discusses the results obtained by the used methods implementing the machine learning algorithms. Finally, the conclusion of this thesis is in Chapter 7.

2 Background

This chapter discusses dyslexia and its traits, as well as the basics of eye movements. The connection between these two is also presented. In addition, other studies similar to this one are discussed.

2.1 Dyslexia

Dyslexia is defined as a neurological learning disorder characterized by reading and spelling impairments despite normal intelligence (Frazier 2016). In more detail, Lyon, Shaywitz, and Shaywitz (2003) explain that dyslexia is "characterized by difficulties with accurate and/or fluent word recognition and by poor spelling and decoding abilities ... Secondary consequences may include problems in reading comprehension and reduced reading experience". These difficulties are generally regarded to be due to impairment in the phonological processing of language (MacFarlane et al. 2010), though Hautala (2012) also points out that alternative views indicate a connection between deficient visual and/or attentional processes and problems in fluent reading. The phonologic-deficit hypothesis states that dyslexics have difficulty in understanding the connection between written letters and the sounds they constitute (Shaywitz 1998). In this study, the focus is on people with developmental dyslexia, which is defined as the inability to develop an effortless reading skill (Hautala 2012).

The prevalence of dyslexia has been estimated to be about 5% to 10% (Shaywitz 1998). This differs depending on the estimation criteria. The study by Katusic et al. (2001) indicates a prevalence of 5.3% to 11.8% depending on the formula used. Hautala (2012) also specifies that approximately 6% to 17% of humans are considered to have at least mild problems in reading. Regarding the gender distribution of dyslexics, Shaywitz (1998) notes that previously "it was believed that dyslexia affected boys primarily; however, more recent data indicate similar numbers of affected boys and girls".

The reading difficulties caused by dyslexia can have many kinds of negative effects. Dyslexia has been shown to greatly reduce the learning ability of students and subsequently make attaining academic success much harder (Undheim 2009). It may also limit the individuals'

life-choices as they may try to avoid studies and jobs that involve reading. According to Undheim, dyslexics also tend to be overachievers, which results in more stress. With the modern days of continual information search in the form of web text, inviduals with difficulties in reading and spelling are also predicted to have problems in creating a relevant query for information retrieval, finding useful terms in documents, and thus in understanding key concepts for refining their search (MacFarlane et al. 2010). The not-so-obvious effects of dyslexia include damaging the pupils' self-esteem and self-concept, creating a sense of helplessness regarding control of success attained by learning, and feelings of isolation (Glazzard 2010). Additionally, dyslexia appears to have a negative impact on working practices and career progression (Morris and Turnbull 2007).

According to Snowling and Hulme (2012), the importance of identifying and providing intervention for children with dyslexia at an early age has been emphasized for many years. Glazzard (2010) stresses the need of an early diagnosis for dyslexia, because it can stop pupils' development of learned helplessness and begin improving their confidence. Glazzard found that pupils' confidence, self-concept, and self-esteem made a change to the better after the diagnosis. Vellutino et al. (2004) also explain that children with reading disorders "can acquire at least grade-level reading skills if they are identified early and are provided with comprehensive and intensive reading instruction tailored to their individual needs". Therefore, developing a fast, reliable, and simple method for screening dyslexia would be highly beneficial. In this study, we chose to use eye movement data captured with an eye-tracker to detect dyslexia.

2.2 Eye movements

The two main types of eye movements in reading are fixations and saccades (Rayner 1998). Fixations are defined as movements made when the eye is relatively still and focused on a target (Deans et al. 2010). These can last from tens of milliseconds to several seconds though usually lasting 200 to 300 milliseconds while reading (Holmqvist et al. 2011). According to Eden et al. (1994), the average duration of a fixation is 225 milliseconds when the participant reads silently. In between each fixation, the eye moves rapidly. This rapid movement is called a saccade. During saccades, the eye scans and processes the information between the

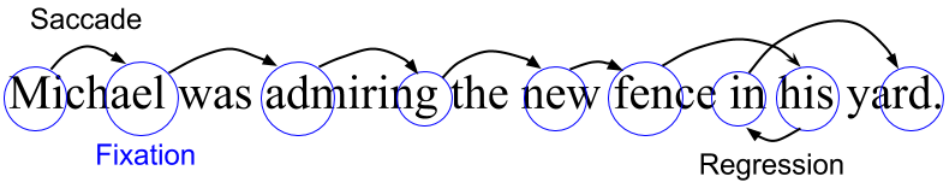

Figure 1. A visualisation of fixations and saccades during reading.

fixation points (Eden et al. 1994). Saccade amplitude refers to the angular distance the eye travels during this movement. Backward saccades are referred to as regressions. Typically, a saccade lasts 30 to 80 milliseconds. Figure 1 shows a visualisation of these eye movements when reading. The circles indicate fixations and the arrows saccades.

Holmqvist et al. (2011) explain that, in addition to fixations and saccades, the eye displays a group of other movements. During a fixation the eye is not actually completely still; it has three types of micro-movements: tremor, microsaccades, and drifts. According to Holmqvist et al., these eye movements are "...mostly studied to understand human neurology". During the end of a saccade the eye usually 'wobbles' slightly before coming to a stop. This movement is referred to as a glissade. Furthermore, when following a moving target people's eyes make a movement called smooth pursuit. It is a slower movement than a saccade and requires a target to follow with the eyes. As fixations and saccades are the most relevant eye movements for understanding reading, these were chosen as the eye movements to examine in this study.

The movements of the human eyes are tracked by using an eye-tracker. The most widely used method to estimate where someone is looking on the stimulus is based on pupil and corneal reflection tracking. The goal is to first detect the position of the pupil and the corneal reflection and then calculate their geometric centres. These centres are then used to calculate their relative distance to each other. Because the pupil moves faster than the corneal reflection, the position of the gaze can be calculated based on their relation. For the eye-tracker to know how this relation corresponds to the points in the stimulus, a calibration must be performed. The calibration gives the eye-tracker examples on how the participants pupil and corneal reflection relation relates to the stimulus area. (Holmqvist et al. 2011)

2.3 Eye movements and dyslexia

The connection between eye movements and dyslexia is a well-established fact (Rayner 1998). People with a reading disability display more and longer fixations, shorter saccade duration and length, and more regressions than normal readers (Rayner 1998; Deans et al. 2010; De Luca et al. 2002). Additionally, Lefton et al. (1979) showed that dyslexic readers' eye movements are "chaotic, frequent, of longer duration, and generally unsystematic" and that the normal developmental gains in eye movements made by children cannot be detected in dyslexics. The underlying reason for the eye movement abnormalities is proposed to be due to difficulties the person has in reading and understanding the text, i.e., the eye movements reflect these problems (Rayner 1998; Hyönä and Olson 1995).

Using the observations on fixation and saccadic differences as a starting point for predicting dyslexia, we can see the possibilities machine learning methods offer. The goal of supervised machine learning as stated by Kotsiantis, Zaharakis, and Pintelas (2007) is "to build a concise model of the distribution of class labels in terms of predictor features". Once the model has been created, it can be used to quickly and efficiently predict the values of the class labels. In this case, the knowledge on the differences in eye movements for a normal and dyslexic reader could be used to select suitable attributes from the eye tracking data to use as the predictor features to separate the two classes. In addition to machine learning being able to help in separating the classes, there are methods that can help in determining the relevant eye movement attributes for this classification. By using these methods, it could also be possible to gain a better understanding of the studied problem.

Despite thorough researching, only three papers were found regarding the use of machine learning methods in detecting dyslexia from eye movements. Nevertheless, these studies show that it is possible to separate dyslexic people from nondyslexic using machine learning with a good accuracy. The first study published by Rello and Ballesteros (2015) presents a model, based on the Support Vector Machine classifier, which is capable of detecting dyslexia with an accuracy of 81,18%. Lustig (2016) compared several machine learning methods; the best results were obtained with Support Vector Machines and Feed-Forward Neural Networks at an accuracy of 83%. Additionally, Benfatto et al. (2016) also used Support Vector Machines to develop classification models capable of separating high-risk

dyslexia subjects from low-risk subjects with a high accuracy. These studies suggest that there is interest in finding new ways to apply machine learning in predicting dyslexia from eye movements, which was the goal of this study.

3 Data

In this chapter the data used for the research is discussed from the perspectives of how it has been obtained and what it contains. Obtaining relevant data is an important step in creating a machine learning model capable of correct predictions.

3.1 Collection of the data

The data used for this research was given by the eSeek project group from the Department of Psychology at the University of Jyväskylä. Their research project was about Internet reading skills among Finnish students with and without learning disabilities. The data had been obtained over the course of three years and contains data of 165 youngsters with an average age of 12.5 years: their results of the tests done, eye-movement data, and partial analysis of these. The students had been chosen from a class of about 400 students. Of the chosen students, 30 (18%) met the criteria for a reading disorder based on choosing the 10th worst percentile of the reading fluency performance score. This criteria was used to label the students as either dyslexic or normal readers. When compared to the general prevalence of dyslexia established in section 2.1, the dyslexics in this data are slightly over-represented.

The eye movements of the participants were recorded using an EyeLink 1000 (SR Research, manual) eye-tracker with a sampling frequency of 1000 Hz. A Dell Precision T5500 workstation with an Asus VG-236 monitor (1920 x 1080, 120 Hz, 52 x 29 cm) at the viewing distance of 60 cm was used for displaying the stimuli. The calibration of the device was performed before the experiment and repeated between trials, if visible head movements were made, a drift was detected on the researcher's screen used for following the eye movements, or the calibration error exceeded .30 visual degrees. (Hautala et al. 2018)

During the experiment, participants completed a practice task and then 10 simulated information search tasks. The tasks consisted of reading the contextualized question and then selecting a search result (out of four options), which would help them answer the question. An example of the given question is "Find out why pandas are endangered?" (Hautala et al. 2018)

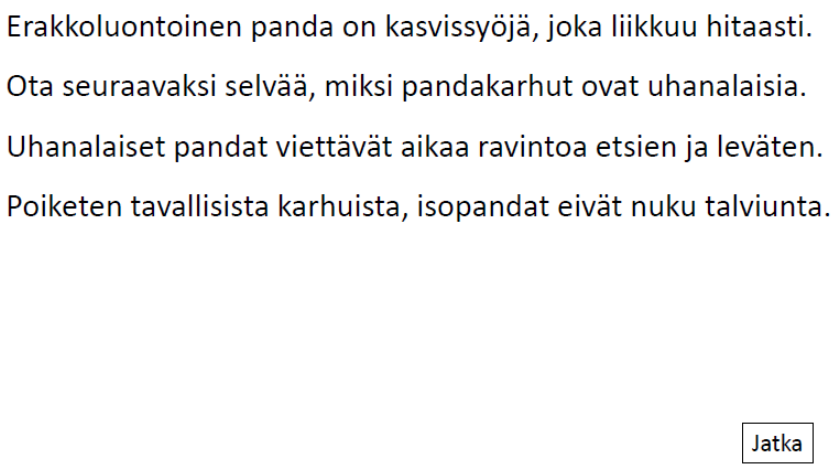

Figure 2. Example of the question page shown during the task. The questions are in Finnish.

The eye movement data that was focused on in this research was obtained from the question page shown to the participants. Figure 2 shows one question page, in which four sentences and a "Continue" button were displayed. The second and third sentence had an important role; one contained the task (question) for this information search, the other was a distractor with irrelevant information regarding the task. The placement of these varied between the tasks, i.e., the task question could also be on the third row and the distractor respectively on the second.

In the case of figure 2, the distractor is the third sentence. The first sentence reads "The reclusive panda is a herbivore that moves slowly". The second sentence, which is the task, reads "Find out why pandas are endangered". The distractor sentence translates to "The endangered pandas spend their time looking for food and resting". Finally, the last sentence reads "Contrary to normal bears, giant pandas do not hibernate".

3.2 Data preprocessing

The obtained data is structured by containing one fixation per row. The 10 tasks and the practice task that the participants completed are henceforth referred to as trials for clarity's sake, as this is their name in the data file.

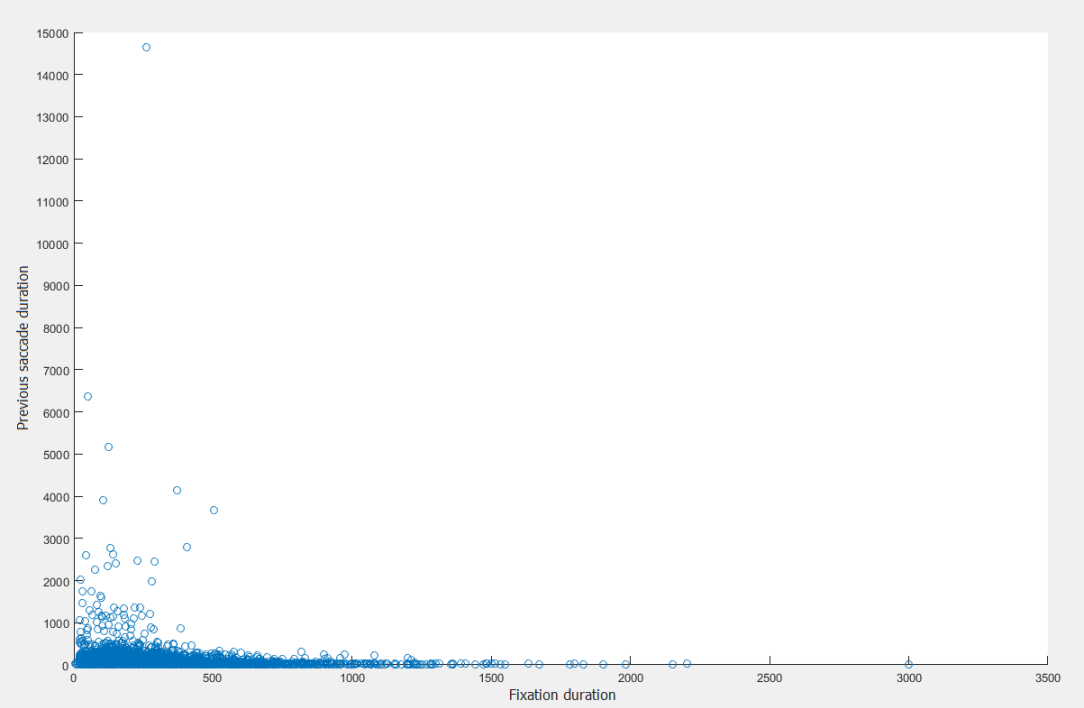

Figure 3. Graph showing the distribution of fixation and saccade durations in the data.

The eye movement data was also cleaned up before being used for creating the feature sets in this study. The SPSS Statistics (IBM Corporation) software was used for the preprocessing. The steps conducted are below:

- The participants with no reading fluency performance score were left out, because this score was needed in establishing whether the participant had a reading disorder or not.
- The data of the practice trial was removed, as it was focused on getting the participants ready for the actual tasks.
- The first fixation of each trial was removed due to the tracking being inaccurate at this stage.
- The rows that contained the value "1" in the BadData column were removed. This label indicated that the particular data row contained bad data.
- Participant with the id 396 was removed due to missing too much relevant data after the above operations.

After the preprocessing, 161 students were left in the data file. Of these, 30 are recognized as having a reading disorder based on their reading fluency performance score.

4 Machine learning methods

The machine learning methods used in this study are presented in more detail in this chapter. The selection of Support Vector Machines and Random Forest as the used methods is also discussed.

4.1 Background

As stated by Alpaydin (2009), machine learning is "programming computers to optimize a performance criterion using example data or past experience". It employs statistics theory to give the computer the ability to learn from data (Alpaydin 2014). Common cases for using machine learning involve problems that cannot be solved directly by any method or the knowledge required to solve them does not exist. In these cases, amassing a lot of data and letting the machine learning model find certain patterns and regularities can produce a solution to the problem. Generally, this process is known as "fitting" the model. It may not be possible to acquire a perfect solution; rather, the goal is to construct a useful approximation. (Alpaydin 2014)

Classification of machine learning methods can be done by task and by application. The tasks are generally divided into supervised and unsupervised tasks. When categorizing machine learning by application, interest is on the desired output. When the goal is to classify the inputs into two or more classes, the machine learner deals with classification. If the desired output is continuous, e.g., numbers, the method type belongs to the regression category. In the case of this study, the goal is to separate dyslexics from normal readers. Additionally, we know what kind of patterns to teach the machine learner, so we have a binary classification problem with supervised learning.

On the advantages of machine learning, Kotsiantis, Zaharakis, and Pintelas (2007) state that "people are often prone to making mistakes during analyses or, possibly, when trying to establish relationships between multiple features. This makes it difficult for them to find solutions to certain problems. Machine learning can often be successfully applied to these problems, improving the efficiency of systems and the designs of machines". The benefits of

machine learning thus include removing a part of the human error involved. For diagnosing dyslexia reliably, removing some of this error could prove to be significant.

The chosen machine learning methods for this thesis were Support Vector Machine and Random Forest. These were selected because they are currently widely used methods. Support Vector Machine has also been previously used in studies of this field (Benfatto et al. 2016; Rello and Ballesteros 2015; Lustig 2016), so this was chosen to establish the baseline results.

4.2 Support Vector Machine

Support Vector Machine (SVM), presented by Cortes and Vapnik (1995), is a widely used and effective classification method. It has succesfully been applied in face detection (Osuna, Freund, and Girosit 1997) and text recognition (Wang, Babenko, and Belongie 2011), among other problems (Noble et al. 2004; Mountrakis, Im, and Ogole 2011; Noble et al. 2004).

SVM separates classes by mapping the input vectors into a high dimensional feature space through the chosen non-linear mapping (Cortes and Vapnik 1995). In this space an optimal hyperplane is found for the separable classes. Figure 4 shows the optimal hyperplane and its margins. This hyperplane is defined as the linear decision function with a maximal margin between the data points of the two classes. Maximising this margin has been proven to reduce the generalization error (Vapnik 1999).

Suppose we are given a set of training data D, with n data points

$$D = \{(x_i, y_i) | x_i \in \mathbf{R}^p, y_i \in \{-1, 1\}\}_{i=1}^n \tag{4.1}$$

where x_i is a p-dimensional real number and y_i the class which x_i belongs to (either 1 or -1). The data points are said to be linearly separable if

$$y_i(w \cdot x_i + b) \geq 1, \ \forall i = 1, \ldots, n \tag{4.2}$$

where w is a vector perpendicular to the hyperplane and $\frac{b}{\|w\|}$ the offset of the hyperplane

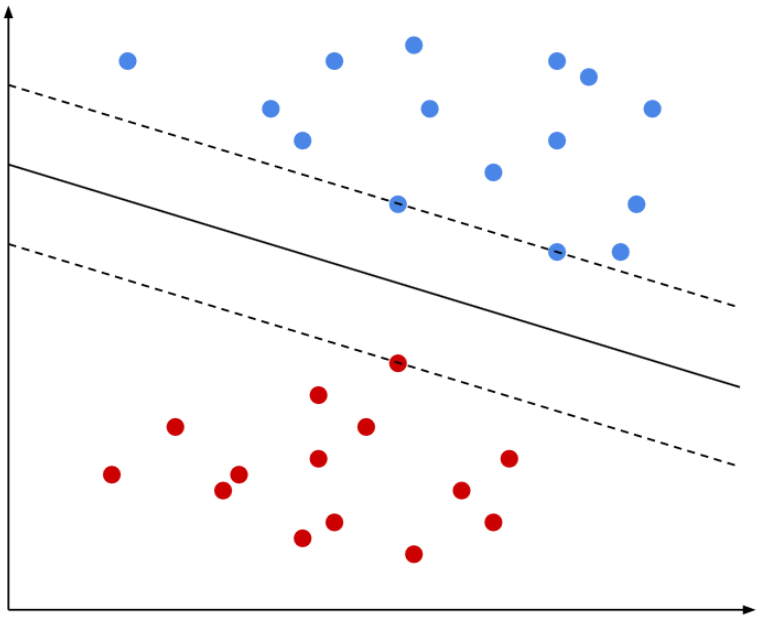

Figure 4. Example of an SVM with a maximum-margin hyperplane separating two classes. The dashed lines represent the margins. The dots on both margins are support vectors.

from the origin along w.

If the training data is linearly separable, the margin hyperplanes can be selected in a way that there are data points between them. The distance between the margins, which is $\frac{2}{\|w\|}$, can then be tried to be maximized. Maximising this distance involves minimizing $\|w\|$. Thus, this is the optimization problem, with the constraint 4.2.

However, in real-world problems the data cannot usually be linearly separated without error. This problem is handled by introducing slack variables $\xi_i \geq 0$, $i = 1, \ldots, n$ (Cortes and Vapnik 1995) so that

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, \ \forall i = 1, \ldots, n \tag{4.3}$$

These slack variables allow the data points to be a small distance ξ_i on the wrong side of the hyperplane. The optimization function thus becomes

$$\min_{w,\xi} \left\{ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \right\}, \tag{4.4}$$

where C is a constant chosen by the user. With a bigger value of C, the penalty for training errors is higher. The constraint of this optimization function is equation 4.3.

The above function is still only for solving linear classifications. In the case of one class being divided by the other, linear classification cannot achieve good results. This can be solved by using a non-linear classifier created by using the kernel trick (originally proposed by Aizerman, Braverman, and Rozonoer (1964)). Boser, Guyon, and Vapnik (1992) proposed this method in 1992. The kernel trick consists of mapping the original space into a much higher dimension, which presumably makes the separation problem easier. In the resulting algorithm, every dot product is replaced by a non-linear kernel $K(x,x_i)$. The form of these functions is

$$f(x) = \sum_{i=1}^{n} y_i \alpha_i K(x, x_i), \tag{4.5}$$

where x_i is the image of a support vector in the input space and α_i is the weight of a support vector in the feature space (Cortes and Vapnik 1995).

In this study, the SVM implementation used was from the Python module Scikit-learn (Pedregosa et al. 2011). Internally this SVM implementation uses the libsvm library, which is wrapped in C (Chang and Lin 2011). In this implementation, the radial basis function kernel is

$$K(x_i, x_j) = \exp(-\gamma \| x_i - x_j \|^2), \; for \; \gamma > 0, \tag{4.6}$$

where γ is a parameter specified by the user with the keyword `gamma`.

Using this kernel the decision function is

$$\operatorname{sgn}\left\{ \sum_{i=1}^{n} y_i \alpha_i \exp(-\gamma \| x_i - x_j \|^2) + \rho \right\}, \; for \; \gamma > 0 \tag{4.7}$$

The other kernel choices available in the Scikit-learn module (Pedregosa et al. 2011) are

- Linear: $\langle x, x' \rangle$
- Polynomial: $(\gamma \langle x, x' \rangle + r)^d$, where d defines the degree and r serves to trade off the influence of higher-order versus lower-order terms.
- Sigmoid: $\tanh(\gamma \langle x, x' \rangle + r)$, where r serves the same purpose as in the polynomial kernel above.

4.3 Random Forest

Random Forests (RF) have been used with good results in various tasks, including data mining (Verikas, Gelzinis, and Bacauskiene 2011), bioinformatics and computational biology (Boulesteix et al. 2012), and remote sensing (Belgiu and Drăguţ 2016). Random Forest is a classifier that consists of an ensemble of randomized decision trees, which vote for the most popular class (Breiman 2001).

The decision tree classifier consists of a rooted tree, which contains nodes $t_0, ..., t_n, n \in N$ that each represent a subspace $X_{t_n} \subseteq X$. The root node t_0 corresponds to the input space X. Each node t is labeled with a split s_t. The splits divide the nodes' subspace X_t into two subspaces, which are represented by the nodes' children. (Louppe 2014)

Formally, a Random Forest is defined (Breiman 2001) as a classifier consisting of a collection of tree structured classifiers $\{h_k(x, T_k), k = 1, \ldots\}$, where T_k are independent identically distributed random vectors, and each tree casts a unit vote for the most popular class at input x.

The goodness of the decision tree classifier splits is specified by the impurity measure, also called the Gini index (Alpaydin 2014). According to Alpaydin, a split in the tree is pure "if after the split, for all branches, all the instances choosing a branch belong to the same class". For more details on the Gini index, see Louppe (2014) page 45.

A part of the user-adjustable parameters for Random Forest involve controlling when the splitting of the nodes is stopped (Louppe 2014). This is important to prevent overfitting of the model. The following parameters are used to control when a node t is set as a terminal

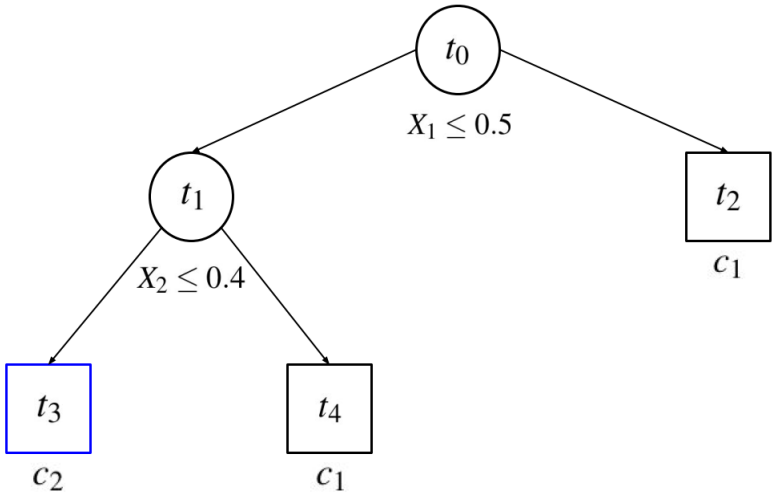

Figure 5. A decision tree created for a binary classification problem. The tree contains five nodes, of which three (t_2, t_3, t_4) are terminal nodes. Two splits partition the input space into three subspaces. (Figure inspired by Louppe (2014))

node:

- `min_impurity_decrease`, the minimum number of samples to set t as a terminal node.
- `max_depth`, the maximum depth of a tree. If t is at the depth of `max_depth`, set t as a terminal node.
- `min_impurity_decrease` the minimum decrease in impurity to set t as a terminal node.
- `min_samples_leaf` the minimum number of samples required to set t as a terminal node.

Additionally, when decision trees are built into a random forest, two more parameters become relevant. The number of trees in the forest is defined by the `n_estimators` parameter. Having a larger number of trees is usually better, but that also increases the computation time for the model. When splitting a node in the decision tree, the feature used for the split is selected from a random subset of features. The amount of features chosen into this subset is determined by the `max_features` parameter. (Louppe 2014)

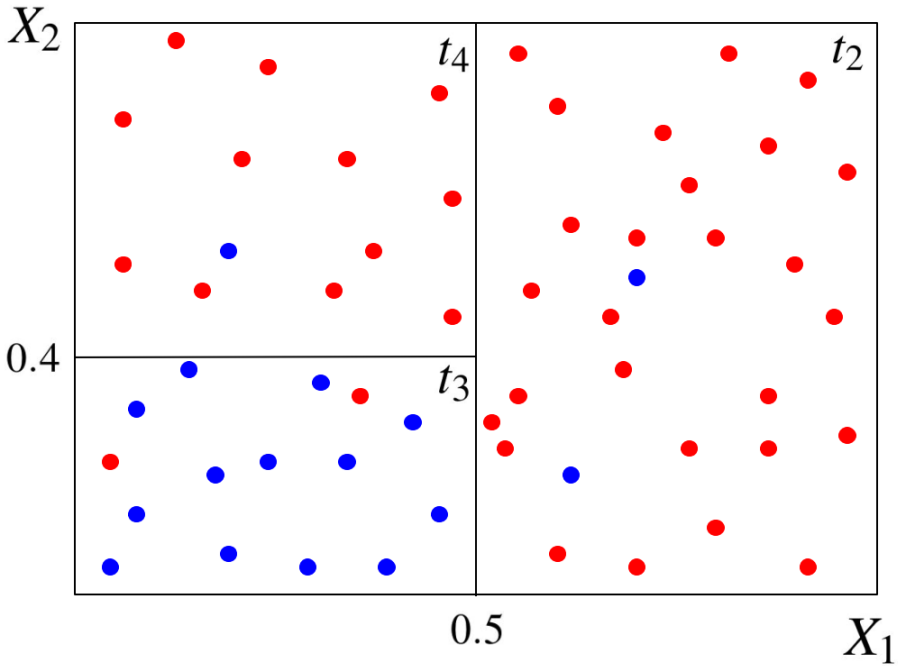

Figure 6. The partitions caused by the splits in decision tree 5. Red dots represent objects of class c_1 while blue dots represent objects of class c_2. (Figure inspired by Louppe (2014))

Decision trees and other tree-based methods are lucrative as explained by Louppe (2014), because they:

- are non-parametric,
- intrinsically implement feature selection,
- are robust to outliers or errors in labels,
- handle heterogeneus data (ordered or categorical variables, or a mix of both).

Additionally, Cutler, Cutler, and Stevens (2012) state that Random Forests are appealing, because they:

- naturally are capable of regression and classification,
- have a built in estimate of generalization error,
- can be used directly for high-dimensional problems.

In this study, the Random Forest implementation used was also taken from the Scikit-learn Python module (Pedregosa et al. 2011). It is important to notice that unlike the original

Random Forest method (Breiman 2001), the Scikit-learn implementation does not let each decision tree classifier vote for a single class. Instead, it combines the classifiers by averaging their probabilistic prediction.

4.4 Hyperparameter optimization

When fitting a machine learning model to the data, the goal is to obtain the best possible fit, i.e., the smallest generalization error. As described in the previous sections 4.2 and 4.3, machine learning models typically have a variety of parameters that affect their behaviour. These parameters are called hyperparameters, and they are set before the model is trained. Appropriate configuration of the hyperparameters is necessary to produce a model with the best performance for the problem in question. (Claesen and De Moor 2015)

The most widely used methods in hyperparameter optimization are grid search and manual search (Bergstra and Bengio 2012). Manually searching for the best hyperparameters involves making educated guesses and comparing the results obtained from the model. Grid search, on the other hand, involves doing an exhaustive search through a manually specified group of hyperparameters and their values for the machine learning algorithm. Below is an example of the grid search parameters used to optimize a SVM model using the radial basis function kernel.

$$\begin{aligned} C &: [1000, 2000, 3000, 5000, 7000, 10000, 20000, 50000, 100000, 500000, 1000000] \\ \gamma &: [0.00004, 0.00006, 0.00008, 0.0001, 0.0005, 0.001, 0.005, 0.01] \end{aligned} \tag{4.8}$$

Bergstra and Bengio (2012) explain that the benefits of using grid search include the simplicity of implementation and trivial parallelization. Additionally, grid search is reliable in low dimensional spaces and does not include technical barriers to manual optimization. The major drawback of using grid search is the curse of dimensionality, as the number of combinations to go through grows exponentially with the number of hyperparameters. In our case, the simplicity of implementing grid search and easy access to manually optimize the hyperparameters made it very lucrative. After initial experiments, grid search was selected

as the chosen method, as the results obtained were satisfactory.

4.5 Cross-validation

When determining how well a machine learning model performs, using only one data set can pose problems. Regarding the general performance of the model, it is difficult to say how well the model will perform in the future on other data sets, if trained and validated on only one set. Also, comparing the expected error of learning algorithms, whether completely different algorithms or the same one using different hyperparameters, is very challenging with just one data set. (Alpaydin 2014)

For the reasons stated above, having a different set of data for validation is important. In fact, having several training sets would be even better. Using only one validation set increases the risk of possible anomalies. Additionally, the machine learning method may contain random factors affecting the result. To reliably evaluate the effect of these factors it is necessary to use more than one validation set. (Alpaydin 2014) One way of achieving this is to use cross-validation.

Cross-validation aims to assess how a machine learning model will generalize to new, unseen data. One round of cross-validation consists of partitioning a data sample into a number of subsets called folds. One fold is selected as the validation data set and the rest form the training data. Most methods involve doing multiple rounds of cross-validation with different subsets to reduce variability. 10-fold cross-validation has been shown to be the best method for comparing models when using real-world datasets (Kohavi et al. 1995). For our study, ensuring that the relative amount of both classes is preserved in the folds was important. This procedure is called stratification.

5 The implementation

In this chapter, the script used to obtain the results of this study is explained. Additionally, the feature extraction methods used are also described. The algorithm was programmed in Python using the Scikit-Learn module for the machine learning algorithms. Additionally, the Pandas module was used for handling the data. For the complete code, see YouSource.

5.1 Overview of the script

The script created for producing our results has the following stages: initialization of the data, feature extraction and generation, training and evaluation of models, and displaying results. The feature extraction and generation stage is explained in more depth in section 5.2.

The initialization of the data contains a few important tasks. The wanted columns from the whole data are selected using Pandas. Simultaneously, the cells containing unknown values are transformed into cells containing a whitespace. These cells are then replaced to cells containing "0". The matrix used to store the extracted data is also prepared depending on the feature set to be created.

The part of the script used for training the model and getting the prediction results uses a self-created cross-validation and grid search method. This method is shown in Algorithm 1. The training and evaluation are done in cycles; each cycle consists of training the model and obtaining the results. The `cycles` constant determines how many times the whole cross-validation cycle is done. `p1` and `p2` are two hyperparameters chosen to be optimized. For SVM, `C` and `gamma` were optimized. Respectively, for Random Forest, `max_features` and `n_estimators` were selected to be optimized. For each new hyperparameter combination, the model is created again to ensure that it does not contain any memory of past trainings.

For our study, we selected to use 5-fold cross-validation, because more folds would have reduced the amount of students with dyslexia in each fold to a too small amount. Additionally, using fever folds was computationally faster. The whole cross-validation process is repeated

100 times to reduce the effect of randomness on the results. A similar approach was used in the study by Mantau et al. (2017), where a 5-fold cross-validation was used to deduce the best parameters, and the whole training and testing process was repeated ten times. This was done to evaluate the performance of the models.

Algorithm 1 The algorithm for training and evaluating the machine learning model with cross-validation

```
for i = 1, ... , cycles do
    for p1, p2 in hyperParameters do
        Create five cross-validation folds
        for each cross-validation fold do
            Create classifier
            Fit model with data
            Store resulting predictions
            Calculate and store confusion matrix
        end for
    end for
end for
Put resulting models in order based on the recall score for predicting dyslexics
```

The combinations of hyperparameters are compared against each other by a performance metric. In our case, we used the recall score of dyslexics predicted. Recall is the fraction of correctly predicted samples out of all the samples of the positive class. This was chosen as the performance metric in this research as it was deemed more important to correctly detect the dyslexics than normal readers. In addition to the recall score, we also observed the overall accuracy of the model. Using only the accuracy score is not enough, because the classes are unbalanced in our data. It would be possible to obtain an accuracy of 81.5% by just declaring all of the test subjects as normal readers. This would give a false picture of the model's performance.

In the case of Random Forest, the algorithm also calculates the feature importances for each model created in the cross-validation folds. The ten most important features for each model are saved into a list. From this list it is possible to obtain a number of the most often occuring

features and use them to create a new feature set. This is one of the methods used to create a feature set in this study; section 5.2 describes this in more detail.

Finally, the script displays the results. The charts are created with the Pyplot module from the Matplotlib library and other relevant information is displayed in the console window. The confusion matrices are calculated for every model for each fold in the cross-validation process. After the cycles have been completed, the mean confusion matrix for each model is calculated and displayed.

5.2 Feature extraction, selection and generation

In the case of a binary classification problem, the goal of feature generation is to choose the features that best separate the two classes. For this study, this meant finding features that help distinguish dyslexics from non-dyslexics by their eye movement patterns. We chose to test several different approaches for generating feature sets. These approaches were based on the results of the research survey conducted, and also on the ideas devised during the meetings of our research group.

The feature sets used and their dimensions (rows x features) are:

- **Averaged (AVG)**: 161 x 4
- **Transition matrix average (TMA)**: 161 x 24
- **Transition matrix (TM)**: 161 x 240
- **Trials on rows (TR)**: 1610 x 24
- **Transition matrix with histograms (TMH)**: 161 x 760
- **Features chosen with RF (RFFn)**: 161 x n

The first feature set generated (AVG) contains the participants' total fixation count, average fixation duration, average saccade amplitude and average saccade duration. This is the most basic feature set created in this study. The saccade amplitude and saccade duration are partially tied to each other, as the larger the amplitude, the longer the saccade lasts.

The rest of the feature sets, apart from RFFEAT, have been created using eye movement transition matrices. The transition matrix is a catalogue of all area of interest (AOI) sequences

	F	T	D	L
F	11	1	2	3
T	2	16	5	6
D	1	7	8	0
L	0	4	8	4

Figure 7. Example of a transition matrix used in this study.

of a length equal to the dimension of the matrix (Holmqvist et al. 2011). The matrix, in figure 7, consists of the AOIs listed in rows and columns, and the cells with numbers indicate how many times gaze has shifted from one AOI to another. In our case, the AOIs are the four sentences on the question page of the experiment. They are labeled in the following way: First sentence (F), Task sentence (T), Distractor sentence (D) and Last sentence (L). For instance, in the example 7, the participant looked from the distractor to the task sentence more often (7 times) than to the first sentence (once).

The difference between the traditional transition matrix, and the one we used, is that in our transition matrix we placed the number of fixations within an AOI on the diagonal. This was considered relevant because dyslexics have been known to have more fixations while reading, as stated in section 2.3. In the example figure 7, we can see that the participants' gaze shifted within the task sentence 16 times.

The chart in figure 8 illustrates all the generated feature sets apart from RFFn. In the lower half are the feature sets generated with their details placed in the order of increasing complexity from left to right. The black arrows indicate what extracted data was used for each feature set. The figures in the top half show the progression of our method used to extract features from the data.

We conjectured that the transitions between these sentences could be useful in separating the readers with difficulties from normal readers. The hypothesis was that dyslexics would have more difficulty finding the task sentence out of the four than normal readers. This would cause them to have a more erratic gaze movement among the sentences, and by comparing transition matrices it should be possible to notice this difference.

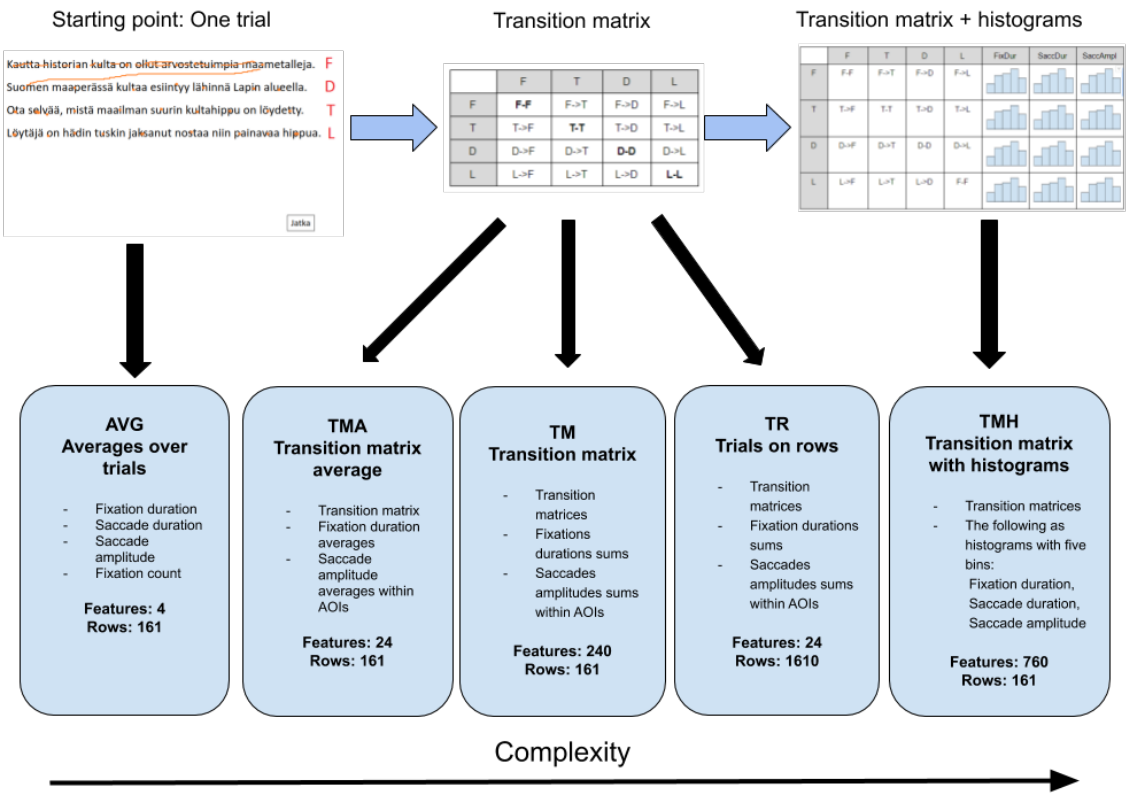

Figure 8. This chart displays the feature sets apart from RFFn and the hierarchy of their generation.

The feature set TM used transition matrices from each trial. It also contained the sums of fixation durations and saccadic amplitudes for each sentence in each trial. On the other hand, the feature set TMA used the mean values of transition matrices. Additionally, the fixation duration averages and saccadic amplitude averages per each sentence over all trials were included. The intention behind using the mean values was to lower the dimensionality of the feature set and possibly reduce the noise.

The feature set TR is created by using the TM feature set and transferring the data of the trials to each row. This causes it to contain $161 * 10 = 1610$ rows, because there were ten trials for each participant. Once the data of each trial has been transferred, each participant's "dyslexia" value is calculated by the trials voting. Each trial of the participant casts a vote; if the trial predicts the participant to be dyslexic, a one is given, else a zero. If the total amount of votes for each participant is greater than five, then the participant is predicted to have dyslexia.

For the TMH feature set, we used the transition matrix data with histograms containing five bins for fixation duration, saccade duration and saccade amplitude. The transition matrix data was the same as in the TM feature set. The histograms were created separately for each sentence in every trial. This causes the feature set to have $transitionmatrixfeatures + bins * values * sentences * trials = 160 + 5 * 3 * 4 * 10 = 760$ features. The bin intervals were calculated beforehand by evenly dividing the entire range of data values into five equally sized partitions.

Each of the feature sets is normalized before saved in to the csv files. The normalization is done with equation 5.1. In the equation, the data values are rescaled to have values between 0 and 1.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{5.1}$$

When selecting variable subsets for better predictive power, the methods used can be divided into three groups: wrappers, filters and embedded methods (Guyon and Elisseeff 2003). Wrappers score the subsets with the machine learner based on their predictive power. Filters

utilize general features such as correlation to remove the least interesting variables regardless of the chosen predictor. Embedded methods incorporate the selection of variable subsets in the process of training, thus perfoming feature selection and classification simultaneously. In this study, the algorithm used resembles the embedded method the most.

The method applied for creating the RFFn feature sets involves using Random Forest to first select the most important features, which are then given to SVM for classifying the data. Similar methods to this have been used in previous studies (Yang et al. 2014). The method used in this study involves first calculating the feature importances with RF for each fold rotation in every cycle for all the hyperparameter combinations. Of these feature importances, the 10 most important ones are saved at each fold rotation. Once the hyperparameter combination with the best recall value for the dyslexia class has been found, the n most frequent features are extracted into their own feature set, where n is the amount chosen. The amounts used were 10, 20, 30, 35 and 40. These were heuristically selected to find the feature set that produces the best results.

6 Results

This chapter discusses the results obtained in this study. These results were all achieved by using the method described earlier in chapter 5 with 100 cycles and a 5-fold cross-validation. The scores presented here for each model are averages of the 100 cycles. The error given is the standard deviation of these results.

Table 1 shows an overview of the best results. The "Method" column indicates the machine learning method used to produce the model. The "Bal" tag indicates that the class weights were balanced for the Scikit-learn library SVM by adjusting them inversely in proportion to class frequencies. The second column holds the names of the feature sets as given in section 5.2. The "Accuracy" column holds the average fraction of correct predictions for all of the hundred models created during the algorithm. The error given is the standard deviation of these accuracy scores. The final column contains the average recall scores for the class of dyslexics of the hundred models created.

Table 1. Best models created with their accuracy and recall scores. These are the average results over 100 cycles.

Method	Feature set	Accuracy	Recall
SVM	RFF35	$89.8\% \pm 4.7\%$	$75.9\% \pm 17.1\%$
	TR	$86.4\% \pm 1.8\%$	$55.7\% \pm 6.4\%$
SVM Bal	RFF35	$89.7\% \pm 4.0\%$	$84.8\% \pm 14.0\%$
RF	RFF35	$86.9\% \pm 4.6\%$	$54.0\% \pm 20.4\%$

6.1 Support Vector Machine

The radial basis function kernel was heuristically selected as the kernel used by SVM, because it appeared to produce the best results.

The best results for SVM were achieved by the RFFn feature sets. These results are displayed in table 2 below. The first column holds the name of the feature set; in this case the number after "RFF" indicates how many of the most important features this feature set contains, e.g.,

"RFF10" contains the top ten most important features. The two last columns contain the hyperparameter values.

By balancing the class weights the recall score of the RFF35 model was improved significantly with a minor decrease in accuracy.

Table 2. Results for SVM using the feature sets generated by Random Forest. These are the average results over 100 cycles.

Feature set	Accuracy	Recall	C	gamma
RFF10	$85.7\% \pm 5.7\%$	$57.5\% \pm 20.6\%$	8000	0.05
RFF20	$86.5\% \pm 5.0\%$	$61.4\% \pm 20.6\%$	30	1.0
RFF30	$89.9\% \pm 4.6\%$	$73.8\% \pm 17.5\%$	30	1.1
RFF35	$89.8\% \pm 4.7\%$	$75.9\% \pm 17.1\%$	30	1.09
RFF40	$89.5\% \pm 4.7\%$	$74.5\% \pm 17.1\%$	30	0.9
RFF35 Bal	$89.7\% \pm 4.0\%$	$84.8\% \pm 14.0\%$	1	1

Table 3 displays the results obtained with SVM by using the rest of the feature sets. The best accuracy and recall score were obtained with the TR feature set.

Table 3. Results for SVM using the generated feature sets apart from RFFn.

Feature set	Accuracy	Recall	C	gamma
AVG	$85.0\% \pm 2.1\%$	$42.8\% \pm 18.1\%$	100000	0.05
TMA	$80.9\% \pm 2.8\%$	$46.5\% \pm 19.6\%$	500	0.09
TM	$78.2\% \pm 3.9\%$	$38.5\% \pm 19.3\%$	1000	0.001
TMH	$85.0\% \pm 3.1\%$	$41.6\% \pm 19.5\%$	200	0.009
TR	$86.4\% \pm 1.8\%$	$55.7\% \pm 6.4\%$	50000	0.1

Presented in figure 9 are the recall score results obtained with SVM and the RFF35 feature set. On the y-axis is the recall score for the class of dyslexics. The x-axis displays the number of the hyperparameter combination. The hyperparameter values used are below

$$C: 1, 10, 20, 30, 50, 75, 100, 300, 500, 1000, 5000, 10000$$
$$\gamma: 0.00001, 0.0001, 0.001, 0.01, 0.1, 0.4, 0.5, 0.55, 0.6, 0.65, 0.8, 0.9, 1, 1.08, 1.09, 1.1, 1.11 \tag{6.1}$$

The large variation of the recall score is explained by the algorithm cycle, which first goes through the values of the C hyperparameter and then changes gamma to the next value. Each dip in the graph is due to a too small value of C.

It is important to notice how much the chosen hyperparameters affect the performance of the model. This exemplifies the importance of doing a thorough grid-search in search of the best hyperparameter values regarding the need of the model. In our case, we deemed the correct classification of the students with dyslexia the most important goal. In addition, we also wanted to achieve a good general accuracy for the model.

6.2 Random Forest

The best results obtained by the Random Forest classifier are displayed in table 4. The two last columns contain the hyperparameters optimized with grid-search and used by each model.

As can be seen, the results are not as good as with SVM. Even by using the RFF35 feature set the results did not improve much.

Table 4. Results for RF using the generated feature sets.

Feature set	Accuracy	Recall	Max_features	n_estimators
AVG	$80.7\% \pm 5.5\%$	$50.2\% \pm 19.2\%$	4	10
TMA	$83.6\% \pm 5.0\%$	$41.3\% \pm 19.2\%$	18	30
TM	$81.7\% \pm 5.3\%$	$36.9\% \pm 20.1\%$	240	20
TMH	$84.5\% \pm 4.6\%$	$39.9\% \pm 19.2\%$	550	20
TR	$86.7\% \pm 1.1\%$	$36.3\% \pm 5.1\%$	24	20
RFF35	$85.4\% \pm 1.1\%$	$42.6\% \pm 19.7\%$	5	20

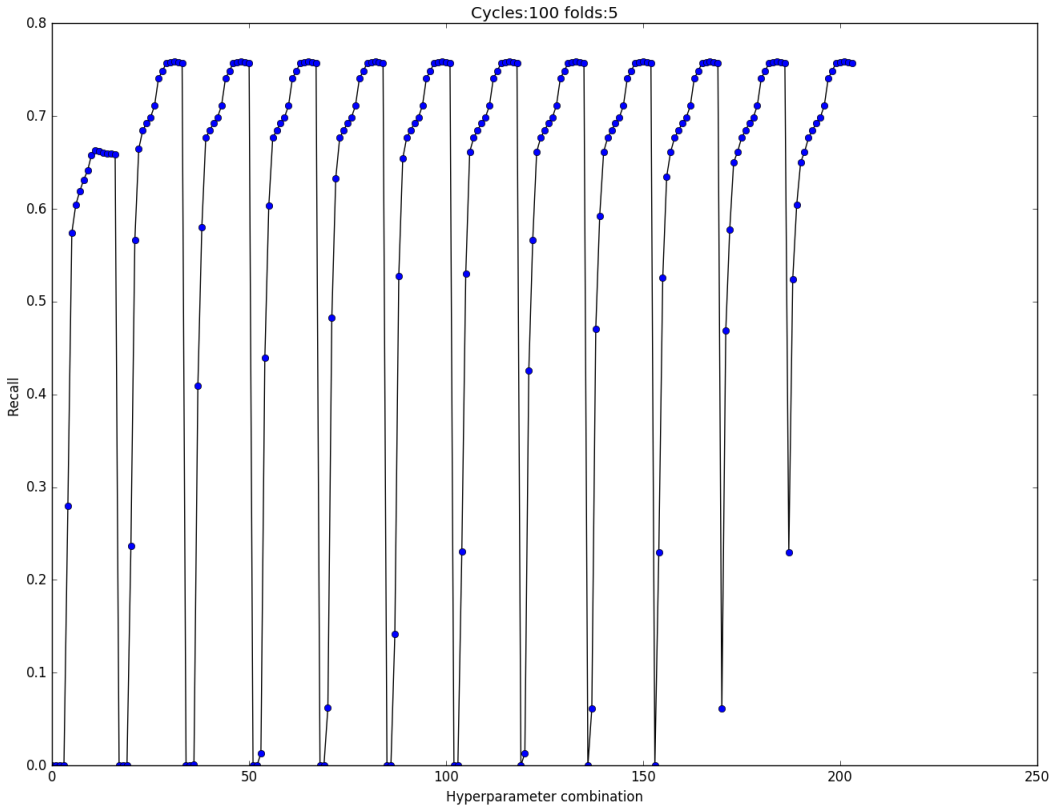

Figure 9. This chart illustrates the variance in the recall score of the dyslexia class with different hyperparameter values.

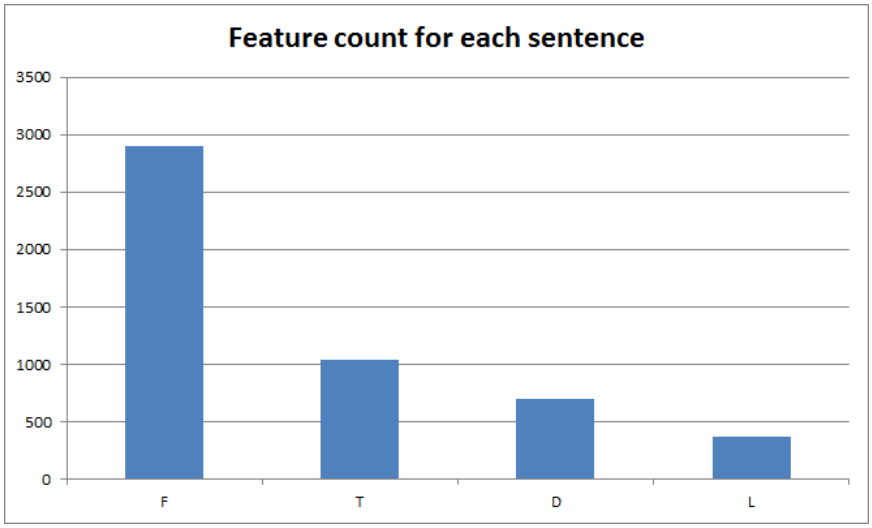

Figure 10. This chart shows how many of the top 10 features in each fold rotation (500 rotations in total) were related to each sentence.

6.3 Feature observations

By analysing the most relevant features obtained in the RFF set, interesting observations were made. Figure 10 displays the amount of features related to each sentence chosen for the top 10 most important features each fold rotation. The total amount of features chosen is $10 * folds * cycles = 10 * 5 * 100 = 5000$. Of these, 2900 (58%) were related to the first sentence on the task question page. This relation means that the feature was generated from gaze activity in the first sentence. Figure 10 shows that the rest of the sentences had a much lesser effect in creating features that help separate the two classes. This is an observation that the psychology department researchers in our team had also noted in their work.

The different trials were also shown to influence feature importance. Figure 11 presents the amount of features related to each trial chosen for the top 10 each fold rotation. The total amount of fold rotations is again 5000. The features generated from T2 data occur most often (32%), indicating a high significance in classifying the two classes correctly. For the rest of the trials, the feature count stays somewhat in the same range, with low points at T6 and T9. The high importance of T2 is speculated to be the result of the participants not having established a context for the text read. Knowing the context helps readers read faster as they are able to predict upcoming words (Hawelka et al. 2015). But readers with a reading

disorder suffer more from not knowing the context of the text than fluent readers. This could explain the importance of T2 trial features in separating the two classes; at this point of the research the participants had not yet seen enough trials to establish the context and form of the question page text. Later on in the experiment, the context is established and thus it is harder to distinguish the readers with difficulties from the normal.

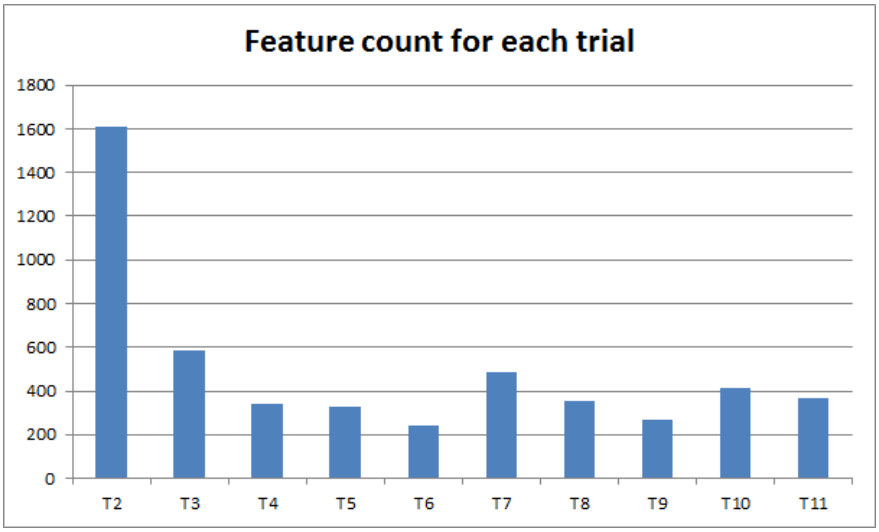

Figure 11. This chart shows how many of the top 10 features in each fold rotation (500 rotations in total) were related to each trial.

The feature occurrences in each fold rotation were also proven to display certain phenomena. Figure 12 shows the amount of times the most important features were picked. We can see that features concerning the first sentence (indicated by an "F") and ones from T2 have occured most often, indicating their importance, as stated above. In addition, by looking at the histogram bin numbers in the feature names, we can also see that in the case of saccadic features, the most frequent bin is the first. Respectively, for the features that are created from fixation data, the most important bin is the last. These observations indicate that the shortest saccades and the longest fixations help the classification the most. This is a conclusion that agrees with the results obtained by Deans et al. (2010) and De Luca et al. (2002).

We can also notice that features extracted from saccadic data are more important than ones from fixation data. The use of transition matrices did not contribute greatly to the classification; only one feature in the 35 most important features is from a traditional transition matrix. The other three features in this list (T2F-F, T10T-T and T3D-D) are the amounts of

fixations made on the indicated sentence.

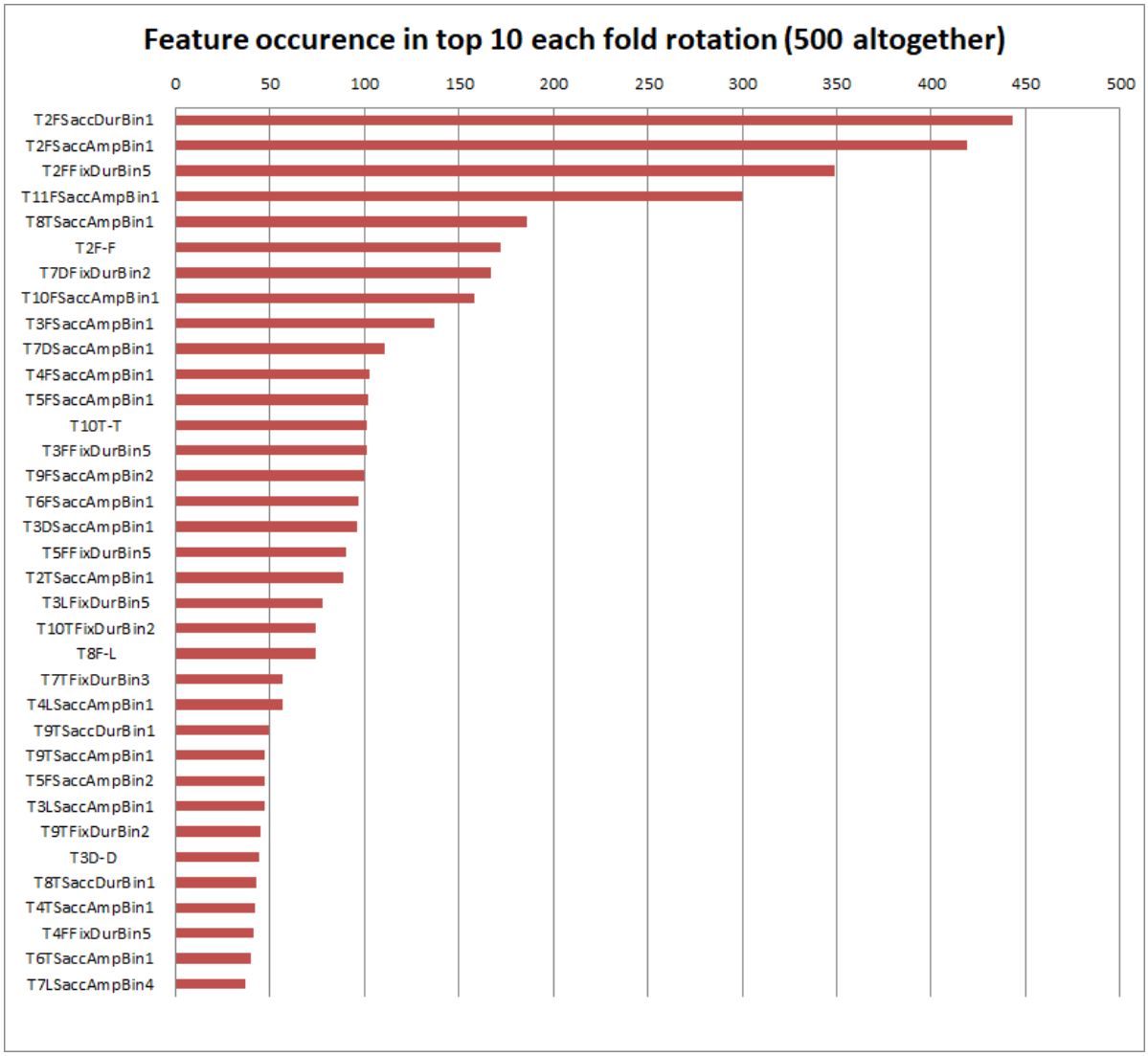

Figure 12. This chart displays the occurence of each feature in the top 10 most important features each cycle (100 in total).

6.4 Additional results

A quick test was conducted on detecting attentional difficulties with the TMH feature set. Random Forest was used, as earlier, to first measure the feature importance's and then the top 30 features were given to SVM for creating models. The best result gained was with an accuracy of $81.4\% \pm 6.4\%$ and a recall of $44.7\% \pm 21.2\%$. The value used to define

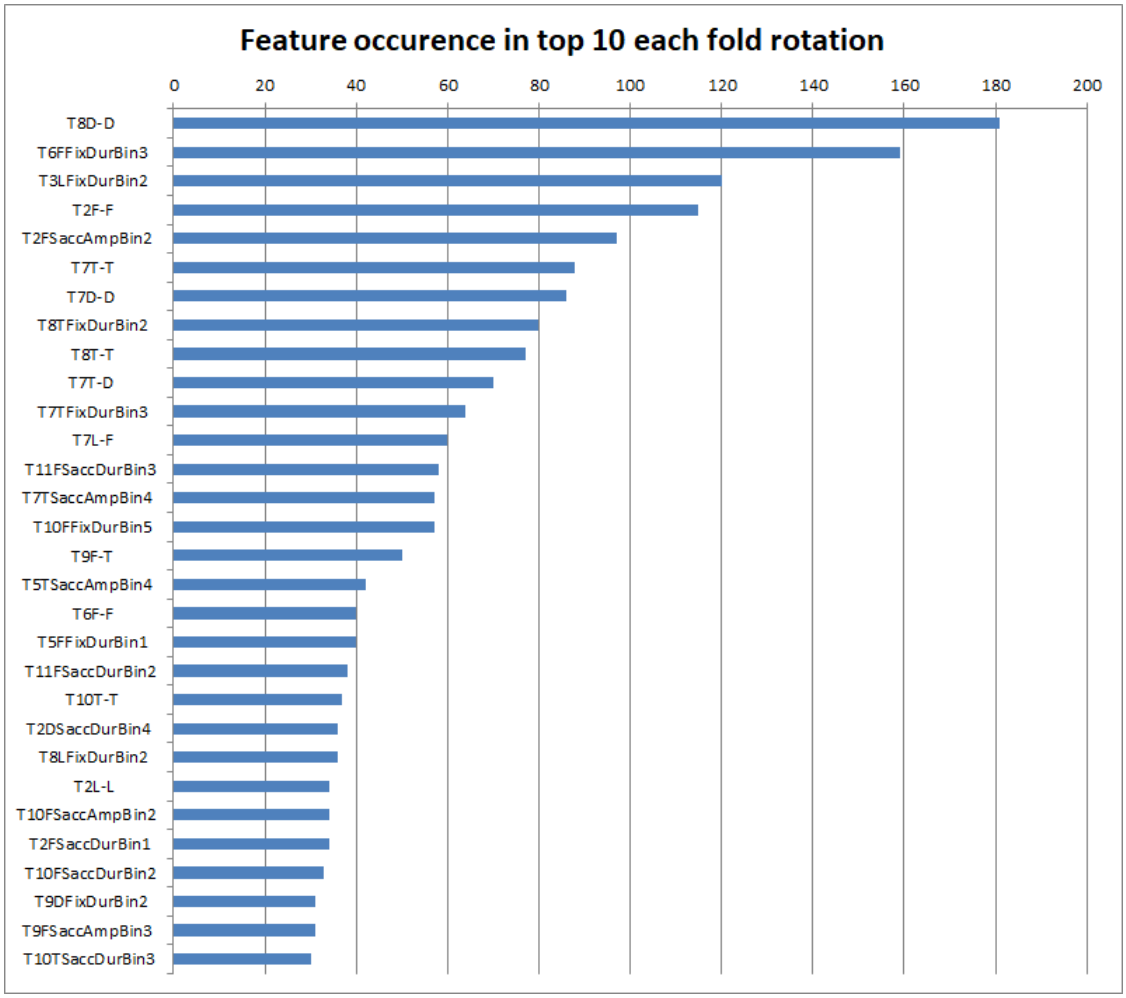

Figure 13. This chart displays the occurence of each feature in the top 10 most important features each cycle (100 in total). These results are for attentional difficulties.

the classes of normal students and those with attentional difficulties was obtained with the ATTEX test (Klenberg et al. 2010). As expected, the accuracy and recall scores indicate that the same features used for predicting dyslexia do not work as well for predicting attentional difficulties. The feature occurences that are presented in figure 13 show that the features from the transition matrix were more useful in detecting attentional difficulties than dyslexia. This indicates that there could be more value in using transition matrices for detecting attentional difficulties.

7 Conclusion and future work

In this study, we set out to produce a machine learning model capable of reliably detecting students with reading disorders using eye movement data. The machine learning algorithms selected were the Support Vector Machine and Random Forest. Using the design science principles in an iterative fashion, we were able to achieve our goal and also gain more knowledge regarding the problem. The best model, with an accuracy of 89.8%, was achieved with the Support Vector Machine by using a feature set created from the most relevant eye movement features selected by Random Forest.

This study showed very promising results in being able to detect students with reading disorders based on their eye movements. A simpler experiment could be set up with the participants concentrating on reading more text. From the eye movement data obtained this way, a similar set of the most important features could be extracted by using the histogram matrix feature set and Random Forests feature importance measurement. The model created this way could then possibly be used as a pre-screening tool for dyslexia detection.

The results obtained by the best RFFn models also reflect the possibilities of fine-tuning. The different hyperparameter combination possibilities were not searched through exhaustively. By, for example, testing values of C between 1.0 and 1.1, and possibly going even further, it could be possible to slightly improve the models accuracy. Regarding the most important features selected (top 35 in the case of the best model), it can be seen from the results that creating models with the top 31, 32, 33 or 34 features could possibly yield better results. In addition to fine-tuning the model, removing possible outliers, i.e., exceptions in the data that do not obey the general rule, from the data could very likely improve the results.

Another factor likely affecting the results are the data cells with missing values. As stated in section 5.1, these missing values were replaced by zeroes. However, methods exist to imputate data into these missing values. These algorithms were considered but left out due to time constraints.

For future work and the use of the results in this study, it is important to take into account the Finnish language orthography. Finnish language, along with Spanish, Greek, Italian and

Germany, has a transparent orthography (Serrano and Defior 2008). This means that the written symbols (graphemes) correspond to the spoken sounds (phonemes) of the language. English, on the other hand, has an opaque orthography, i.e., graphemes may correspond to the same phonemes and vice versa. Serrano and Defior state that "in languages with a more transparent orthography like Spanish, dyslexia seems to involve less severe deficits than those found in opaque writing systems". Additionally, in transparent orthographies, reading accuracy appears to be less important than reading speed in detecting dyslexia (Serrano and Defior 2008). Thus, the results of this study may not be directly applicable to languages of more opaque orthographies.

Additionally, it is worth mentioning that using Random Forest as the feature selection method is most probably not optimal. Better suited feature selection algorithms most likely exist. The article by Dash and Liu (1997) contains a comprehensive overview of the various methods. Selecting a group of different algorithms and comparing the feature sets created by them could prove to be fruitful. Our choice of using Random Forests feature importance measurements for feature selection was based on ease of use as Random Forest was already implemented in our algorithm.

Using the design science principle proved to be useful for this study. The problem-solving process in our research was improved by each new result obtained by the designed machine learning artifact. By using iterations, we were able to gain new knowledge of the problem in each cycle and improve the method used to create the feature sets and the validation algorithm. Without this cyclic motion obtaining the results would have been a great deal more difficult.

The short test conducted to detect attentional difficulties indicated the potential usefulness of transition matrices. It has been shown that in the case of ADHD, individuals have difficulties controlling their eye movements (Munoz et al. 2003). Therefore, by creating a suitable feature set based on transition matrices and other applicatory data, detecting attentional difficulties reliably from eye movements could prove to be possible with a similar approach. Using the design science iterations could also possibly lead to new information regarding this problem.

Bibliography

Aizerman, MA, È M Braverman, and LI Rozonoer. 1964. “Theoretical foundations of potential function method in pattern recognition”. *Automation and Remote Control* 25 (6): 917–936.

Alpaydin, Ethem. 2009. *Introduction to machine learning.* MIT press.

———. 2014. *Introduction to Machine Learning.* MIT Press.

Belgiu, Mariana, and Lucian Drăguţ. 2016. “Random forest in remote sensing: A review of applications and future directions”. *ISPRS Journal of Photogrammetry and Remote Sensing* 114:24–31.

Benfatto, Mattias Nilsson, Gustaf Öqvist Seimyr, Jan Ygge, Tony Pansell, Agneta Rydberg, and Christer Jacobson. 2016. “Screening for dyslexia using eye tracking during reading”. *PloS one* 11 (12): e0165508.

Bergstra, James, and Yoshua Bengio. 2012. “Random search for hyper-parameter optimization”. *Journal of Machine Learning Research* 13 (Feb): 281–305.

Boser, Bernhard E, Isabelle M Guyon, and Vladimir N Vapnik. 1992. “A training algorithm for optimal margin classifiers”. In *Proceedings of the fifth annual workshop on Computational learning theory,* 144–152. ACM.

Boulesteix, Anne-Laure, Silke Janitza, Jochen Kruppa, and Inke R König. 2012. “Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics”. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2 (6): 493–507.

Breiman, Leo. 2001. “Random forests”. *Machine learning* 45 (1): 5–32.

Chang, Chih-Chung, and Chih-Jen Lin. 2011. “LIBSVM: a library for support vector machines”. *ACM transactions on intelligent systems and technology (TIST)* 2 (3): 27.

Claesen, Marc, and Bart De Moor. 2015. “Hyperparameter search in machine learning”. *arXiv preprint arXiv:1502.02127.*

Cortes, Corinna, and Vladimir Vapnik. 1995. “Support-vector networks”. *Machine learning* 20 (3): 273–297.

Cutler, Adele, D Richard Cutler, and John R Stevens. 2012. “Random forests”. In *Ensemble machine learning,* 157–175. Springer.

Dash, Manoranjan, and Huan Liu. 1997. “Feature selection for classification”. *Intelligent data analysis* 1 (1-4): 131–156.

De Luca, Maria, Marta Borrelli, Anna Judica, Donatella Spinelli, and Pierluigi Zoccolotti. 2002. “Reading words and pseudowords: An eye movement study of developmental dyslexia”. *Brain and language* 80 (3): 617–626.

Deans, Pamela, Liz OLaughlin, Brad Brubaker, Nathan Gay, and Damon Krug. 2010. “Use of eye movement tracking in the differential diagnosis of attention deficit hyperactivity disorder (ADHD) and reading disability”. *Psychology* 1 (04): 238.

Eden, GF, JF Stein, HM Wood, and FB Wood. 1994. “Differences in eye movements and reading problems in dyslexic and normal children”. *Vision research* 34 (10): 1345–1358.

Frazier, Marilyn. 2016. *Dyslexia : perspectives, challenges and treatment options.* New York Nova Biomedical. ISBN: 9781634853286 (hardcover).

Glazzard, Jonathan. 2010. “The impact of dyslexia on pupils’ self-esteem”. *Support for learning* 25 (2): 63–69.

Guyon, Isabelle, and André Elisseeff. 2003. “An introduction to variable and feature selection”. *Journal of machine learning research* 3 (Mar): 1157–1182.

Handler, Sheryl M, Walter M Fierson, et al. 2011. “Joint technical report—Learning disabilities, dyslexia, and vision”. *Pediatrics:* peds–2010.

Hautala, Jarkko. 2012. *Visual word recognition in fluent and dysfluent readers in the transparent Finnish orthography.* University of Jyväskylä.

Hautala, Jarkko, Carita Kiili, Yvonne Kammerer, Otto Loberg, Sanna Hokkanen, and Paavo HT Leppänen. 2018. “Sixth graders’ evaluation strategies when reading Internet search results: an eye-tracking study”. *Behaviour & Information Technology:* 1–13.

Hawelka, Stefan, Sarah Schuster, Benjamin Gagl, and Florian Hutzler. 2015. “On forward inferences of fast and slow readers. An eye movement study”. *Scientific reports* 5:8432.

Holmqvist, Kenneth, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. 2011. *Eye tracking: A comprehensive guide to methods and measures.* OUP Oxford.

Holmström, Jan, Mikko Ketokivi, and Ari-Pekka Hameri. 2009. “Bridging practice and theory: a design science approach”. *Decision Sciences* 40 (1): 65–87.

Hyönä, Jukka, and Richard K Olson. 1995. “Eye fixation patterns among dyslexic and normal readers: Effects of word length and word frequency.” *Journal of Experimental Psychology: Learning, Memory, and Cognition* 21 (6): 1430.

Katusic, Slavica K, Robert C Colligan, William J Barbaresi, Daniel J Schaid, and Steven J Jacobsen. 2001. “Incidence of reading disability in a population-based birth cohort, 1976–1982, Rochester, Minn”. In *Mayo Clinic Proceedings,* 76:1081–1092. 11. Elsevier.

Klenberg, Liisa, Sari Jämsä, Taru Häyrinen, Pekka Lahti-Nuuttila, and Marit Korkman. 2010. “The Attention and Executive Function Rating Inventory (ATTEX): Psychometric properties and clinical utility in diagnosing ADHD subtypes”. *Scandinavian Journal of Psychology* 51 (5): 439–448.

Kohavi, Ron, et al. 1995. “A study of cross-validation and bootstrap for accuracy estimation and model selection”. In *Ijcai,* 14:1137–1145. 2. Montreal, Canada.

Kotsiantis, Sotiris B, I Zaharakis, and P Pintelas. 2007. “Supervised machine learning: A review of classification techniques”. *Emerging artificial intelligence applications in computer engineering* 160:3–24.

Lefton, Lester A, Richard J Nagle, Gwendolyn Johnson, and Dennis F Fisher. 1979. “Eye movement dynamics of good and poor readers: Then and now”. *Journal of Reading Behavior* 11 (4): 319–328.

Louppe, Gilles. 2014. “Understanding random forests: From theory to practice”. *arXiv preprint arXiv:1407.7502.*

Lustig, Joakim. 2016. “Identifying dyslectic gaze pattern: Comparison of methods for identifying dyslectic readers based on eye movement patterns”.

Lyon, G Reid, Sally E Shaywitz, and Bennett A Shaywitz. 2003. “A definition of dyslexia”. *Annals of dyslexia* 53 (1): 1–14.

MacFarlane, Andrew, Areej Al-Wabil, Chloe Ruth Marshall, A Albrair, Susan A Jones, and Panayiotis Zaphiris. 2010. “The effect of dyslexia on information retrieval: A pilot study”. *Journal of Documentation* 66 (3): 307–326.

Mantau, Aprinaldi Jasa, et al. 2017. “Multiclass classification of cancer based on microarray data using extreme learning machine”. In *2017 1st International Conference on Informatics and Computational Sciences (ICICoS),* 159–164. IEEE.

Morris, David, and Patricia Turnbull. 2007. “A survey-based exploration of the impact of dyslexia on career progression of UK registered nurses”. *Journal of Nursing Management* 15 (1): 97–106.

Mountrakis, Giorgos, Jungho Im, and Caesar Ogole. 2011. “Support vector machines in remote sensing: A review”. *ISPRS Journal of Photogrammetry and Remote Sensing* 66 (3): 247–259.

Munoz, Douglas P, Irene T Armstrong, Karen A Hampton, and Kimberly D Moore. 2003. “Altered control of visual fixation and saccadic eye movements in attention-deficit hyperactivity disorder”. *Journal of neurophysiology* 90 (1): 503–514.

Noble, William Stafford, et al. 2004. “Support vector machine applications in computational biology”. *Kernel methods in computational biology* 71:92.

Osuna, Edgar, Robert Freund, and Federico Girosit. 1997. “Training support vector machines: an application to face detection”. In *Computer vision and pattern recognition, 1997. Proceedings., 1997 IEEE computer society conference on,* 130–136. IEEE.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. “Scikit-learn: Machine learning in Python”. *Journal of machine learning research* 12 (Oct): 2825–2830.

Rayner, Keith. 1998. “Eye movements in reading and information processing: 20 years of research.” *Psychological bulletin* 124 (3): 372.

Rello, Luz, and Miguel Ballesteros. 2015. “Detecting Readers with Dyslexia Using Machine Learning with Eye Tracking Measures”. In *Proceedings of the 12th Web for All Conference,* 16:1–16:8. W4A ’15. Florence, Italy: ACM. ISBN: 978-1-4503-3342-9. doi:10.1145/2745555.2746644.http://doi.acm.org/10.1145/2745555.2746644.

Serrano, Francisca, and Sylvia Defior. 2008. “Dyslexia speed problems in a transparent orthography”. *Annals of dyslexia* 58 (1): 81.

Shaywitz, Sally E. 1998. “Dyslexia”. *New England Journal of Medicine* 338 (5): 307–312.

Snowling, Margaret J, and Charles Hulme. 2012. “Interventions for children’s language and literacy difficulties”. *International Journal of Language & Communication Disorders* 47 (1): 27–34.

Torgesen, Joseph K. 2000. “Individual differences in response to early interventions in reading: The lingering problem of treatment resisters”. *Learning Disabilities Research & Practice* 15 (1): 55–64.

Undheim, Anne Mari. 2009. “A thirteen-year follow-up study of young Norwegian adults with dyslexia in childhood: reading development and educational levels”. *Dyslexia* 15 (4): 291–303.

Wang, Kai, Boris Babenko, and Serge Belongie. 2011. “End-to-end scene text recognition”. In *Computer Vision (ICCV), 2011 IEEE International Conference on,* 1457–1464. IEEE.

Vapnik, Vladimir Naumovich. 1999. “An overview of statistical learning theory”. *IEEE transactions on neural networks* 10 (5): 988–999.

Vellutino, Frank R, Jack M Fletcher, Margaret J Snowling, and Donna M Scanlon. 2004. “Specific reading disability (dyslexia): What have we learned in the past four decades?” *Journal of child psychology and psychiatry* 45 (1): 2–40.

Verikas, Antanas, Adas Gelzinis, and Marija Bacauskiene. 2011. “Mining data with random forests: A survey and results of new tests”. *Pattern recognition* 44 (2): 330–349.

Yang, Jing, Dengju Yao, Xiaojuan Zhan, and Xiaorong Zhan. 2014. “Predicting disease risks using feature selection based on random forest and support vector machine”. In *International Symposium on Bioinformatics Research and Applications,* 1–11. Springer.