— Grau
en Llengües
Aplicades

Treball de Fi de Grau

From data to pre-diagnosis: an evaluation of Machine learning models for dyslexia detection

Alejandra Duque Maldonado

Tutor/a: Thomas Brochhagen
Seminari 104: Llengües i tecnologia

Curs 2022-2023

upf. Universitat Pompeu Fabra Barcelona
Facultat de Traducció i Ciències del Llenguatge

ABSTRACT

Dyslexia is a learning difficulty related to the processing of language that affects approximately 10% of the Spanish population. Early detection of dyslexia is crucial to prevent academic failure or abandonment. In the present day, there are automated methods, such as machine learning methods, to process data, recognize patterns and detect pathologies. A previous study by Rello (2020) demonstrated that it is possible to apply machine learning to the detection of dyslexia. Rello created an open access tool through which, by performing a linguistic-based test, users can learn whether they might suffer from this disorder. Rello (2020) investigates the use of an algorithm that, from the processing of data, classifies users in the "Dyslexia" or "No dyslexia" classes. The goal of this thesis is to replicate the algorithm by Rello and contrast if other machine learning methods could be effective in detecting dyslexia. By training three supervised learning algorithms, we have obtained an algorithm with 92% of sensitivity for the "Dyslexia" class. More generally, this study demonstrates that other methods can also be satisfactory in the detection of dyslexia, and that the selection of parameters can influence the improvement of the results of a model.

Keywords: machine learning, dyslexia, detection, language pathologies, classification algorithms.

RESUM

La dislèxia és una dificultat d'aprenentatge relacionada amb el processament del llenguatge que afecta aproximadament el 10% de la població espanyola, per tant, la seva detecció precoç és essencial per prevenir el fracàs o l'abandonament escolar. Actualment, hi ha mètodes automatitzats, com els mètodes d'aprenentatge automàtic, per a processar dades, reconèixer patrons i detectar patologies. Un estudi previ realitzat per Rello (2020) ha demostrat que és possible aplicar l'aprenentatge automàtic a la detecció de la dislèxia. Rello desenvolupa una eina de lliure accés gràcies a la qual, en fer una prova amb base lingüística, els usuaris poden conèixer si pateixen aquest trastorn. Rello (2020) investiga l'ús d'algoritme que a partir del processament de dades, classifica els usuaris dins de les categories "Dislèxia" o "No dislèxia". Aquest treball té com a objectiu replicar aquest algoritme utilitzat per Rello i contrastar si altres algoritmes d'aprenentatge automàtic poden resultar efectius en la detecció de la dislèxia. A partir de l'entrenament de tres algoritmes d'aprenentatge supervisat, hem obtingut com a

resultat un algoritme que presenta un 92% de sensibilitat a la classe “Dislèxia”. L’estudi demostra que altres mètodes poden ser satisfactoris en la detecció de la dislèxia i que la selecció de paràmetres pot incidir en la millora dels resultats d’un model.

Paraules clau: aprenentatge automàtic, dislèxia, detecció, patologies del llenguatge, algoritmes de classificació.

RESUMEN

La dislexia es una dificultad de aprendizaje relacionada con el procesamiento del lenguaje que afecta alrededor del 10% de la población española, cuya detección temprana es esencial para prevenir el fracaso o abandono escolar. Actualmente, existen métodos automatizados, como los métodos de aprendizaje automático, para procesar datos, reconocer patrones y detectar patologías. Un estudio previo realizado por Rello (2020) demuestra que es posible aplicar el aprendizaje automático en la detección de dislexia. Rello desarrolló una herramienta de libre acceso donde, al realizar una prueba con base lingüística, los usuarios pueden saber si padecen de este trastorno. Rello (2020) investiga el uso de un algoritmo para la detección de dislexia, donde a partir del procesamiento de datos, se clasifica a los usuarios en las categorías “Dislexia” o “No dislexia”. Este trabajo tiene como objetivo replicar el algoritmo utilizado por Rello y contrastar si otros métodos podrían resultar efectivos en la detección de dislexia. A partir del entrenamiento de tres algoritmos de aprendizaje supervisado, obtuvimos un algoritmo con 92% de sensibilidad de la clase “Dislexia”. El estudio demuestra que otros métodos pueden resultar satisfactorios en la detección de la dislexia y qué la selección de los parámetros puede incidir en la mejora de los resultados de un modelo.

Palabras clave: aprendizaje automático, dislexia, detección, patologías del lenguaje, algoritmos de clasificación.

ACKNOLEDGEMENTS

I would like to thank my tutor, Thomas Brochhagen, for guiding me through the process of writing this thesis and supporting my ideas during the planning of this work.

I would also like to dedicate this work to my family, who always support me and motivate me into giving the best of myself in everything I do.

I want to thank my partner, who has motivated me to make my best effort and supported my initiatives no matter what.

Finally, I want to thank my friends and my classmates, who have motivated me through this process and have accompanied me at all times in this last year.

TABLE OF CONTENTS

1. INTRODUCTION

We live in a digital era where artificial intelligence enhances our lifestyles and helps us solve problems in more effective ways, no matter the application. In recent years, we have experienced the rise of artificial intelligence technologies in our environment, as these technologies have been developed to find solutions to different problems in an automated way. Artificial intelligence (AI) refers to the capability of computers to understand human intelligence and focuses on the use of machines to solve real-world problems (McCarthy, 1999). AI is a combination of different technologies that can go from processing natural language to performing statistical analysis, as we see with machine learning.

Machine learning is a branch of AI that focuses on the use of algorithms to process data, usually in large quantities. These are statistical methods designed to classify, predict, find patterns… It attempts to make a machine "learn" based on the training of an algorithm with data, in order for it to generalize towards new information. Although this has had many applications, the most distinctive have been in the medical field, as these algorithms were initially designed to process large quantities of medical data and perform diagnosis for different conditions, such as cancer. These algorithms have been largely developed throughout the years and can go from simple methods (e.g., k-nearest neighbors) to more complex algorithms (e.g., neural networks). In this thesis, we will study how machine learning algorithms can be applied to the detection of language pathologies, since they are designed to deal with errors in data (noisy data or missing information), extract relevant information and provide transparent results (Kononenko, 2001). For the case of dyslexia, the goal is to replicate a previously developed algorithm and use its training data to determine if other machine learning algorithms can perform accurate detection of dyslexia.

Dyslexia (or specific reading impairment) is defined as a learning disability that affects the process of decoding and comprehension of language. This impairment influences visual and perceptual processes. As dyslexia patients have difficulties in reading (extracting meaning from printed words) and developing phonological awareness, they often struggle to relate sounds of spoken language to their graphemic representation in print (Shaywitz, 1998). In other words, dyslexia refers to the difficulty to identify meaning when reading and producing language. Speakers struggle to segment words into smaller units (phonemes) and to understand the meaning behind the written representation of words. Some symptoms of this pathology include:

difficulty recognizing individual letters (graphemes) and its related sounds (phonemes), slow reading and writing speed, difficulties in learning and processing new vocabulary, or poor orthographic skills. This pathology is typically detected during initial schooling periods, usually motivated by poor academic performance, but is not usually diagnosed before 7 or 8 years of age, as this is considered the period when an average person has reached the basic reading and writing skills used to acquire new curricular knowledge (Garriga & Salas, 2015). Diagnosing dyslexia in early schooling stages is crucial, as difficulties stemming from this impairment can affect academic performance. If dyslexia is not addressed at an early age, it could consequently lead to academic failure or abandonment.

Diagnosing this disorder is not trivial, it requires exhaustive professional analysis that is able to discard other possible disorders and evaluate different linguistic and cognitive abilities that could be influenced by dyslexia. These diagnoses often tests: abilities related to reading, such as oral comprehension and expression, short and long term memory, language processing skills (e.g., speed for word recognition); reading and writing abilities, like fluency, knowledge of the alphabet or recognitions of words and pseudowords; and non-linguistic skills that could be influenced by dyslexia, like arithmetic skills or cognitive abilities, such as, orientation, visual memory or processing skills (Garriga & Salas, 2015). These tests seek to determine what degree of dyslexia a person has, as it can go from influencing the ability to recognize phonemes, to having difficulty in recognizing words, or both.

The application of machine learning for the detection of dyslexia is not new, in fact, there has been previous work that has studied and developed an algorithm to address this pathology. Rello, L. developed a detection tool in her paper "*Predicting dyslexia with an online gamified test*" (2020) with the goal of determining whether dyslexia could be screened through a machine learning algorithm, which is trained with data obtained from gamified linguistic questions. The article considers the difficulties it supposes to screen dyslexia in languages with transparent orthographies, like Spanish, given that the correspondence between letter and sound is more consistent compared to languages with deep orthographies, as English, where the correspondence is not as explicit and makes people struggle in learning how to read, which makes dyslexia easier to screen (Rello, 2020). With this tool, Rello created a way to help Spanish speakers, who are suspected to have dyslexia, get a pre-diagnosis of this pathology. In a nutshell, the contribution of this article consists of a game designed to gather data and a machine learning algorithm that makes predictions from the input data.

It is important to note that, a contribution like the one of Rello (2020) provides great aid for the part of the population who might suspect to have this disorder but does not have direct access to professional help to address it. This is why it is crucial to assure a tool oriented to approach these sorts of problems can provide accurate information, as it could influence the decisions taken by the user when getting a result. In their paper, Rello introduces one algorithm, which in this thesis we want to replicate and contrast with other algorithms. We want to observe whether these models could perform well to the same problem and observe if the use of optimization methods can improve the results. We have found that we can obtain satisfactory results using other algorithms and shown that the optimization of their parameters can lead to a good detection of dyslexia.

2. THEORETICAL FRAMEWORK

There are a series of fundamental concepts behind machine learning algorithms that will help us have a better understanding of how they are constructed and applied to solve different problems. In this thesis, the fundamental concepts are the ones that were used to develop the models: the meaning of supervised learning, how these algorithms are trained and evaluated. The main reference employed to understand the foundations of machine learning are found in the book "*Pattern recognition and machine learning*" by Bishop (2006), who illustrates different kinds of machine learning algorithms (supervised, unsupervised and reinforcement learning) and provides an in-depth approach to all of them.

In this thesis, to face a classification problem regarding dyslexia, we use supervised learning techniques, one of the approaches an algorithm can take when learning from data. With these techniques, when given a set of data, the algorithm already knows which is the output (or label) of the data points. The aim of these methods is to be able to generalize when given new information. Their task is to assign a label to a new piece of information, based on the information it has been trained with. In our models, the supervised learning algorithms were designed to work as classifiers that can distinguish between two classes, "Dyslexia" and "No dyslexia", based on the information used in the training phase.

There are a series of phases in the construction of a machine learning algorithm, as the models need to be trained and validated to make sure that they are properly generalizing. The process of constructing a supervised learning algorithm usually follows the same phases, training and validation. Given a dataset with a determined number of samples, in the training phase, the data is split into sections that will be used to train the model (learn from it) and eventually test it (see how well it performs based on what it learnt). When training, a section of the samples will be processed with its labels (e.g., "Dyslexia" or "No dyslexia" for a given data point), allowing the model to learn from the input data. Once the algorithm is trained, with the test set, a portion of the data that has not been processed will be used to verify if the given model does well at generalizing, as the algorithm would be performing classification on new examples. The model is trained with a set of the data and tested with another, so the model is tested with pieces of data that it has not seen before; we do this so we can observe how the model classifies new data and determine if the classification provided is satisfactory.

Building the models relying on training only can work well if applied correctly, however, to make sure the results are the optimal for this application, it might be necessary to perform a validation phase. In this stage, the goal is to assess the model and its performance, and determine which are the best parameters that lead to a satisfactory predictive performance. This validation phase can have different approaches, depending on the type of assessment a model needs. In this thesis, two cross-validation techniques are used, K-fold and Grid search. With these techniques, it is possible to evaluate the general performance of a model when it is trained with different training sets and test its performance when the parameters are tuned; something we would not be able to see in the training phase, because we would not be able to distinguish if the model had adjusted too closely to the training data, i.e., overfitting.

In K-fold cross-validation, the main objective is to observe how the model performs under different training, where the data is partitioned into K number of folds and iterated K[1] number of times, with variation on the training examples. Here, given a number of samples, the data will be split similarly than in the training phase (as it would have training and test sets). The portion of the data used to "test" the model will become our assessment method, called the validation set. As shown in Figure 1, there are different iterations (labelled "run 1", "run 2",

[1] K being an arbitrary number that represents the number of folds and iterations. Generally speaking, smaller K provides more reliable estimates at the expense of an increase in computational time and resources required to obtain them.

"run 3" …), where a different portion of the data is reserved as a validation set, so in each iteration the model will be trained and validated with different conditions. This technique can help to assess the general performance of a model under different contexts. However, it does not provide information about which is the best combination of parameters that can give better results for the problem stated, it just gives information about the general performance of the algorithm used.

Figure 1. Illustration of K-fold validation (Bishop, 2006).

To find the optimal combination of parameters, one of the approaches cross-validation techniques can offer is the use of a Grid search, a method where different combinations of the hyperparameters of the model are tested. It provides another perspective to the assessment of the model, where we can see how the parameters affect the results. Authors Yang & Shami (2020) define hyperparameters as the data that configures the architecture of a model, as they are the pieces of information that modify the parameters of an algorithm and cannot be directly estimated from data, being the kind of information that shapes how data is processed within the algorithm. With this technique, a range of different values of these hyperparameters are tested with different combinations, in order to find the best combination of hyperparameters in the algorithm that provide an optimal performance of the model. The tuning of hyperparameters can boost the performance and adapt algorithms to have optimal performance in different applications.

For this analysis, it is essential to understand the importance of model selection in the development of a machine learning algorithm, and how the assessment and tuning of a model can lead to good performance for the application it has, which in this case would be a classification problem of dyslexia.

In the article by Rello (2020), they explain which parameters and hyperparameters were used to build the predictive model (further explanation in Section 3) and how they use K-fold validation as their method to assess it. We want to observe if the tuning of parameters would

make a difference in the results of dyslexia classification, and not limit the assessment of the model to a K-fold validation only, but rather perform other cross-validation methods in the hyperparameters that shape the models. We hypothesize that, by combining supervised learning algorithms with more than one validation technique, as the ones mentioned, it could be possible to obtain satisfactory results for the classification of the "Dyslexia" and offer other possibilities, in terms of classification models.

One popular theorem that can support this hypothesis is the No free lunch theorem (Wolpert & Macready, 1997), commonly applied in machine learning. This theorem establishes that the average performance of two different algorithms is the same, which means that one of those algorithms (algorithm A) performs better in a certain problem than the other one (algorithm B), then there must exist another problem where algorithm B performs better than algorithm A. By optimizing an algorithm, the performance for the detection of one problem can be better in comparison to the previous algorithm, however, the reverse does not necessarily mean the same thing; the algorithm can perform well in the detection of one matter and worse in the detection of the remaining problems. From this perspective, this theorem explains how there are no perfect algorithms that can detect problems at the same level, and how through optimization the performance of detection problems can be enhanced, but it does not mean there is a perfect algorithm that can satisfy every type of problem that could emerge.

With this background, what is intended in this thesis is to build supervised learning models different from the one presented in the article (Rello, 2020), perform the model selection techniques previously described to optimize the performance and see how these classifiers perform in the detection of dyslexia. The analysis of these models, in addition to the replication of the algorithm used by Rello (2020), will give further insight on how they perform when being trained with the same data and applied to the same problem, which will allow us to determine which model performs better.

Note that, the focus will be primarily on the algorithms used on the prediction of dyslexia rather than the design of the gamified test. This is because, in a first general assessment of the tool, it was considered that the test used to gather the data was accordingly designed to the context it was applied to. This game does not try to replace a professional diagnosis, but rather cover basic aspects that are considered in a dyslexia diagnosis (i.e., tasks to measure working memory, word comprehension and writing skills, etc.). In contrast, the machine learning

algorithm is a part of the tool that could potentially be optimized to provide better results when predicting from data. Thus, for this task, an optimization of the model will be attempted, along with the trial of other models to make a comparison on the performance.

2.1 State of the art

As mentioned in the previous section, the article "*Predicting dyslexia with an online gamified test*" by Rello (2020) will be taken as a main point of reference. This article proposes a way to screen dyslexia in Spanish by using a gamified test to collect data, which is processed through a machine learning algorithm to predict whether a user has dyslexia. This detection tool can then be partitioned into two main components, the tests and algorithm, which are designed to recognize specific aspects related to dyslexia and detect if a user has a risk of having this pathology.

The practical application of this tool can be seen in a mobile application titled *Dytective*[2], an online environment that offers around 42,000 games that allow users to improve their reading and cognitive abilities. In addition, this application also offers the detection tool mentioned in the article, allowing the users to have the option to perform this test and get an estimate of their risk level of suffering from dyslexia.

This gamified test is designed with a linguistic background and can be applied to speakers of all variants of the Spanish language. It targets language skills and cognitive processes, such as working memory and perceptual processes. The test is composed of 32 tasks that the user must complete in a limit of time, approximately fifteen minutes, with gradual increases in difficulty that facilitate the collection of relevant data. From exercises 1 to 21, the focus is on the auditory and visual discrimination and comprehension of linguistic elements; exercises 22 to 29 centre on the correction of errors in words and sentences; and exercises 30 to 32 are short writing tasks of words and pseudo-words that put cognitive processes to the test, targeting visual and auditory working memory. It is with this test that different indicators can be measured, for the algorithm to distinguish between two fixed classes, "Dyslexia" and "No dyslexia", and provide a prediction of a possible pre-diagnosis.

[2] https://www.changedyslexia.org/

The dataset gathered from the tests is, besides the portion of data that describes the profile of each participant (age, gender, native language, etc.), a set of numeric representations of their performance on the tests, such as the number of clicks they made to solve a task, or how many hits or misses they had on each exercise (further explained in Section 3). The algorithm employed in the article is a supervised machine learning model called Random Forest, chosen because it can easily adapt to the data and be interpreted intuitively. The building of the model follows conventional stages of building a supervised learning model (training and validation phases) and is oriented towards a fixed goal: distinguish between the "Dyslexia" and "No dyslexia" classes and be able to generalize when new data is introduced.

All in all, it is important to mention that the authors put emphasis on the fact that this test cannot be taken as a diagnosis. No matter how accurate the results might be, the authors mention this test should be taken only as a tool to screen dyslexia and cannot replace a professional diagnosis. This test does not consider other factors that could be relevant in a dyslexia diagnosis, i.e., the individual difficulties a child can have when reading or writing. Moreover, this test does not discriminate or consider other types of conditions that could be present, such as ADHD (Attention deficit/Hyperactivity disorder), dyscalculia (difficulty in mathematical comprehension) or other language difficulties, like Specific language impairment (SLI). Therefore, this gamified test is conceived mainly to serve as a "pre-diagnosis" in case there are any suspects of a child having dyslexia.

Overall, the game and the tool proposed in the article offer a very accessible way for users to have some guidance on this matter. The game covers the main aspects involved in a dyslexia diagnosis, a combination of tests that compile information about cognitive processes and linguistics abilities, which provide relevant information about the user's abilities and allow for a model to make predictions that should be used as guidance.

3. METHODOLOGY

This section describes the methodology followed for the implementation of the different models used in this classifying task. As mentioned before, we aim to determine if other machine learning algorithms can provide good results on the detection of dyslexia, for which we trained

different models using the same data, and replicated the model by Rello, to observe which one performed better.

The article (Rello, 2020) provides information about the model used for this task, the parameters that shape this algorithm and the dataset used to train the model, which can be found online. In this thesis, to observe how this algorithm performed on data, the model from this article was replicated with its parameters and trained with their dataset. Subsequently, three other algorithms were trained with the same dataset, using two different supervised learning models and a modified version of the model from the article.

These three models especially emphasize in the use of cross-validation of the hyperparameters for the optimization of results. As mentioned before, the algorithm by Rello (2020) only makes use of one validation technique and, for this analysis, we want to test if the use of other validation techniques lead to a good predictive model.

The algorithms were trained to distinguish between two established classes, "Dyslexia" and "No dyslexia". The process included: preparing the data for processing, training, and evaluation of the model. In the following subsections, we give further explanation of the process, including individual descriptions of the models and the reasoning behind the tuning of its hyperparameters.

3.1 Dataset description

The dataset used for training the models and replicating the algorithm from the article (Rello, 2020) was retrieved from a Kaggle[3] post that stored the data used for the training of Rello's algorithm (also referenced in bibliography). The information in this dataset was obtained by testing several participants with the gamified test, which collected relevant data that summarized the performance of the participants on the test.

This dataset presents categorical and numeric data of a total of 3,644 participants, all of them from native Spanish speakers with an age range of 7 to 17 years old. Around 10.7% of them

[3] https://www.kaggle.com/datasets/luzrello/dyslexia

(392 participants) have been professionally diagnosed with dyslexia (Rello, 2020). For each participant, there are a total of 196 features that illustrate their profile, performance, and interaction with the game, along with a label, which places the participant in one of the categories, “Dyslexia” or “No dyslexia”.

The first four features of the data contain categorical variables, information about the participant’s profiles. This section of the data includes the age of the participants, and three binary variables: gender (female or male), native language (yes or no) and language subject (yes or no), which refers to whether the participant had failed the Spanish subject at school.

For the rest of the data (from data point 5 to 196), there is information about the performance in the 32 linguistic exercises from the test. For each of the tasks, six different measures were taken: *clicks*, number of clicks; *hits*, number of correct answers; *misses*, number of incorrect answers; *score*, sum of hits per exercise set; *accuracy*, number of hits divided by the number of clicks; and finally, *miss rate*, number of misses divided by the number of clicks. As mentioned above, each of the sections (or set of games) is focused on a different skill and measures something different: from linguistic skills, such as phonological and alphabetical awareness, to cognitive skills, like executive functions. This is the data that will provide the most relevant information and help the models discriminate between the two fixed classes.

In this dataset, we already have information about the category where each participant belongs to (“Dyslexia” and “No dyslexia”). Once the data is processed, the algorithm will be able to extract patterns that represent these two categories, therefore, when new data comes in, it will be able to categorize between one class or the other based on the training information it already has.

This data, however, cannot be processed by the models without previous preparation. In the next section, a description of the preprocessing will be presented, where data was previously prepared and normalized, so the algorithms could interpret them correctly.

3.2 Preprocessing of data

Prior to processing the data, we need to make sure the data presented to the model can fit in it correctly and be well interpreted by the algorithm. A common practice in machine learning is

to preprocess the data before applying it to a model, where the data can be scaled or normalized, so the algorithm can detect patterns in an easier way (Bishop, 2006). For the data used in the models, different preprocessing methods were applied, from cleaning to balancing and scaling the dataset. Some of these methods were generally applied to all the models and others were reserved for specific algorithms, however, all these steps are aimed to properly prepare the data for processing.

One important step in preprocessing is cleaning the data that will be used. This means, dealing with possible inconsistencies and removing errors in the dataset to improve the quality of the data and avoid wrong conclusions (Rahm, 2000). Cleaning data can be approached from different perspectives depending on how the data is presented; for this case, the categorical data presents inconsistencies, as it has a different format than the rest of features. As mentioned in the description of the dataset (Section 3.1.), there are two types of features for each participant, categorical and numerical data; one containing information about the user's profile and the other about their performance in the game. Both portions of the data are represented differently, categorical data is represented by words and numbers (e.g., age is represented by a number), performance data is represented merely by numbers and, finally, the labels for each participant are represented by words. This supposes an inconsistency, as not all the data is equally represented, which can become a problem when processing the data, given that some algorithms would not be capable of interpreting non-numerical data.

For this problem, we recurred to a normalization of the data to facilitate the integration of the variables, all the attributes for each of the data points were converted into a consistent and uniform format, which would allow the data to be equally interpreted (Rahm, 2000). Given that machine learning algorithms usually process numerical data only, not morphological representations of data, all the data represented by words were changed to numbers. All categorical data (except for the age) and labels were intervened, as they were represented by "Yes" and "No" labels for each category. As shown in Figure 2, to solve this problem, labels were transformed into binary numerical representations, attributing a number to each value. This way, the algorithm can interpret this data better, whilst standardizing nominal data to the other features of the dataset.

Male	0
Female	1
Native language (No)	0
Native language (Yes)	1
Language subject (No)	0
Language subject (Yes)	1
No Dyslexia	0
Dyslexia	1

Figure 2. Encoding of nominal data.

Another problem found in the dataset was the unequal number of samples for the two classes in the data: a class imbalance problem. This is a common problem in machine learning, given that data is naturally unbalanced, and obtaining data from a minority class can be a limiting factor, as some data is more difficult to sample than others, which is the case of the dyslexia class. Research by Rello (2021) has shown that people affected with dyslexia in Spain range from 3% to 10% of the population, which means that the population diagnosed with dyslexia corresponds to a minority group. In the dataset, the majority class corresponds to the "No dyslexia" class (with 3,252 samples) and the minority to the "Dyslexia" class (with 392 samples), corresponding to 10.7% of the dataset. Taking into account that the dyslexia population is already a minority group, we can see how sampling these cases could be a limitation, because there are few of them, to start with. The people professionally diagnosed with dyslexia are a minority group within the population; thus, we can see how this imbalance impacts the samples collected for this problem.

In classification, having an unbalanced number of samples is problematic, since having a majority of samples in one class could overwhelm the classifier and cause it to ignore the minority class, as there would be a higher chance of getting a correct answer when classifying something as part of the majority group. To address this imbalance problem in the dataset, an under-sampling of the data was performed to achieve a balance between the two classes. Here, the goal was to consider the number of cases in the minority class (392 samples) and reduce the number of the majority class, matching its number of samples to the minority class. We randomly selected 392 samples of the "No dyslexia" class and used those samples to represent

this part of the population. This way, the two classes were represented by an equal number of samples, allowing the classifier to not be overwhelmed by any of the groups, and be able to process both classes at the same level.

As mentioned earlier, preprocessing data is oriented to transform data. We have explained how data can be normalized and balanced, but other processes include changing the distribution and dimensionality of data for the algorithm to interpret it better. Once our data was balanced and normalized, some of the models in this classification problem were found to perform better when the input data was changed in dimension and scaled to a certain distribution. This means that, when data changed its original shape, it became easier for the algorithms to read.

Bishop (2006) exemplifies the scaling of data as the variability reduction of a set of data, by locating the range of certain values within the same range. In machine learning, this is also a common practice, as a non-uniform distribution of values could affect the performance of the algorithms and data could be read differently. This process takes each of the features of a dataset and centers it, so the mean and standard deviation of all the values would be scaled around the same range. In the scaling of the data used for this problem, a standard scaling function was used to center the values of the data points. This function transformed data in a way that the mean of these values was reduced to 0 with a standard deviation of 1, so data was scaled within the same range and had minimal variation.

With the scaled data, another procedure applied in the preprocessing phase for some algorithms was the Principal Component Analysis (PCA), to reduce the dimensions of data. This process can have many applications in machine learning, from dimensionality reduction to the visualization of data or feature extraction. Given big data sets with numerous features, the goal of PCA is to reduce the dimensions of the data whilst maintaining as much information as possible in a determined number of components (Jolliffe, 2002) and evict the curse of dimensionality, which are the problems that could surge from having high-dimensional data. For this thesis, we will not go into detail about how data is reduced to a certain number of components; what is important to understand is that, by reducing the dimensionality of data, the variance will also be reduced, containing a compressed version of the features of each data point that will facilitate the analysis. In addition to this, a whitening of the data is also performed; a process that makes the input data less redundant by making features be less correlated and have the same variance.

All these preprocessing methods were performed on the same dataset prior to processing this data with the models. Not all methods can be applied to all the models, as some models might require more preparation of the data than others. However, this process has proved to be necessary, as the processing of raw data in the models could lead to a poor interpretation of it and, therefore, poor performance.

3.3 Models

To properly analyze how machine learning algorithms can be used to classify information in this particular problem, three different algorithms were tested under the same input data, as it allowed us to see how data could be read differently when using different models and compare the performance. In this section, all the models built for the classification of dyslexia will be thoroughly explained, making emphasis on how they work and the selection of their hyperparameters.

Prior to the development of the models, the algorithm used by Rello (2020) was taken as a reference, so we could observe what optimization possibilities it could have given this problem. The predictive model described in this article is: a Random Forest model (explained in Section 3.3.2) built with Weka, a machine learning software. Rello explains the model used 200 trees with unlimited height and weighted attributes, to face the class imbalance problem previously mentioned; this means that a weight was attributed for the two classes (“Dyslexia” and “No dyslexia”) so they would have a level of relevance when being processed by the algorithms. The weight of the classes is not specified in the article, however, we can assume these weights attributed a higher relevance to the minority class (“Dyslexia”) to avoid having a trivial classifier, as they mention that having a trivial classifier would predict everyone does not have dyslexia (Rello, 2020). What they mean is that, even though having a trivial classifier would result in a high accuracy, because most of the data comes from non-dyslexic subjects, it would most likely tend to only detect the majority class and ignore the minority, given that there is the highest chance to make this assumption and have a correct prediction.

This brief description describes the parameters used for the construction of their predictive model. Taking this as a starting point, the models built for this classification problem will attempt to offer alternative methods and an optimized version of the model proposed in the article. The models presented in these sections were built using Python coding[4], with the help of the Scikit-learn library, a Python library designed to use machine learning. These models focus on the use of Grid search and are oriented towards getting a high level of recall (explained in Section 3.4.1.), which is the rate of true positives within all the positive cases present. Taking these factors into account, the different algorithm proposals will offer different takes to this classification, considering the meticulous selection of its hyperparameters and its performance level to the problem presented.

3.3.1 K-nearest neighbors

The first model implemented in this classification problem was the K-nearest neighbors (KNN) approach. This is one of the simplest algorithms in machine learning, as it classifies new data based on the most frequent class within its neighborhood. In other words, when a new example comes in for classification, the algorithm bases its decision on which class is predominant within the K neighbors surrounding it. As exemplified in Figure 3, when a new example comes (green circle), the algorithm focuses on the three nearest neighbors surrounding the new example (K=3 in this case) and it will be the most frequent class within those neighbors which will determine the decision made by the algorithm (red triangle), thus we would expect this sample to be classified as a red triangle.

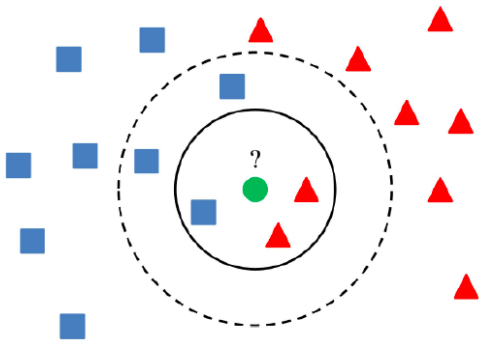


Figure 3. Illustration of K-nearest neighbors algorithm (Kumar, 2022).

[4] All the coding done for the different models is stored in a GitHub repository (also referenced in bibliography and appendix): https://github.com/alejaduque/alejaduque_tfg/.

We decided to evaluate its performance because this is a simple model that is based on analogy, meaning that it bases its decision on the similarity a sample can have with their surrounding neighbors, not on analysis of the features of the individual samples. Given that this is one of the most intuitive machine learning algorithms, it is interesting to see how it performs on a classification problem as it is this one and see if the simplicity of its design influences the results obtained. According to Kononenko (2001), in his article on machine learning for medical diagnosis, this analogous design of KNN shows a poor knowledge of the samples but good performance on practice, given that this analogy is often used to make decisions based on the similarly a sample can have with others. Based on this study, we could say that these sorts of models could perform well in this case, however, the analysis of the data would be poor, as the algorithm would not be basing its decision on the information of each sample, but rather the similarity it has to its surrounding environment.

For the implementation of this algorithm, we used the previously balanced dataset, the samples were split into train and test sets, 20% of this data conformed to the test set; and the data was scaled using a standard scaler (reducing the mean to 0 and unit variance). To determine which number of K would provide better results, a Grid search of the number of estimators hyperparameter was performed, as it would allow us to see which number of neighbors provided the most accurate results for this problem. The cross-validation of the hyperparameters consisted of testing which number in a range of 1 to 50 gave the highest score (which will be later defined in the evaluation section). Based on this score, the optimal K showed to be 37, meaning that the algorithm performed better when taking its decision based on the most frequent class within a neighborhood of 37 neighbors.

With this model, there is no need to perform a Grid search on more hyperparameters, as the only hyperparameter that influences the decision of this model is which number of neighbors is relevant to decide where a new instance would belong in.

3.3.2 Random Forest

Another model used for this classification problem was the Random Forest. This is the same model used by Rello (2020), chosen because of its flexibility and interpretability. By creating

this new version of the model, the goal is to see if this algorithm can be modified to provide better results for the classification of the "Dyslexia" class.

Random Forest models are based on the ensemble of decision trees, which are supervised learning algorithms that consist of hierarchized tree structures that represent all the possible outcomes of decision-making. What these structures do is to, from an initial data point, set different filters or boundaries that explore all the possible decision outcomes and, eventually, decide on the class where this data point belongs to, based on the most probable decision. The structure of a decision tree is exemplified in Figure 4, this tree attempts to classify an animal by asking a series of questions with true or false outcomes that would lead to a final decision. This structure consists of a root, in this case, an initial question about the characteristics of the animal with two outcomes, true or false; this root then splits in two nodes (that contain other questions, or filters) with the same two outcomes, and can either lead to other questions or a final answer, in this case it is the latter. These nodes can recursively split until reaching a label by following a similar structure in each iteration: a question is asked, there are two possible outcomes, if it leads to a final answer, a label is decided; if not, another question is asked.

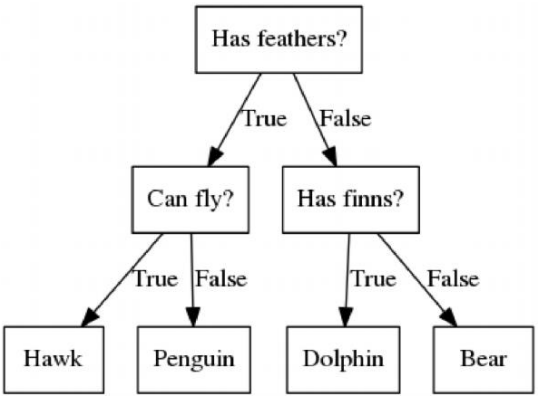


Figure 4. Structure of a decision tree (Towards AI, 2020).

These models are based on the splitting of nodes that can continue until these nodes are "pure", meaning that the data point has passed through enough filters to reach a solid decision. The decision of this tree would depend on the majority class decided on each of the nodes.

In models like Random Forest, decision trees are used as classifiers; they use an assembling of various decision trees that perform as classifiers for certain data points, each of the trees decides on the most probable class for a data point and it is this way that data is labelled on different classes. Chaudhary (2016) describes the advantages of Random Forest algorithms, defining

them as fast models that can handle numerical and categorical data, robust to noise and effective in identifying non-linear patterns in the data.

For the application of this model to this classification problem, a similar process as with the KNN model was followed prior to the application of the data, as it used the same balanced dataset, which was split into train and test sections, leaving 20% of data for testing. However, differently to the previous model, this model did not require a scaling of the data, as it was found that scaling the data did not bring any enhancement to the results, in contrast to the KNN, which did have benefits in processing scaled and centered data.

To select the optimal number of trees, this model required a cross-validation of this hyperparameter to select which number of trees provided the highest accuracy for this problem. From a range of 1 to 500 trees tested with the algorithm, we measured the score obtained with each number of trees and selected the hyperparameter that provided the highest score. This resulted in an optimal number of estimators (the trees) of 133. Similar to Rello's model, these estimators did not have a maximum depth specified, meaning that each tree of the Random Forest could have any extension it needed to reach a decision.

Other parameters were also used in this algorithm, apart from defining a number of estimators, as these parameters proved to enhance the performance of the classifier by making it more stable and effective. A bootstrap method was implemented, as it shows to reduce variance, provide more stable results, and help to avoid overfitting. This technique performs a resampling of the data, by creating a subset of the samples and working with the unique data resulting in this subset; in other words, given a set of *n* samples, what this technique does is to select the data in a way where there is no repetition of values, and the training examples of this sub selection consists of unique samples of the initial set. Similarly, another parameter implemented to this model was a class weight parameter for the subset created for the bootstrap technique. This performs the same weight function as the class weight parameter in Rello's model, but this time attributing different weights to the classes in the resulting subset of the bootstrap, to make it balanced.

We can say, then, that this version of a Random Forest provides a modified version of the one proposed in the article (Rello, 2020). It contains the same parameters used in the article, number of estimators and class weights, with the difference that, (1) the number of estimators

is previously selected through cross-validation and, (2) the class weight parameter is not directly applied to the input data, but rather a resampled version of this data. The resulting model is a tuned version of the initial proposal that includes a modified version of the previous parameters and additional ones to enhance the performance.

3.3.3 Support Vectors Machine

The last model applied to this classification problem was a Support Vector Machine (SVM) classifier, possibly the most mathematically difficult model out of the ones implemented throughout this thesis. We decided to try this method as we wanted to observe how a model designed to work with high dimensional spaces works with a binary classification problem as the one presented in this thesis.

This is a supervised learning algorithm that finds the optimal plane that separates the two classes to make classification. The name Support Vectors Machine is because this plane depends on a small number of samples of the training data, which are the support vectors. In the example shown in Figure 5, we see how two classes are embedded in a two-dimensional space (as an example), where SVM sets a boundary that marks a separation between the two classes. This boundary is set by the colored shapes representing each class, the support vectors. The way this model classifies is by, given a new sample, the SVM model labels this example depending on which side of the boundary this sample lays.

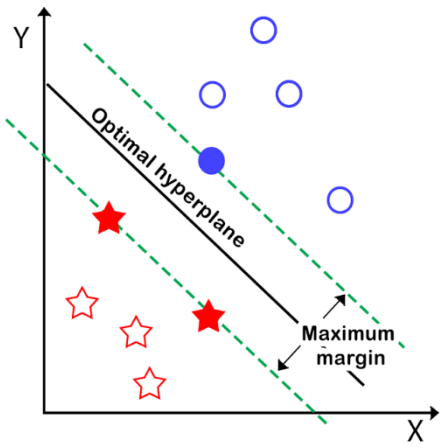


Figure 5. Example of a SVM model (Dabakoglu, 2018).

To implement this model, the dataset used for this classification was, as with the other models, the balanced version of the dataset. This data was scaled and transformed using PCA, as it

helps reduce dimensionality on the data and compress it to the most relevant information. Finally, as with the rest of models, data was split into train and test sets, with 20% of the data conforming to the test set. This model also made use of cross-validation of hyperparameters to select the optimal values, this time applying it to the preprocessing phase, and to the parameters of the model. The process of selection of these hyperparameters was similar for all the parameters in the model, a range of values was defined and tested with each of the parameters to see which values lead to a high score. The final model collects the optimal combination of hyperparameters, the ones that provide the highest score, obtained from cross-validation.

Prior to performing cross-validation in the parameters of the model, we performed cross-validation to the number of components of PCA, as we found that the number of components for the feature extraction of each data point influenced the final score of the model. A range of 1 to 196 components was tested to see which was the highest score, resulting in a number of 40 components as the optimal value.

The parameters in which the cross-validation is performed were the parameters of the kernel of the model, which is the function that takes data and implicitly transforms it in a high-dimensional space (Gupta, 2021). The kernel used for this model was the default kernel set by Scikit-learn (Pedragoza *et al.*, 2011), a Radius Basis Function kernel (RBF). The model uses as parameters, the gamma, a parameter of the kernel that decides how far the influence of a training example reaches, and C, which is a penalty term that allows some misclassification. The functioning of these parameters is out of the scope of this thesis, as our purpose is to observe how a model like SVM can perform in the detection of dyslexia, therefore, we will focus on finding the parameters which can lead to a high score, rather than going in detail about the background of these functions.

The values of the hyperparameters were selected with cross-validation, the optimal values resulting in an optimal C of 1000.0 and a gamma of 10^{-6}. Apart from the basic parameters of the kernel, another parameter implemented in the model was the class weight parameter. Just like with the previous models, this parameter attributes weights to the different classes so they are balanced.

Out of the three models tried for this classification problem, this was the one that contained more parameters selected using Grid search, and the only one that used PCA in the

preprocessing of the data. With this model, we eventually be able to see how the cross-validation of various parameters and use of PCA can influence the performance of a parameter, by optimizing its design in various points of the model.

3.4 Evaluation

Another important procedure followed to measure the performance of the models was a validation phase, which considered various aspects of the performance of the models, from their general results to their rates of true and false positives in the results. This validation procedure consisted of various steps, which allowed us to observe how the models were adapting to this classification problem and if they were processing the data correctly. We could divide the different evaluation steps in three sections: (1) the evaluation of the metrics, to observe the results of the classification and evaluate if they were correctly labelled; (2) a general evaluation of the model's performance using a K-Fold technique; and (3) an evaluation of the performance based on a ROC curve, which will allow us to observe its true positives and false negatives rates.

For this evaluation, the data is handled differently in terms of data splits. Similarly to the training phase, which uses train and test sets, the data is split into different sections that include: train, test and validation splits of the data. With this sectioning, we use the train set to train the model, the test set to test this training and, finally, the validation set to observe the performance of the model and evaluate how the model is addressing this problem.

These different evaluation methods allow us to observe different perspectives on how this model performed in the detection of dyslexia and, from this, assess which of the models provided the best results on this application.

3.4.1 Confusion matrix

To have a better understanding of the results obtained from the predictions of the model, we decided to observe a confusion matrix, as it shows how accurately the results were labelled. A confusion matrix is a very useful and popular method used to evaluate classifiers. This method shows the results of a prediction of the classifier, separating the correctly detected data from

the data that was incorrectly detected. This distribution can be understood better by having a visual representation, shown in Figure 6.

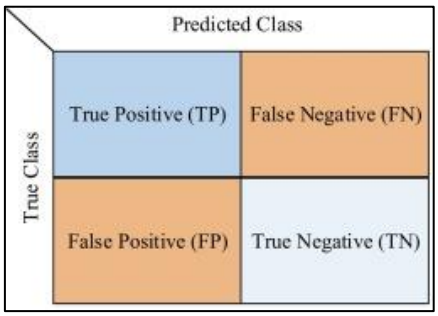


Figure 6. Confusion matrix representation (Demir, 2020).

In Figure 6, we can see how the data is distributed in the number of accurate and inaccurate predictions. Here, we differentiate between true positives (TP) and true negatives (TN), the results predicted accurately, and false positives (FP) and false negatives (FN), the values that were labelled wrong. For the classification of the dyslexia class, we would define the positives as the class 1, and the negatives as the class 0, as these are the labels of the possible outcomes of the classifier. This way, we would consider a true positive to be a correct detection of class 1, and a true negative a correct detection of class 0.

The confusion matrix gives us an insight of how our model is classifying the data, as it explicitly tells us what portion of the data is being interpreted correctly and which one is not. With this information we can, on the one hand, have an initial idea on how well our model performs and, on the other hand, use these values to calculate other metrics that will give us further information about the model's performance. The main metrics used to evaluate performance were mainly recall and f1-score.

The recall (or sensitivity) is a metric that indicates how much of the positive data is accurately predicted in the data. To calculate this, the number of true positives is divided by the number of the true positives and false negatives obtained from the confusion matrix, the formulation is exemplified in Figure 7. In the selection process of the hyperparameters, this calculation was used to select the parameters that provided the highest score. For this dyslexia case, we are

interested in getting a high score in this value, as we prefer to correctly diagnose the positive cases (dyslexia cases) and orient users into getting professional help, rather than missing it, leaving dyslexia patients unattended.

$$Recall = \frac{TP}{TP + FN}$$

Figure 7. Formula for recall calculation

The f1-score refers to the metric that relates the recall and precision metrics. For this score we use the precision, the measure of the rate of true positives out of all the positives detected (Figure 8), and the recall, from which we obtain a harmonic mean of these values (Figure 9). These metrics help to see how the detection of positive cases is handled in general. We do not make use of this metric to select the hyperparameters or tune the algorithm, however, we do observe this metric to have an idea of how the detection of positives is generally handled, as it considers both, the correct detections amongst all the positives detected (accurate and inaccurate), and the accurate cases detected amongst the actual positives.

$$Precision = \frac{TP}{TP + FP}$$

Figure 8. Formula to calculate precision.

$$f1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Figure 9. Formula to calculate f1-score.

3.4.2 K-fold validation

In the theoretical framework, we discussed the need of a phase in the evaluation that helps us assess the model and the performance it has on its application. Bishop (2006) introduced a method to observe the performance of a model under different training conditions. This method proposed the partition of the data into a determined number of folds to train, taking one of those split as a validation set and the rest as train sets, and iterate those splits an equal number of times, varying the training data by using a different test set for every iteration (as seen in Figure 1). What this technique does is to expose the model we are testing to different environments and provide a way to observe how our model would classify under different conditions, and if the results would vary under these changes.

To assess the general performance of our models, this technique was used to determine if the introduction of new parameters and optimization of hyperparameters performed well under different training examples. To achieve this, a 10-K Fold technique was applied to the different models, meaning that the data was partitioned into 10 splits and iterated 10 times, having a variation of the split for testing (validating) in each of the iterations. For each iteration, the data was fit into the model, a prediction was made from the validation set, and, with this prediction, a confusion matrix was created, which allowed us to calculate the recall of each iteration.

Just as with the selection of hyperparameters, we are interested in selecting the models that will result in a higher recall, as it would mean the model generally does well in detecting the true positive cases. We use this K-fold technique to calculate the mean and standard deviation of the recall for each model. For each fold, by knowing the resulting recall, we can calculate the mean and standard deviation, as it will tell us what is the expected percentage of the data that will be correctly predicted with the models, and how much we can expect this value to vary. By getting the mean, we are considering all the different trainings obtained from the K-fold method, in a way that we get a general view on what the expectation of the performance can be.

3.4.3 ROC curve

The final evaluation method was the plotting of ROC curves for each model. The receiver operating characteristic curve (ROC) is a plot commonly used in machine learning, that allows to display the accuracy of a model by plotting its rate of true positives and false negatives. This graphic representation is used to assess the performance and effectiveness of a model, as it gives a visual representation of how satisfactory the results can be. Another measure used as a reference is the area under the curve (AUC), a measurement that tells us the performance of this threshold variation, and the extension this curve has by calculating the area under the ROC. If the resulting AUC is a value close to 1, it means that the classifier has a high accuracy level and a higher rate of true positives; if it is the contrary, it means that the classifier is not performing accurately, and the rate of false positives is high.

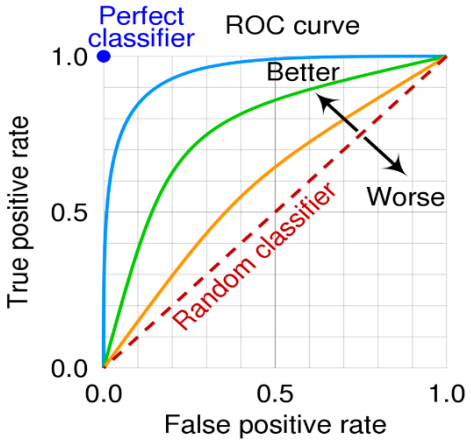


Figure 10. *ROC curve example (Thoma, 2018)*

In Figure 10, a representation of a ROC curve is shown, along with a representation of the meaning behind different curves that could result. The way a ROC curve is constructed is by calculating the probability of getting a true and false positive under different decision thresholds. In other words, given a series of the probabilities of getting a true and false positive, the results are collected and represented in the form of a curve in a two-dimensional plane, with X (false positive rate) and Y (true positive rate) axes.

As shown in Figure 9, by getting a curve that lays closer to the threshold value representation (red striped line), it would mean our classifier is not working effectively, as its rate of true positives its low. Contrary to that, by having a ROC curve that approximates a value of 1 (or close to 1) in the Y axis, means that our classifier is having an optimal performance and correctly detects all the positive cases, as it has a high rate of true positives and a low rate in false negatives.

The use of this representation (ROC) and measurement (AUC) for each of the models will allow us to contrast which classifier provides the best results. In this thesis, ROC curves will be used to observe how the different classifiers performed in the detection of dyslexia and, with these representations, determine which model seemed to have the best performance for this task.

4. RESULTS

The results obtained from the models applied with the methodology were summarized in the form of confusion matrices and ROC curves. These confusion matrices served mainly to calculate the metrics for the assessment, the recall and f1-scores. To assess the performance and contrast each of the tested models, we will use the ROC curves, as these will allow us to see the overall performance of each model.

The present results were obtained from the prediction of the test set, the unprocessed portion of data from the initial splitting in the training phase. For every test set, there are a total of 157 examples, which correspond to 20% of the data set (784 samples; 392 samples for each class). Both classes, 0 and 1, are equally represented in each split, having approximately one half of the data representing each class. The values shown in the confusion matrices will allow us to have a better understanding of how our model process and label samples they have not seen before (i.e., how they generalize). The results obtained from the testing of each model were the following:

	Negative	**Positive**
Estimated negative	48	30
Estimated positive	6	73

Figure 11. KNN confusion matrix.

For the KNN model, represented in Figure 11, we see that 77% of the data was accurately labelled, whilst 23% of data was not detected correctly. 46.5% of the data were true positives, 30.5% true negatives, 19% false negatives and 4% false positives. Out of this confusion matrix, we calculated a recall of 92% and a f1-score 80% and, in the general validation of the model (K-fold validation), a recall of 75% with a standard deviation of 6.5%.

	Negative	Positive
Estimated negative	62	14
Estimated positive	16	65

Figure 12. Random Forest confusion matrix.

In the Random Forest classifier (Figure 12), a total of 81% of data was correctly detected and 19% was labelled incorrectly. The results from the confusion matrix show 41.4% of true positives, 39.5% true negatives, 10% false positives and 9% false negatives. Based on the resulting confusion matrix, we obtained a recall of 79% and f1-score of 80%. In the general validation of the model, we obtained a recall mean of 78% with a standard deviation of 6%.

	Negative	Positive
Estimated negative	42	28
Estimated positive	21	66

Figure 13. SVM confusion matrix.

For the SVM model (Figure 13), the resulting confusion matrix shows a total of 69% of properly detected data, and 31% of incorrectly labelled data. More specifically, we observe 42% of true positives, 27% true negatives, 18% false negatives and 13% false positives. Out of this confusion matrix, we calculate a recall of 75%, f1-score of 72%. In the general validation phase, we obtained a recall mean of 96.8% with a standard deviation of 1.4%.

	Negative	Positive
Estimated negative	654	1
Estimated positive	69	5

Figure 14. Confusion matrix of the replication.

Lastly, regarding the replication of the model by Rello (2020), we observed the resulting confusion matrix from the processing of the test set. As this model made use of the full version of the dataset, which was not previously balanced, this prediction was done on a larger test set that contained more samples (729 samples). The results of this prediction showed that 90.38% of the data was correctly detected and 9.63% of the data was not labelled properly. The confusion matrix (Figure 14) shows 89.7% true negatives, 9.5% false positives, 0.68% true positives and 0.13% false negatives: all this resulting in a recall of 6.7%. However, in the general validation phase, obtaining a recall mean of 66% with standard deviation of 7.6%.

Aside from the results reflected in the confusion matrices, the ROC curves can give us further information about the performance of each model in this classification problem, as it gives us information of the resulting rate of true and false positives. In Figure 15, the ROC curves of the models are superimposed to observe how the models performed in comparison to each other.

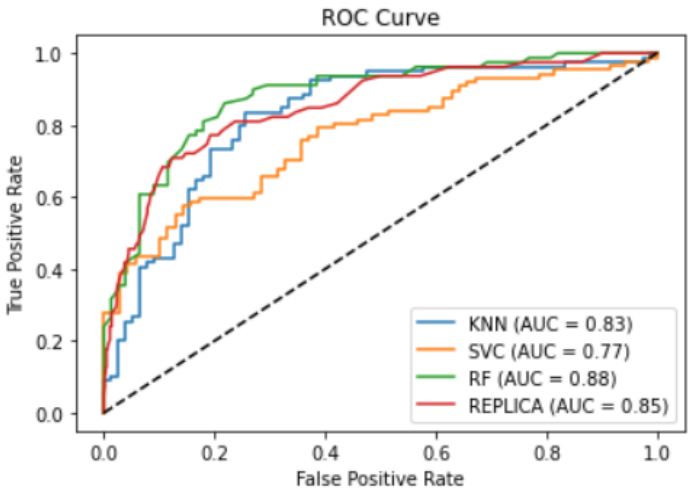


Figure 15. ROC curves of the models.

We can see that the highest true positive rate has been achieved by the Random Forest model, having an AUC of 0.88, and the lowest true positive rate was scored by the SVM (described as SVC[5] in the figure), having an AUC of 0.77. The replication of Rello's model is the one with the second-highest area, having an AUC of 0.85. The KNN model seems to be at a middle point, in comparison to the rest of models, as it has an AUC of 0.83, which means it has a high true positive rate but not as high as other models.

Even though some models seem to work better than others for this classification problem, the ROC shows they all tend to have a high rate of true positives. However, these curves do not necessarily tell us which model is the best option for this classification problem, as we will discuss below, given that the ROC curves slightly differ from the previously performed evaluation.

4.1 Discussion of results

To determine which model is more suitable, we are interested in selecting the model that has the highest recall in general, meaning that we want to select the model that generally performs better in detecting true positives among all the positive cases. The results obtained from the different evaluation methods show different perspectives on the performance. On the one hand, they show metrics obtained from the classification of one specific portion of data (test set) and, on the other hand, metrics obtained in the K-fold validation, which measure the general performance of the models under different training regimes. However, these results do not necessarily match in terms of ranking the classifiers, in fact, the results of these two perspectives differ a lot.

Results from the confusion matrices show that the KNN is the most appropriate model because, regardless of having the third-highest AUC, the metrics indicate it is the model with the most sensitivity (recall). However, based on the recall mean obtained in the K-fold validation, this is the model with the lowest recall score and greatest variability out of all the models proposed (not including the replication of Rello's model).

[5] SVC stands for Support Vectors Classifiers.

As we want to select the model that generally scores the highest recall and lowest standard deviation, if we look at the values obtained from the K-fold validation, we find that the best suiting model is the SVM. Even though it has the lowest AUC and recall score in the confusion matrices (out of the three proposed models), this is the model that generally has the highest sensitivity and more stable results, compared to what is shown in the ROC, which shows less consistency and lower sensitivity.

With this, we can see the ROC does not necessarily state which model is better, as metrics obtained from the K-fold method indicate the best suiting model is the SVM, not the highest ranked model in the ROCs. The ROC curves show the "best model" for this application seems to be the optimized version of the Random Forest model; however, the rest of metrics show this model tends to be in the middle, neither scoring the highest values nor the lowest. As in terms of the replication of Rello (2020), the ROC shows how this model would technically be the second-best model, however, other metrics show the contrary.

In the confusion matrix and metrics of this model, we see how the replicated model has the lowest scores of all models. This is a model that detects mostly class 0 (no dyslexia). Class 1 only corresponds to 1% of the samples in the test set and, although it has detected some of the positive cases, if the model is mostly trained with samples from class 0, we would not know if the model would properly classify examples from class 1 if it does not have enough examples. There is not a balance in the classes, and, consequently, this causes the model to have a bias towards the majority class. The ROC has a high rate of true positives because most of the negative cases, and some of the positives, have been detected well, but this does not tell us if it will generalize well with class 1, because most of the examples it has are from class 0.

Note that, in the confusion matrices, the number of false positives and negatives is generally balanced. Which tells us that, generally, the margin of error is the same for both classes. However, if we had to choose one, we would prefer a model that has a higher false positive rate, as it is preferable to misclassify a person who does not have dyslexia than the contrary.

Based on our results, we could conclude the best option for this application is the SVM model, being the one who generally scores the best recall and has less variation, as the standard deviation is relatively small in comparison to the rest of models.

As mentioned earlier, we want to prioritize the detection of dyslexia, preferring to send a child without dyslexia to get professional help than leaving children with dyslexia unattended. This means that, from all the models, we are giving priority to the one who can detect the most positive cases under different training conditions and tends to generalize properly.

5. CONCLUSION

Throughout this thesis, we have explored how machine learning methods can be successful in the detection of a language pathology, as it is dyslexia. Considering how useful machine learning algorithms have been in the medical field, we have tested its use in the detection of a language pathology by training algorithms that can predict whether a person has dyslexia from the processing of data.

We have analyzed the application of machine learning to the detection of dyslexia by replicating and expanding a previous study on this matter by Rello (2020), as this study proposes one predictive model, and we wanted to see if other algorithms could provide satisfactory results for the same problem. By testing a series of supervised learning algorithms (K-nearest neighbors, Random Forest and Support Vectors Machines) we have proven that satisfactory results can be obtained with the use of different models. These models have been trained in different phases, from preparing the data to evaluating it to ensure results are consistent and optimal. We have also implemented a process of selection of the parameters for each model, as we considered it would lead to better results.

From the results obtained, we have concluded that the model that fits the best for the present problem is the SVM, as it is the model that generally detects the most cases of true positives and has less variance. The replica is successful in detecting class 0 and the SVM in detecting class 1; as this model is biased towards class 1. We see that there is not a perfect algorithm that can detect all cases equally well, but rather, algorithms that perform well in detecting one of the classes.

The conclusion reached confirms the hypothesis previously stated: supervised learning methods that include more than one validation technique can provide satisfactory results. The tuning of hyperparameters leads to better results in detecting the “Dyslexia” class and offers

another possible classification model for dyslexia detection. We have found an algorithm that is able to satisfactorily detect the majority of dyslexia patients, as it gives priority to finding those positive cases and assess the users into addressing this impairment. In the end, the priority of these tools should be the user, who could use this initial assessment as guidance to make future decisions. Thus, providing quality guidance should be a priority, in order to transform these systems into useful and reliable tools that the general public can use.

5.1 Further discussion

In this thesis, we have demonstrated that we can make use of other kinds of algorithms for the detection of dyslexia and provide accurate results. We consider the use of SVM algorithms could be an interesting future line of research for the detection of language pathologies, as these are models that can work with high-dimensional data found in real-life contexts. However, the models presented above should only be taken as a suggestion, as they were only tested to prove if they could improve results towards a specific problem.

Another aspect that can be considered in the development of this tool is its expansion to other languages. In the present day, the *Dytective* tool is currently available in Spanish, English, and will soon be available in Catalan. We consider that, given the linguistic diversity in Spain, this tool could be soon expanded to other co-official minority languages such as Basque or Galician; considering that these are languages with a large number of speakers in the Spanish territory that could benefit from having this tool.

6. REFERENCES

Bishop, C. M. (2006). *Pattern and recognition and machine learning* (pg. 2-33). Springer.

Change Dyslexia. *¿Qué es Dytective de Change Dyslexia?*. https://www.changedyslexia.org/

Dabakoglu, C. (2018) *What is Support Vector Machine (SVM) ?-with Python*. December 23th, 2018 [Image]. Retrieved from: https://medium.com/@cdabakoglu/what-is-support-vector-machine-svm-fd0e9e39514f

Demir, F. (2022). Artificial Intelligence-Based Brain-Computer Interface. *Chapter 14: Deep autoencoder-based automated brain tumor detection from MRI data*. Academic Press. doi: https://doi.org/10.1016/B978-0-323-91197-9.00013-8

Duque, A. (2023) *TFG Repository*. https://github.com/alejaduque/alejaduque_tfg [Github Repository]

Garriga E. F. , Sala M. (2015). Avaluació, diagnòstic i intervenció en la dislèxia. *Revista de Psicologia, Ciències de l'Educació i de l'Esport* (Vol. 33, Núm. 1). Retrieved from: https://recercat.cat/handle/2072/250693

Gupta, A. (2021). *Kernel Tricks in Support Vector Machines*. Retrieved from: https://medium.com/geekculture/kernel-methods-in-support-vector-machines-bb9409342c49

Gwirtz, K. & Morzfeld, M. & Fournier, A. & Hulot, G. (2020). *2.4 Predictions, skill scores and ROC curves*. Can one use Earth's magnetic axial dipole field intensity to predict reversals?. Retrieved from: https://academic.oup.com/gji/article/225/1/277/5981611#227937586

IBM. *Decision Tree*. Retrieved from: https://www.ibm.com/topics/decision-trees

IBM. *What is machine learning?*. Retrieved from: https://www.ibm.com/topics/machine-learning

Jolliffe, I. (2002) *Principal Component Analysis* (pg. 32). doi: https://doi.org/10.1007/b98835

Kononenko, I. (2001). *Machine Learning for medical diagnosis: history, state of the art and perspective*. Artificial Intelligence in medicine. 23(1). pages 89–109. doi: https://doi.org/10.1016/S0933-3657(01)00077-X

Kumar, A. (2022). *K-Nearest Neighbors (KNN) Python Examples*. October 29th, 2022 [image]. Retrieved from: https://vitalflux.com/k-nearest-neighbors-explained-with-python-examples/

McCarthy, J. (1999). *What is AI?*. Retrieved from: http://jmc.stanford.edu/artificial-intelligence/what-is-ai/index.html

R. A. (2020). *A simple introduction to the Random Forest method*. Retrieved from: https://arifromadhan19.medium.com/a-simple-introduction-to-the-random-forest-method-badc8ee6c408

Pedregosa *et al* (2011). *Scikit-learn: Machine Learning in Python*. JMLR 12 (pg. 2825-2830). Retrieved from: https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html

Rello, L., Baeza-Yates, R., Ali, A., Bigham, J. P., & Serra, M. (2020). *Predicting risk of dyslexia with an online gamified test*. PLoS ONE, *15* (12 December). doi: https://doi.org/10.1371/journal.pone.0241687

Rello, L. (2020). *Dataset PLOS ONE Paper- Predicting Risk of Dyslexia with an Online Gamified Test* [Dataset]. Retrieved from: https://www.kaggle.com/datasets/luzrello/dyslexia

Rello, L. & Pavón, C. (2021). *¿Qué porcentaje de la población tiene dislexia?*. Retrieved from: https://blog.changedyslexia.org/que-porcentaje-de-la-poblacion-tiene-dislexia/

Scikit-learn. *Preprocessing: Standard scaler*. Retrieved from: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

Scikit-learn. Random Forest Classifier. Retrieved from: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Scikit-learn. *RBF SVM parameters.* Retrieved from: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html#sphx-glr-auto-examples-svm-plot-rbf-parameters-py

Scikit-learn. *Support Vector Machine: SVC*. Retrieved from: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Scikit-learn. *Support Vector Machines*. Retrieved from: https://scikit-learn.org/stable/modules/svm.html

Shaywitz S. E. (1998). *Current concepts: Dyslexia*. The New England Journal of Medicine.

Shmilovici, A. (2009). *Support Vector Machines. In: Maimon, O., Rokach, L. (eds) Data Mining and Knowledge Discovery Handbook.* Springer, Boston, MA. doi: https://doi.org/10.1007/978-0-387-09823-4_12

Towards AI (2020) . *Decision Trees Explained With a Practical Example* [Image]. May 28th, 2020. Retrieved from: https://towardsai.net/p/programming/decision-trees-explained-with-a-practical-example-fe47872d3b53

UFLDL Tutorial. *PCA whitening*. Retrieved from: http://ufldl.stanford.edu/tutorial/unsupervised/PCAWhitening/

Wolpert, D. H. & Macready, W. G. (1997). *No free lunch theorems for optimization*. IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pages 67-82, April 1997, doi: 10.1109/4235.585893.

Thoma, M. (2018). *Receiver Operating Characteristic (ROC) curve with False Positive Rate and True Positive Rate*. Wikipedia Commons [Image]. Retrieved from: https://commons.wikimedia.org/wiki/File:Roc-draft-xkcd-style.svg#filelinks

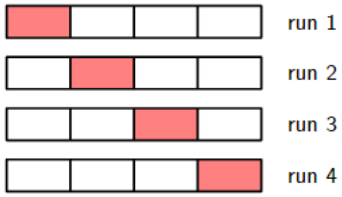
Yang L. & Shami A. (2020). *On hyperparameter optimization of machine learning algorithms: Theory and practice*. Neurocomputing, vol. 415, 20 November 2020, pages 295-316. doi: https://doi.org/10.1016/j.neucom.2020.07.061

7. APPENDIX

1. GitHub repository with coding of the models: https://github.com/alejaduque/alejaduque_tfg [GitHub Repository]

Figures

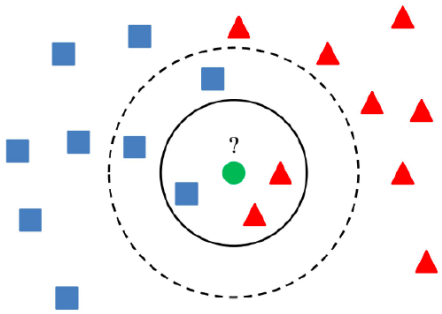
1. Illustration of K-fold validation (Bishop, 2006).



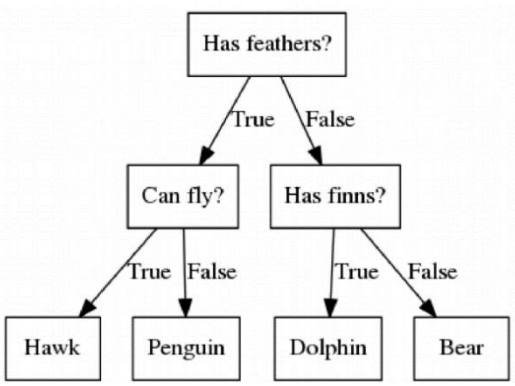
2. Encoding of nominal data.

Male	0
Female	1
Native language (No)	0
Native language (Yes)	1
Language subject (No)	0
Language subject (Yes)	1
No Dyslexia	0
Dyslexia	1

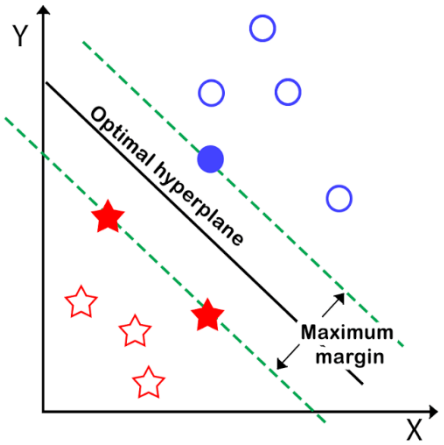
3. Illustration of K-nearest neighbors algorithm (Kumar, 2022).



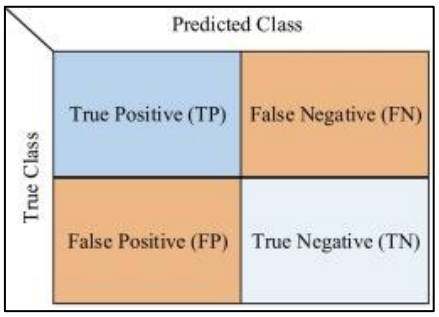
4. Structure of a decision tree (Towards AI, 2020)



5. Example of an SVM model (Dabakoglu, 2018).



6. Confusion matrix representation (Demir, 2020).



7. Formula for recall calculation.

$$Recall = \frac{TP}{TP + FN}$$

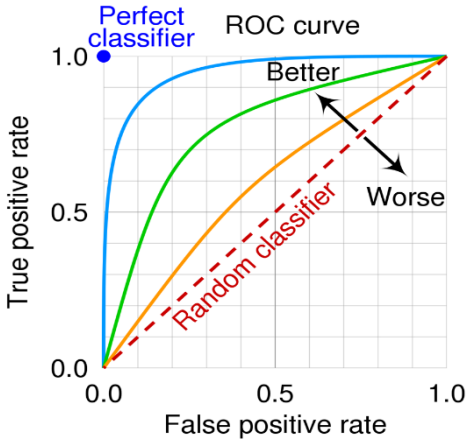
8. Formula to calculate precision.

$$Precision = \frac{TP}{TP + FP}$$

9. Formula to calculate f1-score.

$$f1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

10. ROC curve example (Thoma, 2018).



11. KNN confusion matrix

	Negative	**Positive**
Estimated negative	48	30
Estimated positive	6	73

12. Random Forest confusion matrix.

	Negative	**Positive**
Estimated negative	62	14
Estimated positive	16	65

13. SVM confusion matrix.

	Negative	Positive
Estimated negative	42	28
Estimated positive	21	66

14. Confusion matrix of the replication.

	Negative	Positive
Estimated negative	654	1
Estimated positive	69	5

15. ROC curves of all the models