TÉCNICO LISBOA

Predicting Alzheimer's Disease Progression: a Deep Learning approach

José Carlos André Nobre

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Prof. Pedro Filipe Zeferino Tomás
Prof. Helena Isabel Aidos Lopes Tomás

Examination Committee

Chairperson: Prof. Teresa Maria Sá Ferreira Vazão Vasques
Supervisor: Prof. Pedro Filipe Zeferino Tomás
Member of the Committee: Prof. Alexandra Sofia Martins de Carvalho

January 2021

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I would like to first thank my family, girlfriend, and friends for all the support given along this journey.

I would also like to thank my supervisors Pedro Tomás and Helena Aidos for the help and guidance which was fundamental in this thesis, especially with this change due to the Coronavirus pandemic.

Finally, I would also like to thank the author of the template used in this thesis which is also my supervisor Pedro Tomás, because, without this, the writing would be a lot harder.

Abstract

Alzheimer's Disease (AD) is a neurodegenerative condition that causes a deterioration in cognitive functions, affecting especially people of advanced age. As the disease is considered incurable, it is of the utmost importance to follow the patients as earlier as possible. In particular, as Mild Cognitive Impairment (MCI) is an early stage of Alzheimer's Disease, it is imperative to develop tools to allow predicting if and when a patient will progress from MCI to AD.

In this thesis, deep learning methods were used to predict, from baseline neuropsychological data, whether a patient will remain stable MCI (sMCI) or it will convert into AD (converter MCI, cMCI). A new methodology for automated feature selection on deep learning models was also developed, as well as the use of a missing value imputation technique to perform the oversampling. To evaluate the proposed model, baseline several machine learning methods were used as well as different methodologies to balance the data, perform the missing value imputation (MVI), and Feature Selection (FS). The results obtained through it showed a good capability for the proposed methods, recording high values of AUC and accuracy, and being capable of good predictions as early as 5 years with AUC of 0.86 and accuracy of 77%.

Keywords

Alzheimer's Disease, Neuropsychological Data, Prognostic Prediction, Deep Learning, Feature Selection, Classification

Resumo

A doença de Alzheimer é uma doença neurodegenerativa que causa a deterioração das funções cognitivas, afetando especialmente a população com idade mais avançada. Como a doença é considerada incurável, é da maior importância seguir os pacientes o mais cedo possível. Devido ao Défice Cognitivo Ligeiro ser considerado como uma fase inicial da doença de Alzheimer, é imperativo o desenvolvimento de ferramentas que permitam a predição de se e quando o paciente converte para Alzheimer.
Nesta tese foram utilizadas metodologias de *deep learning* para prever com base em dados neuropsicológicos, se um paciente permanece com défice cognitivo ligeiro (sMCI) ou se converte para AD (converter MCI, cMCI). Além disto, uma nova metodologia para seleção automática das *features* dos dados baseada em *deep learning*, bem como a utilização de uma metodologia de imputação para criação de novos dados foram propostas. Para avaliar as metodologias propostas, diversos métodos de Machine Learning foram usados, bem como diferentes metodologias para equilibrar os dados, realizar a imputação e a seleção de *features*. Os resultados obtidos mostraram boas capacidades para os métodos propostos, obtendo valores elevados de AUC e precisão, e sendo capaz de boas predições até 5 anos antes da conversão, com valores de AUC de 0.86 e de precisão de 77%.

Palavras Chave

Doença de Alzheimer, Dados Neuropsicológicos, Previsão Prognóstica, Deep Learning, Seleção de Características, Classificação

Contents

List of Figures

List of Tables

1

Introduction

Contents

Neurodegenerative conditions affect mostly people of older age and lead to adverse effects on the patients as well as to their closest persons. The deterioration of the cognitive functions is a fact that no one can change, it affects a large part of the population at a certain point in their life, some of them can convert into Alzheimer if the deterioration is big and fast while others might only have a slight decline in cognitive functions. This creates a problem because if all people will suffer from neurological deterioration at some point in their life it is difficult to distinguish these signs from dementia and consequently Alzheimer's Disease (AD). A great loss of memory is usually one of the first indicators of AD but it might not be this straightforward all the time, that is why studies in this area of medicine have grown in the last decades. Better diagnosing techniques have appeared, such as Magnetic Resonance Imaging (MRI), Positron Emission Tomography (PET), and also Neuropsychological Tests (NPT's). All these techniques have proven to help doctors in diagnosing this devastating disease.

This turns out to be important since 60 to 70% of the dementia cases are Alzheimer's Disease. In the US approximately 24 million people suffer from AD, and this number is expected to rise by 4 times until 2050 [3], which corresponds to approximately 100 million cases. However, this estimate accounts for the US population alone. If we think in all the other countries, the number of cases will escalate even more. Also, if we consider the costs associated with the disease this turns to be an economic problem, so there is the urge to find a way to diagnose the disease as soon as possible.

This disease usually starts as a Mild Cognitive Impairment (MCI), which is a stage that lays between the cognitive decline of the usual aging process and a more serious decline of dementia. This impairment usually involves problems for the patient, like loss of memory, difficulties in language and thinking process that are more intense compared to the normal aging process of people. Consequently, if this impairment is not regarded carefully with the proper therapy, these patients can more easily progress into dementia, which in most cases turns to be AD. So, finding a way to predict if a patient will eventually suffer from Alzheimer will allow the medical staff to perform a better follow up of the patient. Also, it can give time and preparation to the patient's closest ones, which can give a more comfortable life to both.

Consequently, it is crucial to find methods to predict the progression of Alzheimer's Disease. Nowadays, new technologies revolve around Machine Learning. Due to the advances in this field and the amount of work dedicated to medical research with it ([4–8]). It certainly seems like the future of medical prediction is with machine learning techniques, these methods can help doctors narrowing the field of patients that can progress to AD and help to see if a patient will probably convert to AD in a given time window. It is expected that these methods will evolve in the following years due to more computational power and the evolution of the machine learning methods.

1.1 Objectives and contributions

The main objective of this work is to find a way of predicting the progression of Alzheimer's Disease that produces reliable results. This will give medical staff more information to help follow their patients the best way possible. However, in contrast with previous approaches, the main goal of this thesis is to exploit deep learning methodologies, as these techniques recently demonstrated superior performance in a wide range of areas. To accomplish this, this thesis is focused on the following areas:

- Finding the best features to predict the conversion to Alzheimer's Disease through Feature Selection techniques, such as by relying on state-of-the-art Neural Networks to remove less important features;
- Find the best methodology to perform the Missing Value Imputation on our time windows, by either using classical methods or new methodologies based on autoencoders;
- Study the best methodology to perform the balance of classes , particularly by relying on oversampling techniques, and analyze the impact of these methodologies on the capability of prediction;
- Predict the progression to Alzheimer's Disease with the help of Machine Learning models, particularly by taking into consideration recent advances in deep learning and by comparing with classical machine learning models;

A new paper is in preparation to communicate the results obtained in this thesis using the methodologies proposed.

1.2 Dissertation Outline

The work will be described in the next chapters as follows. In Chapter 2, an introduction to Alzheimer's Disease is presented, how it is diagnosed and the tests used for this diagnosis, the usual symptoms, and the database used in this thesis. Chapter 3 presents an overview of machine learning methods as well as the metrics to evaluate these classifiers. An introduction will be done to the techniques used for balancing classes and perform oversampling as well as Feature Selection methods and Missing Value Imputation techniques. The methodology used for the classification will be presented in Chapter 4, being introduced with more detail some of the methodologies used as well as the new Feature Selection method created in this thesis. In Chapter 5 a description of the experimental setup for obtaining results will be done as well as the results obtained for every time window. Also a comparison between the results obtained here and the ones obtained by Telma Pereira et al. in [2] will also be made. Finally, Chapter 6 presents the conclusions obtained, and some possibilities for future work are presented.

2

Alzheimer's Disease

Contents

2.1 The Disease

Alzheimer's Disease (AD) is one of the most common causes of dementia, being more focused on elderly people, which are a large part of our population, that is why it is so important to take care of it as soon as possible and start treating the patients. As it was stated in [9], our understanding of AD has developed in three stages, the first was in 1907, when Alois Alzheimer identified the clinical and pathological features of the condition, but it was not until the work of Blessed that the disease was recognized, not as a rare neurological disorder, but as the most common cause of dementia. In second, was the discovery of the frequent histopathological marker lesions in normal elderly individuals and the close relation between the severity of the lesions and the degree of dementia, this physical were the advance in the second phase. Finally, with the evolution of genetics research, the cloning of the gene mutation coding for β-amyloid precursor protein on chromosome 21 was found, and this is known as the third phase.

As we can see, our knowledge about AD as been evolving since more than a century, this allow us to have nowadays a extended amount of research about this disease, which helps us create new solutions and treatments for symptoms, as long as we can 'catch it' soon enough.

As stated in [9], the causes of Alzheimer's Disease are structural abnormalities in the cerebral cortex. This abnormalities can be divided in neurofibrillary tangles and senile plaques, the first ones consist on phosphorylated fibrillary proteins aggregated within the neuronal cytoplasm. These are a natural consequence of the aging process, but it is the high amount and the distribution of them that promotes the pathology and define the stages of the disease.

The second ones consist of cellular deposits of amyloid material and are associated with swollen and distorted neural processes which are called dystrophic neurites. The plaques start as harmless deposits of β -amyloid, but on some individuals they undergo an sequentially transformation into senile plaques which are associated to the development of Alzheimer's. This deposits start appearing on people with around 50 years and nearly 75% of the people with 80 years are affected.

Genetic also sets a role on the appearance of AD, there are several mutations on the gene coding for β-APP on the chromosome 21 that causes a dominance on familial AD. Also some mutations increase the formation of β-amyloid and similar amino acids that aggregate even faster.

This disease brings several consequences for the patients and also their families, which are in contact with the patients on a daily basis. The patients are characterized by a progressive deterioration of cognitive, functional and behavior capabilities, as stated in [3]. This causes, for example, loss of memory, not being capable of doing everyday tasks [10], the impairment of the cognitive system may also manifest through delusions, hallucination, aggression, depression, anxiety, disturbs on the motor functions, sleep and appetite disorder and other more. Since it can create all this problems and even more, this disease

affects all the people around the patient, and so the faster it is diagnosed, the sooner the patient can get treatment to avoid the most of this problems.

2.2 Alzheimer's Disease Diagnosis

The diagnose of Alzheimer's Disease can be done through different exams, either with Brain Imagery, Laboratory Tests, and NeuroPsychological Tests (NPT's).

In Brain Imagery, the most used exams according to [3] are the Structural Magnetic Resonance Imaging (MRI) and the Positron Emission Tomography (PET), other techniques are also used such as Functional MRI and Single Photon Emission Computed Tomography (SPECT).

The MRI is an integral part of the Alzheimer Disease assessment on patients [11] due to its capabilities of estimating tissue damage or loss in characteristically vulnerable brain regions, showing structural markers that are associated with the disease such as the atrophy of medial temporal structures or whole-brain and hippocampal atrophy, which are signs of neural degeneration. The usefulness of the MRI relies on detecting the focus of the atrophy, therefore, helping to rule the different causes of dementia but it is not enough to predict if a patient with MCI will convert or not to dementia.

Other techniques, such as PET have also been used to predict the conversion from MCI to AD [12] with high accuracy rates, rounding 70% to 80%. This technique is helpful because it also measures parameters like brain metabolism and amyloid deposition, which are early signs of AD and occur before the changes that can be detected visually in MRI or impairments detected through NPT's.

2.3 Neuropsychological Tests

The use of Neuropsychololgical tests to assess the progression of MCI to AD is the goal of this thesis, these types of tests are used to evaluate a wide range of domains such as language, sensory functions, and memory. Together with the information obtained from clinical reports and physical tests, this technique has been proved useful to predict the progression of MCI to AD, the only downside is still the similar neural degeneration that affects elderly people and it is part of the natural aging process [13], so the accuracy is not as high as it could be. But this can all be improved by making several follow-up assessments periodically to identify even the slightest cognitive decline, as well as to calculate the rate of this decline which is useful to predict when the patient will reach each stage of the AD.

The NPT's are divided into two different types, the ones who evaluate the Memory Domains and the ones who evaluate the Non-Memory Domains.

Regarding the memory domain, one of the features that are widely seen as a predictor for later Alzheimer's Disease development is the Episodic Memory, which is the recall of events in a specific

place or time. This feature is one of the first aspects to decline on a patient and can be noticed several years before the diagnosis of Alzheimer's Disease.

This memory aspect can be assessed on Neuropsychological Tests with Logical Memory Tests, with Word Recognition and Recall Tasks, Rey Auditory Verbal Learning Test (RAVLT), Verbal Paired-Associate Learning (VPAL) as well as the California Verbal Learning Test (CVLT), which are the features presented on our database that will be explained later.

The other domain, that is part of the NPT's is the Non-Memory domain, this type of domain can be evaluated through tasks like Color-Word Interference Tests, Verbal Fluency, Orientation, Clock Drawing Test (CDT). This type of test aims to evaluate cognitive functions, executive functioning, visuospatial function, and learning capabilities, which are also good predictors to mental decline and consequently Alzheimer's Disease prediction.

2.4 The Database

The Database in which all the work was done is the Cognitive Complaints Cohort (CCC), created by a partnership of Santa Maria Hospital in Lisbon, the Laboratory of Language Studies, Memoclínica and the Neurology Department of Coimbra's University Hospital in order to investigate AD progression in patients with MCI, as stated in [14]. All the patients in this database are evaluated through Bateria de Lisboa para Avaliação das Demências (BLAD) [15], which is a neuropsychological battery validated for the Portuguese population. This database is composed of four different time windows (two, three, four and five years), these are databases on which the patients are grouped based on the information collected about the conversion or not to Alzheimer's Disease within a specific time window.

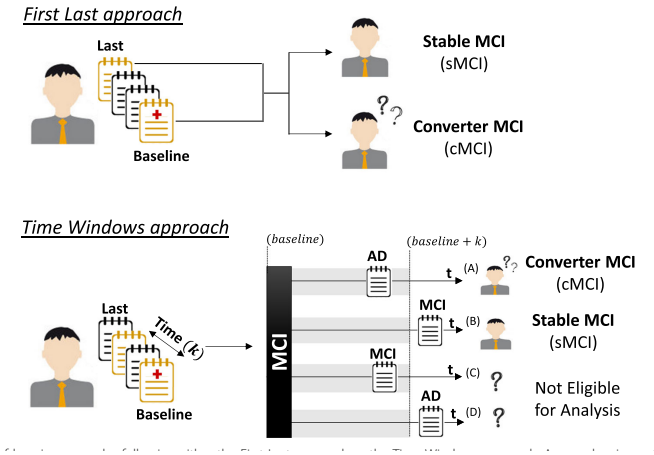


Figure 2.1: Time window construction, from predicting progression of mild cognitive impairment to dementia using neuropsychological data: a supervised learning approach using time windows [1].

These time windows are built as specified in the Figure 2.1, all the assessments from the patient appointments are joined and divided accordingly to the time from the baseline (first medical appointment). For a given time window, a Converter MCI (cMCI) label is created if the patient is diagnosed with AD in a medical assessment that happens in between the baseline and the given time window (baseline + k), which is the first example in the figure. If the patient remains stable it leads into a Stable MCI (sMCI) sample (second example). The two last examples in the figure need to be disregarded when building a time window because a patient that is sMCI in the interval of the time window is not guaranteed to remain sMCI until the end of that period, also if a patient is diagnosed after that time window we can not guarantee that he was either sMCI or cMCI by the end of the previous time window, so it is also discarded. The difference in this approach compared to the First Last Approach is that the same patient can be classified with different labels in two different time windows, it can be sMCI in a smaller time window and cMCI in a larger window because in a later follow-up assessment he can be diagnosed with AD, this allows us to know approximately when the patient will convert.

These windows include features as Mini Mental State Examination (MMSE), California Verbal Learning Test (CVLT), Clock Drawing and Logical Memory Tests. Also, it is included age, years of education and the Z-Scores, which are a measure of how many standard deviations below or above the population mean a raw score is. All these features are static, i.e., they are all present in the different time windows. In total, the database includes 98 different features, 72 of them categorical and 26 numerical, The distribution of patients per time window is shown in Table 2.1.

Table 2.1: Number of patients in each time window

	Number of patients
2 Year	501
3 Year	468
4 Year	431
5 Year	410

Although with the best efforts given to the construction of this database, it is practically impossible to build one like this without any missing values, due to the extension of the tests people get tires and exhausted to complete all the tasks needed. So given that, the database as quite an extensive number of missing values in certain features, as can be seen in the tables present on the Appendix (Tables 7.1 to 7.4). The database also had a great imbalance of classes (Table 2.2), especially on the two and three-year window, this was tackled balance techniques to have a balance of 50/50 on each class for every window.

Due to the lack of information about the converting time of the patient, this work is based on using the time windows methodology, so therefore the First Last Approach was not used.

Table 2.2: Balance of the Database Time Windows.

	sMCI	cMCI
2 Year Window	78.6 %	21.4%
3 Year Window	65.2%	34.8%
4 Year Window	52.7%	47.3%
5 Year Window	42.7%	57.3%

2.5 Related Work Using NPT's

The problem addressed in this work, the prediction of the progression of Alzheimer's Disease, was previously studied by other researchers, although using different approaches and different databases, NPT's, medical images and others. Among all of these works, one of the most important work, is the one done by Pereira et al. [1] because it uses the same database, Cognitive Complaints Cohort (CCC), that is used in this work.

The work in [1] is based on the use of time windows to predict the conversion of Mild Cognitive Impairment (MCI) to Alzheimer's Disease. This reduces the time span of the First Last (FL) approach into a specified temporal frame. In the FL approach, the database combines only the baseline with the last evaluation of each patient, so if a patient in the last evaluation is diagnosed with dementia it is labeled as a converter MCI (cMCI). In contrast, if it is only diagnosed with MCI it is labeled as stable MCI (sMCI), this approach only wants to find if a patient will convert to dementia at some time in the future, not being concerned when it will convert. The time windows approach is specific to one temporal frame, so a converter MCI instance is created when the patient is diagnosed with AD in an evaluation that is done in the specified time window, the patients who remain MCI after the specified time window are labeled as sMCI. So, with this approach, a patient can be sMCI in one time window and cMCI in a larger one. The goal is that we can now predict if and when a patient will convert to dementia. In this work [1], one of the other goals was to make a Feature Selection on the database to have better results, the used FS method used was the Correlation-based Feature Selection (CFS), to balance the classes the SMOTE technique was used and for missing values imputation the values were replaced by the mean or the mode, if the values where numerical or categorical respectively. After this, they produced a pipeline using different classifiers, namely Decision Trees (DT), K-Nearest Neighbors (KNN), Support Vector Machines (SVM) both Polynomial and RBF, Naïve Bayes (NB), Logistic Regression (LR) and Random Forest (RF). The authors reported Area Under the ROC Curve (AUC) values above 0.72, and they were able to predict dementia as early as 5 years with an AUC of 0.88, specificity of 0.71 and sensitivity of 0.88.

Another work done by the same researcher was published one year later [2]. In this work, the time windows approach was still used but this time they focused more on Feature Selection. Hence, instead

of having a Correlation-based Feature Selection they created a feature selection ensemble to combine stability and predictability of various FS methods. The approach used can be combined in two phases, in the first it finds a subset of features sorted and sorts them by relevance using ensemble learning. In the second phase, the subset of features is optimized with regard to the stability and predictability. This subset of optimal features will vary with the classifier used in the second phase for the optimization, but this can also be prevented by using different classifiers with an ensemble-based approach. Another change in this work was the use of two different databases, the CCC and the Alzheimer's Disease Neuroimaging Initiative (ADNI), where patients were selected from both databases 584 from the first and 433 from the second one. With this feature selection approach, they managed to get better results than when using the individual feature selection methods, except in the 4 year time window. This all proves that Feature Selection can help improve the prediction and also simplify the classifier due to the use of fewer features, which consequently helps at the interpretability of the model.

A work that is also important for this thesis was made by Maroco et al. [16], which used an older version of the database used in this work. The study used several different classifiers, some of them are: Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Logistic Regression, Neural Networks (NN), Support Vector Machines, Classification trees (CT) and Random forests. The data used consisted of 921 elderly non-demented patients with cognitive complaints from three institutions in Lisbon, the Laboratory of Language Studies, Santa Maria Hospital, and Memoclínica. On this data, the authors needed to do some selection of the patients. For that purpose, patients that had dementia or other disorder that caused cognitive impairment, having medical treatments that affect cognitive functions and that abused on alcohol and illicit drugs were excluded from the database. After this selection, the database consisted of 400 patients. The authors reported good results based on similar work at the time, most of the classifiers exhibit AUC values greater than 0.7 and they did not find significant statistical differences between 8 of the 10 classifiers used. However, in some of the classifiers, they reported poor performance on sensitivity values which is significantly in works such as this.

Another work done to predict the conversion to Alzheimer's Disease was done by Ye et al. [4]. The authors used the ADNI data collected from 50 different sites and used 319 MCI subjects, where 177 were sMCI and 142 were cMCI. The conversion was analyzed in a 4 year time period, the database in this study includes in addition to MRI and Cerebrospinal Fluid (CSF) measurements, scores like Mini Mental State Examination (MMSE), Clinical Dementia Rating Sum of Boxes (CDR-SB), Alzheimer's Disease Assessment Scale-cognitive subscale (ADAS-cog) and others. To perform the classification the SVM classifier was used and the Feature Selection was performed with Sparse Logistic Regression with stability selection to perform the ranking of the features, which in turn returns the best subset of features. With this process, they managed to achieve good performance and have results of AUC of 0.8587 which at the time of the work was the highest between the works that used similar type of data.

A study made by Chapman et al. [17] had the goal of predicting the conversion from MCI to AD using Neuropsychological tests combined in a method with two levels of empirically derived weighting. The test measures were first reduced to underlying components, Principal Components Analysis (PCA) [18], and then the component scores were combined to classify individuals using discriminant analysis. For this research, they studied 43 elderly individuals diagnosed with Mild Cognitive Impairment. The MCI diagnosis was done by memory-disorders physicians and met the criteria for the amnestic subtype of MCI. The tests performed on the patients were also done by memory-disorders physicians and they include MMSE, clock-face drawing and category fluency task (animal naming). From this group of 43 patients, 14 were diagnosed with AD in the following evaluations while the other 29 remained stable. For the PCA analysis, the authors needed to add more subjects to develop the component structure from the neuropsychological test battery. For this purpose, they added 55 elderly individuals diagnosed with AD, 78 individuals with normal cognition for the control group, 5 individuals diagnosed with Age-Associated Memory Impairment), and 35 more MCI subjects, this sums to a total of 216 subjects. The relevant results obtained from this procedure were that the patients that converted to AD, in general, performed worse than the stable MCI group. From this PCA analysis, they obtained 13 components that showed the high capability of separating the converters from the stable group. The last two components were discarded to maintain a roughly 4:1 ratio between subjects and predictor variables, the remaining components accounted for 72% of the total variance of the data. From the remaining 11 scores entering the step-wise discriminant procedure, six of those components were selected as those that have better discriminability between the stable MCI group and those who converted. According to the authors, these discriminant functions performed well in the initial set of 43 subjects. From that set 36 were correctly classified resulting in an accuracy score of 83.7%. Of the 14 on the conversion group, two were incorrectly predicted as stable which resulted in a sensitivity score of 0.86 and a positive predictive value of 0.71. Finally, on the stable group, 24 of the 29 members were correctly predicted, this resulted in an accuracy score of 0.83 and a negative predictive value of 0.92.

2.6 Summary

In this section, a review of the Alzheimer's Disease was made, how it appears, and the consequences of the disease to the patients and their closest ones. It was also seen how it is diagnosed and the tests used for that purpose, with more emphasis on the tests which are present in this thesis and are in the dataset, the Neuropsychological Tests. Finally, the database and its statistics was presented and some works that were done in this database, on which this thesis is based on. In the next chapter, the methodologies to make the prediction of the progression from MCI to AD will be presented.

3

Background

Contents

A large amount of work has been done for different diseases, one of the most studied besides cancer is Alzheimer's Disease. In this field, there have been works using different kinds of exams, from medical images to Neuropsychological Tests (NPT's) and using different methods, from the First Last Approach, which is the most common and uses all the patients, to Time Windows [1,2] which separates the patients in windows of the time they take to convert to Alzheimer. All this works tells us that this is an area with a large potential of improving, either in the databases or the Machine Learning methods which are always evolving, allowing better approaches for our problems.

Machine Learning, according to [19] is a technology with the goal to develop computer algorithms that try to be really close to human intelligence by learning the environment in which they are put into and previous experiences, and with this, they can make predictions with new data that is presented to them. They have been rising in the last decades due to the agglomeration of big sets of data and the higher computer processing power which together make better methods (better predictions) and faster learning ones. The only things usually needed to get predictions are the data and the classification method, but in some cases, there are more things to add, for example, Missing Value Imputation to have a complete dataset, or Balance of Data in the case that there is a big unbalance on the dataset.

In the case of the prediction of Alzheimer Disease's progression, because the dataset is not complete i.e., it has missing values, and has a large number of features, there is the need to consider a Feature Selection method and a missing value imputation method. Feature Selection is one of the major problems in this work because the features need to be well selected to have the better prediction possible. Because in some of the datasets of the database the data is seriously unbalanced there is also the need to balance the dataset. So, due to all of this, the proposed pipeline for the problem being dealt with is the one presented in Figure 3.1.

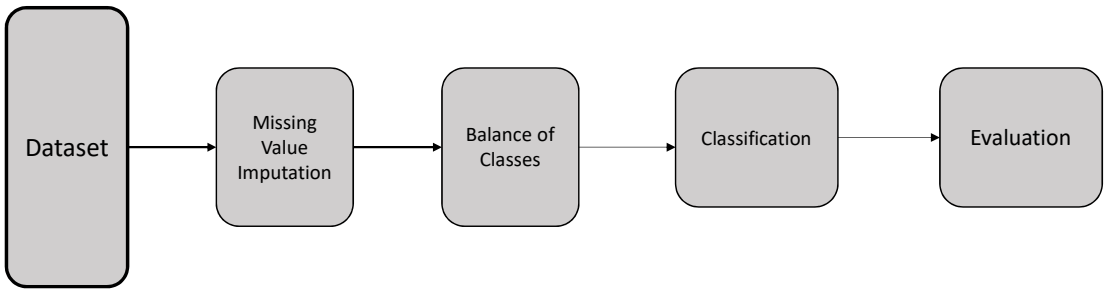


Figure 3.1: Typical classification workflow for medical data.

3.1 Dealing With Missing Values

One of the problems that arise when working with large databases is the existence of missing values, this is a problem that can be dealt with in several different ways. According to [20, 21], there are three different types of missing values:

- Missing Completely at Random (MCAR) - In this, the missing data does not depend on any other values of the dataset. It means that whatever method is used, there will be no bias.
- Missing at Random (MAR) - This means that the missing data does not depend on the other missing data but may depend on the observed data. This means that the data is missing because of other data.
- Missing not at Random (MNAR) - The missing data depends on the missing data itself. This just means that the data is not either MCAR or MAR.

In order to use the database, a method to impute those missing values has to be used. In [20] the most common approaches to deal with missing data are:

- Ignore the features with missing values. If a feature has at least one missing value it is automatically ignored. This is not the best method for large datasets because the probability of having at least one missing value in each feature is very high, so we would be eliminating almost the entire dataset.
- Replace by the most common attribute value, in this, all the missing values are replaced by the most occurred value in the database. It is not the best in the case where we have categorical and numerical values in the same dataset.
- Replace by the most common attribute value in the class, which means that the most common value in a class will replace all the missing values in that feature. It is good for categorical data.
- The mean substitution. Here, the most mean value of the data in a feature is used to replace the missing values in that one. It is best suited for numerical data.
- Replace using regression or classification methods. In this approach, a classification or regression model is used to predict the values that will replace a missing attribute, it would base the predictions on the remaining data in a class.
- Hot deck imputation. In this methodology, the missing values are replaced by similar cases in the database.

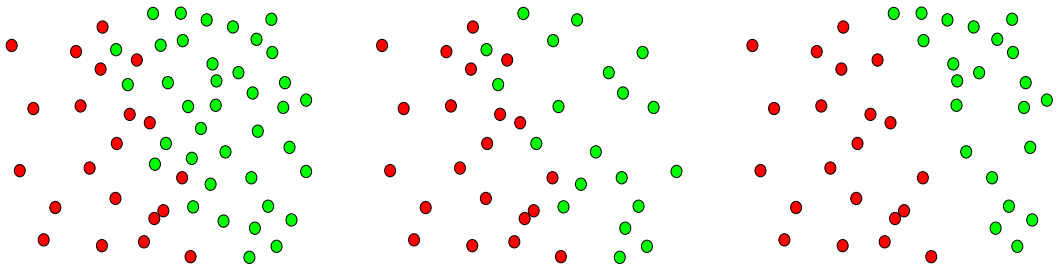
Another option that might be useful in this work would be the replacement of the Missing Values using a dedicated Neural Network [22] instead of a simple classification or regression model. Some works have already focused on using a deep learning approach to perform the missing value imputation [23–26], for the purpose of this thesis one methodology was used [26] and will be explained in a further section.

3.2 Data Balance Techniques

Most of the data in the real-world are imbalanced by nature. This situation occurs when the distribution of the target class (prediction) is not uniform among the different classes. This subject has revealed a lot of interest among the Machine Learning community because most of the Machine Learning Methods are created to work on a perfect dataset, this is a dataset where the classes are equally balanced. To overcome this class imbalance and improve the overall performance of the classifier there are two different types of techniques that can be used, undersampling and oversampling.

3.2.1 Undersampling Techniques

The undersampling method is a non-heuristic methodology, in which the database is reduced to obtain balanced classes, this means that we will remove instances of the database from the class with more instances to achieve the balance. For example, if we have 300 cases of class 0 and 200 cases of class 1, the database will be " cut " to have 200 cases of each class. There are 2 main methods of Undersampling [27], Random Undersampling (RUS) and Focused Undersampling (FUS), in the first method the instances from the majority class are randomly chosen to be removed to balance the classes, while in the second one, the instances of the majority class that are removed are the ones closest to the border between classes, this difference can be seen in Figure 3.2.



(a) Original Dataset. (b) Random Undersampling. (c) Focused Undersampling.

Figure 3.2: Differences between Random Undersampling (RUS) and Focused Undersampling (FUS).

Due to this reduction, the database will become smaller, which is not the best practice, because the

training set will be less ”rich” which can lead to a poorer classification. The upside of this method is that there is not the creation cases that are not real, which leads to a dataset with only real data.

3.2.2 Oversampling Techniques

The oversampling method is the opposite of the one stated before, here there is the creation of more cases to balance the classes, examples of techniques that use this approach are SMOTE [28], SMOTE-NC [28], Random Oversampling [29], Adaptive Synthetic Sampling (ADASYN) [30] and techniques based on auto-encoders [31, 32]. Based on the example from before, the database is now turned into one that has 300 cases for each class. The upside is that there are more samples, which is really good for the training of our classifiers. On the other hand, there was the generation of samples that are not real, which is not the perfect scenario.

3.2.2.A SMOTE

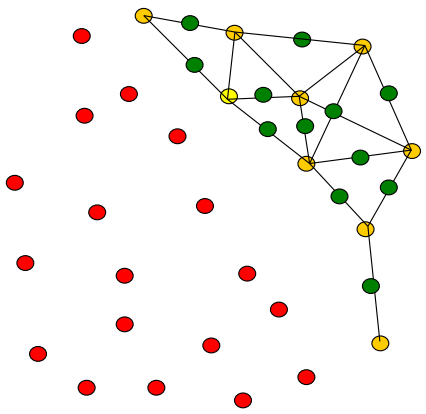


Figure 3.3: Example of SMOTE: Red Dots (Class1). Yellow Dots (Class 2). Green Dots(Synthetic Samples from Class 2).

SMOTE (Synthetic Minority Oversampling) [28] works by creating synthetic examples instead of over-sampling with replacement, also, this operation is performed in the feature space rather than on the data space. Following what is said in [28], the minority class is over-sampled by taking each instance of that class and introduces those synthetic examples along the imaginary lines that connect the k nearest neighbors from the minority class, this can be seen on Figure 3.3 where the class represented in yellow dots is being oversampled with SMOTE, the results of the oversampling are the green dots. The steps to generate the synthetic samples are:

1. Take the difference between the feature vector, also called sample, and the nearest neighbors;

2. Multiply that difference with a number between 0 and 1 and add it to the feature vector.

These steps will cause the creation of a sample that is a random point along the line that connects two of the samples from the minority class. Ultimately, this approach will force the decision region of the minority class to become more general.

SMOTE-NC is a variant of SMOTE that works with datasets that have both numerical and categorical data, this way we can have better synthetic samples for our dataset.

3.2.2.B ADASYN

Another method for oversampling is the adaptive synthetic sampling (ADASYN) [30], this method is based on the idea of adaptively generating minority data samples according to their distributions, this is, there will be more generated samples for instances that are harder to learn than those which are easier to learn. This method cannot only reduce the bias of having an imbalanced dataset as also to shift the decision boundary of the dataset to the samples which are harder to learn.

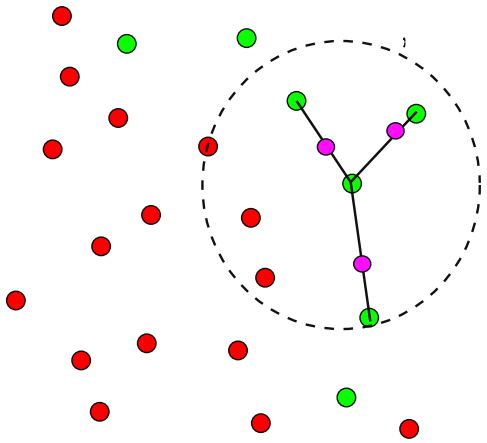


Figure 3.4: Example of ADASYN: Red Dots (Majority Class), Green Dots (Minority Class), Purple Dots (Synthetic Samples from the Minority Class).

The ADASYN algorithm works as follows, it first calculates the degree of class imbalance on the dataset, if the imbalance is less than the tolerated ratio of class imbalance it proceeds with the following execution:

1. Calculates the number of synthetic samples to be created from the minority class

2. For each instance of the database belonging to the minority class the algorithm finds the K nearest neighbors based on the Euclidean distance

3. The ratio of instances from the majority class in the K nearest neighbors r, is calculated and then normalized to create a density distribution

4. Finally the number of synthetic samples that need to be computed for each instance in the minority class is then calculated

The synthetic samples are calculated for each instance in the minority class, so for each one, the algorithm will randomly choose one minority instance from the K nearest neighbors group and then generate the synthetic example by choosing a random point in the imaginary line between those 2 data points, in a SMOTE fashion. The key feature in ADASYN is the criteria to decide the number of synthetic samples to be created for each minority instance, opposite to SMOTE which creates the same number for all instances.

3.2.2.C VAE

Another method of oversampling data that arises in the last years and is still in development is the use of Variational Auto Encoders (VAE). One of the algorithms that use this technique is SMRT, which is still in development by the authors, and is based on works [31, 32] using VAE's. The goal of using this technique is to create a variational autoencoder that fits the minority class data and then to use the same VAE to generate data that will be similar to the data that it has learned.

3.3 Feature Selection

One of the crucial problems in machine learning tasks is to separate the relevant features from the not so relevant in a dataset, this is called Feature Selection (FS) [33, 34]. This separation of features is very important because it allows the reduction of noise in the data we are using as stated in [2]. Also, by reducing the subset of features that are used, we reduce the classification model complexity, which in turn helps to prevent the over-fitting of the model. There are three main types of Feature Selection methods: filter, wrapper and embedded. The first one evaluates feature worth based on the characteristics of the data and is independent of the machine learning algorithm, this is a filter method, so it does not need any classification algorithm associated with it in order to perform the selection. Wrapper methods use the result obtained by a classification algorithm to see the importance of a subset of features. The last ones are a mix between feature selection and classification and the importance of feature is analyzed during the classification algorithm, one example is an L1 Regularization.

One of the most simple methods which was also used in [1] is the correlation feature selection (CFS) and is based on the correlation between the features in a dataset, and those features and the prediction, so it measures the usefulness of those features to predict the class.

The Recursive Feature Elimination (RFE) presented in [35], in opposite to correlation is a wrapper method, so it needs a classifier in order to perform the selection, the most common classifiers associated with this method are SVM's and Logistic Regression. This method starts by creating a set with all the features and then works by recursively removing the least important feature from that set until reaching the best set of features.

The Sequential Feature Selection is similar to RFE but this one uses a more complex methodology to find the subset of features, also, RFE uses weight coefficients or feature performance, while Sequential Feature Selection uses a user-defined classifier or metric to perform the selection. In the case of Forward and Backward Feature Selection, the first one starts with a null set and then starts adding features choosing in each addition the one that brings better performance to the subset, this happens until the desired number of features that we want to be selected is reached or achieved the best performance. The Backward method starts with all the features in the subset and then step by step removes the one which maximizes the performance of the subset in relation to the classifier used.

Another methodology that is widely used is the minimum redundancy maximum relevance (mRMR) [36]. This feature selection method tries to select the best set of features according to the maximal statistical dependency criterion based on mutual information. Due to the difficulty to implement the maximum relevance condition an equivalent form called the minimal-redundancy-maximal-relevance criterion was derived. After this, the criterion is applied with wrapper methods that are more sophisticated. With this combination of the methodologies, a more compact and superior subset of features can be selected with less computational cost.

A methodology that has been known for some years is Relief [37, 38], which is a filter-based method that was invented in the early 1990s. This method designs statistics to measure the importance of each feature and works by first randomly choosing a sample from the dataset, then it searches for the closest sample of the same class and the closest from the opposite class. For each feature, the algorithm follows the next equation.

$$W_i = W_i + (diff(x_i, x_{sameclass})^2 - diff(x_i, x_{oppositeclass})^2). \tag{3.1}$$

So, if the difference between the values of the first sample and instance of the same class is less than the difference with the opposite class it means that the feature is beneficial to distinguish classes, then the feature score will increase, if the opposite happens, then the feature score will decrease. These steps happen the necessary times to go through all the features and are repeated to average the results.

3.4 Feature Importance

Due to the rise of more complex Machine Learning Methods like Support Vector Machines and Neural Networks which are hard to analyze, because they do not provide any explanation for their predictions, there is the need to find methodologies that turn these methods easier to interpret. This is a need because these methods are usually far superior in terms of predicting than simple ones, and especially in health problems, the interpretability of a method is also a must have.

To help in this subject, researchers developed model-agnostic interpretability methods [39, 40], these describe the importance of each feature and their contribution to the prediction of a class, because each feature can have more power on predicting one specific class than others. From all the interpretability techniques, there are two which are the most used, SHAP and LIME.

SHapley Additive exPlanations or SHAP is based on Shapley Values, these are also based on the game coalition theory [41]. The basic methodology of this method is the following, it first combines two features and checks their impact on the prediction and evaluates their importance with a weight, then it combines another feature and assigns a weight to it, and so on until all the features are combined. With this method, we can get the importance of each feature for the prediction.

The other mentioned method is the Linear Model Agnostic Explanation or LIME method, and it is made to analyze a particular instance of the dataset (row). The LIME method uses models like Logistic Regressions and Decisions Trees to fit the predictions made by the black-box model.

This method works by taking one instance of the original dataset and making samples that are close to it, basically changing the values of the features by a small amount so they are in the vicinity of the original sample, these samples are weighted by the proximity of the original instance, this to ensure that errors in the samples that are closest are more weighted than the ones farther. After this step, the samples are sent to the original model and then it gets the predictions, then a simple model like the ones stated before is fitted into those data points. Finally LIME explains the predictions by interpreting the simple model created.

The downside of this method is that LIME is a local algorithm, so we only get the interpretation for one instance of the dataset, this is susceptible to errors because we are not making an interpretation based on the whole dataset.

3.5 Overview of Classification Methods

There are many classification methods that can be used, in this section it is presented the ones used in this work, which are Naïve Bayes, Support Vector Machines (SVMs), Logistic Regression, K Nearest Neighbors and Neural Networks.

3.5.1 Naïve Bayes

The Naïve Bayes [42] relies on the assumption that all the features are independent. It uses the Bayes theorem given by

$$P(C_i|X = x) = \frac{P(X = x|C_i)P(C_i)}{P(X = x)}, \tag{3.2}$$

in which C_i is a class, X is the data, $P(X = x|C_i)$ is the class conditional probability distribution, $P(C_i)$ is the prior distribution and $P(X = x)$ is the data distribution. The Naïve Bayes classifier chooses the class with the highest posteriori probability $P(C_i|x)$, for instance, if $P(C_i|X = x) > P(C_j|X = x), \forall i \neq j$, then it will classify x as belonging to class i. As $P(X = x)$ is the same for all the classes, then it will be ignored, so it only needs to maximize the upper term on the equation (3.2). Also, assuming that the features are independent, then we obtain:

$$P(C_i|X = x) \propto \prod_{j=1}^{N} P(X_j = x_j|C_i)P(C_i). \tag{3.3}$$

This is the case where we know the classes probability $P(C_i)$, if we do not know that, we assume that all the classes are equally likely, then we assume that:

$$P(C_i|X = x) \propto \prod_{j=1}^{N} P(X_j = x_j|C_i). \tag{3.4}$$

3.5.2 Support Vector Machines

Support Vector Machines (SVM) [43] is a method for classification that works by separating the training data with hyperplanes, these hyperplanes divide the data into classes and they can be described by the following equation:

$$(w \cdot x) - b = 0, w \in \mathbb{R}^n, b \in \mathbb{R}. \tag{3.5}$$

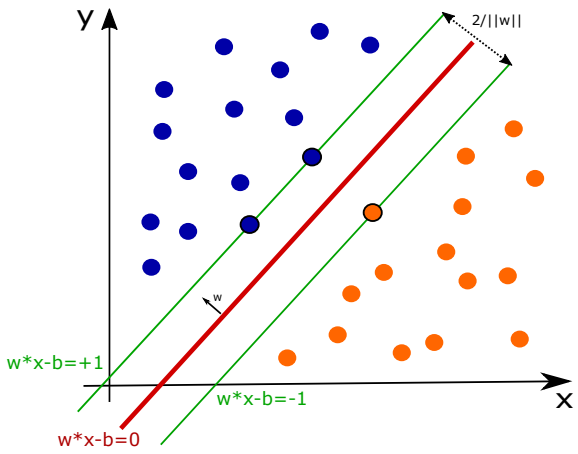


Figure 3.5: SVM Separation of Data Example.

To find the optimal hyperplane, which is the one that provides the maximal margin of separation between the classes, we need to find the combination of support vectors. This combination w can be calculated as

$$w = \sum v_i x_i, \tag{3.6}$$

where $v_i = \alpha_i y_i$ are the support vectors, α_i are the Lagrange Multipliers and $y_i = \{-1, 1\}$ is the class true label. These support vectors (closest points to the hyperplane), in Figure 3.5 they are the dots on the green lines, carry the relevant information about the classification problem. The SVM will try to create the decision boundary in a way that maximizes the margin ($\frac{2}{||w||}$), because with a greater margin there is a smaller probability that the SVM mislabels the instance to be tested. So, as the optimal hyperplane is found, the decision function can be written as $f(x) = sign((w \cdot x) - b)$ which is the same as

$$f(x) = sign\left(\sum_i v_i x_i \cdot x - b\right), b \in \mathbb{R}. \tag{3.7}$$

Since not all the datasets can be linearly separated we need to have another form to use SVM's, this is the case where kernels help. Kernels are a transformation on the feature space which lets us turn a non-linear dataset into one that is by switching the space where the variables are represented, so $\tilde{x} = \phi(x)$, where $\phi()$ is a non-linear mapping function. The non-linear transformation kernel will be $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$. Replacing the kernel in the decision function 3.7 we obtain

$$f(x_j) = sign\left(\left[\sum_i v_i K(x_i, x_j)\right] - b\right), b \in \mathbb{R}. \tag{3.8}$$

The kernels that are usually used are:

- Linear - $K(x_i, x_j) = x_i^T \cdot x_j$;
- Polynomial - $K(x_i, x_j) = (x_i^T \cdot x_j + a)^b, a, b \in \mathbb{R}$
- RBF (Radial Basis Function) - $K(x_i, x_j) = e^{-\frac{1}{2\sigma^2} \cdot ||x_i - x_j||^2}$.

3.5.3 Logistic Regression

The Logistic Regression (LR) [44] classifier is based on a posteriori probability with the use of the logistic function (3.9). It is an adaptation of the linear regression that uses probabilities, which is better for classification problems, because instead of giving a value in an interval, it will give a probability of that value belonging to that class.

$$g(x) = \frac{1}{1 + e^{-x}}, \tag{3.9}$$

For binary classification problems, the model can be written as

$$P(C_i = 1|x) = g(x^T \beta), P(C_i = 0|x) = 1 - g(x^T \beta), \tag{3.10}$$

$$P(C_i = 1|x) = \frac{1}{1 + e^{\beta_0 + \beta_1 x_1 + ... + \beta_n x_n}}. \tag{3.11}$$

The coefficients β can be calculated with a method like Gradient Descent or the Newton Method [45]. In the case of a binary classification problem, if the probability given by the function (3.11) is greater than 0.5 then the instance given belongs to that class, if it is less than 0.5 then it belong to the other class.

3.5.4 K Nearest Neighbors

The K Nearest Neighbors (KNN) [46] classifier is an extension of the Nearest Neighbor classifier. Supposing we have a training set $T = \{(x_i, y_i), i = 1, ..., n\}$, the easiest strategy to make a prediction on y based on x, is by finding the instances x_i nearest to x and approximate y by y_i. Suppose that the nearest neighbor from x is x_i, the outcome of the classifier will be y_i. The nearest neighbor can be obtained using a distance, this distance can be computed, for example with the Euclidean distance.

The K Nearest Neighbors is identical to the last one, but takes into account the prediction based on the K nearest neighbors instead of just one neighbor. So the outcome of the classifier will be either the most voted class, for classification problems, or the average of the values given by each neighbor, for regression problems.

3.5.5 Neural Networks

Neural Networks (NN) [47] are based on the way that the human brain works, this gives them a great pattern recognition capability, which is why they are widely used on several problems such as image recognition tasks. A NN is composed by several perceptrons that are similar to human neurons, these are composed by summations and weight, the equation associated with these are shown in Equations (3.12) and (3.13).

$$z_i = \sum_{j=1}^{k} w_{ij} x_{ij} + bias, \tag{3.12}$$

$$y_i = F(z_i). \tag{3.13}$$

In equation (3.12), w_{ij} is the weight from the connection between neuron i and j, and the bias is the bias from the neuron. The function denoted by $F()$ in (3.13), is called the activation function, this is a differentiable and function and it can be for example a sigmoid or a relu function. The goal of this function is to keep the output values of the network between certain values, in order for the network values not to raise indefinitely, this is why it can be also called squashing function.

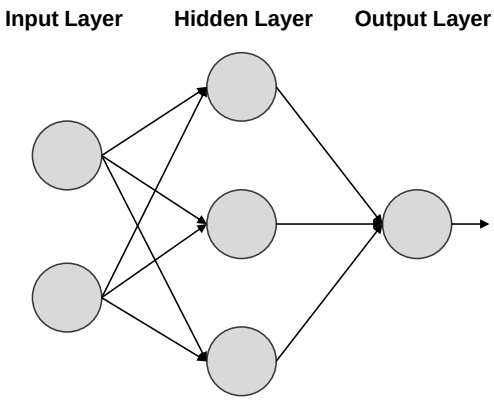


Figure 3.6: Multilayer Perceptron.

The so-called Neural Network is nothing more than simply a set of those neurons organized in layers, as shown in Figure (3.6). There are three types of layers, the input, the hidden and the output layer. The first one is just the training instances from the dataset and the other two include neurons. In several layers case, the outputs from one hidden layer will be connected to the inputs from the next hidden layer, and the output layer will give the final result, the prediction. In binary classification cases we usually just

have one perceptron in the output layer which will give either 0 or 1 for the predicted class, in the case where we have more classes, we will have one perceptron per class in the output layer.

One important step from the neural networks which provides the learning capability is the Bakpropagation step, this step will compare the output value of the network with the real value provided by the user. It will make the comparison between values with the help from a loss function, for example the Mean Squared Error (MSE), after analyzing this error, the network will adjust the weights and bias of the perceptrons to reduce that error. These are made by differentiating the loss to each of the weights $\frac{\partial L}{\partial w_{ij}}$. After this propagation, when the first layer is reached, the network will update the weights to minimize the Loss.

One of the techniques that are being applied in Neural Networks to optimize and ”simplify” these classifiers, reducing the processing power needed to train them and also make better predictions is the Pruning Technique [48–51]. This type of technique is being widely used in heavy networks such as ResNet [50, 51], in order to reduce the training time and consequently provide researchers more time to tune and develop the network. In the works cited before, the results obtained with a large amount of pruning only slightly decreased the accuracy in some cases while in others the accuracy improved together with the training time. The most common approach of pruning [51] can be divided into two steps that happen on each epoch of the training of the network, in the first step after computing the gradient of the weights for the update, the importance of each of the neurons is analyzed using the average gradient, after this, the second step consists on removing the less important neurons on the network.

3.6 Metrics for Model Evaluation

When working on a classification problem, we need to find out which of the classifiers works best or if one is working at all on predicting the results. For this purpose, we need to resort to metrics to tell us if the model we are using is good or not when the model is applied to the test set. Four of the most common metrics that are used in machine learning tasks and the medical environment are: the accuracy, the Area Under the ROC Curve which is usually called AUC, the sensitivity and specificity.

To calculate these metrics it is needed to define the so-called Confusion Matrix, which is shown in Table 3.1. This matrix helps us calculating all the metrics previously mentioned, and there are four elements we need to know:

- True Positive (TP) - Positive predictions that are correctly classified;
- True Negatives (TN) - Negative Predictions that are correctly classified;
- False Positives (FP) - Positive Predictions that are incorrectly classified;
- False Negatives (FN) - Negative Predictions that are incorrectly classified.

Table 3.1: Confusion Matrix for two Classes.

	C_1'	C_2'
C_1	True Positive (TP)	False Negative (FN)
C_2	False Positive (FP)	True Negative (TN)

On the confusion matrix, C_1' and C_2' are the predicted values for the positive and negative classes and C_1 and C_2 are the actual values for those classes.

The **Accuracy** can be measured by the ratio between the correctly classified predictions and all the predictions. The accuracy can be computed as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3.14}$$

The **Sensitivity**, also called True Positive Rate (TPR), is the ratio between the correct predictions of the class C_1 and all the predictions of that class C_1. The sensitivity is given by

$$Sensitivity = \frac{TP}{TP + FN}. \tag{3.15}$$

The **Specificity**, also called True Negative Rate (TNR), is the ratio between the correct predictions of the class C_2 and all the predictions of that class C_2. The specificity is obtained as

$$Specificity = \frac{TN}{TN + FP}. \tag{3.16}$$

The **Receiving Operator Characteritics** (ROC) curve [52], shown in Figure 3.7, is a plot that shows the performance of a model as is discriminant threshold is varied. It is created by plotting the True Positive Rate (Sensitivity) against the False Positive Rate (1- Specificity) at various different thresholds. For instance, if we increase the threshold, then we will have fewer false positives and consequently more false negatives. These are really useful tools to compare models because they are designed to see if a model can distinguish between true positives and negatives. As can be seen in Figure (3.7), a good classifier should have the ROC Curve above the diagonal of the chart. The better the model, the more it approximates to the top left corner of the chart, that is where the perfect separation of data is achieved.

The **Area Under the ROC Curve** (AUC), is another good metric to evaluate a classification model, it is derived from the ROC Curve and it is usually calculated just by measuring the area under the ROC Curve on a plot. The AUC values are between 0 and 1, 0.5 means that there is no separation of data, 1 means that it predicts all the classes correctly and 0 means that predicts them all wrong. This means that a classifier that has an AUC score of 0 has more capacity of separating data than one that as a score of 0.5, although this classifier will give the incorrect result, for example in a binary classification problem if the correct classification is 1 the classifier will predict as 0.

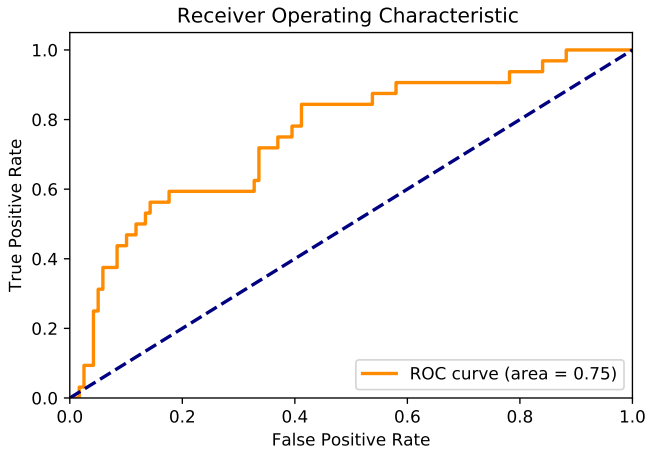


Figure 3.7: ROC Curve Example with Polynomial SVM.

3.7 Summary

In this section we have presented the pipeline that will be used to accomplish the work proposed and analyzed every component of this pipeline. We have seen how the missing values where imputed, like in [1, 2], we saw how to overcome the problem of the imbalanced classes on our Database, which can be fixed using data balance techniques and consequently expand our dataset with the same techniques. Another of the problems that we had to deal was the selection of the most useful features and for this problem we presented the methods that will be used in this work. Finally we saw one of the most important parts in this work, the machine learning methods that will be used, in here we explained how they work and also we have seen the metrics used to evaluate all the methods, because a method is not good until we can prove what he does. All these methods will be applied in predicting the progression of the Alzheimer Disease, a process which will be explained in the next chapter.

4

Proposed Classification Methodology

Contents

To predict the conversion from MCI to AD, a pipeline was implemented with four stages to help achieve a better result, these stages are Missing Value Imputation, Data Balance and Oversampling, Feature Selection, and finally the Classifier which is the stage that makes the prediction. For this specific problem, the pipeline presented in Figure 4.1 was created.

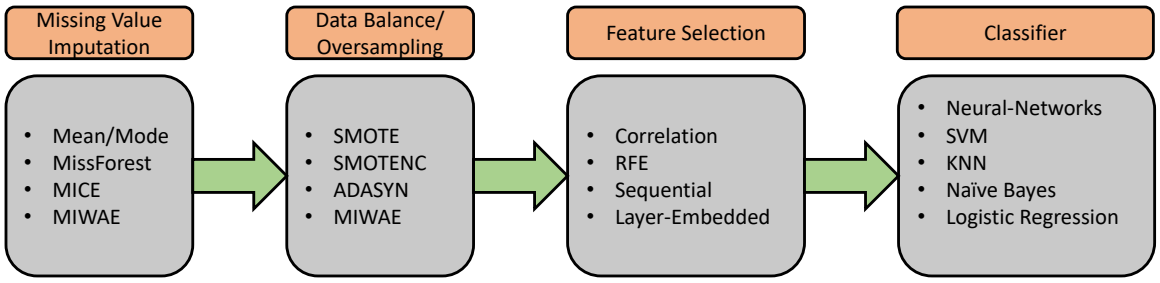


Figure 4.1: Methodology to predict AD Progression.

As can be seen in the Figure 4.1, the first step is to perform the missing value imputation on the dataset, since, like most of the real data, not all the information is presented, either because a patient skipped a test or it was not available at the time. So for our classifiers to work there is the need to fill those missing values.

The following step is to perform the data balance and oversampling. This is an important step because like in the previous step, real datasets are not always class balanced which means that the classifier will not learn both classes with the same perfection. The other problem is that a huge dataset to work with is not always available, it takes a lot of years to build a dataset with a sufficient amount of information to make better predictions, so to improve this problem we need to perform oversampling to help increase the number of samples for each class.

After improving the dataset by performing the missing value imputation, and the data balance there is another step to help improve the classification. This step is the Feature Selection, since there are a lot of features in this dataset there is the need to find a way to reduce them to the most essential ones, i.e., the ones who help more at making the prediction. This is a really important step because it reduces the amount of data being sent to the classifier which makes the classifier less complex and also improves the execution times of the classifiers.

The last step which is the one where we can see the effects of the previous ones is the classification model. With this step, we can finally predict if the patient will convert or not to AD and get the overall metrics for each classifier.

All the code necessary to build this pipeline and obtain the results reported was made in Python (version 3.7.4), this because of the large library that it has for machine learning purposes. The primary libraries used were the Scikit-Learn [53], for the major part of machine learning and feature selection,

missingpy and *fancyimpute* for missing value imputation, and the Tensorflow/Keras [54,55] for the Neural Networks classifier and the Layer Embedded Feature Selection.

4.1 Missing Value Imputation

To perform the Missing Value Imputation four different methods were used and their capabilities were evaluated, Mean/Mode, MissForest, MICE and MIWAE.

The first method analyzed was the mean/mode method, in this methodology, the NaN's were replaced by either the mean or the mode of the features if the data was numerical or categorical respectively.

MissForest [56] was the second method analyzed, this one is based on the Random Forest algorithm. The first step of this algorithm is to make an initial guess for the missing values by replacing them with the mean or the mode of each feature. Then the algorithm fits random forest to the previous imputed dataset which is then used to predict one of the previously imputed missing values and replaces it. The dataset is then updated and the previous step happens again but for another missing value, this happens time after time until a stopping criterion is fulfilled.

Multiple imputations by chained equations or MICE [57, 58] method is another method to replace missing data. This methodology works best when the data is missing at random or completely at random (MAR or MCAR). This method works in a divide and conquer fashion, in a dataset with M features from $x_1, ..., x_M$ the first step is to fill all the missing values with some basic sort of imputation, for example with the mean or the mode, after this imputation, from first feature x_1 are removed all the imputations and with all the other features, using a regression, new values for the missing values are generated. This step is done until all the features from $x_1, ..., x_M$ are imputed, this is called a cycle. This is done usually around 10 times for the data to stabilize and to obtain a better imputation.

The last methodology for Missing Value Imputation is the missing data importance-weighted autoencoder or MIWAE [26] which is based on the importance-weighted autoencoder(IWAE) [59]. The MIWAE goal is to fit a Deep Latent Variable Model (DLVM) into a dataset with missing data. DLVM's are latent variable models that use deep neural network architectures to ensure a higher flexibility on learning the underlying structure of data, such as clusters, patterns or statistical correlations. This type of models have problems when handling datasets with missing values. The usual methodologies when handling DLVM's such as variational autoencoders (VAE) or IWAE assume that the training data is fully observed, so MIWAE tries to overcome this limits. After training the DLVM using the MIWAE bound for missing data applications, this DLVM already knows the data distribution it can know fill the missing values with values that follow approximately the same distribution.

4.2 Oversampling

To perform the balance of the dataset and to increase the training samples, four different mechanisms were used, SMOTE, SMOTENC, ADASYN and MIWAE . The first three methodologies were already explained in section 3 so their mechanism is already known.

The last methodology is using MIWAE to perform the oversampling. MIWAE was created only to perform the imputation of data, but one idea that arises in this thesis was that since the autoencoder already knows the data distribution from performing the missing value imputation, it could be used to generate new data to balance the classes.

As stated in the previous section, the model is built using a deep latent variable model. More specifically it is a Deep Latent Variable Model (DLVM) with a Gaussian prior and a Student's t observation model as in Equation (4.1).

$$p(x_i|z_i) = St(x_i|\mu_\theta(z_i), \Sigma_\theta(z_i), \nu_\theta(z_i)), \tag{4.1}$$

where μ_θ, Σ_θ, ν_θ are functions parametrised by the deep neural network, whose weights are stored in θ and x_i and z_i are respectively, the data instances and the latent variables. After this, a decoder is built to support the three previous functions (μ_θ, Σ_θ, ν_θ), the encoder or inference network is then built using a Student's t approximation with an architecture that is similar to the decoder. The MIWAE bound is defined by the following equation:

$$\mathcal{L}_K(\theta, \gamma) = \sum_{i=1}^{n} E_{z_{i1},\ldots,z_{iK}\sim q_\gamma(z|x_i^o)} \left[\frac{1}{K} \sum_{k=1}^{K} \frac{p_\theta(x_i^o|z_{ik})p(z_{ik})}{q_\gamma(z_{ik}|x_i^o)} \right]. \tag{4.2}$$

Where p_θ is the posterior distribution, q_γ is the conditional distribution, p is the prior distribution and z_{ik} are auxiliary variables.

The optimal imputation will be the conditional mean $E[x^m|x^o]$ that can be estimated with

$$E[x^m|x^o] \approx \sum_{l=1}^{L} w_l E[x^m|x^o, z_{(l)}] = \sum_{l=1}^{L} w_l \mu_\theta(z_l)^m, \tag{4.3}$$

where

$$w_l = \frac{r_l}{r_1 + \ldots + r_L}, r_l = \frac{p_\theta(x_i^o|z_{(l)})p(z_{(l)})}{q_\gamma(z_{(l)}|x_i^o)}. \tag{4.4}$$

By using the same procedure, we can use the same architecture to generate a whole row of data instead of only a few values for imputation,this is done by feeding the VAE with a empty row to evaluate instead of a row of data that only has some missing values, by estimating this one, a whole instance of a class can be produced.

With this methodology, the data generated follows a distribution that is close to the one from the

original dataset, and with this more reliable results can be produced.

4.3 Feature Selection

The third step of our pipeline is to perform the Feature Selection on our dataset, for this purpose, four different methods were used from which three were already explained previously (Correlation, RFE, Sequential).

The last methodology proposed for this thesis and relies on Neural Networks to perform the Feature Selection, this method is based on the concept of pruning techniques and the goal is to eliminate the less useful features influence from the network to improve the overall accuracy of the results.

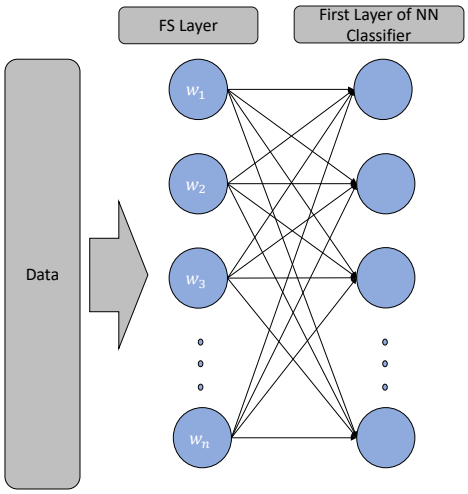


Figure 4.2: Layer Embedded Feature Selection.

This methodology can be seen in Figure 4.2 where the first layer of the network is responsible to assign weights to the features. These weights are multiplied by the input features which are made to tend to zero by the loss function $Loss$.

$$Loss = MSE + \sum_{i=0}^{N} |w_i|. \tag{4.5}$$

So the loss function of the network will tend to zero, and consequently the weights will also tend to zero, this means that the features of the dataset f_i will be turned into f_i' in which

$$f_i' = w_i * f_i. \tag{4.6}$$

In order to make this method more adjustable a threshold t set by the user can be modified, and these

weights w_i will be set to zero depending on this parameter t. This because neural networks do not naturally set the weights to zero, so this is done at the end of every batch on the training step, if a weight is below the threshold it will be set to zero, else it remains the same following the equation:

$$\begin{cases} 0, & \text{if } |w_i| < t \\ w_i, & \text{otherwise.} \end{cases} \tag{4.7}$$

This adjustment in the parameter t changes the amount of features chosen, if the parameter is increased the amount of features chosen will decrease, if t is decreased, the amount of features will increase. The benefits of using this method is that we do not need any independent feature selection algorithm as it is already embedded in the classifier, also as it uses neural networks, it has a high adaptability to the datasets being used. The architecture of the Neural Network on which the FS method will be applied is the following:

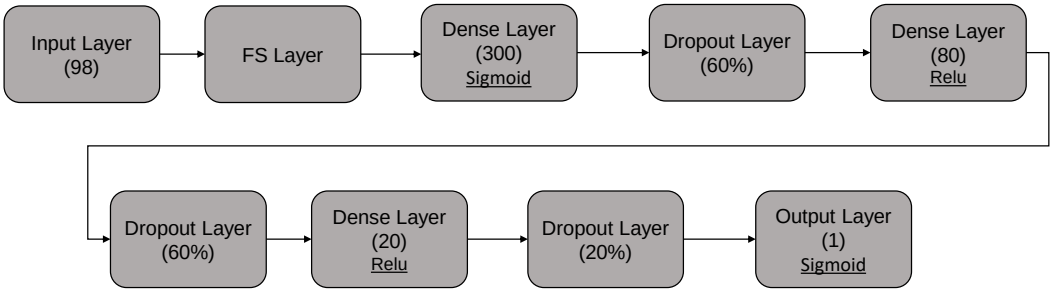


Figure 4.3: Neural Network Architecture.

The activation functions used were the sigmoid for the first and last dense layers and relu for the two middle dense layers, the loss function used was the sum of the Mean Squared Error with the custom loss in Equation 4.5.

In this section, the pipeline used to achieve the goal of this work and how every element of it is important was presented. First, the handling of missing values and the methods used were explained in more detail. Next, the data oversampling importance and how a methodology used for Missing Value Imputation can be used to create more samples for our dataset was explained. Finally, a new methodology for Feature Selection proposed for this thesis was presented as well as how it works. Now in the next chapter, these methodologies will be applied to our datasets and evaluated.

5

Predicting the conversion from MCI to AD

Contents

In the previous chapter, it was introduced all the methods that will be used in this section, the classifiers, the data balance and oversampling, and the feature selection methods. Now an evaluation of their performance will be done in the CCC database presented in section 2.

5.1 Experimental Setup

Following the work previously done in [1, 2] using time windows, the first step was to create a simple pipeline Fig 5.1 using the previously explained methods.

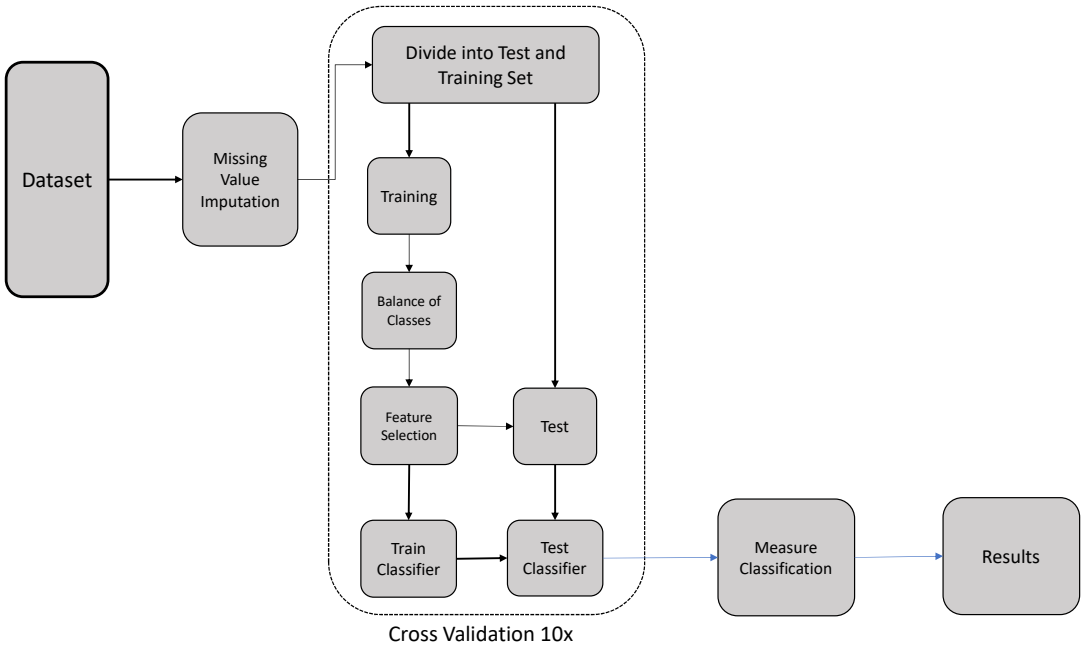


Figure 5.1: Pipeline Created for the prediction of AD Progression.

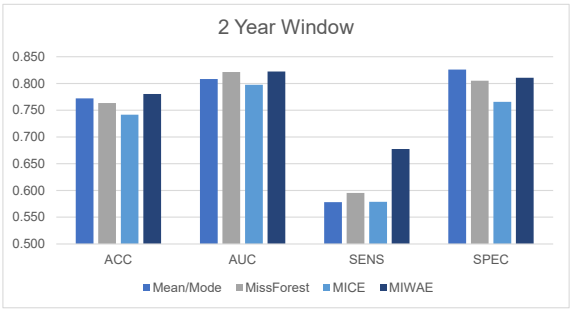
To perform the validation of the algorithms four different datasets/time windows were used, from 2 to 5 years, and a 10-fold cross-validation methodology in each of the datasets to average the results.

To find the best combination of methodologies to predict the conversion from MCI to AD, an extensive number of tests was performed. This consisted of gradually replacing and testing each component (MVI, Data Balance, and FS) until obtaining the best combination.

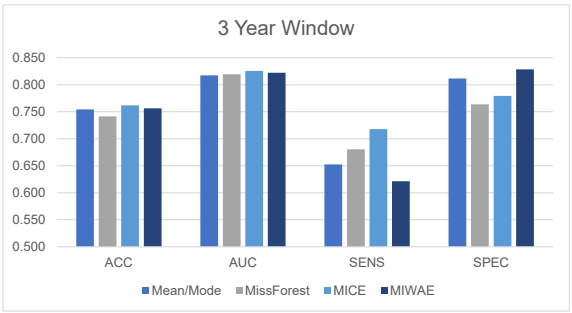
5.2 Missing Value Imputation

The start combination was SMOTE for data balance and the layer embedded feature selection and consequently, the classifier used the proposed neural network. Four different methods were used for Missing Value Imputation, MICE, MissForest, Mean/Mode, and MIWAE, these four methods were analyzed in the four different datasets.

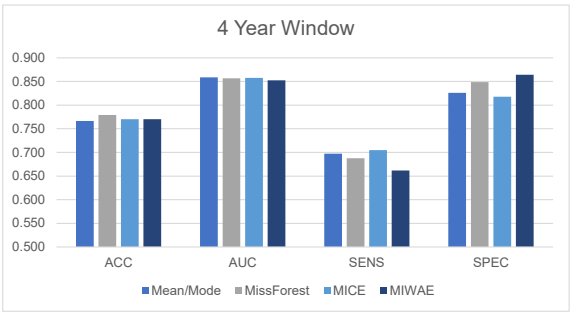
In the following tests, none of the Missing Value Imputation methodologies had parameters to be adjusted, the SMOTE algorithm was used with a sampling strategy of 500 samples for each class to balance and increase the number of samples in the dataset. The feature selection algorithm embedded in the neural network had a value of $t = 0.1$, and the neural network had previously proposed architecture.



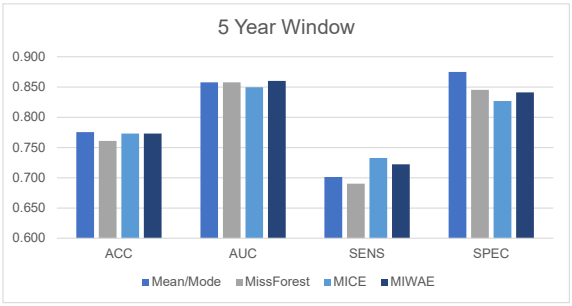
(a) 2 Year Window.



(b) 3 Year Window.



(c) 4 Year Window.



(d) 5 Year Window.

Figure 5.2: Missing Value Imputation Results for the four time windows.

By analyzing the results obtained in Figure 5.2, we can see that overall the results are very similar between the MVI methods, but the MIWAE methodology has a slight overall performance in the AUC score, so that is the method we will use for the next section.

5.3 Oversampling

Now that we have a fixed Missing Value Imputation methodology (MIWAE) the next step is to find the best methodology for oversampling and balance. Four methodologies were used to perform the data balance and oversampling the dataset, SMOTE, SMOTENC, ADASYN, and the MIWAE adaptation proposed in this thesis to create samples. These four methods were also analyzed in four different time windows. In all the methods the sampling strategy was 500 samples for each class, which would increase the samples of the dataset and balance the same.

To be sure that when a oversampling algorithm is applied to the dataset, this one still remains reliable for predicting the conversion, a test was made to prove it. This test consisted of dividing the dataset into training and test, 50% for each set, after this, the training set was gradually reduced maintaining the original class proportions, and two of the most used algorithms of data balance, SMOTE and ADASYN, were used to perform oversampling to balance and to fill the missing part of the training set that was removed as shown in Figure 5.3.

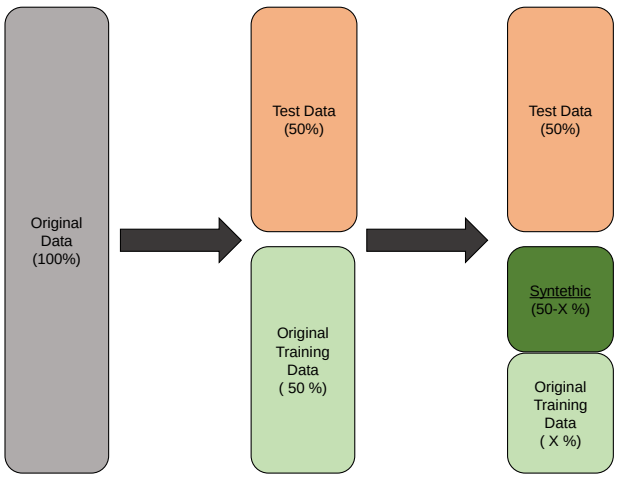


Figure 5.3: Methodology to evaluate Oversampling.

The accuracy results were obtained in the letter recognition dataset using a Neural Network with the following structure:

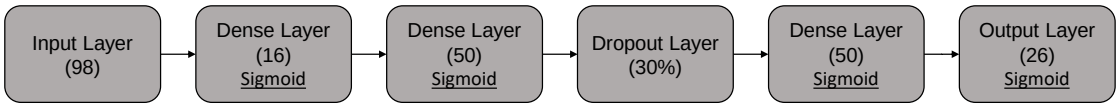


Figure 5.4: Neural Network architecture for evaluating oversampling techniques.

The results obtained with that experimental setup to compare the oversampling techniques are shown in Figure 5.5.

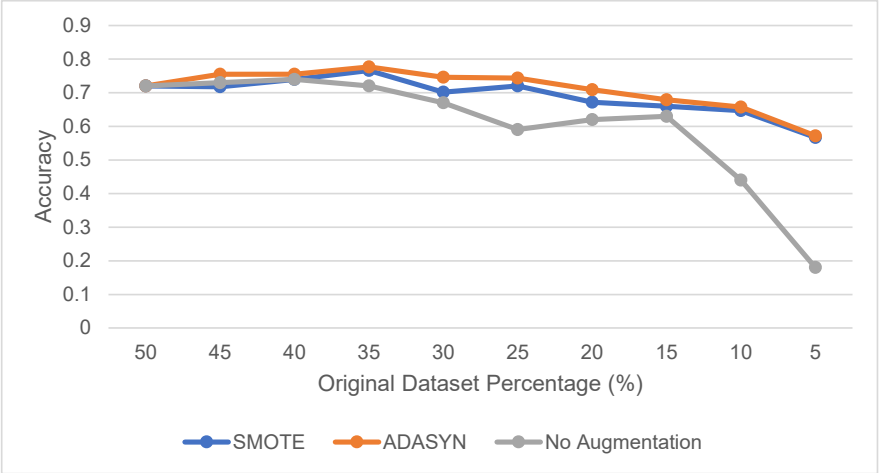
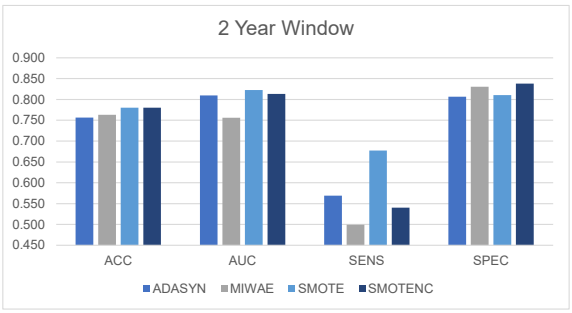
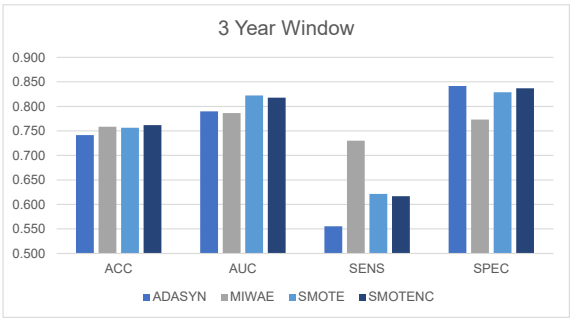


Figure 5.5: Relation between accuracy and the percentage of original training data.

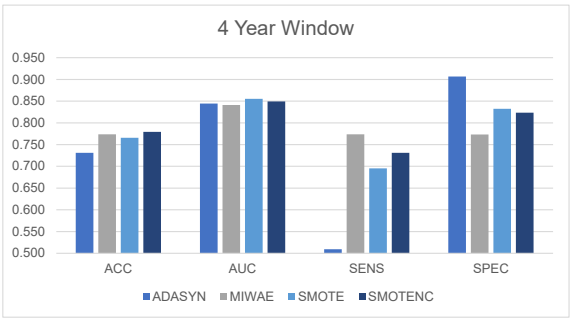
From Figure 5.5, we can see that as the percentage of data used for training the classifier is reduced the accuracy also reduces. This reduction is more substantial if no oversampling method is used as shown in the grey line. If a oversampling method is used, orange and blue lines, the accuracy reduction is not so noticeable, only declining substantially if the original data is reduced to around 20%, with the remaining 80% being synthetic data. This result means that the use of data balance methods to increase the size of the dataset can provide useful results as long as the synthetic data does not exceed 80% of the dataset being used.



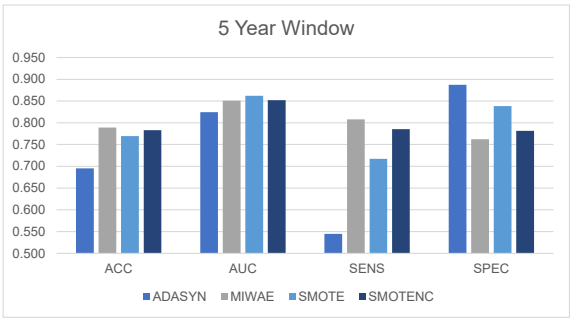
(a) 2 Year Window.



(b) 3 Year Window.



(c) 4 Year Window.



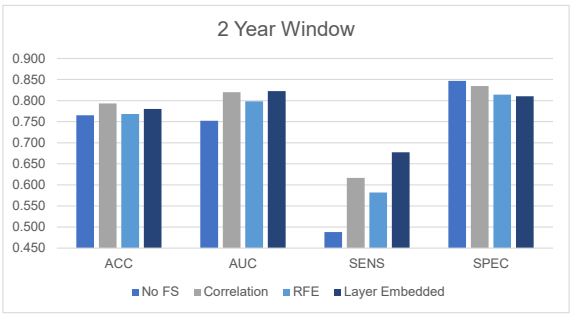
(d) 5 Year Window.

Figure 5.6: Comparison of the four data balance techniques on the four time windows.

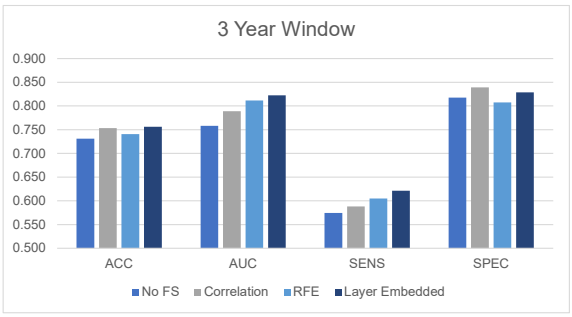
From the analysis of Figure 5.6, we can see that in terms of Area Under the ROC Curve (AUC) the methodology that maximizes this metric is the SMOTE technique, being the best across all four datasets. The proposed technique using MIWAE has a comparable performance on the 3, 4 and 5 year window, but in terms of sensibility it leads across all four windows, being better than the other techniques. Overall, these results indicate that the best methodology to increase the samples on our datasets and to perform the balance is the SMOTE technique because it delivers the more solid results across all windows and metrics, so this is the one which will be fixed for the following tests.

5.4 Feature Selection

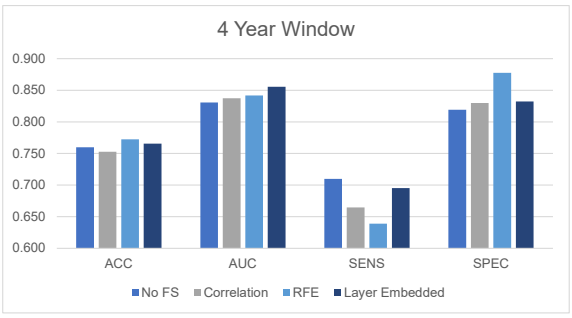
Continuing with the proposed pipeline, and with the methodologies for Missing Value Imputation and Data Balance chosen, being respectively MIWAE and SMOTE, the next stage is to evaluate the best Feature Selection methodology.



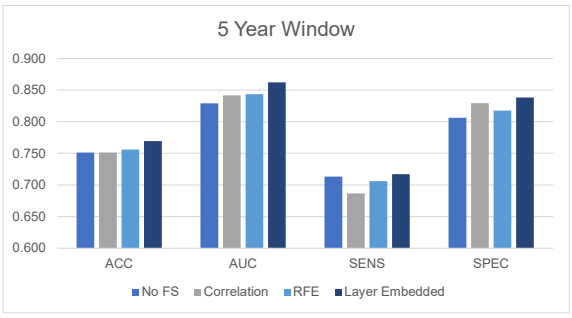
(a) 2 Year Window.



(b) 3 Year Window.



(c) 4 Year Window.



(d) 5 Year Window.

Figure 5.7: Results comparing the three FS methodologies with no use of FS on the four datasets.

As can be seen in Figure 5.7, in all datasets, the methodology that best performed in terms of AUC was our feature selection layer embedded in the neural network. Even in the other metrics, it was one of the best, being on top almost every time. The overall best results can be seen in Table 5.1 for all four datasets.

Table 5.1: Overall best results on the four time windows.

	ACC	AUC	SENS	SPEC
2 Year	0.780±0.03	0.822±0.03	0.677±0.13	0.811±0.05
3 Year	0.756±0.03	0.822±0.03	0.621±0.11	0.829±0.08
4 Year	0.766±0.07	0.855±0.04	0.695±0.12	0.832±0.07
5 Year	0.770±0.04	0.862±0.04	0.717±0.08	0.838±0.07

In Table 5.2 are the features chosen by our methodology which are exclusively common across all the time windows and across three time windows, these are the ones which are more important for the

prediction. The features in bold are the ones which are also common with the ones in [2].

Table 5.2: Most Common Feature selected by the proposed methodology.

Common Across All Datasets	Common Across 3 Datasets
PA_Dif_Total	DS_Forward
MVI_Free	LM_a_Total
MVI_Tot	***LM_a_Cued***
Orient_T	VisualM_B
Fluency_Sem	***Or_Total***
MPR_Total	Orient_P
a1_a5_Total	Proverb_Total
a_cr_int	MMS_Orientation_total
Depressao_GDS	MMS_OrientationTemporal_Total
MVI_Tot_Z	a_lg_int
Orient_T_Z	As_tot_Z
M_Initiative_Z	DS_back_Z
Proverb_Total_Z	TMT_B_temp_Z
LM_a_Total_Z	
LM_a_Interf_Z	

Another technique tested to see the most useful features was SHAP which is a Feature Importance technique. This methodology was applied to the proposed Neural Network with the FS layer in order to see the the importance of the features chosen by our classifier, the results obtained are shown in Figure 5.8 for the 2 Year Window, as similar results appeared for the remaining three datasets.

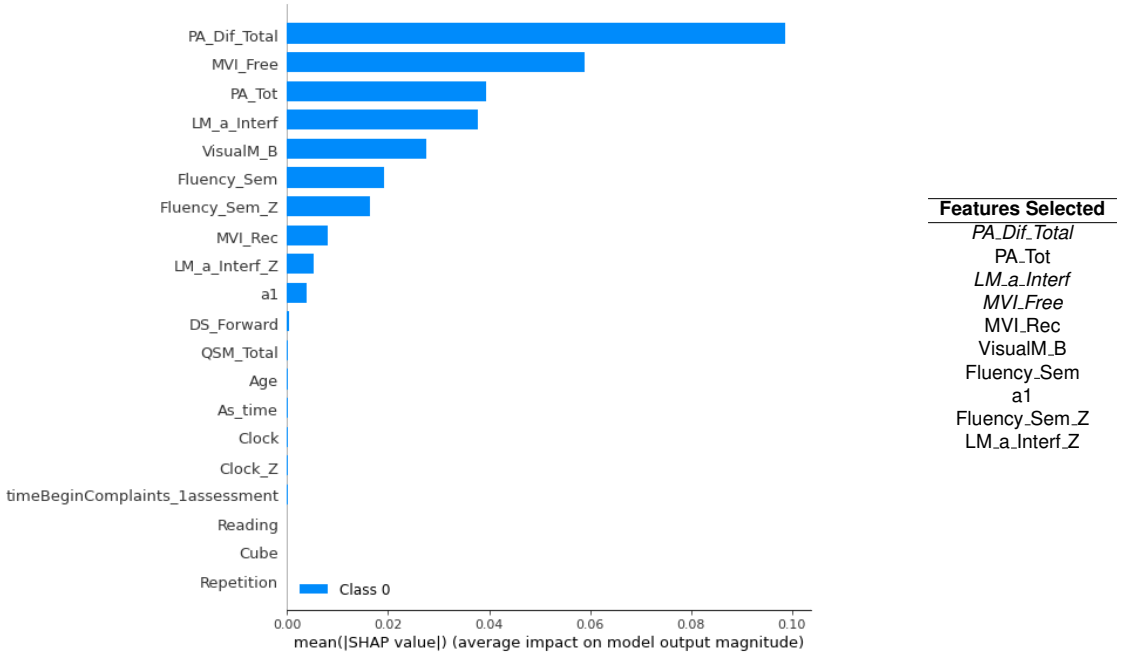


Figure 5.8: SHAP Feature Importance Results on the 2 Year Window.

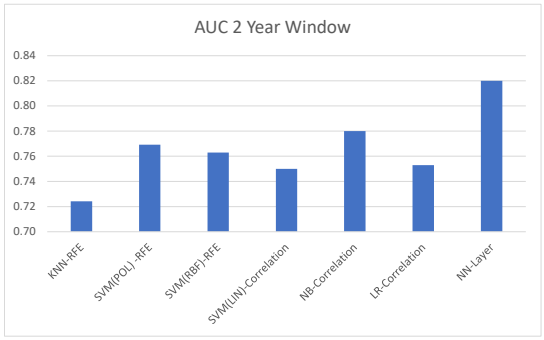
As seen in Figure 5.8 on the right there is the SHAP importance plot and on the left the features chosen by the proposed FS methodology, SHAP performs a classification for the usefulness of the features to predict the class 0. By the analysis we can see that the features which have the best capability are *Pa_Dif_Total* and *MVI_Free*, also we can confirm that the features that are more important are indeed the ones chosen by our method, which means that the FS is working properly. The upside of using this methodology allied with the Feature Selection is that it allows a better interpretability for medical staff to see what exams/tests are better at predicting the conversion to Alzheimer's Disease.

5.5 Comparison with other classifiers

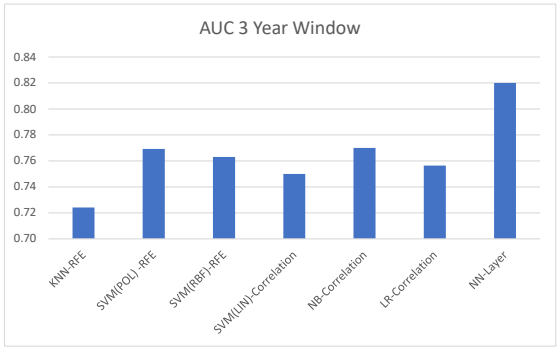
Following the work previously done in [1, 2] using time windows and using a simple pipeline Fig 5.1 consisting of four different feature selection methods and seven different classifiers a comparison was made between these methods and the results obtained in the previous section. To note that for every time window used, gridsearch was used to find the best parameters in an interval for the classifiers, the intervals were :

- KNN - Number of Neighbors $\in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$
- LR - Solver $\in \{`newton - cg', `lbfgs', `liblinear', `sag', `saga'\}$
- SVM Polinomial - Degree $\in \{1, 2, 3\}, C \in \{0.1, .., 1, 2, 3, ..., 10\}$
- SVM RBF - $C \in \{0.1, .., 1, 2, 3, ..., 10\}$, $\gamma = [10^{-2}, 10^{2}]$
- SVM Linear - $C \in \{0.1, .., 1, 2, 3, ..., 10\}$

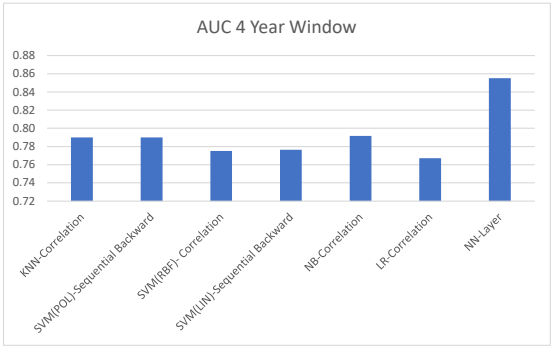
Figure 5.9 shows the AUC results of the previous classifiers with the best combination of feature selection method for every time window, this combination was performed in the previous work in IIEEC by trying every classification methodology with every FS method (RFE, Correlation, Sequential Backward and Forward).



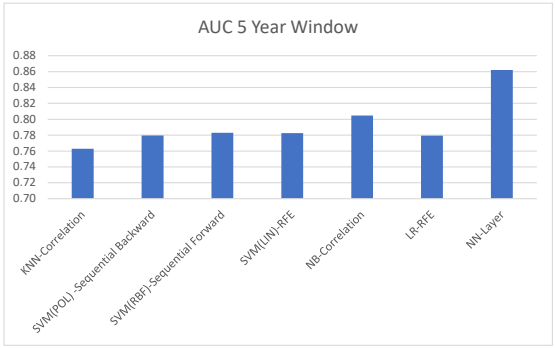
(a) 2 Year Window.



(b) 3 Year Window.



(c) 4 Year Window.



(d) 5 Year Window.

Figure 5.9: Comparison between classifiers.

As seen in Figure 5.9, the overall best methodology is the Neural Network Architecture with the Feature Selection Layer proposed in this thesis. This has proved to be the best in all the four time windows. While Naïve Bayes has proved to be one of the best in the work made by Telma Pereira in which this one is based [1, 2]. A comparison between the results obtained in this thesis and the ones obtained by Telma Pereira in [1, 2] can be seen in Table 5.3.

Table 5.3: Comparison between the results obtained in [2] with FS ensemble on the left and in this thesis on the right.

	AUC	SENS	SPEC		AUC	SENS	SPEC
2 Year	0.821±0.00	0.738±0.02	0.765±0.01	2 Year	0.822±0.03	0.677±0.13	0.811±0.05
3 Year	0.859±0.00	0.778±0.01	0.781±0.01	3 Year	0.822±0.03	0.621±0.11	0.829±0.08
4 Year	0.868±0.00	0.793±0.01	0.788±0.00	4 Year	0.855±0.04	0.695±0.12	0.832±0.07

As seen in the previous table, the results obtained in this thesis are comparable with the ones obtained with the reference work that uses a complex FS ensemble to make the prediction. In terms of

AUC on the 2 and 4 year windows, the results are very close to each other, with a noticeable difference in the 3 year window in which our methodology did not perform as well as the one in [2]. In matters of sensitivity, the results obtained here are not the best in comparison to the ones in the reference work, the same can not be said about the specificity values obtained, which were higher than the values obtained in [2] in every time window, this means that the capability of predicting the patients who will not convert to AD might be better.

5.6 Summary

In this chapter, an approach to predict the conversion from Mild Cognitive Impairment to Alzheimer's Disease using state of the art machine learning methods was studied. In first place, a validation of the proposed methodology was done by evaluating four different stages (Missing Value Imputation, Oversampling, Feature Selection and Classification Methods).

For each of these stages one method was chosen based on the metrics obtained on the four time windows. In the first stage the best method found for MVI was MIWAE, after this the oversampling methodologies were evaluated and the one o provided the best samples was SMOTE. The following stage was the Feature Selection on which the best method was the one proposed for this thesis together with the proposed Neural Network architecture.

After this validation, a comparison between the results obtained with the proposed methodology were compared with the ones obtained in other related works [1, 2], in this comparison similar results were found in some metrics and even accomplished to get slightly better ones in Sensitivity values.

Between the four temporal windows, the best results obtained were from the larger one, the 5 year window, this proves that our methodology can predict with good results as far as 5 years before the conversion.

6

Conclusion

Contents

6.1 Conclusions

The goal of this work was to predict the progression of Alzheimer's Disease, using machine learning methods. While in the search for a methodology that could bring us similar results to the ones obtained by [1], there was the idea to exploit the pruning techniques [48–51] usually applied to reduce the complexity of deep learning models to feature selection. Naturally, Neural Networks have some learning capability to "reject" the less useful features by setting the associated weights close to zero. However, the idea is to improve that learning capability to reject features by eliminating them. The goal was that in the input layer of the Neural Network there were a set of weights, each one belonging to a different feature, and those weights will change to choose the more useful features. These weights would be part of the loss function of the neural network which would tend to zero while also making the predictions better. So the objective of this was to have a limited set of features that brings us to the better result possible.

Several classifiers (Naïve Bayes, Logistic Regression, Support Vector Machines, K Nearest Neighbors and Neural Networks) and different feature selection methods (Correlation, Recursive Feature Elimination and Sequential Feature Selection) were used to compare the proposed methodology. With these different methodologies, a plan was created to step by step combine them to find the overall best methodology for all the four time windows. First established a baseline methodology using MIWAE for missing value imputation, SMOTE as data balance and oversampling method, and the feature selection methodology allied with a deep neural network to perform the classification. In the first step, the missing value imputation method was changed making sure the other methods did not change, after finding out the best combination of MVI which was the MIWAE method the second step was to find the best data balance and oversampling method. After trying the four different methods the one which handled the best results was SMOTE, so this was the chosen method for balancing and sampling data. Finally, different Feature Selection methods were tested and the one who gave the best results was our methodology allied with the Neural Network.

The overall results showed a better classification of our methodology in all but one time window, these results were also compared with the ones obtained in the work made by Telma Pereira et al. in [2]. In this comparison, our methodology gave similar results to the previously mentioned work in terms of AUC and higher specificity values, but sensitivity results were lower than expected. These results have shown comparable capability of prediction to other state of the art works and capable of making predictions as early as 5 years before the conversion with accuracy values of 77%, sensitivity of 72%, specificity of 84% and ROC Area of 0.86.

6.2 Future Work

As this work is not perfect and accuracy of 100% is still far from being achieved, some improvements could be done to enhance the quality of the predictions could be investigated. Another techniques that could also be applied is the use of time series to make the prediction and also the use of clustering techniques to identify groups of patients with similar characteristics in order to improve the classification.

Another idea would be the implementation of a Feature Importance methodology side by side with our Feature Selection method on a Neural Network, which would allow us to have a classifier with all these methods embedded and to exploit the capabilities of Neural Networks into other tasks.

Finally, another proposal for the future would be the creation of a Decision Support System, which would allow clinics to actually use the methodology in a more user-friendly way. This would allow the data to be correctly inserted into the database and automatically create the time windows based on the data inserted, i.e., if the patient converted or not in that given time frame. Also with a tool like this, it would allow the medical staff to schedule appointments and exams at the right time, this would decrease the number of missing values due to missed appointments and exams which would, in turn, provide a better dataset for the classifier.

Bibliography

[1] T. Pereira, L. Lemos, S. Cardoso, D. Silva, A. Rodrigues, I. Santana, A. de Mendonça, M. Guerreiro, and S. C. Madeira, "Predicting progression of mild cognitive impairment to dementia using neuropsychological data: a supervised learning approach using time windows," *BMC medical informatics and decision making*, vol. 17, no. 1, p. 110, 2017.

[2] T. Pereira, F. Ferreira, S. Cardoso, D. Silva, A. Mendonça, M. Guerreiro, and S. C . Madeira, "Neuropsychological predictors of conversion from mild cognitive impairment to alzheimer's disease: A feature selection ensemble combining stability and predictability," *BMC Medical Informatics and Decision Making*, vol. 18, 12 2018.

[3] C. Reitz and R. Mayeux, "Alzheimer disease: epidemiology, diagnostic criteria, risk factors and biomarkers," *Biochemical pharmacology*, vol. 88, no. 4, pp. 640–651, 2014.

[4] J. Ye, M. Farnum, E. Yang, R. Verbeeck, V. Lobanov, N. Raghavan, G. Novak, A. DiBernardo, and V. A. Narayan, "Sparse learning and stability selection for predicting MCI to ADs conversion using baseline ADNI data," *BMC neurology*, vol. 12, no. 1, p. 46, 2012.

[5] O. M. Doyle, E. Westman, A. F. Marquand, P. Mecocci, B. Vellas, M. Tsolaki, I. Kłoszewska, H. Soininen, S. Lovestone, S. C. Williams *et al.*, "Predicting progression of alzheimer's disease using ordinal regression," *PloS one*, vol. 9, no. 8, p. e105542, 2014.

[6] J. A. Cruz and D. S. Wishart, "Applications of machine learning in cancer prediction and prognosis," *Cancer informatics*, vol. 2, p. 117693510600200030, 2006.

[7] S. Sarraf and G. Tofighi, "Classification of alzheimer's disease using fmri data and deep learning convolutional neural networks," *arXiv preprint arXiv:1603.08631*, 2016.

[8] H. Mohsen, E.-S. A. El-Dahshan, E.-S. M. El-Horbaty, and A.-B. M. Salem, "Classification using deep learning neural networks for brain tumors," *Future Computing and Informatics Journal*, vol. 3, no. 1, pp. 68–71, 2018.

[9] D. G. Munoz and H. Feldman, “Causes of alzheimer’s disease,” *Cmaj*, vol. 162, no. 1, pp. 65–72, 2000.

[10] E. Musiek *et al.*, “Neuropsychiatric signs and symptoms of alzheimer’s disease: New treatment paradigms,” 2017.

[11] G. B. Frisoni, N. C. Fox, C. R. Jack, P. Scheltens, and P. M. Thompson, “The clinical use of structural MRI in alzheimer disease,” *Nature Reviews Neurology*, vol. 6, no. 2, pp. 67–77, 2010.

[12] M. Pagani, F. Nobili, S. Morbelli, D. Arnaldi, A. Giuliani, J. Öberg, N. Girtler, A. Brugnolo, A. Picco, M. Bauckneht *et al.*, “Early identification of MCI converting to AD: a FDG PET study,” *European journal of nuclear medicine and molecular imaging*, vol. 44, no. 12, pp. 2042–2052, 2017.

[13] M. S. Albert, M. B. Moss, R. Tanzi, and K. Jones, “Preclinical prediction of AD using neuropsychological tests,” *Journal of the International Neuropsychological Society: JINS*, vol. 7, no. 5, p. 631, 2001.

[14] M. R. J. F. D. P. T. P. Florentino Fdez-Riverola, Mohd Saberi Mohamad, *11th International Conference on Practical Applications of Computational Biology Bioinformatics - 2017*. Springer, 2017.

[15] M. M. G. Guerreiro, *Contributo da neuropsicologia para o estudo das demências*, 1998.

[16] J. Maroco, D. Silva, A. Rodrigues, M. Guerreiro, I. Santana, and A. de Mendonça, “Data mining methods in the prediction of dementia: A real-data comparison of the accuracy, sensitivity and specificity of linear discriminant analysis, logistic regression, neural networks, support vector machines, classification trees and random forests,” *BMC research notes*, vol. 4, no. 1, p. 299, 2011.

[17] R. M. Chapman, M. Mapstone, J. W. McCrary, M. N. Gardner, A. Porsteinsson, T. C. Sandoval, M. D. Guillily, E. DeGrush, and L. A. Reilly, “Predicting conversion from mild cognitive impairment to alzheimer’s disease using neuropsychological tests and multivariate methods,” *Journal of Clinical and Experimental Neuropsychology*, vol. 33, no. 2, pp. 187–199, 2011.

[18] H. Abdi and L. J. Williams, “Principal component analysis,” *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[19] M. M. El Naqa I., Li R., *Machine Learning in Radiation Oncology*. Springer, 2015.

[20] G. Chhabra, V. Vashisht, and J. Ranjan, “A classifier ensemble machine learning approach to improve efficiency for missing value imputation,” in *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*. IEEE, 2018, pp. 23–27.

[21] R. W. Krause, M. Huisman, C. Steglich, and T. A. Sniiders, “Missing network data a comparison of different imputation methods,” in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 159–163.

[22] E.-L. Silva-Ramírez, R. Pino-Mejías, M. López-Coello, and M.-D. Cubiles-de-la Vega, “Missing value imputation on missing completely at random data using multilayer perceptrons,” *Neural Networks*, vol. 24, no. 1, pp. 121–129, 2011.

[23] C.-Y. Cheng, W.-L. Tseng, C.-F. Chang, C.-H. Chang, and S. S.-F. Gau, “A deep learning approach for missing data imputation of rating scales assessing attention-deficit hyperactivity disorder,” *Frontiers in psychiatry*, vol. 11, p. 673, 2020.

[24] S. Phung, A. Kumar, and J. Kim, “A deep learning technique for imputing missing healthcare data,” in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2019, pp. 6513–6516.

[25] I. Gad, D. Hosahalli, B. Manjunatha, and O. A. Ghoneim, “A robust deep learning model for missing value imputation in big ncdc dataset,” *Iran Journal of Computer Science*, pp. 1–18, 2020.

[26] P.-A. Mattei and J. Frellsen, “MIWAE: Deep generative modelling and imputation of incomplete data sets,” in *International Conference on Machine Learning*, 2019, pp. 4413–4423.

[27] T. Elhassan and M. Aljurf, “Classification of imbalance data using tomek link (t-link) combined with random under-sampling (rus) as a data reduction method,” 2016.

[28] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority oversampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[29] A. Liu, J. Ghosh, and C. E. Martin, “Generative oversampling for mining imbalanced datasets.” in *DMIN*, 2007, pp. 66–72.

[30] H. He, Y. Bai, E. A. Garcia, and S. Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 1322–1328.

[31] Y. Zhang, “Deep generative model for multi-class imbalanced learning,” 2018.

[32] Z. Wan, Y. Zhang, and H. He, “Variational autoencoder based synthetic data generation for imbalanced learning,” in *2017 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2017, pp. 1–7.

[33] A. L. Blum and P. Langley, “Selection of relevant features and examples in machine learning,” *Artificial intelligence*, vol. 97, no. 1-2, pp. 245–271, 1997.

[34] J. Tang, S. Alelyani, and H. Liu, “Feature selection for classification: A review,” *Data classification: Algorithms and applications*, p. 37, 2014.

[35] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[36] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[37] K. Kira and L. A. Rendell, “A practical approach to feature selection,” in *Machine Learning Proceedings 1992*. Elsevier, 1992, pp. 249–256.

[38] K. Kira, L. A. Rendell *et al.*, “The feature selection problem: Traditional methods and a new algorithm,” in *Aaai*, vol. 2, 1992, pp. 129–134.

[39] G. Casalicchio, C. Molnar, and B. Bischl, “Visualizing the feature importance for black box models,” in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 655–670.

[40] C. Molnar, “A guide for making black box models explainable,” *URL: https://christophm. github. io/interpretable-ml-book*, 2018.

[41] S. Banerjee, H. Konishi, and T. Sönmez, “Core in a simple coalition formation game,” *Social Choice and Welfare*, vol. 18, no. 1, pp. 135–153, 2001.

[42] I. Rish *et al.*, “An empirical study of the naive bayes classifier,” in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

[43] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, “Support vector machines,” *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[44] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013, vol. 398.

[45] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[46] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012.

[47] G. Daniel, *Principles of artificial neural networks*. World Scientific, 2013, vol. 7.

[48] T. I. Burgess, K. Howard, E. Steel, and E. L. Barbour, “To prune or not to prune; pruning induced decay in tropical sandalwood,” *Forest ecology and management*, vol. 430, pp. 204–218, 2018.

[49] Q. Huang, K. Zhou, S. You, and U. Neumann, "Learning to prune filters in convolutional neural networks," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 709–718.

[50] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," *arXiv preprint arXiv:1810.05270*, 2018.

[51] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 264–11 272.

[52] D. J. Hand and R. J. Till, "A simple generalisation of the area under the ROC curve for multiple class classification problems," *Machine learning*, vol. 45, no. 2, pp. 171–186, 2001.

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[54] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[55] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[56] D. J. Stekhoven and P. Bühlmann, "Missforest—non-parametric missing value imputation for mixed-type data," *Bioinformatics*, vol. 28, no. 1, pp. 112–118, 2012.

[57] P. Royston, I. R. White *et al.*, "Multiple imputation by chained equations (MICE): implementation in stata," *J Stat Softw*, vol. 45, no. 4, pp. 1–20, 2011.

[58] J. N. Wulff and L. Ejlskov, "Multiple imputation by chained equations in praxis: Guidelines and review." *Electronic Journal of Business Research Methods*, vol. 15, no. 1, 2017.

[59] Y. Burda, R. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," *arXiv preprint arXiv:1509.00519*, 2015.

7

Appendix

Table 7.1: Statistics 2 Year Window

Features	Missing Values	Mean	Mode
Gender	0.000	1.611	2.000
Age	0.002	68.774	70.000
School	0.008	9.930	16.000
Start_Age	0.008	66.549	73.000
timeBeginComplaints_1assessment	0.018	2.207	1.000
Depression_Clinical_Interview	0.034	0.273	0.000
Memory_Complains_Clinial_Interview	0.032	0.965	1.000
As_cut	0.140	15.262	16.000
As_time	0.144	41.933	40.000
As_tot	0.146	4.229	3.600
DS_Forward	0.024	5.045	5.000
DS_back	0.026	3.678	4.000
DS_Total	0.026	8.723	9.000
PA_Easy_Total	0.040	13.490	18.000
PA_Dif_Total	0.040	3.482	0.000
PA_Tot	0.036	10.212	8.000
LM_a_Total	0.020	8.116	7.000
Itens_recuperados_MLA	0.120	3.188	2.000
LM_a_Cued	0.120	11.132	8.000
HELP_MLA	0.120	2.063	2.000
LM_a_Interf	0.076	7.060	0.000
Itens_recup_MLAinterf	0.405	2.406	2.000
LM_a_Interf_Cued	0.403	8.936	13.000
Winning_Lost_MLinterf	0.168	2.482	3.000
Help_Effects_MLI	0.184	0.450	0.000
MVI_Free	0.130	3.888	0.000
MVI_Cued	0.130	3.917	4.000
MVI_Rec	0.130	1.222	1.000
MVI_Tot	0.128	9.048	9.000
Informacao_Total	0.206	17.633	20.000
VisualM_B	0.519	2.378	2.000
Or_Total	0.042	13.810	15.000
Orient_P	0.042	4.898	5.000
Orient_S	0.042	2.956	3.000
Orient_T	0.042	5.933	7.000
Fluency_Sem	0.020	15.619	13.000
Fluency_Phon_M	0.495	9.150	9.000
M_Initiative	0.050	2.691	3.000
Gm_Initiative	0.106	1.804	2.000
Token_T	0.363	15.567	17.000
Repetition	0.705	10.919	11.000
Reading	0.822	1.933	2.000
Cube	0.210	2.470	3.000
Clock	0.052	2.611	3.000
Calc	0.132	12.278	14.000
MPR_Total	0.090	8.586	11.000
Proverb_Total	0.034	7.056	9.000
MMSE_Total	0.671	26.600	29.000
MMS_Orientation_total	0.754	9.211	10.000
MMS_OrientationTemporal_Total	0.756	4.508	5.000

MMS_orienT_Espacial_Total	0.756	4.648	5.000
MMS_Retencao	0.758	2.983	3.000
MMS_Calculo_Total	0.758	4.603	5.000
MMS_Evocacao	0.758	1.636	2.000
MMS_Linguagem_Total	0.766	7.846	8.000
MMS_Ling_Nomeacao	0.766	1.991	2.000
MMS_Ling_Repeticao	0.766	0.983	1.000
MMS_Ling_Compreensao	0.766	2.752	3.000
MMS_Ling_Leitura	0.766	0.991	1.000
MMS_Ling_Escrita	0.766	0.974	1.000
MMS_Desenho	0.762	0.807	1.000
TrailMakingTest	0.393	1.000	1.000
TMT_A_temp	0.435	70.329	50.000
TMT_B_temp	0.443	172.480	140.000
CVLT	0.315	1.303	1.000
a1	0.315	4.743	4.000
a5_Total	0.315	9.099	11.000
a1_a5_Total	0.315	37.341	40.000
a_cr_int	0.509	6.650	6.000
a_lg_int	0.549	6.814	8.000
Depressao_GDS	0.325	0.447	0.000
GDS	0.325	4.583	2.000
QSM_Total	0.627	9.807	8.000
Informante	0.507	1.757	1.000
BlessedAVD	0.481	1.248	1.500
BlessedHAB	0.483	0.015	0.000
BlessedPERS	0.481	2.112	1.000
BlessedTOT	0.481	3.383	3.000
Fi_LM_a_m100	0.090	-18.274	-100.000
As_tot_Z	0.148	0.151	-0.592
DS_back_Z	0.028	0.116	0.140
PA_Tot_Z	0.038	-1.189	-2.441
MVI_Tot_Z	0.130	-1.104	-1.391
Orient_T_Z	0.066	-24.768	-27.310
Fluency_Sem_Z	0.022	-0.159	0.376
M_Initiative_Z	0.052	-0.342	0.276
Gm_Initiative_Z	0.128	0.212	0.556
Token_T_Z	0.485	-0.167	0.554
Cube_Z	0.212	0.550	1.373
Clock_Z	0.054	0.310	1.115
Calc_Z	0.134	-0.245	0.532
MPR_Total_Z	0.092	-0.081	1.238
Proverb_Total_Z	0.036	0.824	1.820
TMT_A_temp_Z	0.465	0.926	0.005
TMT_B_temp_Z	0.473	1.233	0.033
a5_Total_Z	0.317	-2.320	-1.768
LM_a_Total_Z	0.044	-1.298	-0.975
LM_a_Interf_Z	0.098	-1.017	-2.651

Table 7.2: Statistics 3 Year Window

Feature	Missing Values	Mean	Mode
Gender	0.000	1.611	2.000
Age	0.002	68.850	70.000
School	0.004	9.766	16.000
Start_Age	0.004	66.603	69.000
timeBeginComplaints_1assessment	0.011	2.234	1.000
Depression_Clinical_Interview	0.034	0.285	0.000
Memory_Complains_Clinial_Interview	0.032	0.958	1.000
As_cut	0.128	15.201	16.000
As_time	0.130	42.020	40.000
As_tot	0.132	4.206	3.600
DS_Forward	0.024	4.991	5.000
DS_back	0.026	3.643	4.000
DS_Total	0.026	8.636	9.000
PA_Easy_Total	0.034	13.302	15.000
PA_Dif_Total	0.034	3.243	0.000
PA_Tot	0.032	9.912	6.000
LM_a_Total	0.019	7.887	7.000
Itens_recuperados_MLA	0.115	3.205	2.000
LM_a_Cued	0.115	10.903	8.000
HELP_MLA	0.115	2.077	2.000
LM_a_Interf	0.073	6.657	0.000
Itens_recup_MLAinterf	0.389	2.367	2.000
LM_a_Interf_Cued	0.387	8.564	10.000
Winning_Lost_MLinterf	0.158	2.538	3.000
Help_Effects_MLI	0.169	0.447	0.000
MVI_Free	0.120	3.692	0.000
MVI_Cued	0.120	3.937	4.000
MVI_Rec	0.120	1.245	1.000
MVI_Tot	0.118	8.896	9.000
Informacao_Total	0.190	17.422	20.000
VisualM_B	0.536	2.230	0.000
Or_Total	0.036	13.650	15.000
Orient_P	0.036	4.885	5.000
Orient_S	0.038	2.956	3.000
Orient_T	0.038	5.809	7.000
Fluency_Sem	0.021	15.384	14.000
Fluency_Phon_M	0.509	8.913	5.000
M_Initiative	0.047	2.679	3.000
Gm_Initiative	0.109	1.787	2.000
Token_T	0.348	15.423	17.000
Repetition	0.705	10.935	11.000
Reading	0.823	1.928	2.000
Cube	0.218	2.445	3.000
Clock	0.058	2.592	3.000
Calc	0.132	12.256	14.000
MPR_Total	0.090	8.411	11.000
Proverb_Total	0.034	6.918	9.000
MMSE_Total	0.667	26.128	29.000
MMS_Orientation_total	0.763	8.964	10.000
MMS_OrientationTemporal_Total	0.763	4.315	5.000

MMS_orienT_Espacial_Total	0.763	4.541	5.000
MMS_Retencao	0.763	2.982	3.000
MMS_Calculo_Total	0.763	4.505	5.000
MMS_Evocacao	0.763	1.486	1.000
MMS_Linguagem_Total	0.769	7.769	8.000
MMS_Ling_Nomeacao	0.769	1.981	2.000
MMS_Ling_Repeticao	0.769	1.000	1.000
MMS_Ling_Compreensao	0.769	2.704	3.000
MMS_Ling_Leitura	0.769	0.991	1.000
MMS_Ling_Escrita	0.771	0.972	1.000
MMS_Desenho	0.765	0.782	1.000
TrailMakingTest	0.410	1.000	1.000
TMT_A_temp	0.451	70.588	50.000
TMT_B_temp	0.464	172.964	140.000
CVLT	0.325	1.326	1.000
a1	0.325	4.680	4.000
a5_Total	0.327	8.959	11.000
a1_a5_Total	0.327	36.686	36.000
a_cr_int	0.530	6.509	8.000
a_lg_int	0.568	6.584	8.000
Depressao_GDS	0.333	0.458	0.000
GDS	0.333	4.651	2.000
QSM_Total	0.620	9.685	11.000
Informante	0.483	1.785	1.000
BlessedAVD	0.453	1.326	1.500
BlessedHAB	0.455	0.020	0.000
BlessedPERS	0.453	2.102	2.000
BlessedTOT	0.453	3.459	3.000
Fi_LM_a_m100	0.090	-22.087	-100.000
As_tot_Z	0.135	0.114	0.429
DS_back_Z	0.028	0.088	0.140
PA_Tot_Z	0.034	-1.266	-2.441
MVI_Tot_Z	0.120	-1.172	0.058
Orient_T_Z	0.062	-24.809	-27.310
Fluency_Sem_Z	0.024	-0.229	-1.504
M_Initiative_Z	0.049	-0.352	0.276
Gm_Initiative_Z	0.132	0.175	0.556
Token_T_Z	0.474	-0.232	0.554
Cube_Z	0.220	0.492	1.373
Clock_Z	0.060	0.289	1.115
Calc_Z	0.135	-0.230	0.532
MPR_Total_Z	0.092	-0.180	1.238
Proverb_Total_Z	0.036	0.737	1.820
TMT_A_temp_Z	0.481	0.916	0.005
TMT_B_temp_Z	0.496	1.226	0.033
a5_Total_Z	0.329	-2.338	-2.873
LM_a_Total_Z	0.043	-1.362	-0.702
LM_a_Interf_Z	0.094	-1.111	-2.651

Table 7.3: Statistics 4 Year Window

Feature	Missing Values	Mean	Mode
Gender	0.000	1.626	2.000
Age	0.002	68.395	70.000
School	0.007	9.827	16.000
Start_Age	0.002	66.160	69.000
timeBeginComplaints_1assessment	0.014	2.228	1.000
Depression_Clinical_Interview	0.030	0.270	0.000
Memory_Complains_Clinial_Interview	0.028	0.957	1.000
As_cut	0.125	15.231	16.000
As_time	0.125	42.478	40.000
As_tot	0.125	4.134	3.600
DS_Forward	0.021	4.986	5.000
DS_back	0.023	3.651	4.000
DS_Total	0.021	8.642	9.000
PA_Easy_Total	0.030	13.252	14.000
PA_Dif_Total	0.030	3.194	0.000
PA_Tot	0.023	9.833	6.000
LM_a_Total	0.016	7.781	3.000
Itens_recuperados_MLA	0.107	3.192	2.000
LM_a_Cued	0.107	10.753	8.000
HELP_MLA	0.107	2.094	2.000
LM_a_Interf	0.070	6.584	0.000
Itens_recup_MLAinterf	0.394	2.322	2.000
LM_a_Interf_Cued	0.392	8.424	0.000
Winning_Lost_MLinterf	0.160	2.552	3.000
Help_Effects_MLI	0.172	0.448	0.000
MVI_Free	0.114	3.644	0.000
MVI_Cued	0.114	3.921	4.000
MVI_Rec	0.114	1.243	1.000
MVI_Tot	0.111	8.815	9.000
Informacao_Total	0.181	17.351	20.000
VisualM_B	0.543	2.188	0.000
Or_Total	0.039	13.621	15.000
Orient_P	0.039	4.879	5.000
Orient_S	0.042	2.959	3.000
Orient_T	0.042	5.782	7.000
Fluency_Sem	0.019	15.248	14.000
Fluency_Phon_M	0.527	8.995	5.000
M_Initiative	0.044	2.704	3.000
Gm_Initiative	0.097	1.784	2.000
Token_T	0.329	15.443	17.000
Repetition	0.698	10.954	11.000
Reading	0.833	1.931	2.000
Cube	0.211	2.429	3.000
Clock	0.053	2.574	3.000
Calc	0.125	12.249	14.000
MPR_Total	0.088	8.412	11.000
Proverb_Total	0.032	6.882	9.000
MMSE_Total	0.680	25.884	27.000
MMS_Orientation_total	0.773	8.827	10.000
MMS_OrientationTemporal_Total	0.773	4.184	5.000

MMS_orienT_Espacial_Total	0.773	4.490	5.000
MMS_Retencao	0.773	2.980	3.000
MMS_Calculo_Total	0.773	4.449	5.000
MMS_Evocacao	0.773	1.449	1.000
MMS_Linguagem_Total	0.780	7.716	8.000
MMS_Ling_Nomeacao	0.780	1.979	2.000
MMS_Ling_Repeticao	0.780	1.000	1.000
MMS_Ling_Compreensao	0.780	2.674	3.000
MMS_Ling_Leitura	0.780	0.989	1.000
MMS_Ling_Escrita	0.782	0.968	1.000
MMS_Desenho	0.775	0.763	1.000
TrailMakingTest	0.406	1.000	1.000
TMT_A_temp	0.450	71.430	50.000
TMT_B_temp	0.457	174.333	180.000
CVLT	0.332	1.313	1.000
a1	0.334	4.742	4.000
a5_Total	0.336	9.003	8.000
a1_a5_Total	0.334	36.836	39.000
a_cr_int	0.529	6.655	8.000
a_lg_int	0.566	6.695	8.000
Depressao_GDS	0.334	0.446	0.000
GDS	0.334	4.575	4.000
QSM_Total	0.622	9.589	8.000
Informante	0.485	1.806	1.000
BlessedAVD	0.457	1.391	1.500
BlessedHAB	0.459	0.021	0.000
BlessedPERS	0.457	2.167	2.000
BlessedTOT	0.455	3.579	3.000
Fi_LM_a_m100	0.090	-22.090	-100.000
As_tot_Z	0.130	0.011	-0.286
DS_back_Z	0.028	0.078	0.140
PA_Tot_Z	0.028	-1.311	-2.730
MVI_Tot_Z	0.116	-1.230	-1.391
Orient_T_Z	0.072	-25.233	-27.310
Fluency_Sem_Z	0.023	-0.302	-1.504
M_Initiative_Z	0.049	-0.324	0.276
Gm_Initiative_Z	0.128	0.165	0.556
Token_T_Z	0.471	-0.228	0.554
Cube_Z	0.216	0.452	1.373
Clock_Z	0.058	0.234	1.115
Calc_Z	0.130	-0.231	0.532
MPR_Total_Z	0.093	-0.191	1.238
Proverb_Total_Z	0.037	0.705	1.820
TMT_A_temp_Z	0.483	1.000	0.005
TMT_B_temp_Z	0.492	1.269	1.904
a5_Total_Z	0.341	-2.322	-2.873
LM_a_Total_Z	0.046	-1.405	-0.975
LM_a_Interf_Z	0.097	-1.145	-2.651

Table 7.4: Statistics 5 Year Window

Feature	Missing Values	Mean	Mode
Gender	0.000	1.624	2.000
Age	0.002	68.954	70.000
School	0.007	9.754	16.000
Start_Age	0.002	66.672	69.000
timeBeginComplaints_1assessment	0.012	2.267	1.000
Depression_Clinical_Interview	0.034	0.263	0.000
Memory_Complains_Clinial_Interview	0.032	0.955	1.000
As_cut	0.120	15.194	16.000
As_time	0.120	42.655	40.000
As_tot	0.120	4.102	5.000
DS_Forward	0.022	4.975	5.000
DS_back	0.024	3.663	4.000
DS_Total	0.022	8.643	9.000
PA_Easy_Total	0.034	13.128	14.000
PA_Dif_Total	0.034	3.028	0.000
PA_Tot	0.027	9.584	6.000
LM_a_Total	0.017	7.464	3.000
Itens_recuperados_MLA	0.107	3.134	2.000
LM_a_Cued	0.107	10.404	8.000
HELP_MLA	0.107	2.055	2.000
LM_a_Interf	0.076	6.227	0.000
Itens_recup_MLAinterf	0.398	2.340	0.000
LM_a_Interf_Cued	0.395	8.097	0.000
Winning_Lost_MLinterf	0.173	2.563	3.000
Help_Effects_MLI	0.188	0.444	0.000
MVI_Free	0.110	3.460	0.000
MVI_Cued	0.110	3.951	4.000
MVI_Rec	0.110	1.274	1.000
MVI_Tot	0.107	8.691	9.000
Informacao_Total	0.185	17.350	20.000
VisualM_B	0.544	2.128	0.000
Or_Total	0.039	13.475	15.000
Orient_P	0.039	4.868	5.000
Orient_S	0.041	2.954	3.000
Orient_T	0.041	5.659	7.000
Fluency_Sem	0.017	15.010	14.000
Fluency_Phon_M	0.546	8.763	8.000
M_Initiative	0.037	2.671	3.000
Gm_Initiative	0.095	1.765	2.000
Token_T	0.341	15.433	17.000
Repetition	0.705	10.975	11.000
Reading	0.834	1.941	2.000
Cube	0.212	2.384	3.000
Clock	0.054	2.577	3.000
Calc	0.132	12.346	14.000
MPR_Total	0.090	8.319	10.000
Proverb_Total	0.029	6.812	9.000
MMSE_Total	0.702	25.713	29.000
MMS_Orientation_total	0.800	8.646	10.000
MMS_OrientationTemporal_Total	0.800	4.024	5.000

MMS_orienT_Espacial_Total	0.800	4.488	5.000
MMS_Retencao	0.800	3.000	3.000
MMS_Calculo_Total	0.800	4.451	5.000
MMS_Evocacao	0.800	1.402	0.000
MMS_Linguagem_Total	0.810	7.679	8.000
MMS_Ling_Nomeacao	0.810	1.974	2.000
MMS_Ling_Repeticao	0.810	1.013	1.000
MMS_Ling_Compreensao	0.810	2.667	3.000
MMS_Ling_Leitura	0.810	0.987	1.000
MMS_Ling_Escrita	0.810	0.949	1.000
MMS_Desenho	0.802	0.778	1.000
TrailMakingTest	0.410	1.000	1.000
TMT_A_temp	0.451	73.049	60.000
TMT_B_temp	0.463	179.423	180.000
CVLT	0.356	1.318	1.000
a1	0.359	4.635	4.000
a5_Total	0.361	8.779	8.000
a1_a5_Total	0.359	36.065	30.000
a_cr_int	0.554	6.492	8.000
a_lg_int	0.580	6.430	8.000
Depressao_GDS	0.354	0.460	0.000
GDS	0.354	4.694	1.000
QSM_Total	0.641	9.558	11.000
Informante	0.473	1.792	1.000
BlessedAVD	0.441	1.421	1.500
BlessedHAB	0.441	0.017	0.000
BlessedPERS	0.441	2.155	2.000
BlessedTOT	0.439	3.589	3.000
Fi_LM_a_m100	0.098	-26.886	-100.000
As_tot_Z	0.124	0.017	-0.286
DS_back_Z	0.029	0.107	0.140
PA_Tot_Z	0.032	-1.374	-2.296
MVI_Tot_Z	0.112	-1.288	-1.391
Orient_T_Z	0.066	-25.337	-27.310
Fluency_Sem_Z	0.022	-0.365	-0.752
M_Initiative_Z	0.041	-0.388	0.276
Gm_Initiative_Z	0.120	0.133	0.556
Token_T_Z	0.473	-0.212	0.554
Cube_Z	0.217	0.403	1.373
Clock_Z	0.059	0.256	1.115
Calc_Z	0.137	-0.188	0.532
MPR_Total_Z	0.095	-0.221	0.048
Proverb_Total_Z	0.034	0.682	1.820
TMT_A_temp_Z	0.480	1.037	1.075
TMT_B_temp_Z	0.495	1.364	1.904
a5_Total_Z	0.366	-2.446	-2.265
LM_a_Total_Z	0.041	-1.484	-2.615
LM_a_Interf_Z	0.098	-1.220	-2.651