TÉCNICO LISBOA

Time series data imputation - Comparison of Dynamic Time Warping with Needleman-Wunsch algorithm

Guilherme Reis de Moura

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor(s): Prof. Alexandra Sofia Martins de Carvalho
Prof. Susana de Almeida Mendes Vinga Martins

Examination Committee

Chairperson: Prof. Teresa Maria Sá Ferreira Vazão Vasques
Supervisor: Prof. Alexandra Sofia Martins de Carvalho
Member of the Committee: Sara Alexandra Cordeiro Madeira

November 2019

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.

I want to thank my supervisors, Prof. Susana Vinga and Prof. Alexandra Carvalho, for all the support, patience, availability and mainly for providing an incredible experience during this project.
I would also like to thank Catarina Santos for all her help during this work, as well as my family for all the support they have given me.
Lastly, I also want to thank to IST for giving me the chance to learn amazing contents and experience great things.

Resumo

Os algoritmos de aprendizagem automática estão a ser desenvolvidos e aplicados a dados que ajudam a população nas suas necessidades diárias. Estes algoritmos trazem muitos benefícios a diferentes áreas e possibilitam análises preditivas, formação de grupos de classes (clustering) e classificação dos dados. Podem ser usados, por exemplo, em bases de dados médicas para facilitar o diagnóstico e o tratamento de um paciente. Em relação à eficiência destes algoritmos, existe uma preocupação acrescida no que se refere a valores em falta nos dados. Os algoritmos que existem não estão preparados para lidar com a falta destes dados, pelo que existe a necessidade de se abordar este tema. Assim, neste trabalho, pretende-se comparar dois algoritmos de imputação de dados que poderão ser usados para completar os dados em falta. Os algoritmos usados são o Dynamic Time Warping e Needleman-Wunsch. Verificou-se que estes dois algoritmos são capazes de preencher os dados em falta, onde o Dynamic Time Warping se revelou mais preciso, enquanto que o Needleman-Wunsch se revelou mais rápido. Com este trabalho também se verificou que estes dois algoritmos podem ser testados em mais profundidade devido ao seu potencial para preencher os dados em falta. Também são feitas sugestões para melhorar alguns pontos menos positivos em relação à performance dos mesmos.

Palavras-chave: Aprendizagem automática, data mining, valores em falta, Dynamic Time Warping, Needleman-Wunsch.

Abstract

Machine learning algorithms are now being designed and applied to data to help humans in their everyday needs. These algorithms can bring major benefits to many areas and are capable of conducting predictions, clustering and classification on data. They could be used, for example, on medical databases to help in treatments and diagnosis of patients. One major concern that threatens the efficiency of these algorithms are missing values. Many algorithms which are in place today are not prepared to handle these missing values, which means they have to be handled in other ways. In this paper it is aimed to compare two imputation algorithms that could be used in filling these missing values. Both methods use sequence alignment to find matches with which the missing values could then be imputed. One of the algorithms uses dynamic time warping while the other uses Needleman-Wunsch. Both of these algorithms were suitable when it came to data imputation. Imputation done using dynamic time warping was accurate, although it lacked in speed, while the Needleman-Wunsch imputation was faster, but not quite as accurate as the dynamic time warping imputation. The results show that both of these algorithms should be further tested due to their potential in the imputation of values, as well as some suggestions to strengthen the weaknesses of both of these algorithms.

Keywords: Machine learning, data mining, missing values, Dynamic Time Warping, Needleman-Wunsch.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Nowadays, there is an increased necessity in handling large volumes of data, particularly when it comes to medical data. This data can be used to extrapolate useful information and, as a consequence, aiding medical professionals [1]. Many algorithms were designed in order to extract useful patterns from medical databases, which would be impossible for a medical team to analyze due to the immensity of the data. This process is called data mining [2, 3] and one of the fields that is used to perform these operations is machine learning [4]. This field is dedicated to the classification, prediction and clustering of data, among others [5]. All of these can be applied to medical databases in various situations requiring data handling, depending on the circumstances. Besides medical databases many other fields benefit from the use of these techniques in many ways [6].

Regarding the usefulness of these algorithms in the medical fields there is a number of ways on which they excel:

- **Diagnosis** - Many classification algorithms have been developed to provide with an answer with haste [7]. By applying these, many lives could be saved, not only because of the speed on which a diagnosis could be achieved, but also on the accuracy, preventing a wrong diagnosis. Of course these can be merely suggestive, being regarded as a second opinion to the medical professionals. Another important tool for diagnosis is image recognition which can lead to a quick diagnostic by analyzing image-related diagnostic tools.On the field of prediction, many options are being explored, one of which is the capability to predict the appearance of a certain disease or an injury, based on data from the patient. Thus, diagnosis is possibly the one with the most direct impact regarding the health of a patient, since these will define the treatment applied.
- **Treatment** - There have been many changes when it comes to treating patients as well, such as making use of patient data, so that professionals can choose a specific treatment for an individual with specific and similar features [8]. Research and drug manufacturing is also being affected with these algorithms, since they are starting to be produced not only based on past results, but also base on predictive data.

As time passes, a lot of applications using these machine learning techniques have encountered some issues regarding data quality. Given the fact that the amount of data collected is enormous (and regarding the data itself, the amount of variables/features captured is also very big), the probability that the collected data is one hundred percent accurate is very low [9]. This effect is even more critical when it comes to medical data, given the sensibility the algorithms have to have with these type of data. Mistakes are bound to happen, whether it is by the hand of man, which is considered to be very common, whether it is done by machine errors. These mistakes can range from non-existing values to simply absurd values that were found because someone misplaced a simple comma. The problem that is faced nowadays is that with the amount of data in existence it is impractical to identify and correct all mistakes by hand. Thus, methods to identify these situations and correctly handle them are needed, or else, whenever these kinds of situations occur, the data would need to be disposed of, losing valuable information.

Due to the fact that many of the machine learning algorithms require their data intact, a lot of effort is being put into creating accurate imputation algorithms, capable of filling out data with values that could represent the missing value, while still maintaining data coherence [10, 11].

1.2 Objectives

The purpose of this work is to find a suitable way of dealing with missing values in time series data, particularly in medical databases. To do so, two methods of imputation will be compared which will focus on a single feature, making both of these algorithms usable with multivariate and univariate time series, by using an approach based on the similarity measurement of time series using the Dynamic Time Warping (DTW) algorithm [12], while drawing a comparative analysis with the Needleman-Wunsch (NW) algorithm [13, 14] due to the similarity in behaviour of both of these algorithms.

1.3 Thesis Outline

On the background chapter, the state of the art regarding this theme is discussed, starting with general notions on the theme of machine learning algorithms and tasks as well as a more profound analysis of the algorithms used in this work. On the proposed method chapter, it is explained the way in which the two algorithms used to perform the data imputation (DTW and NW) will be used. On the results chapter, an overview of the performance of the two algorithms implemented is shown. Finally, conclusions on this work will be discussed and suggestions on new approaches to develop will be remarked.

Chapter 2

Background

Nowadays, databases are used to store all kinds of information, such as transactions, financial or even simple client information. Amongst these there are medical databases, that have certain importance due to the content displayed by them. The amount of information regarding medical conditions is enormous, given the number of features that can be extracted from patients. In order to ease data comprehension, many datasets are created for a specific purpose, having a reduced amount of patient's data that relates only to the desired problem at hand, such as a specific disease. To solve such problems, there is a need for ways to reach viable conclusions using the available data. For this purpose, the subject of data mining emerged.

Data mining can lead us to anticipate the appearance of conditions, or the likelihood of the use of a certain drug improving the patients' state. It can obviously be applied in other sorts of problems such as: Marketing, Investment, Fraud Detection, Manufacturing, among others. However, with the amount of data at our disposal, data patterns can prove to be quite difficult to be extrapolated from the data. For this task, methods have been developed in a field called machine learning, which mainly solves problems related to: classification, regression and clustering.

2.1 Preprocessing

Before applying machine learning techniques, the data may need to go through a series of operations to prepare it for the analysis needed to infer the data patterns as can be seen on Figure 2.1. These operations can be: data cleaning, data integration, data selection and data transformation. Of all of these operations, the most important for this work would be data cleaning.

Data cleaning is a process that deals with missing values, outliers and errors within the data [15]. These problems can happen in any database, and tend to be associated with human error. Missing values occur when there is no input for a given field, and in medical databases it is a recurring problem. As for errors and outliers, they can also happen due to human typos, which are bound to happen in any kind of database, making the data inaccurate, or even machine related errors. These errors, when found, can receive the same kind of treatment the missing values do, or even be changed by hand,

although this is not a practical solution.

On the subject of missing values, there can be many ways to deal with them [16, 17]. The easiest way would be to completely ignore the missing data. However, by doing so, the information regarding this value would be completely lost. The other way of dealing with this problem, is by attempting to fill in the missing values. To do so, many methods can be considered, mainly statistical and machine learning methods [18], which use a predictive model based on the available data to infer the missing values. Some of these methods include: k-nearest neighbours [19, 20] and Bayesian approaches [21]. Other simple approaches can be used to tackle this issue, as for example using the mean of the non-missing values to fill in the missing values. When dealing with errors and outliers, some of the common techniques are: binning [22], regression [23, 24] and clustering [25]. This last one is specially used for the identification of outliers, since clustering attempts to identify groups of values that could belong to the same cluster, and by leaving values out of these clusters these could be considered outliers. Also, by having previous information regarding the received data, also known as metadata, as for example, maximum and minimum values of certain fields, some of these outliers and errors can be easily identified [26].

Data integration is required when using multiple databases in order to solve any discrepancies that may arise from the combination of various databases into a single one [27]. Problems may include: redundancies, object matching and data value conflicts. Of these problems, perhaps only redundancies would be of any relevance to our problem and it could be solved by correlation analysis of all the attributes.

Data transformation consists in altering the data to accepted values so that it can be properly dealt with [15]. These changes can be implemented by resorting to methods such as: normalization, aggregation, generalization and attribute construction. Normalization is used when certain values should be scaled to fit between certain pre-determined values, and it can be performed using various methods, such as min-max normalization, mean normalization and others. Both aggregation and generalization are used in order to move up in the concept hierarchy, although aggregation does so on numeric attributes, while generalization acts on nominal attributes. Attribute construction, as the name suggests, creates new attributes from already existing attributes. All of these modifications are made so that the process of data mining can be done in a smoothly manner.

Data reduction has the purpose of condensing the available data, while still getting the same results when it comes to discovering data patterns. This can be attained by reducing: the number of attributes, attribute values and tuples. Operations that can be performed in order to achieve these results include: data cube aggregation [28], attribute subset selection [29], dimensionality reduction [30] and numerosity reduction [31].

2.2 Learning tasks

After the data goes through the previous steps successfully, several machine learning techniques can be applied depending on the type of conclusions that need to be reached. Machine learning relies

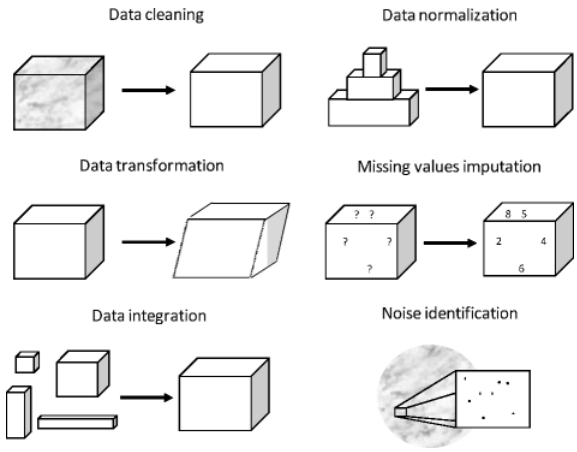

Figure 2.1: The preprocessing steps [32].

on models that can “learn” with the given data and seek out patterns, so that it can predict certain behaviours or give a classification when new data is presented. According to the way data is given there can be two types of learning performed by systems. The first, supervised learning, consists of analyzing given inputs with the corresponding outputs as a learning experience. Having both of these components will enable the system to figure out a way of classifying future data based on the observed “examples” given previously. Then there is unsupervised learning, which, instead of giving inputs and the corresponding outputs, will only give the inputs, and from these it will need to extract patterns.

2.2.1 Classification and Prediction

As previously discussed, machine learning can be used for a variety of tasks, being one of those classification and prediction. Classification consists in the assignment of a pre-determined class to the input data, taking into consideration a particular model. This is usually done with the use of supervised learning, by giving existing inputs and their corresponding labels, called the training set, in order to build a decision model to infer future cases. While classification deals with the attribution of classes/labels, prediction is used for analyzing continuous functions and determining their new values. After a classifier is ready, the respective classification accuracy can be measured by using a test set independent of the training set, because the data would tend to overfit if data related to the training set were to be used. Another method of approaching this issue is cross-validation, on which the training data is divided into two different segments, preferably in a way that the whole data is represented in both segments (stratified cross-validation), being the first segment used to train the data to find a predictive model, and the second used on the process of validation of the model. If the accuracy is considered acceptable the classifier can be used in labeling new data.

Regarding classification, several ways of addressing this issue were created throughout the years. One of those methods would be decision trees [33, 34]. In a simplified way decision trees consist of 3 types of structures: internal-nodes, branches and leaves. Internal-nodes represent the tests made to the data. Depending on the results of those tests, a branch would be chosen which could lead to more tests or to a leaf, which represents the class label and the end of the decision tree. To create these

trees, algorithms such as ID3 [35], C4.5 [36] and CART [37] were created.

Bayesian classifiers are a way of classifying data based on probabilities. As the name suggests it is based on the Bayes theorem, thus relying on posterior and prior probabilities. A variant of this method, showing similar results in terms of classifying, though with less computational power involved, would be the naive Bayesian classifier [38], which has a key aspect to it of considering all attribute values to be independent of one another, making the task of classifying data much simpler.

Neural Networks are another classification method [39]. They are constituted by input and output units, and also hidden units. These units are connected to each other and each of these connections has a weight associated to it. The idea underlying this method is that an extensive training period is performed to, given the input, calculate the weights in order to classify, in the best possible way, the future inputs.

Besides these methods, support vector machines are also used in classification problems [40]. This method uses a nonlinear mapping to bring the data to a higher dimension. In this new dimension it will attempt to find a way to separate the data using a linear hyperplane, while trying to find the optimal solution to do so. To find this hyperplane there is the need to use certain data, called support vectors, that are critical to finding the best solution to separate the data, which will establish the best possible margins to work with.

These are some of the classification methods, although many more exist [41, 42]. The examples given could be considered to be the most widely used ones, being that some of these can also be used as prediction methods.

Regarding prediction, one method stands out for being used in various occasions. Regression techniques are widely used whenever numeric prediction is being attempted [43]. These methods try to estimate a relationship between a response variable and one or more predictor variables. When there is only one predictor variable, simple linear regression is used. An example of a simple regression model would be straight-line regression, which tries to fit the data to a straight line, as the name suggests, using the following equation: $y = b + wx$, where y represents the response variable, and x the predictor variable. Both b and w can be regarded as weights and can be estimated using the method of Least Squares. There can also be more than one predictor variables, implying the use of multiple linear regression, and there is also Nonlinear regression [44]. There are many types of regression used for nonlinear functions that include: Polynomial Regression, Logistic Regression and Poisson Regression.

2.2.2 Clustering

Another machine learning field that is very used is clustering [45]. Clustering is used to group data into clusters, being that the objects within a cluster need to be similar among themselves, and on the other hand, need to be sufficiently different from other clusters' objects. These sort of tasks are used when there are no pre-defined classes, contrary to what happens with classification, thus having the necessity for the clustering algorithms to create their own classes, which are called clusters. Due to these circumstances clustering operations are considered to be an unsupervised learning method because

class labels are not provided due to the fact that they do not exit, having to be created as the algorithm is applied. The algorithms, in order to separate the clusters, tend to use distance measures between the data, which is an intuitive way of performing clustering on data.

Many methods for clustering have been developed, and they can be sorted into different categories. One of those categories is partitioning methods [46]. These methods use a certain number of partitions, which will form the future clusters, to sort the objects among them. The objects would then be relocated through the partitions to find the best fit, usually found through the use of distances as previously said. One of these algorithms is K-means [47]. The K-means algorithm behaves in the way that has been described and uses as similarity measure the mean value of the objects within the clusters. The name of this algorithm comes from the similarity measure used, and the fact that there will be used a K number of clusters.

Hierarchical methods are also used frequently [48]. These methods consist of the grouping of clusters in a tree form and can be developed in two ways: agglomerative or divisive. In agglomerative clustering, all of the objects are clusters in the beginning. As the algorithm progresses, the objects will merge until certain conditions are met, providing the final form of the clusters. The divisive methods are similar, but instead of having all the objects become clusters individually, all of the objects are a single cluster. As the method iterates, the objects will begin to separate creating new clusters. Some examples of these methods are minimal spanning trees, and simple linkage algorithms, and also BIRCH [49], and ROCK [50].

Density-based methods are another type of methods used for clustering [51]. They consist in a growing cluster, that will continue to expand until a certain threshold of density is reached. The density is in relation with the amount of objects contained in the cluster, and while the density stays above the threshold indicated previously, the assumption that the data contained in that cluster belongs to it will remain true. Two popular density-based clustering algorithms are DBSCAN [52] and DENCLUE [53]. The first one uses a density-based connectivity analysis to expand the clusters, and DENCLUE makes use of density distribution functions in order to perform clustering on the objects.

Grid-based methods can also be used to cluster objects. These methods make a finite number of cells, creating a grid structure, upon which the clustering operations are performed. As a result of this sort of method, the whole process of clustering will have a speed dependent only on the number of cells of the grid, thus having the ability to be faster than most algorithms. STING [54] is one of these methods, which acts based on statistical information of each cell of the grid. Another method is CLIQUE [55], although this can be considered a mix of grid-based methods, with density-based methods.

Last but not least, model-based methods are another option when it comes to clustering. These methods resort to mathematical models in order to fit the data into clusters. For this purpose a variety of models can be used, as for example using the logic of K-means, but using mathematical models to infer the distribution probability, in a process called Expectation-Maximization [56]. Other methods make use of neural networks, as for example Self-organizing feature maps [57]. In order to choose which of all of these methods to use, there is a need to know the data on which clustering is to be performed to choose the one that would best adapt to it.

2.2.3 Evaluation measures

In order to check if any sort of data mining operations are correctly performed, whether it is classification, clustering or even imputation, there is a need for evaluation methods. These methods include Root Mean Squared Error, ROC curve and AUC [58], accuracy, and some of the more specific performance metrics when it comes to imputation, prediction accuracy, and coefficient of determination. These are just some examples of the many evaluation measures that exist. Let O_i be the i^{th} observed value, and E_i be the i^{th} estimated value.

The Root Mean Squared Error (RMSE) is used as way of measuring precision and accuracy, by calculating the deviation between the real values, x and the prediction made x', and it can be used on imputed values, being T the amount of values imputed. The lower the RMSE the better the data has been imputed. It is represented mathematically as:

$$RMSE(x, x') = \sqrt{\frac{1}{T}\sum_{i=1}^{T}(x'_i - x_i)^2}.$$

A good way of measuring a binary classifier (classifier with two output classes) is by using the ROC curve and by calculating the respective AUC. ROC refers to Receiver Operating Characteristic and in short represents the trade-off between the true positive rate and the false positive rate when changing the classification threshold. AUC is short for Area Under the Curve, in this case the Curve delivered by ROC. This is a good efficacy metric that will have a value between 0.5 and 1. The closer the AUC is to 1, the better the model behaved.

The accuracy is another good method to evaluate the performance of a classifier. To calculate the accuracy, a proportion between the correctly assigned classification and the wrongfully assigned classifications need to be done. The higher the accuracy, the better the classification model is.

Prediction Accuracy (PA) is a performance indicator that can be used to evaluate whether the imputation is done correctly or not. It can take values that range from 0 to 1, being a value closer to 1 a better fit than if it were closer to 0. Using $\overline{O}$ and $\overline{E}$ as the average of the observed and expected (in this case imputed) values, and σ_O and σ_E as their standard deviations we get:

$$PA = \sum_{i=1}^{T}\frac{(E_i - \overline{E})(O_i - \overline{O})}{(T-1)\sigma_E\sigma_O}.$$

Finally, the last evaluation measure discussed will be the coefficient of determination, which is also used to evaluate imputation processes. In similarity with the previous performance indicator, it can take values that range between 0 and 1, being 1, once again, a better fit than 0. This coefficient can be used to assess the variability in the imputed data in relation to the actual values, and it is given by:

$$R^2 = \left(\frac{1}{T}\frac{\sum_{i=1}^{T}(E_i - \overline{E})(O_i - \overline{O})}{\sigma_E\sigma_O}\right)^2.$$

2.3 Time series analysis

This work will focus on the imputation of missing values in time series, which have a component that differentiate them from the rest of the data - time. Data is to be acquired over time and indexed in accordance to the passage of time, usually with equal time intervals, so that it forms a sequence of time-stamped data. This kind of data is used on many applications nowadays as, for example, in the stock market or medical records, amongst many other fields [59, 60] . Data mining applied to time series share the same concerns as if it were applied to the generality of data: classification, clustering, and forecasting. This last one refers to forecasting future values in time series and is a highly explored field.

A time series is a sequence of data points indexed in time $t \in \{1, \ldots, N\}$ with the result $X = (x_1, x_2, \ldots, x_N)$. Thus, a forecast of a time series, at a further time t, would be x_{N+t}.

Whichever is the goal of the data mining process, there is another procedure that can help filling a time series data. It is called trend analysis [61] and mainly consists of 4 components: trend, seasonal, cyclic and random variations.

- Trend variations refer to the general direction taken over a time interval as, for example, the trend to rise at a certain moment.

- Seasonal variations represent a certain event that re-occurs within a time interval, with a time interval associated with a calendar period, such as monthly or daily routines.

- Cyclic variations are very similar to seasonal variations, being the main difference the interval of time associated - these events use to have a duration longer then. a year

- Random variations, as the name implies, is associated with random events that occur, possibly modifying the previously discussed components.

The existence of these components in time series is very important, since they will enable the use of many algorithms that can analyze the patterns made by them. This is especially important when it comes to matching or aligning time series with one another. Medical databases will benefit greatly from these components because of the big sample of patients they hold, which in turn will increase the diversity of behaviour of many features, making it easier to find connections between the features of some patients.

Time series may need to be handled differently depending on the number of variables, especially when it comes to forecast an event. When a time series is only represented by a single variable, the forecast can only depend on the past and present values. These are called univariate time series. As opposed to this, the multivariate time series are composed by two or more features that are registered along time [62]. An example of this type of time series are medical records, which can hold many variables taken from the variety of exams patients undergo.

2.3.1 Forecasting

When it comes to forecasting several models can be used, being some of the most used the Auto-Regressive Integrated Moving Average (ARIMA) model [63], and the Exponential Smoothing [64]. ARIMA acts based on the autocorrelation of the data which in time series can be very useful for forecasting. This approach uses two procedures: a linear combination of the past values of the variable (auto-regressive model), and, instead of using the past values, it uses the past forecast errors (moving average model).

2.3.2 Clustering and Classification

Clustering in time series can be made with two different clustering types in mind. Whole-series clustering, where the objective is to take the whole time series and apply clustering techniques to it. This can be done by using any of the previously discussed techniques, or even any kind of general clustering approach, having only to define an appropriate distance measure. Methods such as Self-Organizing Maps, Auto-Regressive models and k-means can be used. The second type is called subsequence clustering and is performed on subsequences found in the time series, which will be used to generate the clusters. Although usually, Euclidian distances are used as similarity measures, many alternative methods have been created, dealing with other aspects of the time series such as the shape it takes.

As for classification, in similarity to what was described previously, operations will be done with two different types. One that deals with the whole sequence, and the other deals with subsequences. Either way, once again, any kind of classification technique can be applied and even other methods such as ARIMA can be used.

2.3.3 Data representation and summarization

Another field approached on time series is data representation [65]. The goal to achieve on this field is to reduce the dimensionality of time series, by using representation techniques, while still keeping intact all of the available information on the data set. One of the ways of achieving this is by using the Discrete Fourier Transformation [66]. Another known algorithm is the Symbolic Aggregate Approximation (SAX) [67]. One of the most important aspects of these methods is the distance definition between time series. Among the distance measures for the time series similarity, the Euclidian Distance and Dynamic Time Warping (DTW) [12, 68]. DTW is used to compare and align two time series due to the ability of this algorithm in altering the time series by “compressing” or “decompressing” these.

Data summarization is another of the fields of interest in time series. It strives for summarizing and describing the data in time series, in order to obtain a better performance on the other data mining tasks to be applied on the time series. Many methods have been developed using existing techniques, such as clustering methods, to perform the data summarization.

2.4 Missing data imputation

As previously discussed, imputation is a preprocessing task that should not be overlooked. If the missing values were to be ignored a lot of information would be lost. This becomes an even bigger issue when time series are being handled because future values may depend on the past values, and by erasing certain past data, the operations of forecasting, classification and clustering will become more error prone. The best course of action would be to adopt imputation techniques so that the data could be filled with plausible values. Three types of missing values can be considered: Missing data are not random (NMAR), Missing Data are Random (MAR), and Missing Completely at Random (MCAR). By defining the type of missing data, an appropriate imputation algorithm can also be chosen [69]. Of these types of missing data the most complicated to deal with is MCAR due to the difficulty in finding patterns to impute this type of data.

2.4.1 Statistical Methods

Some of the imputation techniques resort to statistical methods. The most simple of these methods replaces the missing values with the mean or median of the variable [70]. This leads to biased data, diminishing the correlations in these variables, having a negative impact on the learning tasks to be performed. This becomes an even bigger issue when multivariate data is being used. Another statistical method used is the hot-deck imputation [71], which will use the value of a random instance of the required variable to fill in the missing value. An example of this method is the Last Observation Carried Forward [72]. This method makes an ordered version of the data set (according to some variables), and whenever missing values are detected, the previous value will be used to impute them. Once again, this may lead to bias problems. One method that achieves good results in terms of predicting variables is regression. A regression model is built to predict values, based on other variables, in order to impute the data. Although these methods are fairly simple to use, they still lack in terms of efficiency.

One of the most used imputation techniques is called *multiple imputation* [73]. This method has three phases: the first phase is to create n copies of the database, each one with separate imputed values. These values are obtained by using an appropriate model, and also by giving them some variability, in order to include the uncertainty of the imputation. If the data present in the database is highly relatable, making it easier to predict the data to be imputed, there is a smaller need for the variability, although there will still exist some to account for the uncertainty. In cases where the imputed data is not as relatable to the existing data, the variability given to it will be increased, in order to account for a bigger number of cases, making it more likely to achieve the desired results. The second phase refers to the analysis of the n created databases. At this point, it is to be expected that the analysis will result in different outcomes. This is related to the given variability imputation-wise. The third and final phase is the merging of all the information collected on the previous step in order to perform statistical analysis on them, thus inferring on the best possible imputation process.

A particular case of this method is MICE (Multivariate Imputation by Chained Equations) [74]. MICE runs a series of regressive models, that will be performed on variables with missing values, being mod-

elled against the other variables on the data set, so that each variable is modelled according to the way it is distributed. In order to perform this method, firstly a basic imputation method should be applied, such as replacing all the missing values with the mean values. This holds the purpose of while performing the regression for a certain variable, the other variables will have complete data to enable the operations. After this, a single value of one variable that was imputed will be removed, and the remaining values, which are known, will be regressed on the other variables. After the creation of this regression model, the missing values belonging to this variable can finally be filled, resorting to the created regression model. This will then be repeated for all variables, until all variables have been imputed resorting to the regression model. This will be regarded as a cycle, being more cycles performed in order to give a better approximation to the values of the imputed data. In resemblance to what was previously discussed, this whole process will be repeated n times, forming n databases, each with different imputed values due to the variability that was added. The rest of the process will hold similar to what was previously described.

2.4.2 Time Series Imputation

One of the easiest techniques to apply to impute time series is a simple moving average [75]. This consists of a window of data that will use a certain amount of values, from either way of the missing value, to calculate a mean that will be used to impute the missing value. A problem that can arise from this method is when a window takes up more than one missing value. The solution found for this is the automatic increase of the window until the necessary amount of values is found. Other variations of this algorithm exist such as the Exponential Weighted Moving Average.

Multiple imputation is a technique that is commonly used to impute data in time series, particularly, when it comes to multivariate time series. Other of the already described algorithms can also be used in multivariate time series imputation such as the Nearest Neighbour [76] and the Expectation-Maximization methods [77].

Another method used on time series revolves around the notion of the previously described time series characteristics. By removing the components related with trends and seasonality, the imputation process becomes easier, due to the lack of “noise” derived from these components [69]. After removing the components any of the imputation methods described can be applied to perform the imputation. Afterwards, the removed trend and seasonality components will be added in order to deliver the final imputation results. The performance of this method is greatly enhanced when the data sets on which it operates has strong seasonality and trends, while if it does not, the results will be close to what would happen if the seasonality decomposition has not been applied.

When it comes to univariate time series, the methods used must differ from these, in the sense that the univariate methods can only depend on data from the one variable it possesses, thus only being able to analyze past and future values on the time series to impute the data. Linear/Spline interpolation are some of the methods used for univariate time series imputation [78].

In the following section, the sequence alignment algorithms which will be used in this work will be introduced.

2.5 Dynamic time warping - DTW

The DTW algorithm consists in the alignment of two data series by trying to explain variability in the Y-axis with variability in the X-axis [12]. An example of this can be seen in Figure 2.2.

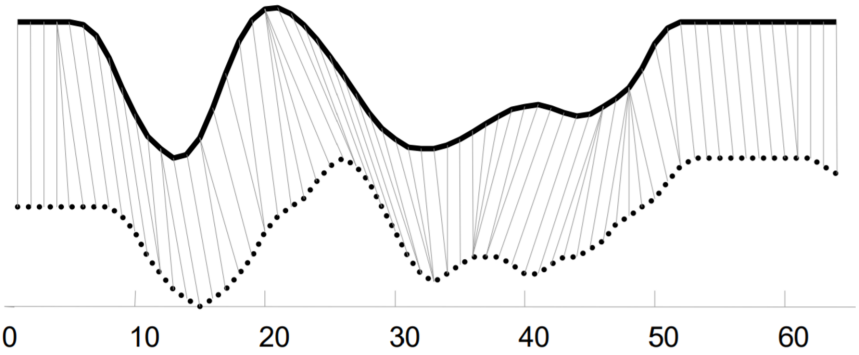

Figure 2.2: Example of the dynamic time warping algorithm [79].

This algorithm takes two time series (Equations (2.1) and (2.2)), not necessarily with the same size, and builds an m-by-n matrix in order to align both of these sequences. Each of the elements of this matrix will correspond to a distance $\delta(a_i, b_j)$ between two points a_i and b_j, for $i \in \{1, ..., n\}$ and $j \in \{1, ..., m\}$. The goal of this matrix is to help to determine a path in which the distance between both sequences is minimized. This is called a warping path and an example of one is given in Figure 2.3, and it would be represented as in Equation (2.3).

$$A = a_1, a_2, ..., a_i, ..., a_n \tag{2.1}$$

$$B = b_1, b_2, ..., b_j, ..., b_m \tag{2.2}$$

$$W = w_1, w_2, ..., w_k, ..., w_t \tag{2.3}$$
$$\max(n, m) \leq t < n + m - 1$$

The warping path elements w_k represent the alignment of two points of the sequences to be aligned $(i, j)_k$. A warping path must obey certain conditions [12]:

- **Continuity** - There is a limit in the steps that can be given: If $w_k = (i, j)$ and $w_{k-1} = (i', j')$ then we must have $i - i' \leq 1$ and $j - j' \leq 1$. In short terms this means the warping path must only progress in the matrix through adjacent cells (horizontally or diagonally).
- **Monotonicity** - The points of the warping path must be monotonically ordered in time such that for $w_k = (i, j)$ and $w_{k-1} = (i', j')$ we have $i \geq i'$ and $j \geq j'$

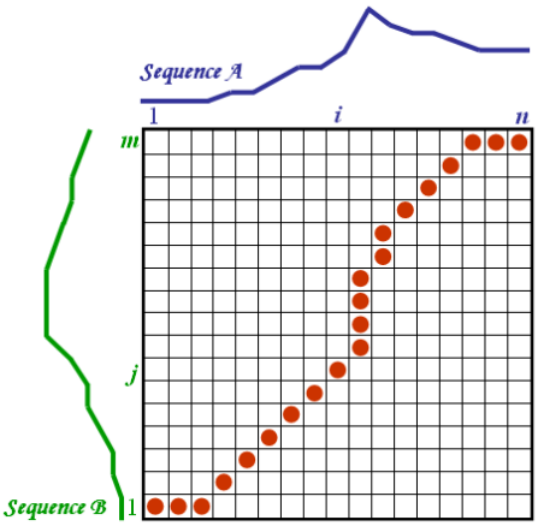

Figure 2.3: The alignment of two sequences (Equations (2.1) and (2.2)) in a matrix, and the respective warping path (Equation (2.3)) [80].

- **Boundary Conditions** - The first point of the warping path and the last should refer to the respective first and last points of the sequences to be matched: $w_1 = (1, 1)$ and $w_t = (n, m)$. Although this is one of the conditions, sometimes there are exceptions that are introduced by giving offsets that could be used to initiate/terminate the warping path.

Many paths can be determined using these conditions, however the goal is to minimize the warping path as much as possible, as presented in Equation (2.4).

$$DTW(A, B) = \min \left(\frac{\sqrt{\sum_{k=1}^{t} w_k}}{Z} \right), \tag{2.4}$$

Z is a coefficient used to compensate for the difference in size of the warping paths. One of the possibilities for this value is the size of the found warping path $Z = t$. Having w_k as the distance between elements of the time series $w_k = \delta(i, j)$, two commonly used distance measures are presented in Equations (2.5) and (2.6).

$$\delta(i, j) = |a_i - b_j| \tag{2.5}$$

$$\delta(i, j) = (a_i - b_j)^2 \tag{2.6}$$

The path can then be found by applying dynamic programming to calculate the cumulative distance $\gamma(i, j)$ for each point, which will be the distance for the current points $\delta(i, j)$ added to the minimum of the cumulative distances of the adjacent elements in the matrix (both horizontally and diagonally), as shown in Equation (2.7).

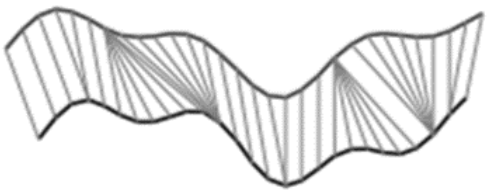

Figure 2.4: DTW alignment.

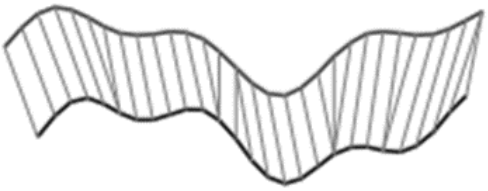

Figure 2.5: DDTW alignment.

Figure 2.6: The alignment of two sequences by using DTW 2.4 and by using DDTW 2.5 [79].

$$\gamma(i,j) = \delta(a_i, b_j) + min[\gamma(i-1,j), \gamma(i,j-1), \gamma(i-1,j-1)] \tag{2.7}$$

When the algorithm is completed, the optimal warping path will be found by tracing backwards through the minimum found values. When the warping path is found a score will be attributed to the match, which will reflect upon the distance of the full warping path to both sequences (in this case reflecting how good the fit was).

2.5.1 Derivative Dynamic Time Series - DDTW

Although DTW is able to align many sequences with great proximity, excelling particularly when it comes to variations in the X-axis (variations in time), there are many sequences in which it will encounter certain issues. One of the problems it will encounter is when it comes to variations in the Y-axis. With the presence of local features, such as peaks and valleys, the alignment will be affected. The DTW algorithm will attempt to justify these local features by making corrections in the X-axis, thus producing singularities which will affect the alignment in the places where these features are located. The issue falls on the alignment being made depending solely on the distance of the sequence points, and not considering the waveform/shape of the sequence. To mitigate this problem, DDTW was created [79]. This algorithm would be using the derivative of the sequences, which will have more significant results in terms of aligning the waves using their shape. In Figure 2.6 it is visible the differences when aligning the same sequence with DTW and DDTW.

The algorithm will behave in the same way as before, however the alignment will happen on the derivative of the sequences. Usually, for simplicity, the derivative is calculated based on the slopes of the point in question with the point on the left and the one on the right. Thus, the derivative, $D[a_i]$ at any

given point, a_i, of the sequence will be as presented in Equation (2.8).

$$D[a_i] = \frac{(a_i - a_{i-1}) + \frac{(a_{i+1} - a_{i-1})}{2}}{2} \tag{2.8}$$

By having the derivative calculated as shown in Equation (2.8), there could be loss of information when it comes to the first and last points of the sequence. Another problem is the exact issue trying to be solved: missing values. Missing values will not only have a direct impact in the point where they are missing, but also in their surroundings, specifically to their immediate right and left. This can have a big impact particularly when it comes to short sequences.

2.5.2 Dynamic time warping-based imputation - DTWBI

Having both of the previous sections in mind we can now look at Dynamic time warping-based imputation or DTWBI for short [81]. This method was developed with imputation of gaps (a continuous group of missing values) as a goal. This method was originally applied in univariate time series, and would compare the sequence with the missing gap with other time series of the same dataset, trying to find the most similar sequence to the one with the missing value.

To perform this imputation, the first step would be to extract a sub-sequence, before or after the gap. Then, the DDTW algorithm would be applied in other time series, locally, as a sliding window, to find the best match possible. The final step would be to copy the values associated with the best match, which was found in the previous step, at the relative position of the gap and the sub-sequence used to find the best match. For instance, if there was a gap from positions a_{20} to a_{25} and the sub-sequence a_{14} to a_{19} was used to find a match, then if the best match were to be found (on a different time series) at positions b_7 to b_{12} then the values to be used for the imputation would be from b_{13} to b_{18}. An example of this algorithm can be seen in Figure 2.7.

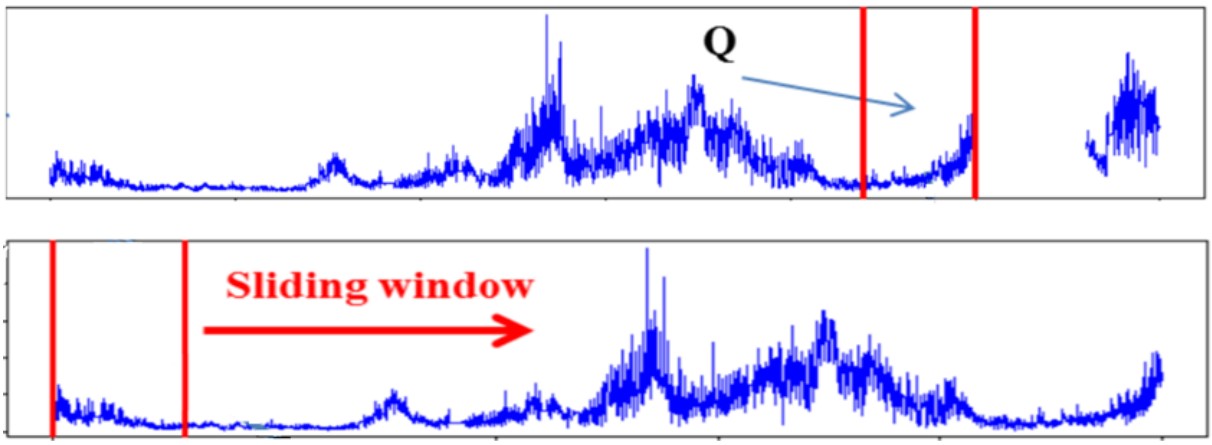

Figure 2.7: An example of the DTWBI algorithm using a sliding window to find the optimal match. [81].

2.6 Needleman-Wunsch algorithm - NW

This algorithm was chosen due to the similarity in terms of behaviour, comparing to DTW algorithm, in the sense that both perform sequence alignment. The NW algorithm [14] was mainly created with the purpose of aligning discrete sequences in the field of bioinformatics such as protein or nucleotide sequences (related with DNA).

It will also behave in a similar way to the DTW algorithm, with the construction of a matrix using both sequences. The difference will be that, in DTW, the distance between data points is used to build the alignment matrix. In NW, a similarity matrix will be used to build the alignment matrix, with given scores between two data points a_i and b_j, for $i \in \{1, ..., n\}$ and $j \in \{1, ..., m\}$.This similarity matrix must be consulted to define the scores of each cell of the matrix. The similarity matrix can be custom made or it can be built based on the scores attributed to matches, mismatches and gaps. Typically the score for matches is 1, while for mismatches is 0 (this is considered a penalty). Besides these factors there are also two types of gap penalties. These can be gap openings, which refer to when a gap must be introduced to a sequence, and the gap extension penalty which is used when a gap (which was already open) extends. When using a custom similarity matrix certain matches of characters could be assigned a greater score, thus favoring the matching/mismatching with certain characters.

When the matrix is filled with all the values, the traceback phase will occur, in similarity to what happens in DTW. Then, it will generate the alignment by finding the path, starting at the bottom right cell of the matrix and ending at the top left cell. To find this path the movements from one cell to another can be made upwards, diagonally or to the left. When a move is made diagonally it will correspond to a match/mismatch, and when it moves either to the up or left a gap is to be introduced. In Figure 2.8 a representation of the Needleman-Wunsch algorithm is shown. On the left the scoring matrix is represented along with the traceback (path in orange), which was made by following the blue lines which were added during the building of the matrix, while on the right side we have the similarity matrix with the scores for the matches and mismatches.

An alternative to this method could be the Smith-Waterman algorithm [82], which is a variation of the NW algorithm. The main difference between these algorithms is that, while the NW algorithm operates globally on a sequence, the Smith-Waterman will act using a local approach. In the traceback phase, instead of aligning both sequences globally it will analyze the best score in the matrix and build the best local path it can find. This means that certain parts of both sequences may not belong to the alignment if they negatively impact the best score.

The next chapter describes will describe the implementation of the imputation algorithms which will make use of DTW and NW.

		G	C	A	T	G	C	U
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	1	0	-1	-2	-3	-4	-5
A	-2	0	0	1	0	-1	-2	-3
T	-3	-1	-1	0	2	1	0	-1
T	-4	-2	-2	-1	1	1	0	-1
A	-5	-3	-3	-1	0	0	0	-1
C	-6	-4	-2	-2	-1	-1	1	0
A	-7	-5	-3	-1	-2	-2	0	0

	A	G	C	T	U
A	1	-1	-1	-1	-1
G	-1	1	-1	-1	-1
C	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
U	-1	-1	-1	-1	1

Figure 2.8: Example of the result of the Needleman-Wunsch algorithm applied to align two DNA sequences with a gap value of -1.

Chapter 3

Proposed Method

This work will consider two main approaches to data imputation, focusing on time series, which will be explained in this section. These methods were chosen due to the similarity in methodology, and will both be compared in a later section.

3.1 DTW imputation

The first method will be based on DTW to perform imputation of missing values. The algorithm will resemble the DTWBI algorithm explained in the previous chapter, although it will suffer some adjustments.

Firstly, the number of points of the window that will be used to find a match will be chosen, and this number will also have impact in the amount of windows to be analyzed. For instance, by analyzing a window of size 5 we will have a maximum of 6 different windows to analyze placed around the missing value. An example of this is represented in Figure 3.1. The size of the window should be influenced by the size of the time series analyzed, as well as the amount of missing values in the series. If the missing value is placed in the beginning or ending of a time series the amount of windows to be used will be reduced due to the non-existence of points to analyze surrounding the missing value. If there are other missing values surrounding the analyzed missing value, the amount of windows will also be adapted to whichever number of windows that are able to utilize in the algorithm.

After the windows which are to be used for the comparison are chosen, they will be compared against other samples of time series. The objective is to find (regarding the same feature) the most similar subsequence in all of the data available against the chosen windows. Needless to say that the more data to be compared, the better due to increased diversity of samples that this could bring. The algorithm must of course not be applied in the presence of missing values in the other data for the alignments to be accurate. Additionally, the position from which the value to fill for the missing value will be retrieved must not contain a missing value.

In this algorithm, the DDTW will be applied, and as such the derivative of the time series must be calculated. One of the problems which was already described would be the loss of information regarding the initial and final points of the time series, as well as loss of information on the points surrounding

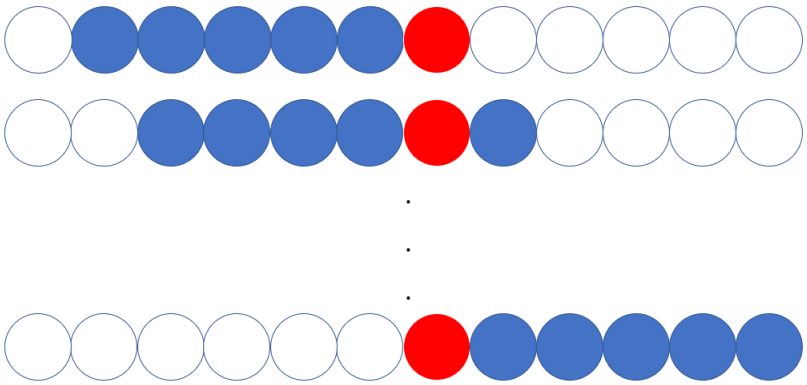

Figure 3.1: Example of the result of the windows to be compared with DTW. Blue circles represent the window to be analyzed, red circles represent the missing value, and the white circles are the remaining data points.

missing values. To counter this, the derivative was calculated by using two points whenever it was necessary. This was done because, in order for this work to be applied in short time series, the amount of information that would be lost because of the derivative using three points would be impactful in the efficiency of the algorithm.

Regarding the imputation of the actual value, instead of copying the exact values as done in the DTWBI algorithm, the value will be calculated by using (2.8). The objective will be to calculate a_i, having $D[a_i]$ as the value found (in the same place of the missing value when compared to the window to be searched), and the values of a_{i+1} and a_{i-1} as the values surrounding the missing value to be imputed.

3.2 Needleman-Wunsch Imputation

The other method to be analyzed will rely on the Needleman-Wunsch algorithm. In order to use this method on the imputation of continuous time series, a pre-processing step of discretization must be taken. The dataDiscretize function from the R package bnspatial was used. The discretization was made with equal sized classes, being the number of classes to be used based on the number of data points available in the sequences. Another important factor is that the discretization for each individual is made independently, which will be beneficial in terms of creating matches between sequences with similar shapes instead of only considering the values of the data points.

To perform this algorithm on a sequence with missing values, the missing values will be assigned a special character: '?'. Also, when building the similarity matrix, we must consider that the objective of this algorithm is to match the missing values ('?') with another character (a suitable one) in order to discover its value. To do so, the similarity matrix must be built with such objective in mind, which means the matrix should consider the following:

- All matches between characters (except for the matches between '?') will have a score of 1.
- All mismatches between characters (except for the ones involving '?') will have the score of -1.
- The match between '?' characters is unwanted, because it would mean that two missing values

are being aligned, which is not the objective of the algorithm. As such, the value assigned to the match of '?' will be the same as a mismatch (-1).

- Mismatches with '?' must be considered beneficial in order to find the values used to replace the missing values. As such, this value must be higher than the normal mismatch value, being the chosen value -0.5.
- The gap introduction penalty was set to -0.8 in order to prioritize finding a mismatch for the missing value instead of opening a gap.
- The gap extension penalty was set to -0.3, so that if a gap opened beforehand a missing value is not mismatched. This means that the missing values will not have a correspondence to a data point inside a gap.

Having the matrix built in such a fashion the algorithm will behave in the usual way, and the best possible match will be the one used to retrieve the missing values. However, there is a possibility that the missing values could align with a gap, which would not be useful to this algorithm. When this occurs the algorithm will have to run again with a feature enabled which will not allow the algorithm to align missing values at the first missing value. It will run as many time as necessary to fill all the missing values.

In the next chapter, the tests and results obtained with these algorithms are presented.

Chapter 4

Results

In this section the results obtained with the previously described methods will be presented. Besides this, some information on the used data will be given as well as a comparison on the performance of both algorithms.

4.1 Datasets

In this work, three datasets were used: two synthetic datasets, ECG and CMUsubject16 [83] and a real dataset, Epileptic Seizure Recognition [84]. This last dataset is a recording of brain activity in several patients under different situations with the main goal of studying people suffering of epileptic seizures, which is helpful when trying to perform the imputation, due to the variety of data, even though this dataset holds only one feature. This dataset is also the one with the highest range, with values from -1885 to 2047. The ECG dataset contains data on 200 individuals with only two features. Although the time series are short (39 datapoints), there is a wide range of values in the sequences (going from -438 to 430). CMUSubject16 is a database with few individuals but with many features (62). The data has a range from -137.54 to 437.35, although the range is shorter on each characteristic. A brief overview on the features of these datasets is presented in Table 4.1.

Dataset	Individuals	Length	Features
CMUsubject16	58	127	62
ECG	200	39	2
Seizures	11500	178	1

Table 4.1: Overview of the used datasets.

4.2 Single missing values imputation

Firstly, tests were made with a single missing value (picked at random) in a certain individual and feature (also picked at random). To test this, a missing value was inserted in each of the datasets, and

once found these would be compared against the original values. In order to be able to compare both procedures, the missing values used for the DTW imputation will be the same used in NW imputation.

As for evaluations measures, the Root Mean Squared Error (RMSE) will be used as a way of measuring precision and accuracy by calculating the deviation between the real values, x and the prediction made x'.

The prediction accuracy (PA) indicator will also be used which is a performance indicator that can be used to evaluate whether the imputation is done correctly or not. It can take values that range from 0 to 1, being a value closer to 1 a better fit than if it were closer to 0.

Finally, the last evaluation measure discussed will be the coefficient of determination (R^2), which is also used to evaluate imputation processes. In similarity with the previous performance indicator, it can take values that range between 0 and 1, being 1, once again, a better fit than 0. This coefficient can be used to assess the variability in the imputed data in relation to the actual values.

The following subsections will discuss the tests made in both approaches.

4.2.1 DTW Imputation

The results of the single missing values imputation using DTW is shown on Table 4.2.

Dataset	RMSE	PA	R^2
CMUsubject16	0.307	0.999	0.766
ECG	13.72	0.993	0.756
Seizures	3.491	0.999	0.875

Table 4.2: Evaluation measures of the single imputation using DTW.

By analyzing Table 4.2 it is shown that the algorithm worked well overall on all the datasets. The ECG dataset was the one with which this method was least effective, and this could be due to the high range of the data (in the case of ECG data ranges from -438 to 430), as well as the amount of data available to compare in order to search for the missing values. Even so, the obtained result for this dataset was mostly accurate. The seizures dataset also had good results even though the RMSE is higher when compared to the CMUsubject16 dataset. This could be once again caused by the range of the data (which in the case of the seizure dataset ranges from -1885 to 2047). Regardless, considering this dataset contained real data, the algorithm could impute data with good precision. In Figure 4.1 an example of an alignment can be visualized. The sequence in blue is the one with the missing value, represented by the cross, and the sequence in orange is the best match found in the dataset. This work used a window size of 5 in the algorithm, and because it analyses the surroundings of the missing value (with the window size given) the charts will have the missing value position in the middle, and will also have the 5 prior and following datapoints. Take in consideration also that any of the windows could have been used to find the best match, and that this match is based on the shape of the waves instead of the values.

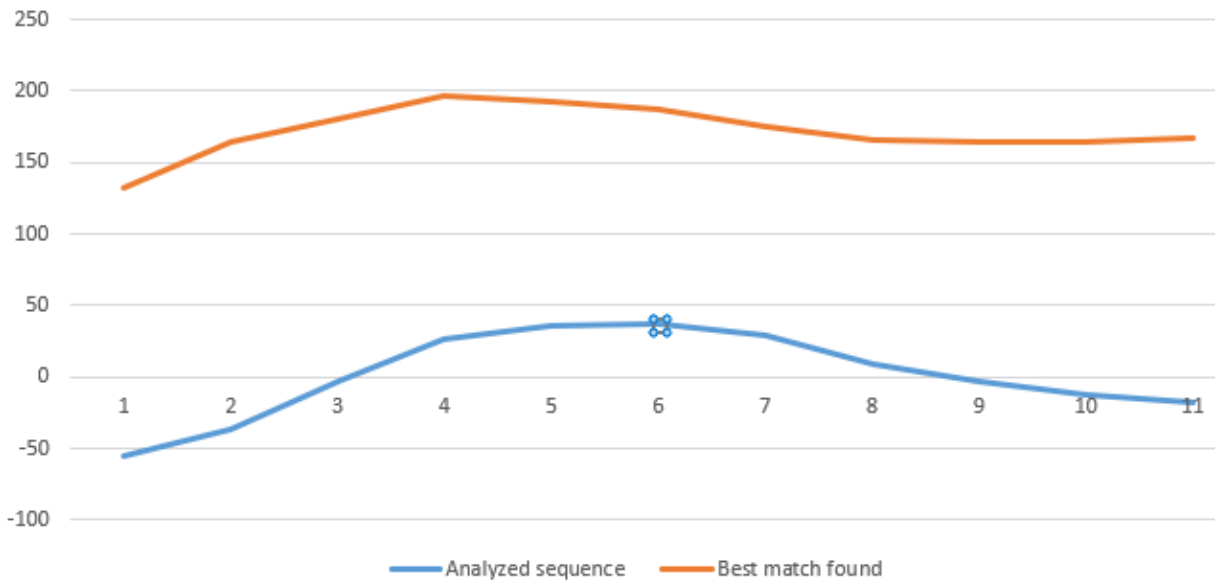

Figure 4.1: Example of the result of the DTW algorithm applied to align two sequences of the Seizures dataset.

NW Imputation

The results of the single missing values imputation using NW are shown on Table 4.3.

Dataset	RMSE	PA	R^2
CMUsubject16	0.523	0.999	0.765
ECG	39.02	0.946	0.685
Seizures	12.99	0.997	0.872

Table 4.3: Evaluation measures of the single imputation using NW.

The NW imputation algorithm shown in Table 4.3 shows that for the CMUsubject16 dataset the results were quite good. Regarding ECG and Seizures datasets the results were not as good. Once again, the ECG dataset suffered the worst imputation, which could be associated with the amount of datapoints for each individual. The size of the sequences directly affects the discretization, as was mentioned in the earlier chapter, because the amount of classes created depends directly on the size of the sequences. Because this particular dataset has short sequences and a big range, the boundaries of each class will have a big range, which will make this method more inaccurate (because the values will be imputed with the intermediate value of the range of the class assigned to the missing value).

As for the Seizures dataset, the results could be explained by the same reason. Because this dataset has bigger sequences, the effects of the discretization were not as noticeable as on the ECG dataset, nonetheless the effects of the range of the data of the Seizures dataset and the size of the sequences also affected the performance of the algorithm. The CMUsubject16 dataset had the best imputation out of the three, which can be explained by the size of the sequences (having a size that facilitates a good amount of classes) and the range of the values, which in this case is smaller than the other two datasets (ranges from -137.54 to 437.35, although the ranges for each feature it has are much smaller). Although the results for the NW were the best for this dataset, when examining the alignment of the

discrete values, we could observe that some of the matches were not correct, but due to the proximity in class (and having each class a low range of values), the imputation was not as affected by the wrongful alignment of the sequences, as the other two datasets. In Figure 4.2 an example of the discretized alignment is given. The missing value is represented by a "?".

Analyzed sequence	Best match found
d	d
c	c
b	b
b	b
b	b
c	c
c	c
d	d
?	e
-	e
-	f
g	g
g	g

Figure 4.2: Example of the result of the NW algorithm applied to align two sequences of the Seizures dataset.

4.2.2 DTW vs NW

Comparing both of these algorithms it can be said that DTW is the most accurate one, although on large sequences with low range of data the results are quite close. Regarding speed, the NW imputation is faster than DTW imputation. It was more noticeable on the Seizure dataset, due to the amount of individuals it had. The NW algorithm also had an issue due to the alignments of missing values with gaps. This was surpassed when forcing the algorithm on finding the best possible alignment which would not align the missing value with gaps.

4.3 Multiple missing values imputation

On the second part of this work , the imputation of data when multiple missing values are present in a sequence will be analyzed. This will be done by following the same steps as before, although instead of removing only one datapoint at random, an individual and feature will be chosen at random, and from the elected sequence we will have 5%, 10%, 15% and 20% of missing values in randomly picked positions. This will be done in four different runs for each missing percentage, with the exception of the Seizure dataset which will be done in a single run, due to the high computation time of the DTW imputation on this dataset. For effects of comparison the same positions will be applied on DTW and NW imputation.

4.3.1 DTW imputation

The results of the DTW imputation with multiple missing values are shown on Table 4.4.

Dataset	Missing %	RMSE	PA	R^2
CMUsubject16	5%	0.221	0.999	0.934
	10%	1.802	0.997	0.965
	15%	2.08	0.997	0.973
	20%	1.8	0.997	0.978
ECG	5%	24.01	0.979	0.734
	10%	36.24	0.934	0.766
	15%	45.82	0.965	0.856
	20%	42.05	0.948	0.844
Seizures	5%	13.4	0.998	0.787
	10%	53.44	0.997	0.888
	15%	29.66	0.907	0.764
	20%	25.98	0.894	0.757

Table 4.4: Evaluation measures of the multiple imputation using DTW.

The CMUsubject16 had the expected behaviour with the increase in the RMSE, even though for the 20% missing values the RMSE value decreased. Overall the values were in accordance to what was seen in the single imputation for this dataset, having a good accuracy and low errors, even in the presence of many missing values. Regarding the ECG dataset, there were also no surprises, because in similarily to what happened with the single imputation, this was the dataset which had the worst performance. Both of these datasets had a relatively fast execution time, which was not the case for the Seizures dataset, which required a lot of processing time to handle the imputations. Due to these long execution times, only one execution of the algorithm was done, causing the values on Table 4.4 to be not as compliant to the expected when compared to the ECG and CMUsubject16 datasets. Although it was to be expected for the RMSE to be the lowest for the 20% missing values, this was not verified, with the highest RMSE on the 10% missing values. In terms of the prediction accuracy, the values were particularly good in the CMUsubject16 dataset. Both the Seizures and ECG dataset also had good results, although the Seizures dataset struggled with a higher missing percentage. An example of the imputation with 20% missing values is shown on Figure 4.3.

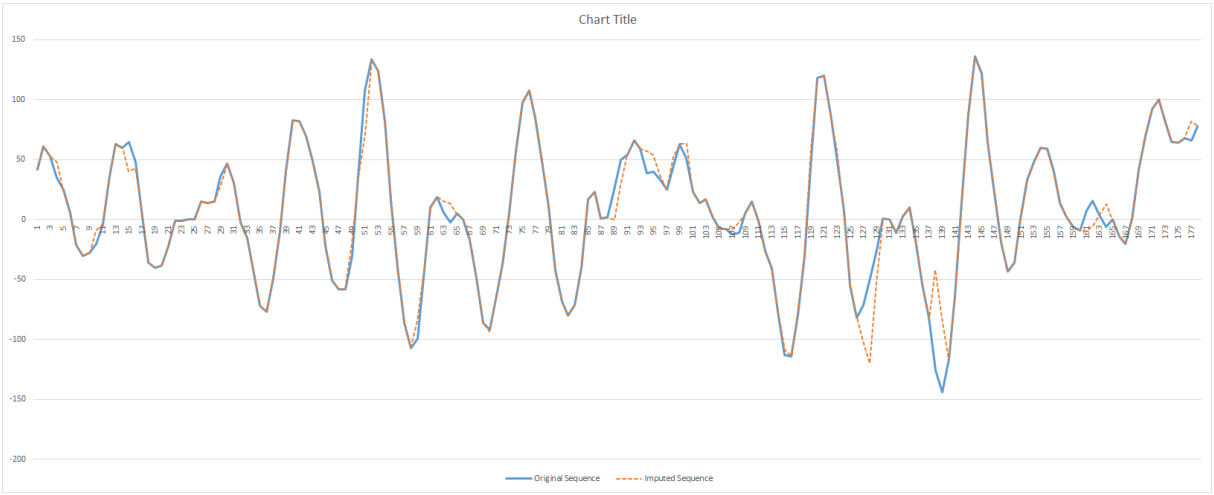

Figure 4.3: Alignment of the sequence using DTW with 20% missing values for the Seizures dataset.

4.3.2 NW imputation

The results of the NW imputation with multiple missing values are shown on Table 4.5.

Dataset	Missing %	RMSE	PA	R^2
CMUsubject16	5%	1.873	0.998	0.931
	10%	3.784	0.991	0.952
	15%	3.622	0.992	0.963
	20%	3.088	0.994	0.973
ECG	5%	39.8	0.942	0.78
	10%	40.54	0.889	0.727
	15%	45.5	0.925	0.809
	20%	57.81	0.862	0.713
Seizures	5%	39.76	0.998	0.787
	10%	289.2	0.855	0.652
	15%	36.17	0.891	0.627
	20%	55.19	0.502	0.238

Table 4.5: Evaluation measures of the multiple imputation using NW.

During the tests, in order to impute all missing values in a sequence, the algorithm had to run multiple times, due to the alignment occurring globally, which means that sometimes a certain amount of missing values would align with gaps. In the following runs the already imputed values would be used and a restriction to find the best matching sequence which could have a match for any missing value would be implemented in order to fill the missing value.

By inspecting Table 4.5 we can verify that for the CMUsubject16 dataset the RMSE was generally low. It is also to be noticed the accuracy for the 20% missings was better than the 15% and 10% missings which was not according to expectations. Although this behaviour was not expected, even though this result may have happened due to the low number of tests performed, this could mean the algorithm can still show acceptable results with a large amount of missing values. Regarding the ECG dataset the same could be verified, although with lesser accuracy due to the size of the classes made by the discretization, leading to imputed values with less accuracy. This behaviour also repeats itself in the seizures dataset, although the 10% RMSE results are far from the expected because of the presence of an outlier in the data (and because this algorithm ran only once), and having difficulties with the imputation with the 20% of missing values. An example of the imputation the Seizures dataset is done on Figure 4.4

4.3.3 DTW vs NW

By analyzing the results on Tables 4.4 and 4.5, it is possible to see that both algorithms behave in a similar way (although there are some exceptions such as the results for the Seizures with 10% missing values). The DTW algorithm is mostly the one with the biggest accuracy, although the NW algorithm has shown good accuracy when facing a big percentage of missing values in the ECG and CMUsubject16 datasets. Although DTW had the biggest accuracy it was the slowest when compared to the NW algorithm. With multiple values missing the difference in execution time is more noticeable, because the DTW algorithm analyzes the missing values one by one, while the NW algorithm matches the sequences

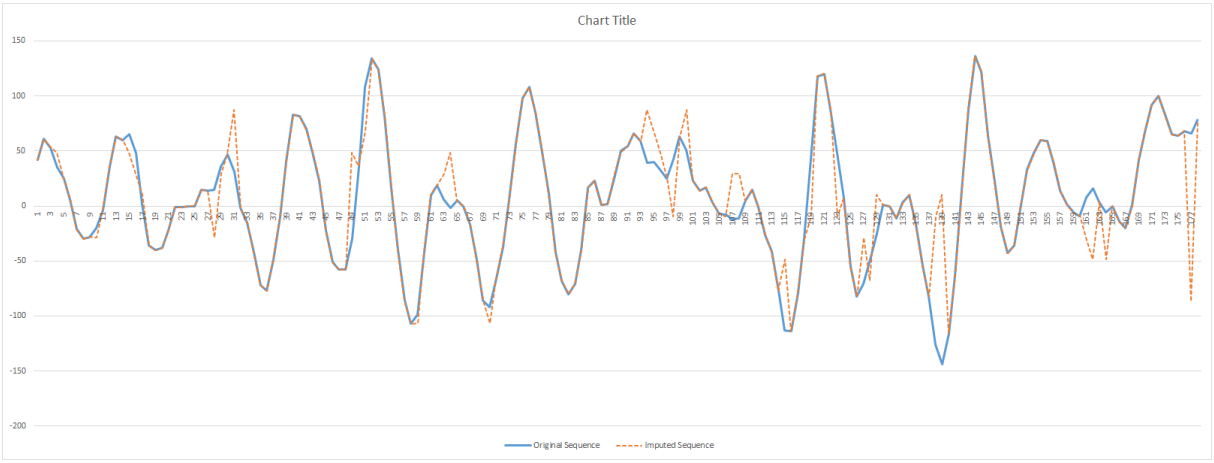

Figure 4.4: Alignment of the sequence using NW with 20% missing values for the Seizures dataset.

globally, although more runs of the algorithm may be necessary to fill the missing values aligned with gaps.

While analyzing the imputations in the DTW algorithm it was seen that in the presence of gaps of missing values the imputation accuracy would drop the most, which can be explained by the calculations made for the derivative using only two available points (which is less effective than the derivative using three points). In the NW algorithm these effects were not as noticeable.

4.4 DTW Discussion

This algorithm gave off the best accuracy overall when compared to the NW algorithm. A few factors could have influenced this such as the fact that this algorithm fills the missing values, and then uses these filled missing values to proceed in finding the next ones. Also, due to the algorithm being applied locally and with several windows, better results were found, which could be overlooked if the algorithm acted on a global scale. That being said, because this algorithm has an exhaustive local analysis, it will also take a long time to complete, especially with multiple missing values on big sequences and a lot of individuals to analyze, which was the case for the Seizures dataset. In order to improve on this aspect a threshold could be established on the scores of the DTW match. Another solution could be comparing all sequences globally in order to determine the ones most likely to succeed in an alignment, and use those on the algorithm. There should also be made attempts to discover a relation between the window size and the size of the sequences being analyzed, as well as the number of windows to be analyzed.

4.5 NW Discussion

In terms of accuracy, the NW algorithm was lacking when compared to DTW, even though the results were still appropriate when compared to the original values. The positive point is that it takes less time than the DTW algorithm. This is because the algorithm compares sequences globally which makes the process faster. A downside of this is that, when comparing sequences globally, some missing values

might only match with gaps, which was countered by adding certain validations, although this means the algorithm will need to process more alignments. Even with this extra step it was still much faster than the DTW algorithm. One of the biggest problems could be the discretization of the sequences. When applying discretization, information will be lost, which could be troublesome when retrieving the final values for the imputation. This issue comes from the fact that the number of classes are chosen based on the size of the sequences being analyzed, which will affect short series with high variability in their data. For example, regarding the ECG and Seizures dataset the size of the sequences and the high variability resulted in a discretization in which classes had a wide range (this happened specially in the ECG dataset).

An additional improvement would be re-designing the similarity matrix. Instead of using the same mismatch value for whenever a mismatch between two characters occur, a score could be assigned to reflect the proximity between the classes. For example, class ‘C’ would have a higher mismatch score with classes ‘B’ and ‘D’ and the mismatch scores would worsen from these on out. This would promote mismatches with the nearest classes, instead of with any class.

Another suggestion of an upgrade for this project is the use of a local version of this algorithm to be compared with DTW which acts locally. To achieve this, the Smith-Waterman algorithm could be used.

Chapter 5

Conclusions

In this chapter the achievements of this work are listed, as well as the future work that can be done.

5.1 Achievements

In this paper, it was intended to analyze the behaviour of two algorithms when applied to sequences with missing values in a continuous time series. This analysis is considered extremely important, given the fact that there are many machine learning algorithms applied to data that have missing values, and this happening is not being taken in proper consideration, which could have a negative impact on these algorithms. Thus, this work implemented two algorithms based on DTW and NW alignments, applied to three different datasets. In addition, some tests were also performed regarding the number of missing values: single missing values imputation and multiple missing value imputation, placing 5%, 10%, 15% and 20% of the dataset as missing values. Results showed both algorithms can be used for imputation, each with their respective strengths - accuracy, when speaking about DTW, or speed, when speaking about NW.

5.2 Future Work

There are many ways in which the proposed algorithms can improve, which were mentioned in the previous chapter. Regarding the DTW based algorithm:

- The biggest problem this algorithm faces is the amount of time expended in performing the imputation. A threshold could be implemented, or even a sort of preprocessing step to eliminate the less likely sequences to have a match.
- Another thing to consider is the size of the windows used in searching for the alignments.

As for the NW based algorithm, which struggled with the accuracy:

- Consider doing a local version of this algorithm by using Smith-Waterman.

- Applying a discretization algorithm that is able to generate the classes in such a way that the algorithm will not be affected by the lack of suitable classes to perform the correct imputation.
- The values of the similarity matrix could also be tweaked, especially the mismatch values, as well as the gap and gap extension values.

Besides these points, both algorithms need further testing to verify their usefulness in imputing real data.

Bibliography

[1] Jyoti Soni, Ujma Ansari, Dipesh Sharma, and Sunita Soni. Predictive data mining for medical diagnosis: An overview of heart disease prediction. *International Journal of Computer Applications*, 17(8):43–48, 2011.

[2] Riccardo Bellazzi, Fulvia Ferrazzi, and Lucia Sacchi. Predictive data mining in clinical medicine: a focus on selected methods and applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(5):416–430, 2011.

[3] Boris Milovic. Prediction and decision making in health care using data mining. *Kuwait chapter of arabian journal of business and management review*, 33(848):1–11, 2012.

[4] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109, 2001.

[5] Rahul C Deo. Machine learning in medicine. *Circulation*, 132(20):1920–1930, 2015.

[6] John S Zdanowicz. Detecting money laundering and terrorist financing via data mining. *Communications of the ACM*, 47(5):53–55, 2004.

[7] A Kusiak, KH Kernstine, JA Kern, KA McLaughlin, and TL Tseng. Data mining: medical and engineering case studies. In *Industrial Engineering Research Conference*, pages 1–7, 2000.

[8] Mai Shouman, Tim Turner, and Rob Stocker. Using data mining techniques in heart disease diagnosis and treatment. In *2012 Japan-Egypt Conference on Electronics, Communications and Computers*, pages 173–177. IEEE, 2012.

[9] Gustavo EAPA Batista and Maria Carolina Monard. An analysis of four missing data treatment methods for supervised learning. *Applied artificial intelligence*, 17(5-6):519–533, 2003.

[10] Thomas V Perneger and Bernard Burnand. A simple imputation algorithm reduced missing data in sf-12 health surveys. *Journal of clinical epidemiology*, 58(2):142–149, 2005.

[11] Ibrahim Berkan Aydilek and Ahmet Arslan. A hybrid method for imputation of missing values using optimized fuzzy c-means with support vector regression and a genetic algorithm. *Information Sciences*, 233:25–35, 2013.

[12] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[13] Marshall A Beddoe. Network protocol analysis using bioinformatics algorithms. *Toorcon*, 2004.

[14] Loris Nanni and Alessandra Lumini. Generalized needleman–wunsch algorithm for the recognition of t-cell epitopes. *Expert Systems with Applications*, 35(3):1463–1467, 2008.

[15] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

[16] Trivellore E Raghunathan, James M Lepkowski, John Van Hoewyk, and Peter Solenberger. A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey methodology*, 27(1):85–96, 2001.

[17] Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[18] José M Jerez, Ignacio Molina, Pedro J García-Laencina, Emilio Alba, Nuria Ribelles, Miguel Martín, and Leonardo Franco. Missing data imputation using statistical and machine learning methods in a real breast cancer problem. *Artificial intelligence in medicine*, 50(2):105–115, 2010.

[19] Evelyn Fix and Joseph Lawson Hodges. Discriminatory analysis. Nonparametric discrimination: consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3):238–247, 1989.

[20] Leif E Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009.

[21] Marco Di Zio, Mauro Scanu, Lucia Coppola, Orietta Luzi, and Alessandra Ponti. Bayesian networks for imputation. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 167(2):309–322, 2004.

[22] Dorian Pyle. *Data preparation for data mining*, volume 1. morgan kaufmann, 1999.

[23] Jay L Devore. *Probability and Statistics for Engineering and the Sciences*. Cengage learning, 2011.

[24] Peter J Rousseeuw and Annick M Leroy. *Robust regression and outlier detection*, volume 589. John wiley & sons, 2005.

[25] Antonio Loureiro, Luis Torgo, and Carlos Soares. Outlier detection using clustering methods: a data cleaning application. In *Proceedings of KDNet Symposium on Knowledge-based Systems for the Public Sector. Bonn, Germany*, 2004.

[26] Malik Agyemang, Ken Barker, and Rada S Alhajj. Mining web content outliers using structure oriented weighting techniques and n-grams. In *Proceedings of the 2005 ACM symposium on Applied computing*, pages 482–487. ACM, 2005.

[27] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246. ACM, 2002.

[28] Jim Gray, Surajit Chaudhuri, Adam Bosworth, Andrew Layman, Don Reichart, Murali Venkatrao, Frank Pellow, and Hamid Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data mining and knowledge discovery*, 1(1):29–53, 1997.

[29] George H John, Ron Kohavi, Karl Pfleger, et al. Irrelevant features and the subset selection problem. In *Machine learning: proceedings of the eleventh international conference*, pages 121–129, 1994.

[30] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10:66–71, 2009.

[31] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11. ACM, 2003.

[32] Salvador García, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez, and Francisco Herrera. Big data preprocessing: methods and prospects. *Big Data Analytics*, 1(1):9, 2016.

[33] Wei-Yin Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

[34] Vili Podgorelec, Peter Kokol, Bruno Stiglic, and Ivan Rozman. Decision trees: an overview and their use in medicine. *Journal of medical systems*, 26(5):445–463, 2002.

[35] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[36] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[37] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[38] Kevin P Murphy. Naive bayes classifiers. *University of British Columbia*, 2006.

[39] Howard B Demuth, Mark H Beale, Orlando De Jess, and Martin T Hagan. *Neural network design*. Martin Hagan, 2014.

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

[41] Fabien Lotte, Marco Congedo, Anatole Lécuyer, Fabrice Lamarche, and Bruno Arnaldi. A review of classification algorithms for eeg-based brain–computer interfaces. *Journal of neural engineering*, 4(2):R1, 2007.

[42] Tjen-Sien Lim, Wei-Yin Loh, and Yu-Shan Shih. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine learning*, 40(3):203–228, 2000.

[43] George AF Seber and Alan J Lee. *Linear regression analysis*, volume 936. John Wiley & Sons, 2012.

[44] Douglas M Bates and Donald G Watts. *Nonlinear regression analysis and its applications*, volume 2. Wiley Online Library, 1988.

[45] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[46] Pavel Berkhin et al. A survey of clustering data mining techniques. *Grouping multidimensional data*, 25:71, 2006.

[47] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[48] Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[49] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM Sigmod Record*, volume 25, pages 103–114. ACM, 1996.

[50] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. *Information systems*, 25(5):345–366, 2000.

[51] Hans-Peter Kriegel, Peer Kröger, Jörg Sander, and Arthur Zimek. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240, 2011.

[52] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[53] Alexander Hinneburg, Daniel A Keim, et al. An efficient approach to clustering in large multimedia databases with noise. In *KDD*, volume 98, pages 58–65, 1998.

[54] Wei Wang, Jiong Yang, Richard Muntz, et al. Sting: A statistical information grid approach to spatial data mining. In *VLDB*, volume 97, pages 186–195, 1997.

[55] Mohammed Javeed Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, Wei Li, et al. New algorithms for fast discovery of association rules. In *KDD*, volume 97, pages 283–286, 1997.

[56] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[57] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

[58] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[59] Kyoung-jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

[60] Eamonn Keogh, Jessica Lin, Ada Waichee Fu, and Helga VanHerle. Finding unusual medical time-series subsequences: Algorithms and applications. *IEEE Transactions on Information Technology in Biomedicine*, 10(3):429–439, 2006.

[61] Robert M Hirsch, James R Slack, and Richard A Smith. Techniques of trend analysis for monthly water quality data. *Water resources research*, 18(1):107–121, 1982.

[62] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. springer, 2016.

[63] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[64] Everette S Gardner. Exponential smoothing: The state of the art. *Journal of forecasting*, 4(1):1–28, 1985.

[65] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

[66] Christos Faloutsos, Mudumbai Ranganathan, and Yannis Manolopoulos. *Fast subsequence matching in time-series databases*, volume 23. ACM, 1994.

[67] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[68] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[69] Steffen Moritz, Alexis Sardá, Thomas Bartz-Beielstein, Martin Zaefferer, and Jörg Stork. Comparison of different methods for univariate time series imputation in r. *arXiv preprint arXiv:1510.03924*, 2015.

[70] A Rogier T Donders, Geert JMG van der Heijden, Theo Stijnen, and Karel GM Moons. A gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.

[71] Teresa A Myers. Goodbye, listwise deletion: Presenting hot deck imputation as an easy and effective tool for handling missing data. *Communication Methods and Measures*, 5(4):297–310, 2011.

[72] Frank J Molnar, Brian Hutton, and Dean Fergusson. Does analysis using "last observation carried forward" introduce bias in dementia research? *Canadian Medical Association Journal*, 179(8):751–753, 2008.

[73] Donald B Rubin. Multiple imputation after 18+ years. *Journal of the American statistical Association*, 91(434):473–489, 1996.

[74] Stef Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45(3), 2011.

[75] Paul Mac Berthouex and Linfield C Brown. *Statistics for environmental engineers*. Lewis publishers, 1994.

[76] Jiahua Chen and Jun Shao. Nearest neighbor imputation for survey data. *Journal of official statistics*, 16(2):113, 2000.

[77] Federica Barzi and Mark Woodward. Imputations of missing values in practice: results from imputations of serum cholesterol in 28 cohort studies. *American journal of epidemiology*, 160(1):34–45, 2004.

[78] Ziv Bar-Joseph, Georg K Gerber, David K Gifford, Tommi S Jaakkola, and Itamar Simon. Continuous representations of time-series gene expression data. *Journal of Computational Biology*, 10(3-4):341–356, 2003.

[79] Eamonn J Keogh and Michael J Pazzani. Derivative dynamic time warping. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pages 1–11. SIAM, 2001.

[80] Computation Biology. Dtw algorithm. `https://www.psb.ugent.be/cbd/papers/gentxwarper/DTWalgorithm.htm`.

[81] Émilie Poisson Caillault, Alain Lefebvre, André Bigand, et al. Dynamic time warping-based imputation for univariate time series data. *Pattern Recognition Letters*, 2017.

[82] Lukasz Ligowski and Witold Rudnicki. An efficient implementation of smith waterman algorithm on gpu using cuda, for massively parallel scanning of sequence databases. In *2009 IEEE International Symposium on Parallel & Distributed Processing*, pages 1–8. IEEE, 2009.

[83] Samuel David Pelaio Arcadinho. Model-based learning in multivariate time series. 2018.

[84] UCI. Epileptic seizure recognition data set. `https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition,`.