Université de Montréal

Exploratory and predictive methods for multivariate time series data analysis in healthcare

par

Adrien Andréas Aumon

Département de mathématiques et de statistique

Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en mathématiques

Orientation mathématiques appliquées

31 août 2022

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Exploratory and predictive methods for multivariate time series data analysis in healthcare

présenté par

Adrien Andréas Aumon

a été évalué par un jury composé des personnes suivantes :

Morgan Craig
(président-rapporteur)

Guy Wolf
(directeur de recherche)

Guillaume Lajoie
(membre du jury)

Résumé

Ce mémoire s'inscrit dans l'émergente globalisation de l'intelligence artificielle aux domaines de la santé. Par le biais de l'application d'algorithmes modernes d'apprentissage automatique à deux études de cas concrètes, l'objectif est d'exposer de manière rigoureuse et intelligible aux experts de la santé comment l'intelligence artificielle exploite des données cliniques à la fois multivariées et longitudinales à des fins de visualisation et de prognostic de populations de patients en situation d'urgence médicale. Nos résultats montrent que la récente méthode de réduction de la dimensionalité *PHATE* couplée à un algorithme de regroupement surpasse d'autres méthodes plus établies dans la projection en deux dimensions de trajectoires multidimensionelles et aide ainsi les experts à mieux visualiser l'évolution de certaines sous-populations. Nous mettons aussi en évidence l'efficacité des réseaux de neurones récurrents traditionnels et conditionnels dans le prognostic précoce de patients malades. Enfin, nous évoquons l'analyse topologique de données comme piste de solution adéquate aux problèmes usuels de données incomplètes et irrégulières auxquels nous faisons face inévitablement au cours de la seconde étude de cas.

Mots-clefs: *Santé, Apprentissage automatique, Données multivariées longitudinales, Visualisation, Prognostic*

Abstract

This thesis aligns with the trending globalization of artificial intelligence in healthcare. Through two real-world applications of recent machine learning approaches, our fundamental goal is to rigorously and intelligibly expose to the domain experts how artificial intelligence uses clinical multivariate time series to provide visualizations and predictions related to populations of patients in an emergency condition. Our results demonstrate that the recent dimensionality reduction tool PHATE combined with a clustering algorithm outperforms other more established methods in projecting multivariate time series in two dimensions and thus help the experts visualize sub-populations' trajectories. We also highlight traditional and conditional recurrent neural networks' proficiency in the early prognosis of ill patients. Finally, we allude to topological data analysis as a suitable solution to common problems related to data irregularities and incompleteness we inevitably face in the second case study.

Keywords: *Healthcare, Machine Learning, Multivariate Time Series, Visualization, Prognosis*

Contents

List of figures

Abbreviations

2D	2-dimensional, or 2 dimensions
AI	Artificial Intelligence
APP	Awake Prone Positioning
ARR	Annualized Relapse Rate
BGD	Batch Gradient Descent
ED	Euclidean Distance
CondRNN	Conditional Recurrent Neural Network
DDO/ddo	Days since Disease Onset
DTW	Dynamic Time Warping

EDSS	Expanded Disability Status Scale
GD	Gradient Descent
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
MBGD	Mini-Batch Gradient Descent
MDS	Multi-Dimensional Scaling
ML	Machine Learning
MRI	Magnetic Resonance Imaging
MS	Multiple Sclerosis
MTS	Multivariate Time Series
NCV	Nested Cross-Validation

NN	(k)-Nearest Neighbours
PCA	Principal Component Analysis
PH	Persistent Homology
PHATE	Potential of Heat-diffusion for Affinity-based Trajectory Embedding
RF	Random Forest
RNN	Recurrent Neural Network
RRMS	Relapsing-Remitting Multiple Sclerosis
SDE	Stochastic Differential Equation
SGD	Stochastic Gradient Descent
SPMS	(Active) Secondary Progressive Multiple Sclerosis

TDA	Topological Data Analysis
t-SNE	t-distributed Stochastic Neighbour Embedding
UMAP	Uniform Manifold Approximation and Projection for Dimension Reduction
UTS	Univariate Time Series

Remerciements

L'aboutissement de ce mémoire n'aurait sans nul doute jamais vu le jour sans l'implication de mon entourage tant sur les plans académiques que personnels. Cette faveur qui m'a été accordée était d'autant plus cruciale dans un contexte de crise sanitaire m'ayant contraint à effectuer la totalité de mes études de maîtrise à distance, sur le sol européen, me dispensant ainsi de tous les bénéfices d'une vie universitaire normale.

De ce fait, je remercie Prof. Guy Wolf, mon superviseur, d'être à l'origine de mon insertion dans la recherche au sein de l'Université de Montréal et de l'institut Mila, de savoir m'aiguiller activement dans mes travaux malgré la distance, et surtout de croire en mes aptitudes de recherche. Je remercie Dr Bastian Rieck d'avoir profité de ma présence en Europe afin de m'intégrer à l'institut de recherche en biologie computationnelle au centre Helmholtz à Munich, expérience unique ayant grandement élargi mes horizons de recherche. Par la même occasion, je remercie mes collaboratrices et collaborateurs pour le travail mutuel qui a été fourni dans les projets faisant l'objet d'une grande partie de mon mémoire, ainsi que le personnel académique contribuant au bon déroulement global de mes études.

Enfin, je n'oublierai jamais le soutien moral et social apporté par ma famille et mes amis proches qui, par le biais de partages, de confidences, de conseils, de confiance, et par-dessus tout d'amour plein et sincère, me conféraient sans cesse cet élan essentiel au passage de cette page importante de ma vie. Merci infiniment.

Introduction

With recent advancements in computing power, learning algorithms and availability of large datasets collected from medical records and wearable health monitors, artificial intelligence (AI) is booming in medicine and healthcare [**6**]. Indeed, through machine learning (ML), AI-based systems have the capacity to collect, examine, and derive conclusions from complex and numerous data that cannot be easily processed by humans. This can aid in patient mortality or disease prediction, which is, in the best-case scenario, a difficult assignment for physicians. For instance, AI can help to determine the likelihood of developing breast cancer by converting images of digital mammography into pixel-level variables which are unrecognizable to the human and combining them with clinical data and any known risk factors in order to limit the high number of predicted false positives that has been observed from usual screening predictions [**138**]. AI applications relate to an uncountable number of other medical tasks, including magnetic resonance imaging (MRI) segmentation [**7**], assistance in diagnostic radiology [**117**] or the promising Google Brain Project [**61**] for retinal eye diseases detection, making room for infinite possibilities.

This thesis aligns with this fast-growing machine learning trend in healthcare by focusing on the analysis of multivariate time series (MTS) medical data and their specific challenges. First, there is a urge to exploit data sampled over time as they are more and more available thanks to the worldwide spread of wearable intelligent devices such as smartwatches [**113**] capable of recording data without clinical supervision. Then, due to the inherent complexity of MTS data and an "AI chasm" limiting the clinical application of more advanced machine learning models [**74**], clinicians tend to conform to more established approaches. For instance, in data visualization, the common practice sticks with star charts, parallel coordinates or scatter plot matrices [**32, 144**] that require mental efforts to draw the right conclusions as the dimension grows, or that oversimplify the feature set to counterbalance the information overload [**25**]. Considering MTS as a whole may actually uncover some properties that are hardly found by breaking them into univariate time series (UTS) because UTS may not fully capture the inherent correlations between variables in MTS data [**150**]. To illustrate, MTS are able to reveal hidden correlations between multiple predictors and outcomes, because individual predictors might seem uncritical if considered independently, but in conjunction, can indicate an alarming condition [**25**]. Unfortunately, organizing multivariate time series data for presentation to a domain expert is a challenging task, because they are often numerous and human cannot mentally represent data in more than three dimensions [**60**]. In the context of data prediction, MTS do not naturally suit traditional predictive models in medicine such as logistic regressions that can either predict on UTS or single multivariate points, but do not handle MTS without feature hand-engineering [**59**].

Through two current real-world healthcare applications in collaboration with biomedical domain experts and other ML scientists, this thesis first intends to show how recent ML methods can help in the two above-mentioned important medical tasks, namely the visualization of complex MTS to improve domain knowledge, and the prediction of future medical events based upon multivariate historical data, useful in instant decision-making to ensure a future well-being. Then, to a wider extent, while applying conventional and more modern ML methods for the same purpose, we follow a comparative analysis with the aim of showing the benefits of recent ML advancements in healthcare.

In the first chapter, we will provide the theoretical foundations of our study with explanations about data visualization and predictions through machine learning, as well as a review of the models used in our applications. This part is essential to our thesis since we attach importance to the demystification of ML in a clinical context.

The second chapter is dedicated to a global application and evaluation of our proposed ML models on a hypoxaemic population of patients with COVID-19 who had been taken in charge in hospital centre to receive awake prone positioning (APP) for faster and safer recovery, and whose features measuring different aspects of their cardiorespiratory condition have been collected daily up to three days post-enrolment [**68, 135**]. On the one hand, we will demonstrate the efficiency of the recent dimensionality reduction method *Potential* *of Heat-diffusion for Affinity-based Trajectory Embedding* (PHATE) (see Section 1.7) in visualizing this time-varying database, with the intention of giving the experts new insights into the APP response of different groups of patients and potentially find distinguishing phenotypes that may predetermine treatment failure, that is, the intubation or the death within 28 days post-enrolment. On the other hand, we will conduct a comparative analysis of four different predictive models based upon k-Nearest Neighbours (see Section 1.4) and Recurrent Neural Networks (see Section 1.5) that perform "trackable" outcomes predictions on newly admitted patients' time series data from enrolment in order to provide an early prognosis system that could be used in clinical environment as an interesting decision-making support.

The final chapter is about the study of multiple sclerosis (MS) [**3**], a chronic disease well-known for its highly varied and unpredictable course over time [**51**]. Considering a database of sick patients whose magnetic resonance imaging (MRI) features and other MS symptoms have been recorded over the past decades, we will follow the same visualization framework as in the previous chapter and try to perform subtyping on common group progressions. However, the purpose of this chapter is rather to point out strong issues directly related to the database, such as mixed types of variables, missing data, unevenly spaced measurements or different time ranges, which restrict the production of interpretable visualizations and that are not easily handled by traditional data processing. After partially solving those issues thanks to a supervised version of PHATE using random forest proximities [**119**], this will be the perfect occasion to state a future direction through topological machine learning [**62**], an approach that bypasses frequent data irregularities by studying the global shape of the collected database instead of its traditional Euclidean vector representation.

Chapter 1

Theoretical background

1.1. Data analysis and machine learning

In practice, multiple data scientists each specializing in one aspect of data science conduct data-oriented projects. In this work, as scientists with a strong mathematical background, we try to bring our own expertise with the study and application of *machine learning*, one of most famous modern approaches to analyze data. Machine learning (ML), as stated in the 1950s by artificial intelligence pioneer Arthur Samuel, is “the field of study that gives computers the ability to learn without explicitly being programmed” [**122**]. It is of course a mean to analyze data, but this definition rather classifies machine learning as an application of artificial intelligence, a concept that refers to the capability of a computer system to mimic human cognitive functions such as learning and problem-solving. This aspect is what makes machine learning programs different from traditional ones, because their purpose is to extract knowledge from a wide variety of situations without being specifically designed for those situations. In other words, machine learning programs are prepared to solve the unknown, as people do when they drive in a foreign country, read a new book or solve a mathematical equation. To explain their particular ability, we need to dive into their construction. As ML models are every time motivated by goals of practical end-use, we are about to present two classic examples of predictive and exploratory problems encountered by humans and present how ML models are constructed around them.

1.2. Predictive machine learning

Prediction problems are one of the most common class of problems encountered by humans. Let us formulate a prediction problem to show how ML can help humans. Consider a couple who purchased a house a few years ago and has just given birth to triplets. Under this unforeseen situation, they are interested in selling their house for a bigger one. Before contacting the estate agency, they would like to estimate their house price by themselves to know if it is worth building an extension instead. Many parameters are known to determine house prices: square footage, number of rooms, has fitted kitchen or not, has garden or not, location and so on. Once collected, those features represent the data of the house, and the final price is the house label. Naturally, our couple, who does not know their house price, wants to make use of their house properties to get an estimation. In ML terms, they would like to *predict* their house label from their house data. However, sole house features are still not sufficient to produce a good price estimation because we have nothing to rely on. Usually, we also have to inspect the current housing market trend. Therefore, our couple collects features and prices of surrounding houses from the local newspaper; then, they compare their house properties with all the collected house ads and find a neighboring house with similar properties; finally, they use the latter to estimate their house price. It is definitely one way to estimate a house value, and likely what people tend to do. Still, it is not the only approach. One may believe that relying upon the most similar house does not give a realistic estimation because this house price could be biased by hidden factors, such as an overestimation of the owner. The couple should instead consider the five most similar houses and estimate the mean of their corresponding prices to produce an estimate. One may also wonder how the couple measures the similarity between two houses. For instance, suppose their house is 2400 square feet, has four rooms, fitted kitchen and a garden. One of the collected house is 2200 square feet, has four rooms, fitted kitchen and similar garden, and another is 2400 square feet, has four rooms, fitted kitchen but smaller garden. Which one of these two houses is the most similar to theirs? It is a rather difficult question, and we only take few house features into account.

In this situation, an appropriate *supervised* machine learning algorithm, a type of ML algorithms adapted for prediction problems, might help our couple to get a better price estimation. The process of a supervised ML system is analogous, but has the benefit of being automated and designed by ML experts from theoretical grounds. To be more accurate, a supervised ML model is defined as the resulting model from three equally important phases, as shown in Figure 1.1:

(1) the training phase considers a labeled data set, the *training* set, and aims to fit an algorithm to our current knowledge data in order to make future predictions. This is analogous to a student learning a course to prepare an examination;

(2) the validation phase considers another set of labeled data points, the *validation* data set, but does not make use of the known target variables to learn. Instead, it consists of inputting only the predictors into our previously learnt algorithm and comparing its output to the true targets. Usually, the learning algorithm of the training phase is built upon a set of parameters, called *hyper-parameters*, which is not internally optimized in the learning process and rarely leads to the best possible output model. Therefore, the validation phase indicates if we need to re-run the training phase with a different setting to output a better model. This is analogous to a student doing a pre-examination test to decide if it is worth re-learning the course with a new method before the final examination;

(3) the testing phase happens after the validation and picks a model which is supposed to be the best. It also outputs a performance score based on comparisons between predicted and true labels but is a mean to evaluate this final model as if it was the chosen one for future applications. This is equivalent to a student taking a final examination whose resulting grade will be the one appearing on the transcript and being subsequently used by other people to have an idea of the student's performance in this discipline.

After confirmation of its test performance and its relevance to the targeted task by domain experts, the final model is saved and ready to be used on new incoming data points during the deployment phase.

1.2.1. Nested Cross-Validation

Nested Cross-Validation (NCV) is a training-validation-evaluation framework that performs the above steps (1), (2) and (3) when setting up a predictive model, in a way that reduces biased validation and evaluation scores. It is based upon the encapsulation of

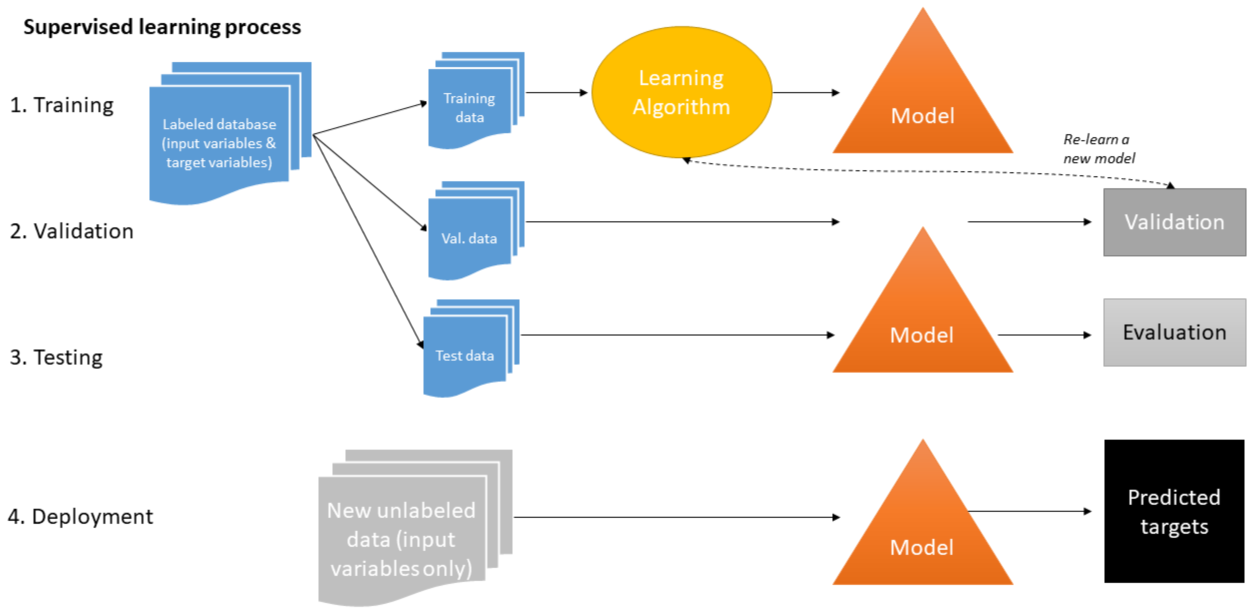


Fig. 1.1. Simplified scheme of the supervised learning process, inspired by a figure from [**71**].

two k-fold cross-validation processes [**20**] dedicated to the validation and the evaluation separately. It is motivated and works as follows.

On the one hand, to determine the best hyper-parameter combination that we need to define a model, we can perform a grid search. This is a brute force method that, given a model architecture, a training set, a validation set and a grid of all the possible hyper-parameters, launches the learning algorithm of the model on the training set and computes the validation loss on the validation set for each possible hyper-parameter combination to finally return the combination corresponding the lowest loss. While becoming very time consuming with large hyper-parameter grids, especially on complex models like neural networks, grid search prevails as the state of the art for hyper-parameter optimization because of its ease of execution and parallelization and its durability in low-dimensional spaces [**99**]. To reduce the bias introduced by a single train/validation split that does not use all of our observations for testing, we perform grid search in combination with a k-fold cross-validation [**20**], a technique that splits the available data into k equally sized subsets (or folds) such that each of the k subsets is used once as the validation set and the $k-1$ remaining subsets form together the training set. Thus, grid search is performed on each of those k training/validation splits and the reference score to determine the best hyper-parameter combination is now based upon the aggregation of the losses across the k possible validation splits.

On the other hand, we need to get insights into the performance of the model with its hypothetically optimized hyper-parameters as described above (see also the analogy in 1.2). k-fold cross validation is also popular when it comes to test a model, but we have to be careful when hyper-parameter optimization is also involved in the process. Indeed, if we just performed one k-fold CV to optimize hyper-parameters and measure performance at the same time, it would automatically output an overestimated performance score because we would test the model on "new" data that has already been observed by the model to optimize its hyper-parameters, so the concept of evaluating a model with a real-world situation is completely lost. To deal with this issue, we opt for a Nested Cross-Validation (NCV), a validation/test framework for supervised ML models that fits a model iteratively using a pair of nested loops, with the hyper-parameters adjusted to optimise a model selection criterion in the outer loop (model selection) and the parameters set to optimise a training criterion in the inner loop (model fitting/training) [**31**] (Figure 1.2[1]).

1.3. Exploratory machine learning

Unlike supervised machine learning that makes predictions for new data outcomes based on previous experience, *unsupervised* machine learning algorithms, another group of ML algorithms, generate knowledge from unlabeled data by themselves (Figure 1.3); they primarily help scientists to *explore* huge and complex data sets. To illustrate, let us take the classic example of customer segmentation [**126**]. Nowadays, the shopping ways of customers have dramatically changed, as they are more and more opting for E-commerce platforms. Compared with traditional sales, E-commerce has a unique characteristic that all the transaction information including the shopping time, items, and prices can be tracked and stored accurately. This allows E-commerce companies to make use of unsupervised ML models to extract knowledge from all this data in order to improve their selling strategies. In the cited paper, the author presents a ML process which allows companies to detect groups of customers based on their collected characteristics and behaviours

[1]by Casper Hansen: `https://mlfromscratch.com/nested-cross-validation-python-code/`

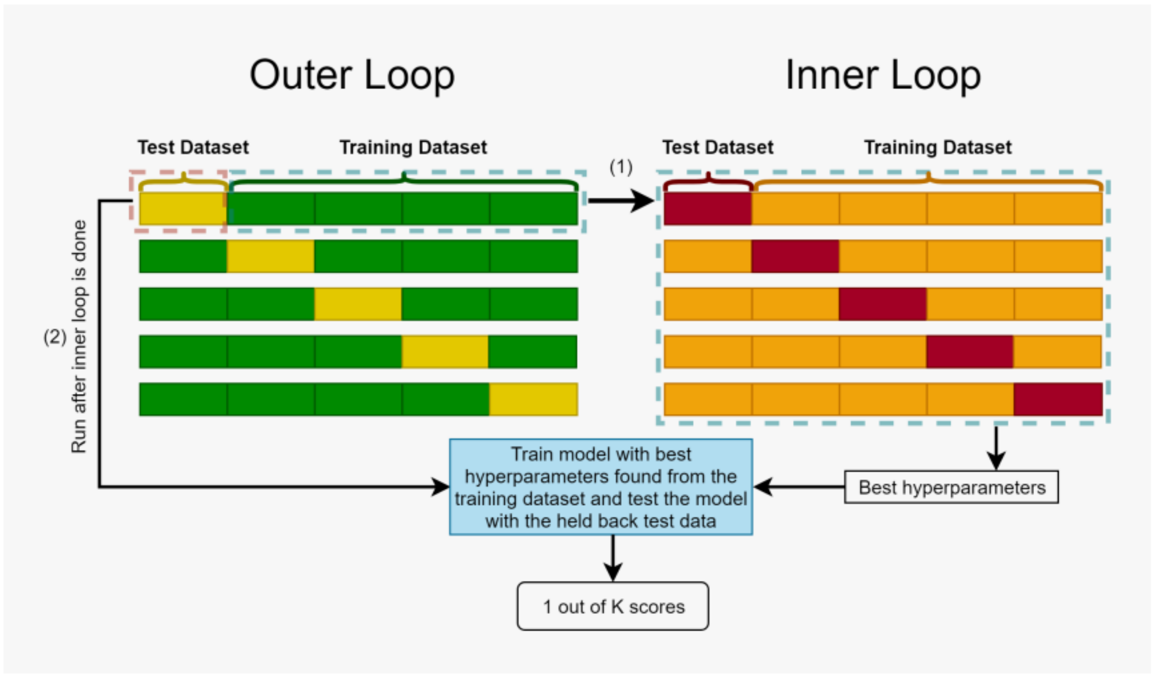

Outer Loop
Inner Loop
Test Dataset
Training Dataset
Test Dataset
Training Dataset
(1)
(2)
Run after inner loop is done
Train model with best hyperparameters found from the training dataset and test the model with the held back test data
Best hyperparameters
1 out of K scores

Fig. 1.2. Illustration of NCV. During one outer iteration, the inner loop optimizes the model's hyper-parameters on the first training data set with a grid search k-fold cross-validation and the outer loop tests the optimized model on the remaining test data set, giving us 1 out of K scores because the outer loop consists of K iterations (i.e. K different training-test splits of the data set). After the K outer iterations, the mean of those K scores gives an unbiased generalization performance of the selected model. It is important to note that the K scores from the outer loop are probably generated by different hyper-parameter combinations, which is precisely the purpose of NCV: we want to evaluate a model seen as an architecture rather than a combination of hyper-parameters.

and link each of them to preferences in terms of purchased products. Doing so by hand is unthinkable, considering the enormous amount of collected data. Thus, a well-designed unsupervised ML algorithm can provide a fast and efficient decision-making support for E-commerce companies.

Remark 1.3.1. *Sometimes, the labels of the data points may be partially or totally available. When the visualization task is led by those targeted labels, there exist supervised ML techniques taking those targets into account in their algorithm with the aim of improving the visualization (see Section 3.3.3 for a real-world example). In other words, not all exploratory ML methods are unsupervised, but the task remains the same.*

From now on, it is clear that these two classic examples distinguish supervised and unsupervised ML algorithms which are, in general, respectively designed to solve two types of problems encountered by humans: prediction and exploration. There are other types of

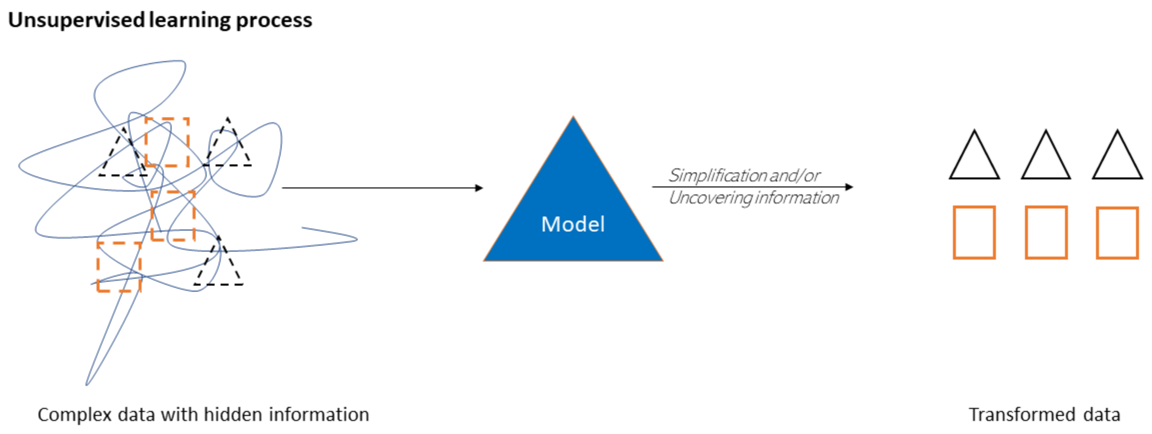

Unsupervised learning process
Model
Simplification and/or
Uncovering information
Complex data with hidden information
Transformed data

Fig. 1.3. Simplified scheme of the unsupervised learning process.

ML framework, but this work is mainly about those ones. The next sections are about to review all the main ML methods that are used in our biomedical applications. In order to be concise and theoretically accurate in our ML algorithm descriptions, we assume some underlying mathematical knowledge, including standard notation, differential analysis, vector and linear algebra as well as probability.

1.4. k-Nearest Neighbours

k-Nearest Neighbours (kNN) models are simple yet widely used supervised ML algorithms [**81**] when the goal is to predict a target variable of new data instances based on previously known data, as in our house price prediction example in Section 1.2. This is probably the most straightforward predictive model because it does not implement any learning algorithm, making the learning phase trivial. Indeed, it considers that there is no need to extract complex relationships when we can just memorize the entire training dataset as the model. In other words, it assumes that a new incoming data point x^* can be labeled only with the help of the training set $\{x_1, \dots, x_m\}$ and their labels $\{y_1, \dots, y_m\}$.

Assuming all our working database has been translated to N-dimensional vectors such that each component represents an attribute, the kNN labeling of a new point $x^* \in \mathbb{R}^N$ first needs to set k to a predefined integer value along with a measure of proximity $s : \mathbb{R}^N \times \mathbb{R}^N \longrightarrow \mathbb{R} \geq 0$ between our data points. Then, the k-neighbourhood of x^*, $\mathcal{N}_k(x^*)$, consisting of the k nearest neighbours of x^*, is computed according to s. Finally,

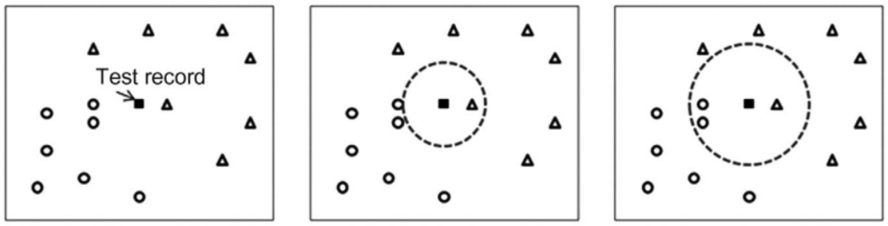


Fig. 1.4. kNN prediction on 2-dimensional synthetic data, with Euclidean distance (ED) and majority vote label assignment for different values of k (from [**81**], Chapter 4). Left: new data point to be labeled “triangle” or “circle”. Center: computation of the $k = 1$-ED-neighbourhood. The predicted label is “triangle”. Right: computation of the $k = 3$-ED-neighbourhood. The predicted label is “circle”.

a label is assigned to x^* thanks to the labels of points in its neighbourhood $\mathscr{N}_k(x^*)$ according to a predefined labeling assignment algorithm. Each combination of k, s and assignment algorithm is arbitrary and defines a unique kNN model. They are typical examples of hyper-parameters as mentioned in Section 1.2, and thus could be optimized through validation depending on our specific needs. In Figure 1.4, we show how varying the hyper-parameter k of a kNN approach affects the output label of a test record, while the other hyper-parameters, namely s and the label assignment algorithm, are respectively fixed to Euclidean distance (ED) and majority vote.

This flexibility in terms of hyper-parameters makes kNN able to suit many other types of input and output data. For instance, if $\{y_1, \ldots, y_m\}$ were continuous instead of discrete labels as shown in Figure 1.4, a labeling assignment method such as the mean label of the k-neighbourhood would provide a consistent continuous output. This is particularly useful when the expected label of a new point is a house price as in Section 1.2, or a probability as in our application in Chapter 2. Moreover, one of the most important kNN’s benefits is its ability to handle time series data, the major axis of our study. This is made possible by setting a proximity measure s valid on sequences of N-dimensional vectors instead of single N-dimensional points. Several measures, with their own strengths and weaknesses, have already been proposed [**4**], including the ED matching which extends the standard ED to time series by summing the distances between pairs of points recorded at the same time step.

While being intelligible and easily implemented, the latter proximity measure and its variants suffer from several drawbacks that make inappropriate their use in certain applications [**30**] : it only suits time series of the same length; it does not handle outliers or noise; it is very sensitive to six signal transformations: shifting, uniform amplitude scaling, uniform time scaling, uniform bi-scaling, time warping and non-uniform amplitude scaling. In other words, ED would not capture time series of similar shapes if they were subject to the latter transformations. This is why we want to briefly introduce Dynamic Time Warping (DTW) [**76**], a proximity measure being heavily used in our next applications thanks to its robustness to the similarity computation, such as its suitability to perform early predictions on truncated time series (Chapter 2). DTW is defined as follows: given two time series Q and C, of lengths n and m, respectively, where

$$\mathrm{Q} = q_1, q_2, \ldots, q_i, \ldots, q_n$$
$$\mathrm{C} = c_1, c_2, \ldots, c_j, \ldots, c_m,$$

we first construct an n-by-m matrix where the $\left(i^{\text{th}}, j^{\text{th}}\right)$ element of the matrix contains the distance $d\left(q_i, c_j\right)$ between the two points q_i and c_j (i.e. usually $d\left(q_i, c_j\right) = \|q_i - c_j\|_N^2$ for N-dimensional sequences). Each matrix element (i, j) corresponds to the alignment between the points q_i and c_j. This highlights the big difference between ED and DTW: while ED restricts to (i, i) alignments, DTW allows one-to-many matching (Figure 1.5 [**4**]). A warping path W is a contiguous (in the sense stated below) set of matrix elements that defines a mapping between Q and C (Figure 1.6 [**76**]). The k^{th} element of W is defined as $w_k = (i, j)_k$. Thus, we have

$$W = w_1, w_2, \ldots, w_k, \ldots, w_{\mathrm{K}} \quad \max(m, n) \leq \mathrm{K} < m + n - 1$$

The following conditions ensure that a warping path captures a "correct" sequences of (q_i, c_j) with respect to time, otherwise the time aspect of a time series comparison would be completely lost:

(1) boundary conditions: $w_1 = (1, 1)$ and $w_{\mathrm{K}} = (m, n)$, i.e. the warping path starts and finishes in diagonally opposite corner cells of the matrix. This avoids the possibility that the time warping degenerates to a tiny part of the sequences;

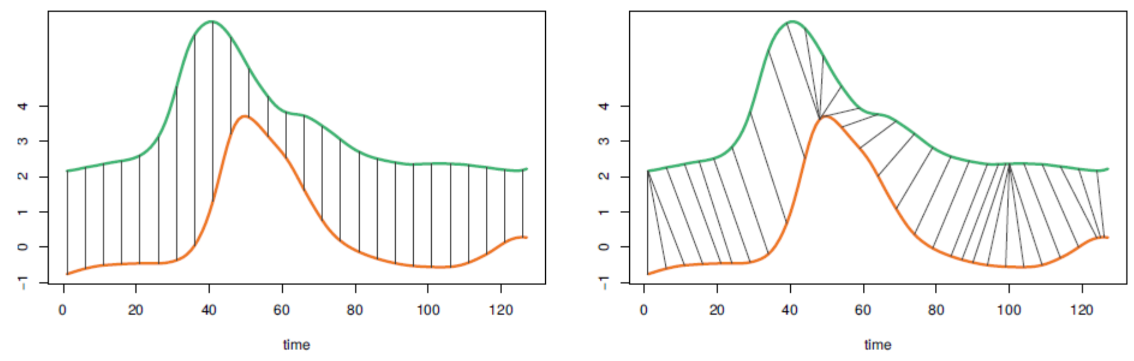


Fig. 1.5. Comparison of ED (left) and DTW (right) distances between the same two time series in green and orange. ED computes the distance between the two time series by summing the time-to-time distances between matching points, while DTW allows time distortions. With its restricted alignment, ED misses the similar shape shared by those two time series.

(2) continuity: given $w_k = (a, b)$, then $w_{k-1} = (a', b')$, where $a - a' \leq 1$ and $b - b' \leq 1$. This restricts the allowable steps in the warping path to adjacent cells (including diagonally adjacent cells), as to avoid "time jumps";

(3) monotonicity: given $w_k = (a, b)$, then $w_{k-1} = (a', b')$, where $a - a' \geq 0$ and $b - b' \geq 0$. This ensures that the warping path does not go backwards in time.

Finally, if $\mathbf{W}$ is the set of all possible warping paths between Q and C, the DTW distance between Q and C is obtained from $W \in \mathbf{W}$ minimizing the warping cost $\sqrt{\sum_{k=1}^{\mathrm{K}} w_k}$:

$$s_{DTW}(\mathrm{Q}, \mathrm{C}) = \min_{W \in \mathbf{W}} \left\{ \sqrt{\sum_{k=1}^{\mathrm{K}} w_k} \right\} .$$

Remark 1.4.1. *Under the above constraints (1), (2) and (3), DTW remains symmetric* [**83**], *that is,* $s_{DTW}(\mathrm{Q}, \mathrm{C}) = s_{DTW}(\mathrm{C}, \mathrm{Q})$, *so DTW provides an intuitive proximity between time series objects and can be used similarly to ED matching in a wide range of applications.*

1.5. Recurrent neural networks

A recurrent neural network (RNN) [**45**][2] is also a machine learning model that aims to predict the labels of new data points. RNNs are similar to traditional feedforward neural networks, except that they handle a memory for sequential inputs by connecting hidden

[2]In this work, we focus on the use of RNNs in the context of supervised classification. In reality, RNNs are useful for many other tasks, even in unsupervised setting [**78**].

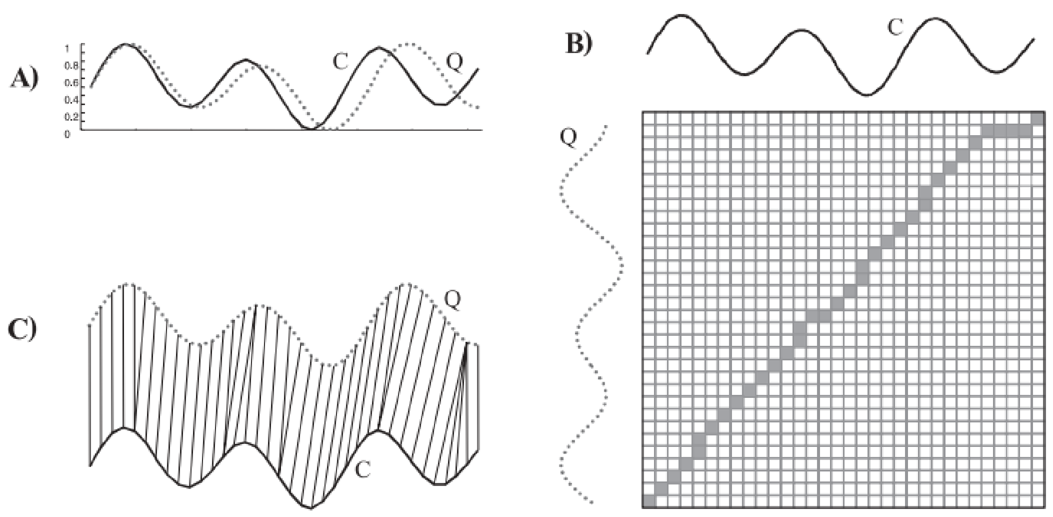


Fig. 1.6. Illustration of the Dynamic Time Warping distance computation between two 1-dimensional time series Q and C. A) Two sequences Q and C that are similar but out of phase. B) To align the sequences, we construct a warping matrix and search for the optimal warping path, shown with solid squares. C) The resulting alignment.

states associated with each time step. As opposed to kNN which directly learns from the data itself thanks to its spatial structure (non-parametric model), a RNN incorporates internal parameters that are optimized through a learning algorithm on the training data set before making predictions on new instances (parametric model). Indeed, the output of a RNN depends on the network weights that are far from being optimal initially because they are previously randomly initialized to produce first estimations. Formally, in the context of classification, we consider RNNs whose outputs $\{o_t\}_{t=0}^{T}$ are computed as[3]

$$o_t = \phi(W_{ho}h_t) \tag{1.5.1}$$

where:

- $h_t = \Phi(W_{xh}x_t + W_h h_{t-1})$;
- $h_{-1} \in \mathbb{R}^{n_{neurons}}$ is the initial hidden state (usually random or null);
- ϕ and Φ are (non-linear) differentiable functions;
- $x_t \in \mathbb{R}^{n_{features}}$ is the t-step input vector;
- $W_{xh} \in \mathbb{R}^{n_{neurons} \times n_{features}}$;
- $W_h \in \mathbb{R}^{n_{neurons} \times n_{neurons}}$;

[3]Internal biases are omitted for simplification.

- $W_{ho} \in \mathbb{R}^{n_{labels} \times n_{neurons}}$;
- $(n_{features}, n_{neurons}) \in \mathbb{N}^2$ are respectively the number of input features at each time step t and the number of neurons in the hidden states of the RNN.

A visual representation of the RNN architecture and its equations is illustrated in Figure 1.7. Thus, the goal of the learning process is to update the weight matrices W_{xh}, W_h

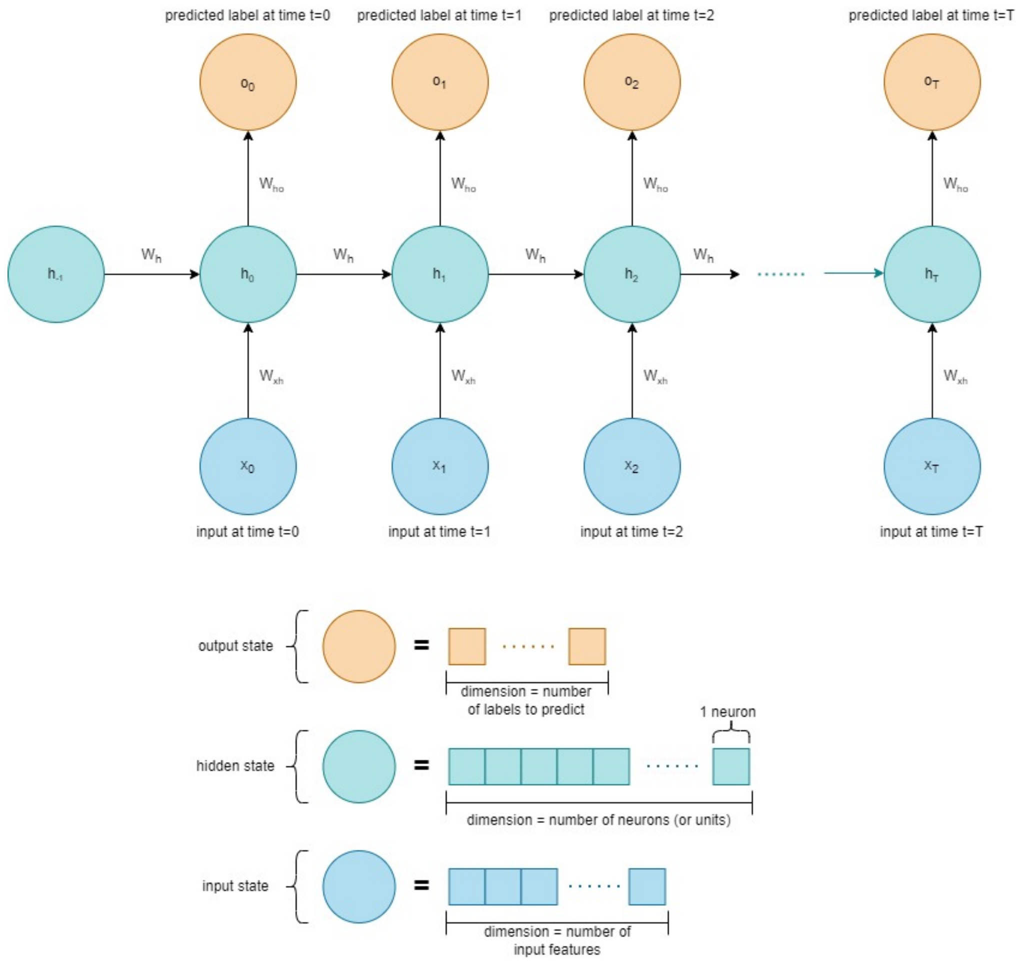

predicted label at time t=0
predicted label at time t=1
predicted label at time t=2
predicted label at time t=T
o0
o1
o2
oT
Who
h-1
Wh
h0
h1
h2
hT
Wxh
x0
x1
x2
xT
input at time t=0
input at time t=1
input at time t=2
input at time t=T
output state
dimension = number of labels to predict
1 neuron
hidden state
dimension = number of neurons (or units)
input state
dimension = number of input features

Fig. 1.7. Schematic illustration of a basic RNN architecture, without the biases for simplification. The trainable parameters are the weight matrices $W_{xh} \in \mathbb{R}^{n_{neurons} \times n_{features}}$, $W_h \in \mathbb{R}^{n_{neurons} \times n_{neurons}}$ and $W_{ho} \in \mathbb{R}^{n_{labels} \times n_{features}}$. Except h_{-1} that needs to be initialized, each hidden state h_t depends not only upon the input, but also the previous hidden state: $h_t = \Phi(W_{xh}x_t + W_h h_{t-1})$. Φ is the hidden activation function, usually a differentiable non-linear function to allow more complex relationships compared to basic regressions. The output cells take the value $o_t = \phi(W_{ho}h_t)$ where ϕ is a differentiable function defined to fit the expected labels.

and W_{ho} such that the output of the network after inputting each training instance better approximates their labels, in the sense that the new weights minimize a loss function $L =$

$L\left(W_{xh}, W_h, W_{ho}\right)$ predefined according to the targets of the case study. This is done in an iterative gradient descent optimization process in which we compute the partial derivative of the loss function with respect to each individual weight at the current configuration of the RNN. This computation indicates the direction of each individual weight update: if the slope is negative, we increase the weight to “go downhill", otherwise we have to decrease it to “go uphill". In matrix notation, we iteratively update weight matrices W_{xh}, W_h and W_{ho},

$$\begin{aligned} W_{xh} &\leftarrow W_{xh} - \lambda \frac{\partial L}{\partial W_{xh}} \\ W_h &\leftarrow W_h - \lambda \frac{\partial L}{\partial W_h} \\ W_{ho} &\leftarrow W_{ho} - \lambda \frac{\partial L}{\partial W_{ho}} \end{aligned}$$

where $\lambda \in \mathbb{R}^+$ is the learning rate that weight the size of the updating step. Gradients are computed with Backpropagation Through Time (BPTT) [**146**], which is the time dependent equivalent of the classic backpropagation for FNNs. Since BPTT precisely demonstrates how a RNN learns from data, we want to provide more intelligible insights regarding those computations because we were very unsatisfied by the inconsistency in notation and explanations across the existing literature about BPTT. First, let us define

$$L = \sum_{t=0}^{T} \ell(o_t)$$

which is the sum of the step-wise prediction losses of the system given at the current weight configuration after one forward pass of a single input, and let us omit biases for simplification. If f is, as usual, the summation function, and if ϕ is the activation function of the output layer, we can write $o_t = \phi(f(W_{ho}, h_t))$. Note that W_{ho} does not depend on time because weights are shared in the same layer, no matter the time step. Thus,

$$L = \sum_{t=0}^{T} \ell(\phi(f(W_{ho}))$$

The main ingredient in BPTT is to successively apply the chain rule backwards thanks to the dependence between a state and the previous ones. Here, by applying the chain rule twice, we have

$$\frac{\partial L}{\partial W_{ho}} = \sum_{t=0}^{T} \frac{\partial \ell}{\partial \phi} \cdot \frac{\partial \phi}{\partial f} \cdot \frac{\partial f}{\partial W_{ho}}$$

Since f is the linear combination of the previous hidden state involving the weights W_{ho}, that is, $f = W_{ho}h_t$, we can re-write the last factor:

$$\frac{\partial L}{\partial W_{ho}} = \sum_{t=0}^{T} \frac{\partial \ell}{\partial \phi} \cdot \frac{\partial \phi}{\partial f} \cdot h_t.$$

All the partial derivatives of the above expression are computable, because each function in our RNN architecture are supposed to be differentiable. For instance, in binary classification, we use the binary cross entropy and the sigmoid functions as the loss and output activation functions [**128**], and the hidden activation functions are commonly the sigmoid or the hyperbolic tangent [**42**]. We can also easily evaluate them at the current weight configuration because we already know the outputs, the summation and the hidden states at each time step after our first forward pass. This is why BPTT is particularly efficient: it remembers each computation at each time step, so we do not have to perform useless computations. We can repeat the process to compute $\frac{\partial L}{\partial W_h}$ and $\frac{\partial L}{\partial W_{xh}}$ by applying the chain rule more times to go deeper backwards into the network.

However, one could notice that some parameters are not learnt through BPTT. They are basically hyper-parameters (see Section 1.2), and the resulting trained model may be sub-optimal since they directly affect the learning process. An important hyper-parameter is the above-mentioned learning rate λ, whose a too small value may lead to slow convergence of the gradient descent process while a too big value may cause the loss function to fluctuate and get stuck in a local minimum or even to diverge [**13, 17, 24**]. Another hyper-parameter that does not appear in the above discussion is the batch size. When learning through BPTT, we can either update the weights after a forward pass of a single training instance, as we did in our example (Stochastic Gradient Descent, SGD), multiple training instances (Mini Batch Gradient Descent, MBGD) or the whole training set (Batch Gradient Descent, BGD). When considering multiple instances, the loss function now takes the average of the each output loss into account. To be more accurate in our notation, if $\mathbf{B}$ is the set of all batches of equal size, if B is one of those batches containing patients' indices sampled from the data set $\mathbf{X}$, if $|B|$ is the batch size hyper-parameter setting the size of all the batches, and if o_t^i is the t-step output of patient i after a feedforward pass into the RNN with the current weight configuration, we update network weights given the loss function

$$L_B = \frac{1}{|B|} \sum_{i \in B} \left(\sum_{t=0}^{T} \ell(o_t^i) \right)$$

and we re-iterate the update until all the batches in $\mathbf{B}$ have been covered with the aim of minimizing the global loss

$$L_{global} = \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=0}^{T} \ell(o_t^i) \right)$$

where $N = |\mathbf{X}|$ is the number of training sequences. The batch size has to be carefully chosen because it also directly affects the learning process. Indeed, BGD is very time consuming but ensures convergence of the loss function to a local minimum, whereas SGD is faster but introduces noise in the weight setting [**29**], so an in-between with MBGD is often needed. There is also number of epochs, corresponding to the number of complete passes through the training dataset. Even the learning algorithm can be seen as a hyper-parameter because recent optimization methods has been proposed to improve the convergence of the classic BPTT. For instance, one of the most recent ones is Adam [**79**], in which the learning rate is adapted for each weight parameter by running averages of both the gradients and the second moments of the gradients.

Of course, we do not want to overwhelm our thesis with the optimization of every possible hyper-parameter. During our application in Section 2.4.3, we end up with a compromise by following common practice in the existing literature in our hyper-parameters optimization.

1.6. k-means and k-medoids clustering

Clustering is an unsupervised machine learning process that seeks to find groupings, or *clusters*, in data in such a way that data points within a cluster are more *similar* (in the sense of a well-defined (dis)similarity measure [**53**]) to each other than to data points in the other clusters [**12**]. It is particularly useful for large data sets described with many features whose groupings are not easily detectable by hand, as in the customer segmentation of Section 1.3 where we intuitively presented an example of clustering application. In this work, we focus on *exclusive partitioning*, the most common and intelligible form of clustering approach that assigns each data point to an exclusive cluster. As one of the most popular exclusive partitioning methods ([**81**], p. 226), k-means [**88**] technique identifies clusters based on central prototype records. In k-means, the prototype record of a cluster is its $centroid$, defined as the mean of its data points. Given a database $X = \{x_1, \ldots, x_n\}$, this method aims to find a partition of $k \leq n$ clusters $\mathbf{C} = \{C_1, \ldots, C_k\}$ minimizing the sum of squared error (SSE) of all data points in a cluster to its centroid. That is, the partitioning $\mathbf{C}$ is the solution to the minimization problem

$$\arg\min_{\mathbf{C}} \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - c_i\|_2^2$$

where c_i is the cluster i's centroid. This objective function may also be viewed as the attempt to minimize the variance of the Euclidean distance of the points to their nearest cluster centers [**21**], giving the output clusters a meaningful geometric property. The traditional algorithm starts by initializing k random centroids $c_1, \ldots, c_k$ in the database X, and repeats the following steps until the centroids/clusters stabilize (i.e., do not change):

(1) Assignment: perform 1-NN classification to (re)assign each $x \in X$ to one of the cluster centroids.

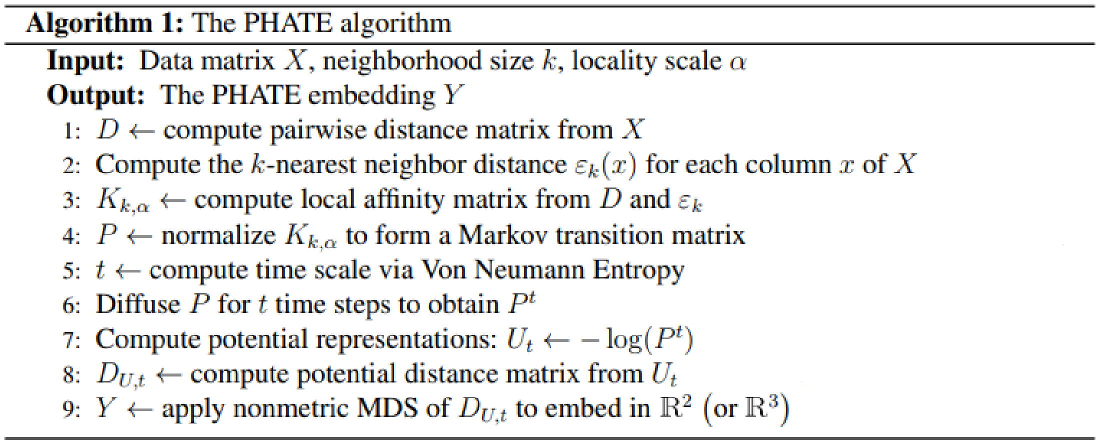
(2) Update: recompute each c_j to be the centroid of all the points assigned to its cluster.

While convergence is guaranteed by SSE decreasing at each update, this often leads to a local optimum highly depending on the initialized centroids ([**81**], p. 232), so multiple runs are often performed to alleviate the risk of having badly representative clusters.

This clustering approach based upon iterative updates of a central prototype makes room for derived methods such as k-medoids [**1**]. In contrast to the k-means algorithm, k-medoids chooses actual data points as central prototypes (named *medoids*), and thereby allows for even greater interpretability of the cluster centers than in k-means, where the center of a cluster is not necessarily one of the input data points. The algorithm is very similar, except that c_i are now medoids and their update is performed by computing the SSE induced by swaping c_i with points in the associated cluster and selecting the one with the lowest score. In the context of our study, we mostly use k-medoids instead of k-means because it increases the visualization ability of our application in Section 2.3.3 by allowing the projection of the medoids into the 2-dimensional PHATE (see Section 1.7) embedding space, which is impossible with k-means since its high dimensional centroids do not have natural 2D counterparts. Moreover, k-medoids better suits the analysis of time series data because its objective function extends to non-metric distances such as DTW (see Section 1.4). Finally, other than those specific needs, k-medoids is generally more robust to outliers, noise and initialized centroids [**11**], while keeping the high interpretability of the k-means algorithm.

1.7. Potential of Heat-diffusion for Affinity-based Trajectory Embedding

Potential of Heat-diffusion for Affinity-based Trajectory Embedding (PHATE) [**102**] is an unsupervised ML model specifically designed to provide a denoised, two or three-dimensional visualization of high-dimensional branching progression structure which is often non-linear and arises from underlying biological processes such as cell differentiation. PHATE defines this high-dimensional branching structure as multiple one-dimensional manifolds (i.e., trajectory curves) that cross each other. The PHATE algorithm is presented in Figure 1.8. Let $X = \{x_1, \ldots, x_N\}$ be the set of high-dimensional original vectors and $Y = \{y_1, \ldots, y_N\}$ be the PHATE output of corresponding 2-dimensional (or 3-dimensional) embedding vectors. The succession of all the steps in Figure 1.8 defines the PHATE embedding function f that maps X onto Y: $Y = f(X)$.



```
Algorithm 1: The PHATE algorithm
  Input: Data matrix $X$, neighborhood size $k$, locality scale $\alpha$
  Output: The PHATE embedding $Y$
   1: $D \leftarrow$ compute pairwise distance matrix from $X$
   2: Compute the $k$-nearest neighbor distance $\varepsilon_k(x)$ for each column $x$ of $X$
   3: $K_{k,\alpha} \leftarrow$ compute local affinity matrix from $D$ and $\varepsilon_k$
   4: $P \leftarrow$ normalize $K_{k,\alpha}$ to form a Markov transition matrix
   5: $t \leftarrow$ compute time scale via Von Neumann Entropy
   6: Diffuse $P$ for $t$ time steps to obtain $P^t$
   7: Compute potential representations: $U_t \leftarrow -\log(P^t)$
   8: $D_{U,t} \leftarrow$ compute potential distance matrix from $U_t$
   9: $Y \leftarrow$ apply nonmetric MDS of $D_{U,t}$ to embed in $\mathbb{R}^2$ (or $\mathbb{R}^3$)
```

Fig. 1.8. Step-by-step summary of the PHATE algorithm [**102**]. In the context of our work, we also have the diffusion power t as input in addition to k and α.

Let us point out additional information about what each of those steps is actually processing.

(1) D is simply the Euclidean distance matrix of X.

(2) For each vector point x in X, we compute the distance between x and its k-nearest neighbor, namely $\varepsilon_k(x)$, and save them to be used in step 3.

(3) $K_{k,\alpha}(x, y)$ transforms distances D into affinities (the "Affinity" of PHATE) similarly to a Gaussian kernel in order to capture the local proximity between our data points. More exactly, given two original data points x, y,

$$K_{k,\alpha}(x, y) = \frac{1}{2} \exp\left(-\left(\frac{\|x-y\|_2}{\varepsilon_k(x)}\right)^{\alpha}\right) + \frac{1}{2} \exp\left(-\left(\frac{\|x-y\|_2}{\varepsilon_k(y)}\right)^{\alpha}\right)$$

where $\frac{1}{2}\exp\left(-\left(\frac{\|x-y\|_2}{\varepsilon_k(x)}\right)^{\alpha}\right)$ and $\frac{1}{2}\exp\left(-\left(\frac{\|x-y\|_2}{\varepsilon_k(y)}\right)^{\alpha}\right)$ are equally weighted terms derived from the traditional Gaussian kernel $K_\varepsilon(x, y) = \exp\left(-\|x-y\|^2/\varepsilon\right)$ incorporating a locally adaptive bandwidth $\varepsilon_k(\cdot)$ that emphasizes on local neighbourhood without being sensitive to points in sparse data regions, and a rate of decay of the tails α that increases the discrimination between points within and outside the bandwidth $\varepsilon_k(\cdot)$ to counterbalance very similar affinities due to big values of $\varepsilon_k(\cdot)$.

(4) We compute

$$P(x, y) = P_{k,\alpha}(x, y) = \frac{K_{k,\alpha}(x, y)}{\nu_{k,\alpha}(x)}$$
$$\text{such that } \nu_{k,\alpha}(x) = \sum_{z \in X} K_{k,\alpha}(x, z)$$

for each couple $(x, y) \in X \times X$ to form a valid Markov transition matrix P where the probability of moving from x to y in a single time step is given by $\Pr[x \to y] = P_{k,\alpha}(x, y)$. We call P the *diffusion operator* (see step 6).

(5) This optional step relates to the optimization of t which powers the diffusion operator P^t using Von Neumann Entropy [**10**]. Similarly to k and α, we can also consider t as another hyperparameter to fit specific needs, as in Section 2.3.2.

(6) We raise the diffusion operator P to the power of t in order to spread the Markov chain so that $P^t_{k,\alpha}(x, y)$ is now the probability of moving from x to y in t time steps. As k and α, t is a hyper-parameter to tweak according to the case study. In particular, if we want to learn more about the global structure without restricting ourselves by the locality from our single-time step affinity-based probabilities, we should set larger values for t. This step refers to the "Heat-diffusion" in PHATE,

because it can be intuitively thought of as the spread of heat in a room going from a warm source (high affinities) to a less warm area (low affinities), which can be modeled mathematically as the heat equation and whose solution is the heat kernel [**105**]. Through the powering process, small probabilities are quickly reduced to zero, providing a means of filtering or denoising the learned manifold.

(7) We take the negative log of the powered diffusion operator $U_t = -\log(P^t)$ that is used and justified in the next step. These potential representations, inspired by information theory, refers to the "Potential" in PHATE.

(8) Instead of probability-based distances contained in P^t, we make use of the *potential distance* matrix $D_{U,t}$ defined as $D_{U,t}(x,y) = \|U_x^t - U_y^t\|_2$, where $U_x^t = -\log(p_x^t)$ refers to the row x of U_t. These alternative distances intend to stabilize the embedding near the boundaries, because $D_{U,t}(x,y)$ is sensitive to differences in both the tails and the more dense regions of the diffused probabilities, resulting in a distance which preserves both local and global relationships. To illustrate, in the synthetic example of Figure 1.9 [**102**], we observe that embedding points of the half circle's extremities are more expanded in the PHATE embedding than in the Diffusion Maps [**41**] which directly makes use of diffusion distances. Thus, we avoid embedding branches to be squeezed and hardly detectable.

(9) The 2-dimensional (or 3-dimensional) embedding vectors Y are finally obtained by using the potential distances as input for a non-metric multi-dimensional scaling (MDS) [**82**], an optimization process that, given initial embedding candidates $\{\hat{x}_1, \ldots, \hat{x}_N\}$ obtained randomly or with classical MDS [**84**], refines this embedding by minimizing a stress function through an iterative process until a predefined threshold. In PHATE, the selected stress function is the popular Kruskal normalized stress 1 [**84**]

$$\mathrm{Stress}_1\left(\hat{x}_1, \ldots, \hat{x}_N\right) = \sqrt{\sum_{i,j} \left(f(D_{U,t}(x_i, x_j)) - \|\hat{x}_i - \hat{x}_j\|\right)^2 / \sum_{i,j} \|\hat{x}_i - \hat{x}_j\|^2}$$

minimized over homologous 2D or 3D vectors $\hat{x}_i$ of our original data points and weakly monotone relations $f : \mathbb{R} \longrightarrow \mathbb{R}$ between potential distances and embedded Euclidean distances. This shows why PHATE makes use of this non-metric MDS instead of a traditional MDS. Indeed, classical MDS assumes that the distances between high-dimensional points must be strictly equal to their homologous low-dimensional distances, which may be overly restrictive. From the stress function, we observe that Euclidean distances in the 2D or 3D PHATE embedding do not necessarily reflect distances between original data points because this is not what we are looking for. Instead, PHATE visually recovers (dis)similarities through the connected tree structure of its output: data instances sharing very similar features should belong to the same branch, more distant points should be located on other branches and so on.

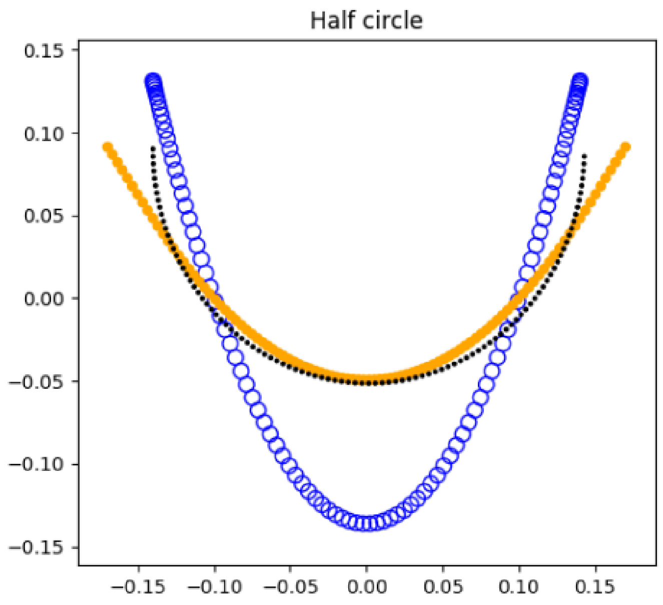


Fig. 1.9. Comparison of Diffusion Maps (blue) and PHATE (orange) embeddings on data (black) from a half circle, which naturally contains two endpoints, using classical MDS for both embedding methods. The Diffusion Maps embedding exhibits instabilities that generate significantly higher densities near the two end points. Thanks to the use of the potential distance matrix $D_{U,t}$ instead of diffusion distances as in Diffusion Maps, PHATE stabilizes embedding densities, resulting in a visualization that does not suffer from squeezed regions.

The algorithm, as described above, has 3 hyper-parameters k, α, and t that are not meant to be theoretically optimized, but rather specifically set to balance between the local and global structure we want to capture in the final embedding in the context of a case study. The diffusion process described in steps (6), (7) and (8) has mathematical foundations similar to Diffusion Maps. It has been shown in several works that manifold geometries are closely related to heat diffusion such as differential Laplace-Beltrami operators [**69**]. Indeed, solutions of the heat equation over a manifold capture its intrinsic properties, while providing embeddings, affinities, and distance metrics that capture intrinsic manifold relations. It has further been shown that these can be robustly discretized for empirical observations that correlate with hidden (or latent) manifold models, e.g., by considering diffusion maps embedding of the data [**103, 104**]. PHATE extends the Diffusion Maps approach by considering an underlying geometry consisting of multiple one-dimensional manifolds that cross each other while alleviating boundary-condition instabilities (see Figure 1.9). This is done by assuming that the distribution p of the data X is the steady state solution of the stochastic differential equation (SDE) $\dot{x} = -\nabla U(x) + \sqrt{2}\dot{w}$ where $U(x)$ is a potential and $w(x)$ be an d-dimensional Brownian motion process. Since the SDE is governed by Fokker-Planck equations, the PHATE potential distances $D_{U,t}(x,y)$ naturally originate from the steady state solution of the SDE which satisfies $U(x) = -\log(p(x))$.

Practical benefits of using PHATE on time varying data have been observed through single-cell data examples [**102**]. In these examples, PHATE successfully identified different cell differentiation stages as branches on the 2-dimensional embedding (Figure 1.10 [**102**]). Still, as a relatively new method, PHATE remains of a niche use. In our work, we aim to supplement what has been done so far to potentially make PHATE a strong alternative in time series medical data visualization.

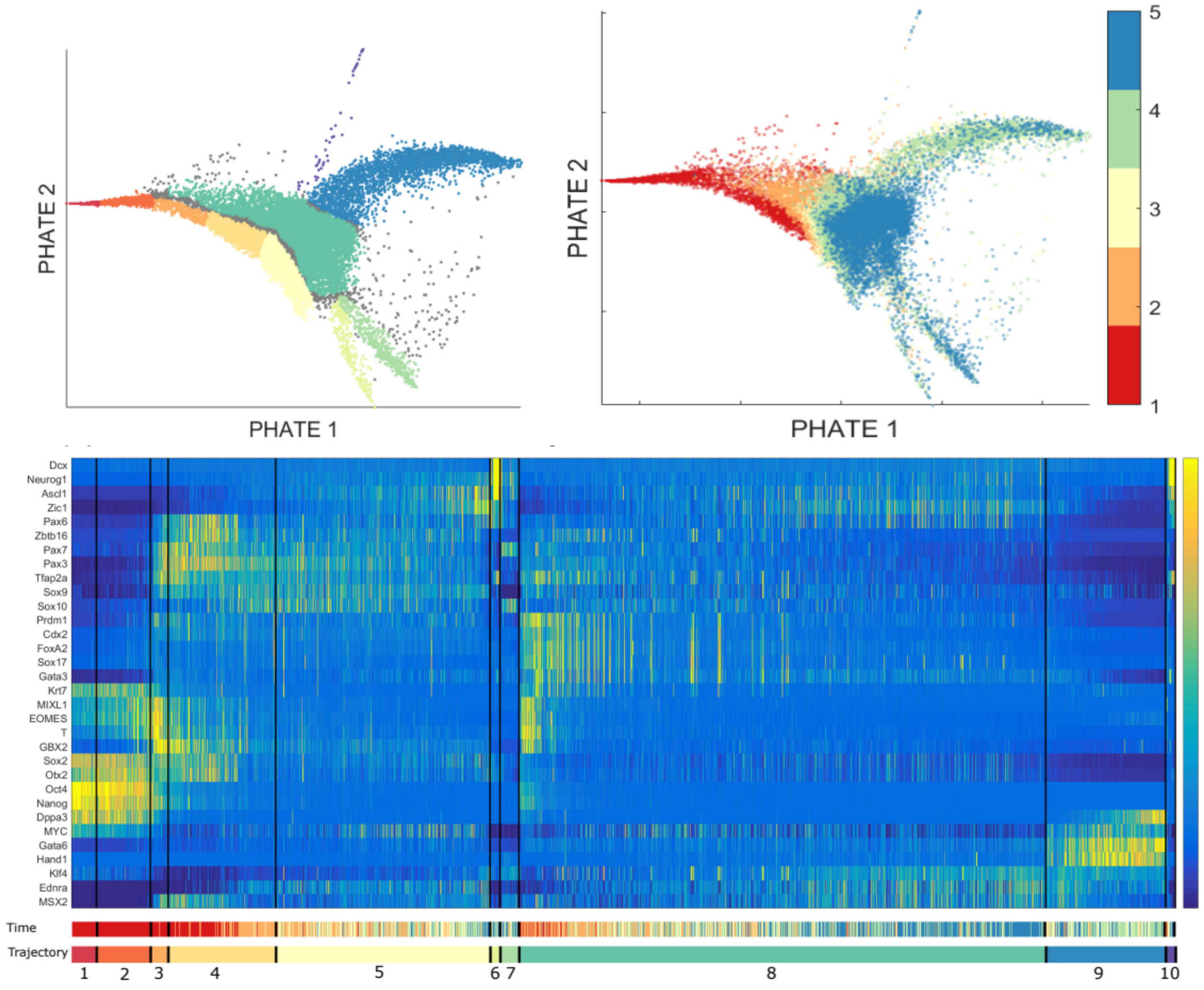


Fig. 1.10. Example of PHATE applied on time varying data. Top left: 2D PHATE plot of a database consisting of multiple gene expressions levels (features) of a human embryonic stem cell population at different time steps (samples) using single-cell RNA sequencing [**66**], colored by detected branches. Top right: same plot, colored by sample batch number (i.e. recording time). Bottom: gene expression matrix of the cells in each detected branch. Trajectories are associated with different gene expression levels as well as different time, which is biologically interpreted as an initial cell population splitting in different differentiation stages over time. This shows how PHATE can be used to highlight meaningful progressions.

Chapter 2

Awake prone positioning for COVID-19: short-term meta-analysis of treatment response

2.1. Context and motivation

Our first example of a real-world machine learning application is about the study of hypoxaemic patients with COVID-19 who had been taken in charge in hospital centre and subject to awake prone positioning (APP). By favoring a more homogeneous distribution of tidal volume and involving the recruitment of dorsal areas of lungs, APP improves oxygenation, lung compliance, and ventilation/perfusion matching [**54, 55, 67**]. Those biological benefits naturally motivates the use of APP in the context of the COVID-19 pandemic to improve the outcome of the hypoxaemic patient flow in emergency departments. In a meta-trial combining data from six open label superiority trials of patients requiring high flow nasal oxygen for COVID-19, Ehrmann et al. [**135**] found that APP significantly reduced the risk of intubation or death compared to standard care. In parallel, a new pragmatic randomised controlled trial conducted by Fralick and al. [**47**] found no differences in the primary outcome (death, invasive mechanical ventilation, or worsening respiratory failure requiring at least 60% fraction of inspired oxygen) between the APP group and the control group. Nevertheless, Barker et al. [**15**] did not interpret those contradictory conclusions as an argument against a systematic use of APP on COVID-19 patients. Indeed, they tried to explain this discrepancy by underlying factors that significantly differed between the two studies. In particular, they primarily observed a much higher APP duration and disease severity in the meta-trial. In this direction, Ibarra-Estrada et al. [**68**] performed a predictive analysis on cut-off values for different predictors on patients subject to APP after hospital enrolment and concluded that longer daily duration of APP, lower respiratory rate before APP, and positive response to APP in the first three days post-enrolment were associated with more treatment success. To sum up, APP may be helpful, but the existing literature to properly supervise its use in patients with COVID-19 remains very scarce. Thus, additional contributions to understand the different treatment responses that may occur between APP patients are required for the establishment of a unified APP protocol that better fits the needs of targeted sub-populations.

Through the analysis of the database from Ehrmann et al.'s meta-trial [**135**], this chapter intends to supplement Ibarra-Estrada et al.'s work on the discovery of success/failure predictors [**68**] regarding patients subject to APP. With a general machine learning point-of-view, our task is to provide domain experts with other visualization and predictive tools on APP patients' follow-up data since their hospital enrolment. In other words, we do not pretend to set exhaustive frameworks, but rather show how the medical domain could benefit from recent methods that have not been yet significantly applied in the existing literature. Our methodology aims to reconsider the problem as a whole multivariate time series data analysis unlike the related paper [**68**] which overall misses either the multivariate nature or the time component of the data. Indeed, in the second figure of the official Mexican study [**68**], we only keep track of each predictor separately across different time steps since enrolment, thus missing a global visualisation of all the variables at the same time which would better highlight common progressions among the variables or simultaneous correlations with the targets. Moreover, on the predictive side, only conventional logistic regressions [**112**] have been performed. Thus, we propose recent machine learning methods for both visualization and prediction, and compare them to more conventional approaches to better demonstrate the current advancements in machine learning.

2.2. Description of the database

The database comes from the aggregation of the participating research groups in six different countries: USA, Canada, Ireland, France, Spain and Mexico [**135**]. Each country collected follow-up data of patients subject to the APP treatment from the day of their hospital enrolment. The numbers of participating patients in each country are respectively 112, 7, 12, 200, 17 and 216. We summarize a detailed the descriptions of the collected features in Appendix A.1. They can be separated in 3 types:

- the longitudinal features, namely $SpO2$, $FiO2$, RR, SF, ROX and $TimePP$ measurements, recorded at day 0 (D0), day 1 (D1), day 2 (D2) and day 3 (D3), such that D0 is the day of admission at hospital center. All of them are clinically and biologically motivated since they measure different aspects of the cardiorespiratory condition of patients with hypoxemia. $SpO2$ (percentage) refers to the peripheral oxygen saturation monitoring by pulse oximetry which estimates the oxygen saturation of arterial blood, that is, the proportion of hemoglobin in the blood which is carrying oxygen [**34**]. $FiO2$ (decimal) is an estimation of the oxygen content a person inhales and is thus involved in gas exchange at the alveolar level [**49**]. RR refers to the respiratory rate (number of breaths per minute). SF and ROX respectively refers to the $SpO2$ to $FiO2$ ratio and the SF to RR ratio. They have already been confirmed as strong predictors of treatment failure/intubation in previous studies [**8, 121**]. $TimePP$ (minutes, or hours) refers to the total time of all prone sessions during a day. These features are considered as the six main features of our study, because they are the only ones recorded on a daily basis, thus describing multivariate time series;
- the static features, consisting of demographic features ($Gender$[1], Age...), comorbidities or other features somehow describing patient's condition during the trial;
- the three outcomes of interest, namely $Death28d$, $Intub$ and $Primary\_out$. $Death28d$ (0/1) indicates death before 28 days post enrolment. $Intub$ (0/1) indicates intubation before 28 days post enrolment. $Primary\_out$ (0/1), the combined outcome, indicates intubation or death before 28 days post enrolment.

Remark 2.2.1. *Some prefixes originating from the original labels of our working database, in the form $X^*\_$, may appear when referring to features during experiments.*

[1]To be more accurate, $Gender$ should be written as Sex. We choose to follow the notations of the original study [**135**] anyway.

Formally, if N and K are respectively the number of considered patients and longitudinal features, our working multivariate time series data set is denoted by

$$\mathbf{X} = \{X_i\}_{i=1}^{N}$$

such that $X_i = \{x_j^i\}_{j=0}^{3} \subset \mathbb{R}^K$ encodes the K-dimensional time series of length 4 for the i^{th} patient, starting from day $j = 0$ to day $j = 3$. In other words, x_j^i represents K longitudinal features of patient i collected at day j. To be concise, we refer to every x_j^i as a single patient's recording day. Additionally, each time series X_i is linked to a vector of its corresponding static features $C_i \in \mathbb{R}^{K_{static}}$, where K_{static} is the number of considered static features, as well as its binary outcomes of interest $Y_i = \{y_D^i, y_I^i, y_P^i\}$.

Furthermore, we separate the visualization and predictions on the database in three groups: the Mexican, the non-Mexican and the whole group. This distinction is motivated by two major considerations. First, it has been observed, in the previous analysis of the predictors [**68**], that $TimePP$ has a strong positive correlation with the primary outcome, which is not the case in the non-Mexican data. After expert investigation of the protocol in the different countries, we suspect, co-jointly with I. Pavlov, one of the contributors, that it could be due to a very different APP protocol within the Mexican trial. Indeed, it appears that the Mexican patients were compelled to execute prone positioning whereas it was only suggested in other countries. Consequently, the Mexican data is more likely to have its own structure, and we aim to observe this difference by comparing its output with the outputs of the non-Mexican and the whole population. A second consideration, and not the least, is the restricted data sharing. Indeed, even though all the necessary has been attempted to get a copy of the whole database, strict legal and ethical requirements to protect patient privacy, a broad concern of ML applications in the medical domain [**70**], limit us to a local copy of the Mexican database and a remote access for the rest via E. Tavernier, the authorized data holder. Consequently, we perform extensive experiments on the Mexican data and then provide the script to run with E. Tavernier on the non-Mexican and the whole database. These permission issues dramatically complicate the analysis of the whole data, especially the visualization, but we hope to get access to the rest of the data in the near future.

2.3. Visualization methodology

Detecting subgroups of patients associated with treatment failure would allow domain experts to investigate their phenotype and find underlying causes of failure. Ultimately, this would lead the specialists to reconsider their awake prone positioning protocol to better suit those sensitive phenotypes. To this end, we need to provide a visual representation of each patient's response to the APP treatment after enrolment. In the APP paper [**135**], APP response has been primarily measured through changes in terms of $SpO2$, $FiO2$, RR, SF and ROX. In addition, $TimePP$ appeared to be strongly correlated with treatment success, especially in the Mexican database [**68**], where a longer APP duration has been concluded to provide better treatment success, so we have a total of six predictors measured periodically after enrolment to describe each patient. Instead of describing patients' progressions with separate univariate time series of each predictor or simple after$-$before deltas, our approach is to take into account those six variables simultaneously to get the most out of our multivariate time series data. Since $SpO2$, $FiO2$, RR, SF and ROX have been recorded daily up to day 3 after enrolment, and $TimePP$ up to day 14, our database naturally provides a description of each patient in the form of a 6-dimensional time series of length 4, one time point for day 0, day 1, day 2 and day 3 respectively. Still, because of their high dimensionality, we cannot visually extract information from those MTS. Hopefully, reasons mentioned in Section 1.6, such as the ability to catch groupings with medoids and project them onto a lower-dimensional representation space, naturally motivates the use of k-medoids on the multi-factorial patients' trajectories in combination with PHATE.

Thus, we propose a two-step process to increase the interpretability of our MTS data. First, we flatten $\mathbf{X}$ to consider a data set in which each data instance is a single patient's recording day instead of its full time series,

$$\mathbf{X}_{flattened} = \{x_j^i\} \subset \mathbb{R}^6,$$

where i is the patient identifier and $j \in \{0, 1, 2, 3\}$ the j^{th} recording day. We need to consider each single recording day instead of time series because PHATE does not accept time series as input; instead, we can input $\mathbf{X}_{flattened}$ into PHATE to reduce and project all patients' data on the 2-dimensional Euclidean space. Consequently, each patient's recording day $x_j^i \in \mathbb{R}^6$ is associated with an homologous 2-dimensional vector $\tilde{x}_j^i \in \mathbb{R}^2$. Additionally, each formerly 6-dimensional time series $X_i = \{x_j^i\}_{j=0}^3 \subset \mathbb{R}^6$ now has a corresponding $\tilde{X}_i = \{\tilde{x}_j^i\}_{j=0}^3 \subset \mathbb{R}^2$ in two dimensions. Finally, thanks to the PHATE branching structure, we apply k-medoids (Section 1.6) on the original high dimensional time series and project the resulting clusters of time series on the 2-dimensional PHATE embedding to highlight common group progressions. Thanks to a proper coloring for each cluster, we are then able to track and distinguish common group progressions over the four days after enrolment. An outline of this 2-step visualization framework is illustrated in Figure 2.1.

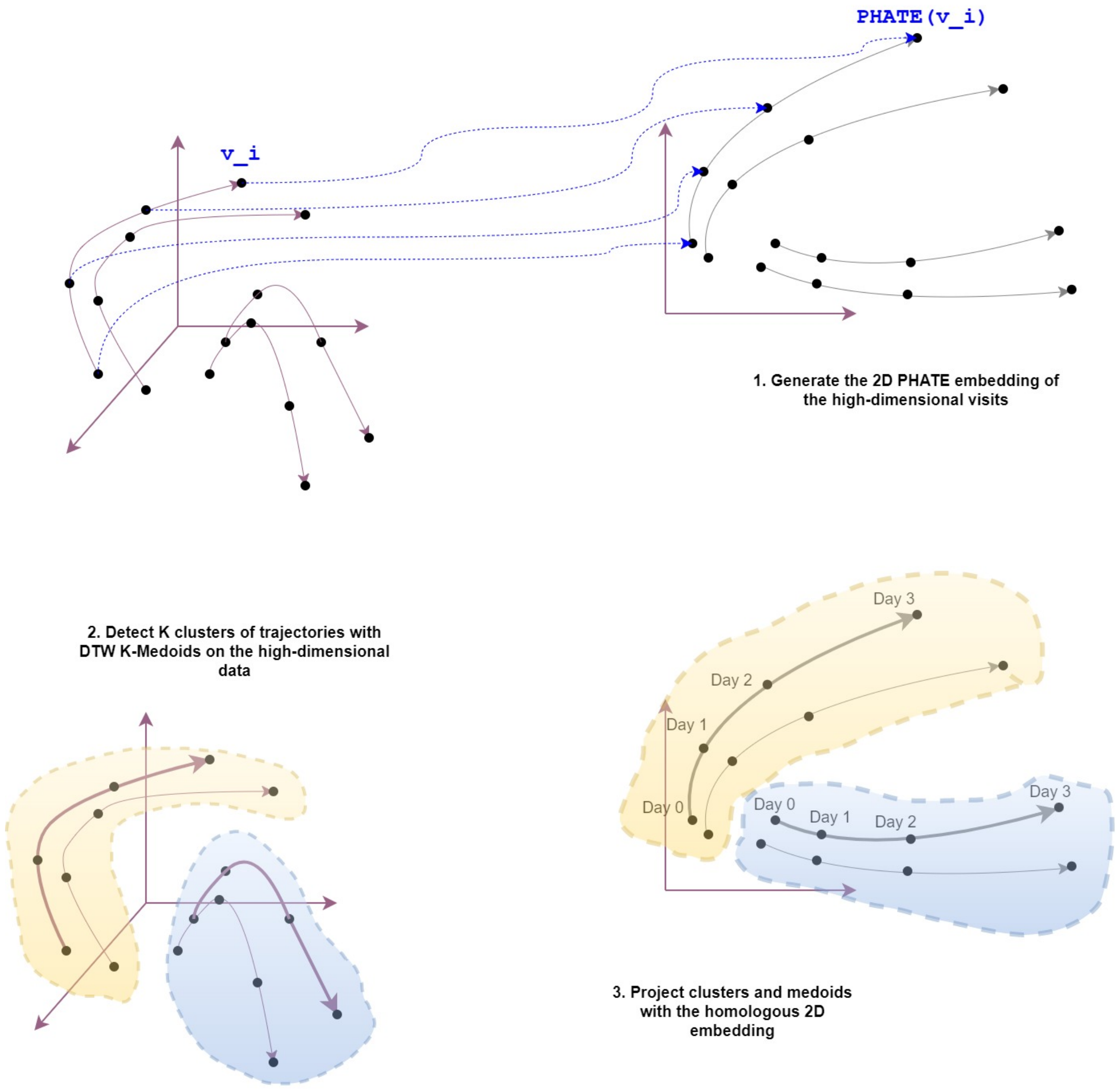

PHATE(v_i)
v_i
1. Generate the 2D PHATE embedding of the high-dimensional visits
2. Detect K clusters of trajectories with DTW K-Medoids on the high-dimensional data
Day 3
Day 2
Day 1
Day 0
Day 0
Day 1
Day 2
Day 3
3. Project clusters and medoids with the homologous 2D embedding

Fig. 2.1. Scheme of our visualization steps.

2.3.1. Data processing

We remove patients whose $SpO2$ and RR are outside their respective realistic ranges $60 - 100\%$ and $5 - 60$ breaths/min. The database contains few missing values (about 7% of the database) and evenly spaced daily measurements. Moreover, those missing values are mainly caused by patients failing the trial very early, i.e. intubated or dying before the third day post-enrolment. To prevent having medoids of fewer than 4 time points that do not properly reflect a temporal trend, we impute this small amount of missing values with forward filling. That is, we impute the tail of the incomplete time series with the last available feature value. For our purpose, this imputation is completely safe as it does not alter the visualization of the database (Appendix A.1). Compared to Z-transformation, its direct alternative, namely Min-Max scaling $x \mapsto \frac{x-\min(x)}{\max(x)-\min(x)}$ is by definition more sensitive to outliers or a lot of unusual spread in the data because the minimum and the maximum of a distribution often are outliers [**96**]. This is a concern in our data because abnormal values of $TimePP$ have been observed at Day 0: some values are either very small due to the bias of having way shorter enrolment days than the next full days, or exceeding the duration of Day 0 itself, a phenomenon that remains unexplained at the moment. Moreover, it is important to note that Z-transformation does not require normality to be useful. It effectively more or less squishes or expands the data to fit a standard deviation of 1, but the the original distribution is not affected. This is why we finally choose to Z-transform the data before inputting it into PHATE.

Finally, because our emphasis is on visualizing temporal patterns, we only consider longitudinal features as input. However, as opposed to our prediction task (see Section 2.4.4), data processing in the context of exploratory data analysis is not subject to a strict feature selection. In fact, exploratory data analysis can be seen as an early step after data collection and pre-processing (i.e. the latter paragraphs), where the data is simply visualized, plotted, manipulated, without any assumptions, in order to help assessing the quality of the data and building models [**80**]. Thus, all the 6 main longitudinal features $TimePP, FiO2, SpO2, RR, SF$ and ROX that have been suspected in the former APP studies to be potential predictors are considered as input into our PHATE embedding.

2.3.2. Embedding data in two dimensions with PHATE

After processing, we have respectively a total of 216 and 310 patients for the Mexican and the non-Mexican intervention populations, respectively. In matrix notation, multiplying by the number of days, we consider the flattened Mexican, global and non-Mexican input data

$$\mathbf{X}^{MEX}_{flattened} \in \mathbb{R}^{864\times 6}$$
$$\mathbf{X}^{ALL}_{flattened} \in \mathbb{R}^{2106\times 6}$$
$$\mathbf{X}^{NOMEX}_{flattened} \in \mathbb{R}^{1240\times 6}$$

where rows are patients' vector representations of the 6 recorded features during a single day. It is important to notice that we input single days (matrix rows), not patients' full time series, because PHATE does not handle time series as input. In other words, each single point in a PHATE embedding is associated with a single patient's recording day, and a full patient's time series is thus associated with four embedding points.

2.3.3. Drawing trajectories with projected DTW k-medoids clusters

Once we have successfully embedded patients' time points with PHATE, all the information of the 6 variables of interest is visually accessible in the 2-dimensional space. However, because we have only plotted discrete time points, we still miss the time component of our database that is contained in each patient's underlying time series in terms of the 6 longitudinal variables. Moreover, we are rather interested in exhibiting common progressions among patients than drawing individual trajectories.

To this end, we can perform a clustering algorithm on our patients' time series such that the output shows groups of patients with common progressions. Since clustering methods directly inherit properties within a data set and since PHATE's algorithm ineluctably distorts data, we preferably apply clustering on the 6-dimensional processed data instead of its 2-dimensional PHATE representation. Reasons mentioned in Section 1.6, especially the ability to map cluster prototypes (i.e. medoids) onto the corresponding 2D PHATE embedding, naturally lead us to perform k-medoids on the 6-dimensional processed patients' time series. After projecting the k medoids time series on the 2-dimensional PHATE embedding, we label each of their time steps according to the corresponding day so that we have a proper idea of the global motion within each cluster. Thus, those k medoids can be interpreted as reference trajectories from Day 0 to Day 3 of k sub-populations. Distances between original time series are computed with DTW because of its increased flexibility as opposed to a strict ED time-to-time matching (see Section 1.4), and we refer this time series clustering method to "DTW k-medoids". To better understand how we produce and visualize the clusters of trajectories, our general visualization framework is synthesized in Figure 2.1.

The choice of the parameter k, which determines the number of expected groups of trajectories to be detected, remains strictly arbitrary and mostly depends upon some domain experts' expectations, but we can already have a rough idea just by inspecting the 2D PHATE embedding. For instance, since the Mexican plot is structured in four connected clusters, $k = 4$ looks like a good first guess, because we expect trajectories along the edges. Performing a $k = 4$-medoids clustering (Figure 2.5) results in 4 clusters of almost equal size and with distinctive enough patterns, so we decided to base our cluster analysis on this clustering. For the global data (Figure 2.7), given that the Mexican and non-Mexican populations seemingly originate from very different distributions, we expect more groups of trajectories and $k = 6$ looks like a good compromise to look at both trajectories specific to the Mexican and non-Mexican data as well as non-specific global trajectories. For the non-Mexican population only (Figure 2.9), we also use $k = 4$ even though the clusters of trajectories are clearly not as defined as in the Mexican population, indicative of noisier and more heterogeneous data. Along with the resulting plots of trajectories, we provide insightful statistical descriptions of each cluster through box plots (Figures A.12, A.15, A.18), bar plots (Figures A.13, A.16, A.19) and tables (Figures A.14, A.17, A.20). Note that $TimePP$ are in minutes for the Mexican population only.

Remark 2.3.1. *The degree of freedom related to the desired number of clusters of trajectories k is the essence of exploratory analysis. Low values of k are prone to show a zoomed out overview of the data while big values are meant to increase the resolution of the exploration by breaking the big trajectories into smaller ones and potentially uncover subtle differences between distinct responses. As a first step into our data exploration, since there are few* *assumptions about the data, we need to set a relatively small k value to avoid dealing with noisy patients while keeping it big enough to show interesting patterns that have not been observed yet in the former APP studies* [**135**][**68**]. *This is why our k values are mostly chosen between 4 and 6, a good "in-between" to discover generalized patterns and more subtle ones that need further investigation.*

2.3.4. Evaluation

We evaluate two main aspects of our visualization framework: the ability of PHATE to embed the high-dimensional structure into a 2D phase space on which trajectories can be properly drawn, and the clinical relevance of the k-medoids clusters of trajectories.

On the one hand, we show the benefit from using PHATE over other existing methods in visualizing high-dimensional time series. To do so, we compare it with more commonly used approaches in the existing literature, namely Principal Component Analysis (PCA) [**97**], t-distributed Stochastic Neighbour Embedding (t-SNE) [**139**] and Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) [**98**] (with random initialization), applied on the exact same processed data, with the aim of pointing out how PHATE excels in plotting high-dimensional data in a branching structure, even on relatively small data sets. To lighten the number of figures, we only compare the different embeddings on the Mexican intervention group (Appendices A.6, A.7, A.8, A.9, A.10 and A.11).

On the other hand, because the database has already been studied, we can make use of this background knowledge to assert the quality of our groups of trajectories. For instance, the goodness of the resulting clusters can be quantitatively asserted thanks to their homogeneity with respect to the available labels [**56**], and, at the same time, we need to retrieve well-known groups of trajectories among our clusters to gain the confidence of the domain experts.

2.4. Prediction methodology

We provide alternatives to the logistic regression with memory-based ML models such as kNN with DTW distance and RNN variants to predict the death, intubation and combined outcome of new patients in a "trackable" way. This "trackable" predictive process is defined as follows: let f be a classification model, let $X := x_0 \in \mathbb{R}^K$ be a newly admitted patient described by some K features recorded at day 0, including $SpO2$, $FiO2$ and respiratory rate RR in our case, and let Y be the binary outcome of interest, let us say the combined outcome, with $Y = 1$ if the patient is intubated or dies before 28 days after enrolment, $Y = 0$ otherwise. We would like f to predict the outcome of the patient as a likelihood,

$$f(X) = p_0 \in [0, 1],$$

where p_0 estimates the probability that $Y = 1$ at day 0. From the clinician point-of-view, the value of p_0 helps a lot to decide what to do next with this patient: should we keep performing APP or directly transfer to reanimation and intubate? Generally, a high value of p_0 means that X would be prone to fail the APP treatment and X should be intubated directly; at the same time, we need to restrict unnecessary transfers to reanimation because the intensive care unit capacity is limited and patients with more serious conditions would better benefit from APP. There is no perfect threshold for p_0 to deal with this crucial trade-off: it is up to the clinician to use p_0 as a prognostic support only and consider external variables such as the available space in reanimation to tip the scales in favor of one decision over another. This is why we should extend awake prone positioning on patient X to an extra day for sufficiently small values of p_0 according to the clinician. In that case, the same K features of X are re-recorded by the end of day 1, let's say $x_1 \in \mathbb{R}^K$, and f should also outputs a probability of failure from those features to know if we keep awake prone positioning or transfer to reanimation after day 1. However, instead of considering only x_1 as input for f, we conduct our study with the intuition that keeping track of past patients' features has an influence on its current prognosis. Thus, after day 1, we also include day 0 measurements to describe patient X's condition that now consists of a K-dimensional time series of length two: $X := \{x_0, x_1\} \subset \mathbb{R}^K$. Afterwards, we input the new description of patient X into f again to get another estimation p_1 of the failure probability after day 1,

$$f(X) = f(\{x_0, x_1\}) = p_1 \in [0, 1]$$

and we repeat the procedure: we either stop APP or extend to day 2 depending on the value of p_1 etc. Our "trackable" predictive process is then made possible thanks to an appropriate classifier f allowing different input lengths, and we perform a kind of *early* *prognosis*, because f outputs a prognostic support as soon as the patient has been enrolled at the hospital, without the need of having its full time series.

Unfortunately, logistic regressions do not suit early prognosis for two main reasons. First, they require inputs of the same length for both training and testing: a test data instance must be padded to the same length as the training data set, implying that we cannot predict the outcome of a newly arrived patient without imputing its features for the following days. Moreover, they do not naturally handle time series that are multivariate, because fitting the model requires to embed each training instance into a single multidimensional vector. Different means such as concatenation [**64**] or time series feature extraction [**123**] have been studied to feed multivariate time series into logistic regressions, but they still do not directly train on multivariate time series and could not encode all relationships between variables or temporal patterns related to our specific database. Beyond these issues related to the structure of the database, another downside of logistic regression is the linearity in the logit for continuous variables which is often not met in many papers that make this assumption [**133**]. Our non-linear models derived from kNN and RNN do not make this assumption and allow more complex relationships.

Ultimately, in addition to this "trackable" process, our RNN-based models, called Conditional Recurrent Neural Networks (CondRNNs), allow static features as input. The relevance of adding this background information into a predictive system is going to be verified by a rigorous cross validation process in which all our proposed methods are confronted to each other.

2.4.1. Early prognosis with kNN

Our very first strategy to predict the endpoint of a newly admitted patient is a kNN lazy classifier (see Section 1.4). At first glance, this approach seems naive, but kNN algorithms are still widely used and are considered to be among the top-10 data mining algorithms [**148**]. They are easy to understand and implement and due to the sensitivity of making decisions in emergency cases, medical personnel tends to have trust in data scientists employing comprehensible methods. Thus, even if more advanced models such as neural networks have already demonstrated high performance in many domains, they still struggle to be accepted as a common support in medicine. kNN classifiers are known to be among the most robust and well-performing methods on a wide array of data sets [**94**], and are often used as a baseline to validate the efficiency of more advanced methods such as neural networks [**9**], especially in time series classification [**14**]. Therefore, we claim that it would be a mistake not to consider a kNN classifier as a solid starting point in our predictive analysis.

Additionally, we use DTW as the proximity measure for multiple reasons. First, in the context of our early prognosis objective, DTW is a natural choice when it comes to classify a new patient automatically even with its first time step whereas ED requires to truncate the available time series or impute missing values since it is a strict time-to-time comparison. Moreover, on small data sets, elastic measures such as DTW are empirically more accurate than the Euclidean distance [**149**].

The main downside is about the inclusion of the static features, which may potentially improve the model performance. Unlike RNNs which automatically balance the importance of the features in the learning process, we have to be careful when adding this static information into a kNN classifier. Indeed, since kNN has no learning algorithm and only takes the data Euclidean structure into account, the only way to also consider static information is to add the static features as stationary features of the time varying data, either by directly adding each feature to the existing time series or by empirically aggregating them into a single feature (without knowing exactly how to aggregate them). Because of the distance-based classification algorithm of kNN, this might overemphasize on static features that are not as relevant as longitudinal ones to predict an outcome. This is why we provide other approaches based upon RNNs to properly include static information in the next section.

2.4.2. Early prognosis with variants of recurrent neural networks

Another way to predict the endpoint of a new patient as early as possible is to make use of RNNs. RNNs and their more advanced variants Long Short-Term Memory (LSTM) [**63**] or Gated Recurrent Unit (GRU) [**36**] have already demonstrated state-of-the-art performance in similar clinical applications, such as heart failure early detection [**35**] or predictions of kidney rejection/loss or death on patients with transplanted kidney [**46**]. It has also been empirically checked that standard RNNs cannot remember events that occurred around five to ten time steps into the past [**100**]. Since we would like to remember sequences of length up to four (from D0 to D3), it is not necessary to make use of the common alternatives LSTM or GRU that were specifically designed to remember longer sequences. Thus, we believe that constructing a RNN-based model looks like a good start to expect high performance, but also to be used for comparison with other models. We still have to keep in mind that our working database is relatively small. Whereas it may be true that neural networks do not perform well on small data in general [**22, 77, 86**], that does not mean it is the case in our specific study. Shaikhina and Khovanova [**125**] even proposed an efficient framework to train neural networks on small medical data sets consisting of less than 10 observations per predictor variable, which is far below the size/dimensionality ratio of our database. In the end, experimenting on our data remains the best way to check those hypothetical conclusions, and this is our direction.

Due to their nature, medical data will always contain both static and dynamic features, and therefore it is fundamental to develop algorithms that can combine and exploit both types of data. Esteban et al. [**46**] proposed a very similar early prognosis study in which they found the highest performance for predicting endpoints on patients that underwent a kidney transplantation with an architecture combining a RNN with an independent Feed-forward Neural Network (FNN) that processes the static information. This was done by concatenating the hidden states of both networks and providing this information to the output layer before its activation.

Here, we prefer a less intrusive architecture, named Conditional Recurrent Neural Network (CondRNN), a RNN-based model that also handles static features as input, depicted by Karpathy and Fei-Fei [**72**] and Vinyals et al. [**142**]. To the best of our knowledge, no clinical application using this approach has been officially published yet, contributing to the novelty of our thesis. The idea is to reshape the vector of static features (or conditions) with an independent neural net through linear transformations in order to fit the dimension of the RNN's hidden layers. Thus, we can use this new vector to initialize the very first hidden state of our RNN instead of affecting zeros as usual. This results in a more theoretically correct modelling of the conditional probabilities $p(y_t \mid x_0, \ldots, x_t, conditions)$ for $0 \leq t \leq T$, since it properly conditions the RNN on non-temporal inputs, naturally solves the shape problem, and also avoids polluting inputs timesteps with additional non-temporal information so that static features are appropriately handled as background information. Formally, the equation for CondRNN is the same as the equation 1.5.1 for RNN, except that, instead of being randomly initialized set to zero, the initial hidden state is computed as $h_{-1} = W_3 \cdot \texttt{Conc}(W_1 c_1, W_2 c_2)$ where, as shown in Figure 2.2,

- $c_1 \in \{0,1\}^{n_{cat}}$ and $c_2 \in \mathbb{R}^{n_{cont}}$ are the static categorical and continuous input vectors, respectively;
- $W_1 \in \mathbb{R}^{n_{neurons} \times n_{cat}}$;
- $W_2 \in \mathbb{R}^{n_{neurons} \times n_{cont}}$;
- $W_3 \in \mathbb{R}^{n_{neurons} \times (n_{cat}+n_{cont})}$;
- $\texttt{Conc}(\cdot,\cdot)$ is the concatenation function;
- $(n_{cat}, n_{cont}) \in \mathbb{N}^2$ are respectively the numbers of static categorical and continuous input features.

Our second model is a modified version of the previous one, called Binary CondRNN, with the same conditional structure but now operating on discretized inputs. We also take this inspiration from Esteban et al. [**46**] who produce the best results with this approach. The idea is to generate three binary variables for each feature representing three different levels of intensity. For instance, feature *SF* splits into *SF Low*, *SF Medium* and *SF High*. If the originally collected value of *SF* is considered "low", then *SF Low* $= 1$ and *SF Medium* $=$ *SF High* $= 0$. We use each feature's mean and standard deviation of the training database to generate the three different levels. In our example, let μ_{SF} and σ_{SF} be the mean and standard deviation of the attribute *SF* among patients in the training database, and let X_{SF} be the attribute domain. To have a consistent definition of what should be a "low", a "medium" or a "high" *SF* value of a patient based upon the known database, Esteban et al. assume that the three levels of intensity are defined by the partition $X_{\mathit{SF}} = \mathscr{P}_{\mathit{Low}} \cup \mathscr{P}_{\mathit{Medium}} \cup \mathscr{P}_{\mathit{High}}$, where

$$\mathscr{P}_{\mathit{Low}} = \{x \in X_{\mathit{SF}} \mid x \leq \mu_{\mathit{SF}} - \sigma_{\mathit{SF}}\}$$

$$\mathscr{P}_{\mathit{Medium}} = \{x \in X_{\mathit{SF}} \mid \mu_{\mathit{SF}} - \sigma_{\mathit{SF}} < x \leq \mu_{\mathit{SF}} + \sigma_{\mathit{SF}}\}$$

$$\mathscr{P}_{\mathit{High}} = \{x \in X_{\mathit{SF}} \mid \mu_{\mathit{SF}} + \sigma_{\mathit{SF}} < x\}$$

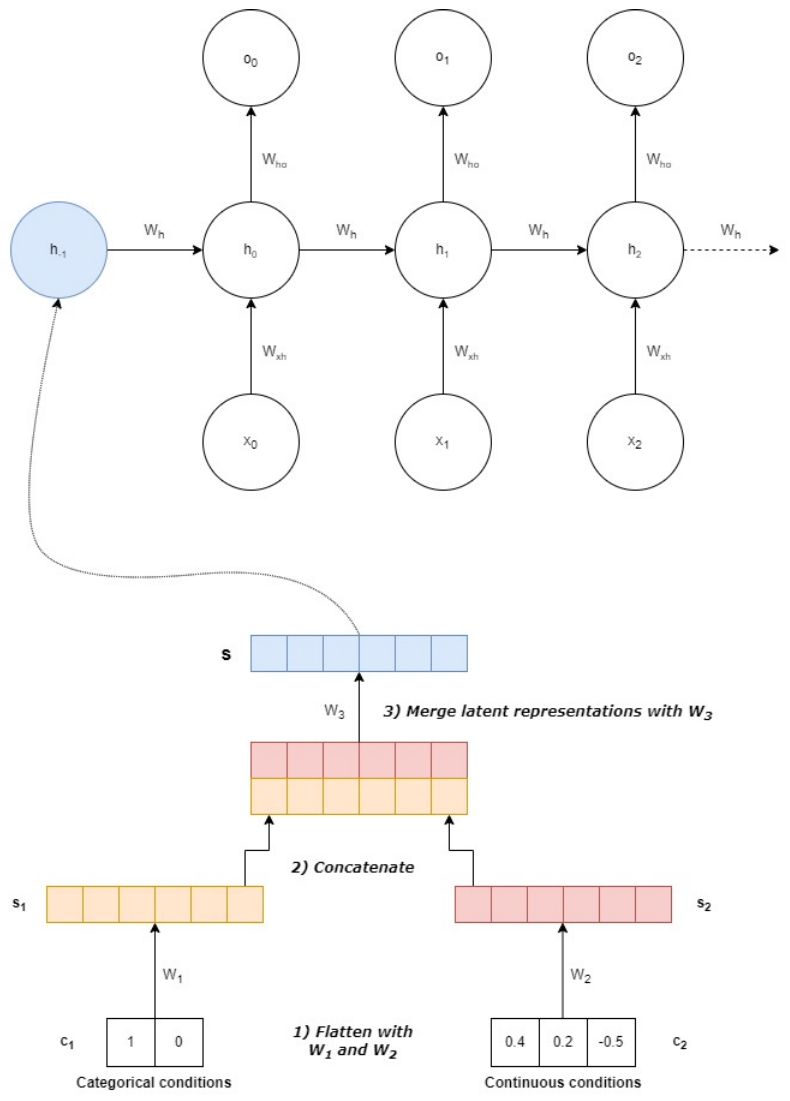


Fig. 2.2. Schematic illustration of our conditional RNN architecture, inspired by Philippe Remy [**118**]. Categorical and continuous static features are first embedded to fit hidden states' size, here $n_{neurons} = 6$ to simplify. Then, we concatenate those latent representations before merging them to the same dimension. Finally, we use those values to initialize the neurons of the initial hidden state h_{-1} of our RNN block.

Therefore, a "low" value x of *SF* happens when $x \leq \mu_{SF} - \sigma_{SF}$, a "medium" value when $\mu_{SF} - \sigma_{SF} < x \leq \mu_{SF} + \sigma_{SF}$ and a "high" value when $\mu_{SF} + \sigma_{SF} < x$. This strategy has two main benefits. First of all, if the binarization thresholds were clearly defined and valid on any unseen patient, it would allow the clinician to categorize measurements quickly and give its own estimation of the new patient's condition. To illustrate a practical case, if a new patient's SF is measured slightly above the "low" threshold $-$ thus falling into the "medium" category $-$, the clinician may put it into the "low" category instead to not underestimate the patient's condition, since a low SF generally indicates bad condition. However, doing so excessively may lead to a higher false positive rate, so the clinician should be aware of the hospital's surge capacity as well as the measuring instrument's error before qualifying any new patient's measured feature. Finally, since missing measurements are naturally translated by a zero value in each subcategory, it gets around the problem of setting an appropriate imputation method.

Our third and final model is a traditional RNN (see Section 1.5), named SimpleRNN, with equation 1.5.1. This is a mandatory baseline that is compared to the above conditional architectures and assert their usefulness.

The three above-mentioned approaches have only been presented as conceptual architectures and we still need their underlying hyper-parameters (learning rate, number of neurons, loss function etc.) to fully define them. In Section 2.4.3, we propose a rigorous framework to optimize them such that the resulting model satisfies a minimum validation loss. However, hyper-parameter optimization is itself an important research field in machine learning with its own challenges and advancements, so we wisely refer the reader to specific papers for a more detailed study [**151, 154**]. That being said, even though hyper-parameter tuning is a often a necessary step in evaluating architectures like RNNs defined by many hyper-parameters, please keep in mind that we do not want to focus on hyper-parameter optimization and are aware of possible improvements in this direction (see Section 2.8). Therefore, we have to assume some hyper-parameters in our above-mentioned RNN architectures.

To this end, we set fundamental hyper-parameters that suit binary classification problems, similarly to the study of Esteban et al. [**46**]. The numbers of input and output units are respectively $n_{features} = 3$ (see Section 2.4.4) and $n_{labels} = 1$. For CondRNN and Binary CondRNN only, $n_{cat} = 1$ and $n_{cont} = 5$ (see Section 2.4.4). Since all our RNN models are supposed to output probabilities, we use the sigmoid as the output activation function ϕ. The step-wise loss function ℓ guiding the iterative weight update of the learning algorithm naturally inclines towards the binary cross-entropy, which, for all $t \in \{0, 1, 2, 3\}$ and for all patients' indices i, has the form

$$\ell(o_t^i) = -y_i \log\left(o_t^i\right) - (1 - y_i) \log\left(1 - o_t^i\right)$$

where o_t^i and y_i are respectively the step-wise predicted outputs and the true binary end-point of patient i. As in [**46**], we also set the hidden activation function Φ to the default hyperbolic tangent.

Moreover, even though many optimizers have been proposed to train RNNs, such as the vanilla gradient descent through BPTT (see Section 1.5), we stay with Adam [**79**] for its high speed [**37**] and even better training and test losses than other existing methods in some RNN applications [**115**].

Finally, we also make use of *early stopping* [**116**], a popular regularization method which stops the learning process when the validation loss starts increasing. Its main goal is to prevent overfitting that often occurs when the number of training epochs is too high. Figure 2.3 [**152**] motivates the use of early stopping by showing the overfitting phenomenon occurring after too much epochs: while the training loss keeps decreasing, the test loss starts increasing, because the learning algorithm is fitting noisy points of the training set, resulting in a trained model that do not generalize well on new data. Early stopping tries to avoid this by inputting the holdout validation set into the model at the end of each training epoch and stopping the learning process when this validation loss starts increasing. To do so, we introduce *patience iterations* and *minimum delta value* parameters defining the number of epochs of stagnancy before stopping and the minimum change in the validation loss to qualify as an improvement. As stated by Goodfellow, Bengio and Courville, early stopping regularization is so easy to use that there is no doubt that it "should be used almost universally" [**52**]. However, it is important to note that, in our case, we primarily use early stopping for its reduced running time since our inner k-fold cross-validation implementation (see Section 2.5) already keeps the number of epochs with the lowest validation loss. It may appear artificial to use early stopping in our case, but we found it useful when it came to reproduce experiments multiple times. Besides the benefits of early stopping, using it in the inner loop may bias the hyper-parameter optimization as it takes advantage of the validation folds to find the stopping points. According to Prechelt [**116**], early stopping should take place apart in a distinct validation process, but our database is rather limited and it would over-complicate our framework. Therefore, we assume that this bias is small enough to produce reasonable results when performed in the inner loop; still, we are aware of this bias and its consequences when it comes to interpret any result. The *patience iterations* and *minimum delta value* parameters for early stopping are set to 10 and 0.0001 in our experiments. In this setting, we observe that early stopping never occurs above 100 epochs across different folds and hyper-parameter combinations, with peak values around 70-80 epochs. This is why we set the maximum number of epochs to 100 in the hyper-parameter grid (see Section 1.2).

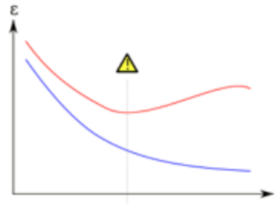


Fig. 2.3. Idealized progressions of the training (blue) and validation (red) losses versus the number of epochs. In most neural networks, training loss keeps decreasing with the number of epochs whereas validation loss reaches a minimum value from which the model starts overfitting. Early stopping is a popular method to determine this minimum point.

2.4.3. Validation and evaluation

Each of our proposed strategies in Sections 2.4.1 and 2.4.2 includes a lot of hyper-parameters, and we think that the optimization is crucial since several studies show that a set of optimal hyper-parameters improves model performance [**18, 130**]. Once each model has found its best hyper-parameter combination, we evaluate its performance at predicting on unseen data. In other words, we need to properly define the validation phase and the testing (or evaluation) phase of our four predictive models (see Section 1.2).

In practice, the validation and testing of each model is performed with NCV (Section 1.2.1) with the following features. We set the number of folds for the outer testing loop to 10 which ensures a reasonable running time while estimating an unbiased prediction error in most cases [**127**], and is also the standard value used in a very similar prognosis study with a less than a thousand patients [**106**]. The number of folds for the inner validation loop lowers to 5 because we do not want to overemphasize on hyper-parameter optimization. Folds are determined by stratified random sampling so that the class ratios are preserved to avoid the bias induced by learning, validating and testing on highly unbalanced folds that are not representative of the general case. For the grid search of the inner validation loop, the hyper-parameter grids of our three neural networks and kNN are respectively

$$\begin{aligned}
\text{number of hidden units } n_{neurons}\text{:}\ & [16, 32, 64] \\
\text{learning rate:}\ & [0.1, 0.01] \\
\text{number of epochs:}\ & 100 \\
\text{batch size:}\ & [25, 50] \\
\text{recurrent dropout:}\ & [0, 0.1] \\
& \& \\
\text{number of neighbours } k\text{:}\ & [5, 10, 25] \\
\text{classification method:}\ & [\text{majority vote}, \text{weight adjusted vote}],
\end{aligned}$$

where the recurrent dropout is the fraction of the hidden units to drop for the linear transformation of the recurrent state, a well-known regularization technique to prevent overfitting [**132**], and where the weight classification method of kNN chooses between majority vote using uniform neighbours' weights and an adjusted majority vote that weight points by the inverse of their distance, such that closer neighbours of a query point will have a greater influence than neighbours which are further away. This choice of hyper-parameter grid results from the combination of common practice in hyper-parameter optimization [**46, 129, 132**], computational limitations, and truncated experiments in which we track the test losses of the best combinations on different validation folds in order to have a global idea of what should be "good" range of possible values. The best hyper-parameter combination provided by one inner loop is the one with the highest mean validation loss over all the 5 possible training/validation splits, except for kNN where it is the one with the highest mean accuracy.

The evaluation metric of the outer loop is the traditional Area Under Receiver Operating Characteristic Curve (AUROC) [**57**], a gold standard in medicine also used in the former APP predictive analysis [**68**]. To evaluate our models at predicting early prognosis, i.e. our "trackable" process as described in Section 2.4, AUROC scores are computed on cropped test data. More precisely, for any outer test fold $\mathbf{X}_{test}$, given that each data instance $X \in \mathbf{X}_{test}$ has the form of a multivariate time series $X = \{x_0, x_1, x_2, x_3\}$ with a corresponding binary true label y, we truncate X such that we have $X_{D0} = \{x_0\}$, $X_{D1} = \{x_0, x_1\}$, $X_{D2} = \{x_0, x_1, x_2\}$ and $X_{D3} = \{x_0, x_1, x_2, x_3\}$ that represent patient X as if we only had its measurements up to Day 0, Day 1, Day 2 and Day 3 respectively. Then, we successively input the four truncated time series into the optimized and trained model and only keep the last step outputs (i.e. o_0 for X_{D0}, o_1 for X_{D1}, o_2 for X_{D2} and o_3 for X_{D3}) as the reference predicted labels to be compared with the true label y and produce the step-wise AUROC scores. In order to provide AUROC scores for the kNN as well, we switch from (weighted) majority vote to (weighted) class proportion within the k neighbours such that we properly output probabilities. After the completion of the outer loop, the final step-wise model performance is defined as the mean of the step-wise AUROC scores over the 10 folds. For the sake of simplicity, we refer to the final step-wise model performances as AUROC scores, even though they are technically averaged AUROC scores.

2.4.4. Data processing

First, we have to decide which features to include in our predictive models. Ideally, we would like to include the input variables resulting in the best performing model after the validation process. A direct approach is to optimize the selection of features by including it into our NCV in which each possible combination of features is another hyper-parameter of the inner grid search [**143**]. Despite being exhaustive in the sense that we train and test over every possible combination of features and hyper-parameters, including a set of features $\mathscr{F}$ into our grid search would multiply the initial number of combinations to check by $2^{\#\mathscr{F}}$, resulting in a huge increase in running time–even for small sets of features–that we cannot afford. Hopefully, unlike hyper-parameters, we have at least some expert knowledge about a subset of features that seem important or not in predicting our three targets. This is why we instead select features as part of data processing and first start by considering the same six time series features as for the visualization in Section 2.3.1: $\{RR, SpO2, FiO2, SF, ROX, TimePP\}$. However, we can reduce this set and get even better results. Indeed, although SF and ROX contain more information on a patient's condition than single RR, $SpO2$ or $FiO2$ measures, they are still computed from the latter ground features. Thus, by using all features in a training process, we would implicitly emphasize on some features through redundancies. In parallel, since we want to make the input of our model as easy as possible to allow on the fly use by a clinician, the number of input variables must be small. After simplified experiments, that is, with a reduced hyper-parameter space and number of folds, and by comparing NCV scores of models trained on the three non-redundant longitudinal features subsets $\{RR, SpO2, FiO2, TimePP\}$, $\{RR, SF, TimePP\}$ and $\{ROX, TimePP\}$, the first set slightly outperforms the others, so we choose to consider those four features in the final evaluation. Note that this conclusion is not surprising, because non-linear models are prone to recover the relationships captured by SF and ROX during the training phase. In addition, for our conditional RNN models, we also include the static binary feature $\{X04\_Gender\}$ and the 5 continous variables

$$\begin{aligned}&\{X05\_Age,\\ &Nb\_Comorbidities,\\ &X44b\_Delta\_Post\_PP\_RR,\\ &X45a\_Delta\_Post\_PP\_SpO2,\\ &X45b\_Delta\_Post\_PP\_FiO2\}.\end{aligned}$$

where $X04\_Gender$ is a binary indicator for Male (0) and Female (1), $X05\_Age$ denotes the age of the patient at inclusion, $Nb\_Comorbidities$ counts the number of existing comorbidities in a patient among 7 selected diseases/disorders (see Appendix A.1.2), and $X44b\_Delta\_Post\_PP\_RR$, $X45a\_Delta\_Post\_PP\_SpO2$ and $X45b\_Delta\_Post\_PP\_FiO2$ refer respectively to the difference between respiratory rate, SpO2 and FiO2 values after and before the first prone positioning session. As previously mentioned, this selection is not meant to be *a priori* optimal. In contrast to the main time varying features that are selected with some strong expert knowledge and other validation experiments, we basically select the static features based upon more hypothetical relevance. In fact, the point is to know, through experiments, if learning from a set of static variables results in better predictions, no matter the intrinsic predictive quality of this set.

Moreover, while learning through BPTT does not require normalized inputs, it could improve performance [**28**]. As a distance-based classifier, DTW-kNN also needs normalized inputs. We choose to be consistent with the visualization and decide to Z-transform features before inputting them into our four predictive models. However, as described in Section 1.2, the supervised learning process splits the available database into training and evaluation/test sets in order to simulate new incoming data points. Consequently, we cannot scale the whole database all at once as in the visualization, otherwise we would use information from validation/test splits which are supposed to be unknown prior training. This phenomenon is a form of *data leakage*, defined as the use of information in the model training process which would not be expected to be available during the test prediction, causing the predictive scores to be overestimated and not reflecting a reliable performance [**73**]. To avoid this, data scaling is instead performed separately on the training and validation/test sets: on each training set, we fit the scaling method, save the scaling parameters μ and σ and transform the data before inputting it into the learning process; afterwards, incoming data points from the complementary validation/test set are scaled with μ and σ, assuming they originate from the same distribution as the training set.

Finally, we have to deal with patients missing clinical attributes. Unlike kNN (see Section 2.4.1) and Binary CondRNN (see Sections 2.4.2), SimpleRNN and CondRNN require data imputation for training. We cannot simply exclude those missing attributes because data points have to be in the same dimension to be exploitable by any machine learning algorithm. Moreover, we may not completely remove recording days with missing values because our RNN implementation only accepts a fixed number of time steps as training input and we would like to take advantage of the other available attributes. Thus, we require data imputation. To determine how to perform it, let us take a close look on missing values. We observe that they only occur on some patients' last recording days, suggesting that values are missing for an underlying reason. Indeed, referring to the original study [**135**], missing recording days are most often related to patients being intubated or dying prematurely. That is, missing values are mainly associated with bad outcomes and must be treated as relevant information for better predictions. Making use of generic imputation methods like forward filling as in Section 2.3.1 or mean imputation would not take this contextual meaning into account. A solution is to replace missing values with a fixed numerical value highlighting the absence of data, as long as it is not already a meaningful value [**38**]. Lipton, Kale and Wetzel [**87**] achieved top results in a similar study by imputing with zeros. Instead of zeros, we similarly imputed with -4. Since we standard scaled our variables and observed that this value was not reached by any of the available data points, our choice is motivated by the intuition that a -4 value sufficiently distinguishes the absence of measurements for all the features without being too discriminating, which would disturb the learning process.

2.4.5. Measuring model confidence

Even though CondRNN is seemingly the best candidate to get accurate early prognoses for incoming patients (see Section 2.7.2 for our empirical proof), it still needs further investigation to be approved and potentially deployed for clinical use. Indeed, the non-linearity of deep learning models makes the optimization process difficult to follow by non-ML experts, in contrast to linear methods such as linear regressions which are now admitted as reliable options in medical research [**153**]. Moreover, this reluctance to use "black-box" models is even more present in healthcare where they are to affect patient care directly, leading more and more studies to emphasize on building trust in complex models that not only goes through the demonstration of high performance but also their interpretability [**2**]. Thus, in the hope of deploying our model for effective early prognosis, we need to add metrics which unroll the complexity of CondRNN and fit current domain specific research.

One way to do so is input feature perturbation [**114**]. Given a trained CondRNN model over a training set of time series $X_{train} = \{x_1, \dots, x_N \mid x_i \in \mathbb{R}^{T \times K}\}$ and their associated binary labels $Y_{train} = \{y_1, \dots, y_N\}$, input feature perturbation consists of adding white noise to each input feature data to observe the effect of a small feature perturbation to the model output. More formally, it consists of first feeding the trained model with X_{train} to produce the output $\hat{Y} = \{\hat{y}_1, \dots, \hat{y}_N\}$. Note that the latter are not the real labels that were used to train the model: they are the step-wise outputs $\hat{y}_i = \{o_t^i\}_{t=0}^T$ of the training patients and aim to show how the model transforms the inputs to approximate the real labels. Then, we add a small Gaussian perturbation to the k^{th} column of each $x_1, \dots, x_N$ to form K new data sets $\tilde{X}^{(1)}, \dots, \tilde{X}^{(K)}$, where each $\tilde{X}^{(k)} = \{\tilde{x}_1^{(k)}, \dots, \tilde{x}_n^{(k)}\}$ differs from

X_{train} only in its perturbed data entries of the k^{th} feature. Finally, we input each $\tilde{X}^{(k)}$ into the model, store their output $\tilde{Y}^{(k)} = \{\tilde{y}_1^{(k)}, \ldots, \tilde{y}_n^{(k)}\}$ and measure the importance of the k^{th} feature with the root mean squared error $RMSE^{(k)} = \sqrt{\frac{\sum_{i=1}^{N}\left(\tilde{y}_i^{(k)} - \hat{y}_i\right)^2}{N}}$. Since CondRNN returns T step-wise outputs for each data instance, it implies that the final $RMSE^{(k)}$ is a $T-$dimensional vector. Thus, the T entries of $RMSE^{(k)}$ quantify the importance of the k^{th} feature for different step-wise predictions.

Remark 2.4.1. *Each $RMSE^{(k)}$ does not provide the intrinsic importance of a variable because it does not confront the perturbed outputs to the real labels y_i. In other words, a single $RMSE^{(k)}$ value does not help us determine if the k^{th} feature is important. As stated in* [**114**]*, "the resulting change in a chosen error metric for each input perturbation represents the relative importance of each input variable". This implies that a variable importance computed this way is only relevant when compared to other ones. Moreover, input perturbation is a model specific approach showing variable importances in the context of one model. Therefore, the resulting importance scores on a bad performing model would be prone to highlight important variables in making bad predictions, which is obviously not reliable. Hopefully, our model has already shown high performance, which allows us to proceed this way.*

The motivation behind this choice comes from its high interpretability in itself, which was already mentioned above as being an important criteria when building confidence metrics, but also its fast and easy implementation. Unlike methods such as backward stepwise elimination or forward stepwise addition [**114**] consisting of measuring prediction differences when removing or adding one variable at a time, input perturbation does not need to re-train our model.

2.4.6. Deployment

Of course, our predictive analysis is motivated by a potential practical use in clinical environment. We would like to give the clinician the ability to input a new patient's features into a program returning the associated prediction. Unfortunately, at the cost of giving a generalized comparison between different models, our NCV does not output a set of best performing hyper-parameters because the optimal combination may vary across outer loops. This implies that we still cannot implement the program we want to provide, because the ML predictive models (DTW-kNN or RNN-based) need hyper parameters to be fully defined before training. The only reason to use our NCV framework is to decide which model architecture to use among our four proposed solutions, which overall seems to be CondRNN (see Section 2.7.2). Thus, to make it workable for practical use, since we no longer compare models, we set a random seed and re-run a simple cross validation, i.e. without the outer test loop, with the same hyper parameter grid search. Once the optimal combination is found, we can use it to train the CondRNN model on the full available data set, so that our model is ready to be used on "true" incoming data, i.e. not artificially introduced as we did for the cross validations.

Remark 2.4.2. *Note that the considered training data may vary depending on the location of use. Of course, in general, it is be better to build a single flexible model trained on all the available data to better generalize, but since we have pointed out a specific APP protocol in Mexico, we should probably train the model exclusively on the Mexican data if we plan to apply it locally, because incoming patients in Mexico might be subject to that specific APP protocol as well.*

2.5. Experimental setup

Both visualization and prediction have been implemented in the programming language Python [**140**], version 3.9, in the integrated development environment (IDE) PyCharm[2]. Although we preferably emphasized on methodological explanations because of the context of the thesis, the hidden programming process was probably the most costly in time aspects of our work, full of experimental trials-errors and extensive documentation

[2] `https://www.jetbrains.com/pycharm/`

research. This is why we mention and acknowledge the authors of the respective general data analysis Python tools and other more specific third-party libraries that we have included in our implementation, without whom the applied dimension of this thesis would be tremendously missing:

- `pandas` [**110**][**147**] and `numpy` [**58**] for general data extraction and matrix manipulation;
- `phate` [**102**] for the PHATE embedding;
- `tslearn` [**134**] for the DTW distances computation between trajectories and `pyclustering` [**107**] to perform k-medoids on those distances;
- `sktime` [**92**][**93**] for other time series manipulations;
- `tensorflow`'s `keras` module [**39**] for setting up our RNN-based classifiers and its implementation of the Adam optimizer;
- `scikit-learn` [**111**] for standard scaling, for the PCA and t-SNE embeddings, for setting up our DTW kNN classifier and its NCV pipeline, as well as the calculation of the AUROC scores;
- `cond-rnn` [**118**] for its original implementation of the conditional RNN variant;
- `keras-hypetune`[3] for handling grid search on our multi-input conditional RNNs, which is impossible with `scikit-klearn`'s `GridSearchCV`[4];
- `matplotlib` [**65**] and `seaborn` [**145**] for plotting.

Our project is also publicly available at `https://github.com/AdaGHub/prone-positioning` for a more detailed overview of its structure and transparency with respect to the abovementioned libraries.

[3]`https://github.com/cerlymarco/keras-hypetune`

[4]This forces us to build our own NCV pipeline for the conditional RNN models with `scikit-klearn`'s `StratifiedKFold` in contrast to the all-in-one encapsulation provided by `scikit-klearn`'s `GridSearchCV` for kNN.

2.6. Results

2.6.1. PHATE embedding plots and projected trajectory clusters

The resulting 2D PHATE embedding plots after inputting $\mathbf{X}^{MEX}_{flattened}$, $\mathbf{X}^{ALL}_{flattened}$ and $\mathbf{X}^{NOMEX}_{flattened}$ (Section 2.3.2) into the PHATE algorithm with the above-mentioned parameters are depicted in Figures 2.4, 2.6 and 2.8, respectively. The 2-dimensional PHATE plots for the Mexican, global and non-Mexican processed inputs are respectively produced with the triplets $(k = 30, t = 60, \alpha = 10)$, $(k = 20, t = 40, \alpha = 15)$ and $(k = 20, t = 60, \alpha = 30)$, along with the square root potential instead of the log potential. Each of them consists of 6 sub-plots illustrating the corresponding 2D PHATE embedding that have been colored according to the values of the 6 inputs features. This coloring allows us to understand how PHATE organizes its 2-dimensional embedding points such that the separability in terms of the 6 inputs features is preserved and embedded in a single plot. In addition (see (Appendices A.3, A.4, A.5), we color the PHATE embedding according to the outcomes and other features that have not been included as input in order to quickly spot some input-target correlations.

Remark 2.6.1. *Thanks to PHATE's robustness to perturbations of its parameters* [**102**]*, the overall data structure extracted from different parameter combinations remains visually similar. Thus, it is safe for a clinician that uses PHATE naively to stick with the default PHATE input parameters and still get the same overall structure (Appendix A.2).*

Projected time series clusters following our DTW k-medoids approach (Section 2.3.3) on each of the Mexican, global and non-Mexican intervention groups are shown Figures 2.5, 2.7 and 2.9, respectively.

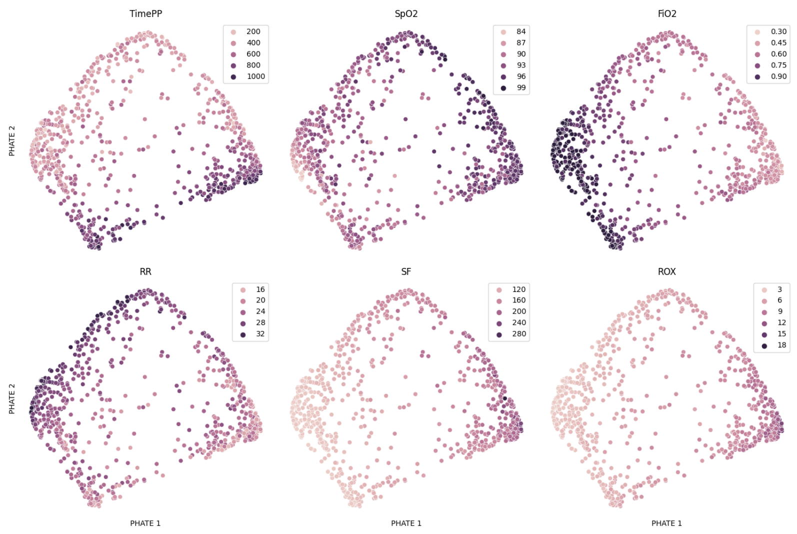


Fig. 2.4. 2-dimensional PHATE embedding of $\mathbf{X}^{MEX}_{flattened}$ following our processing steps in Section 2.3.1. Each of the six sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1.

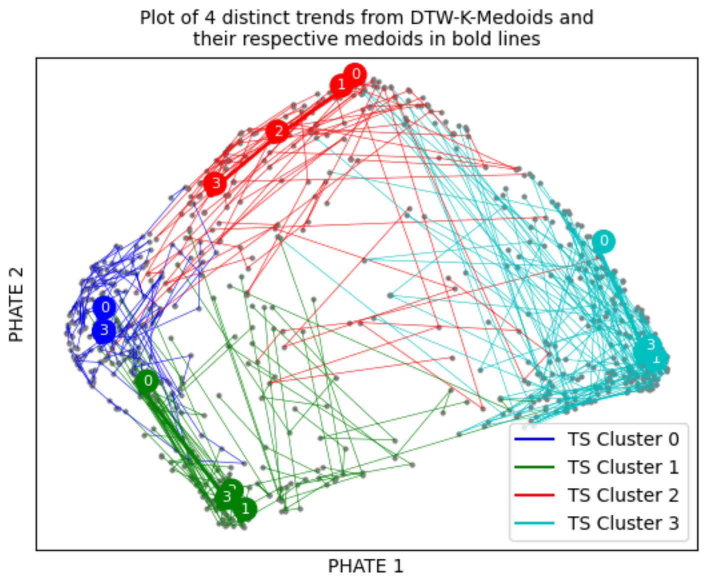


Fig. 2.5. Groups of trajectories of the Mexican intervention group detected by 4-medoids and projected on our 2-dimensional PHATE embedding, with the associated medoids in bold lines.

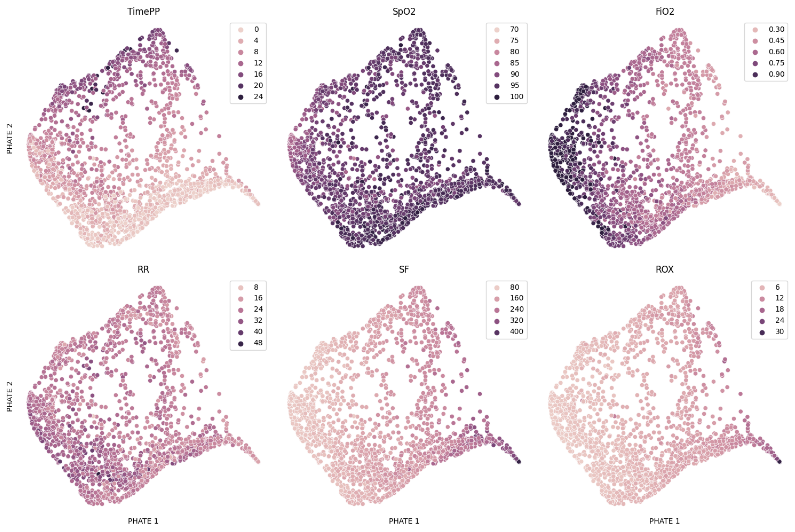


Fig. 2.6. 2-dimensional PHATE embedding of $\mathbf{X}^{ALL}_{flattened}$ following our processing steps in Section 2.3.1. Each of the six sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1.

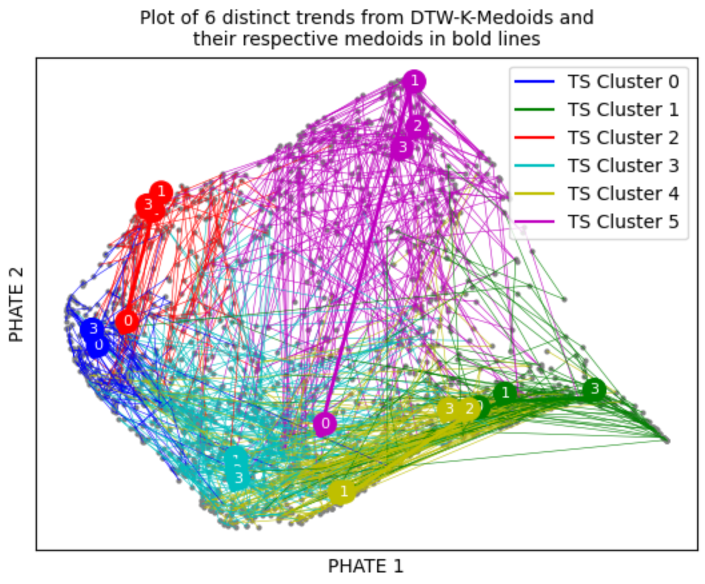


Fig. 2.7. Groups of trajectories of the whole intervention group detected by 6-medoids and projected on our 2-dimensional PHATE embedding, with the associated medoids in bold lines.

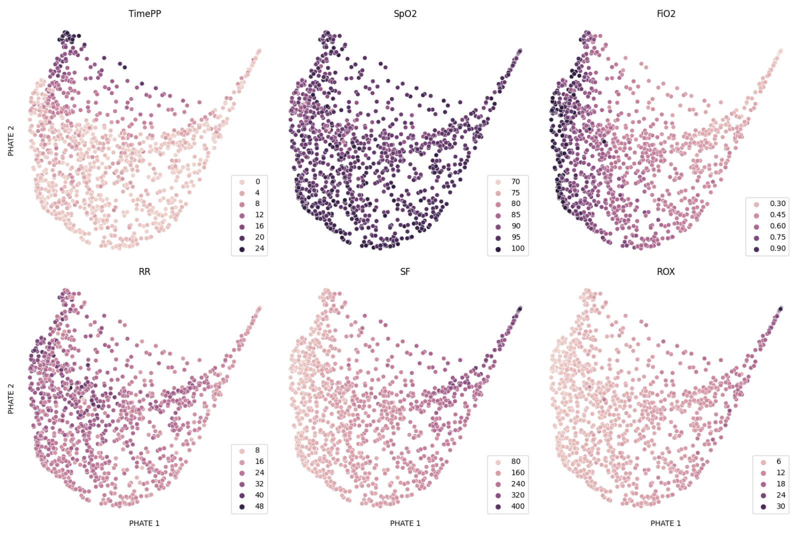


Fig. 2.8. 2-dimensional PHATE embedding of $\mathbf{X}^{NOMEX}_{flattened}$ following our processing steps in Section 2.3.1. Each of the six sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1.

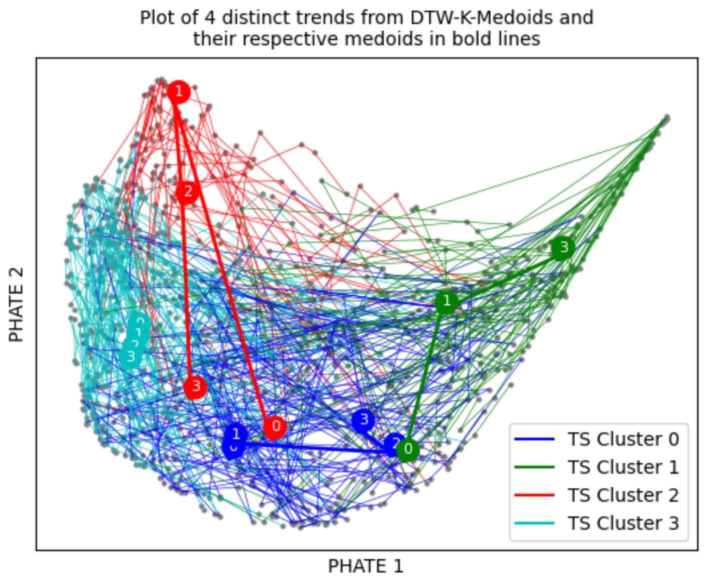
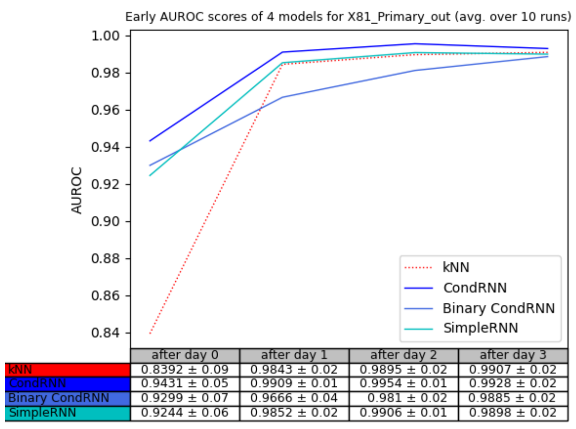


Fig. 2.9. Groups of trajectories of the non-Mexican intervention group detected by 4-medoids and projected on our 2-dimensional PHATE embedding, with the associated medoids in bold lines.

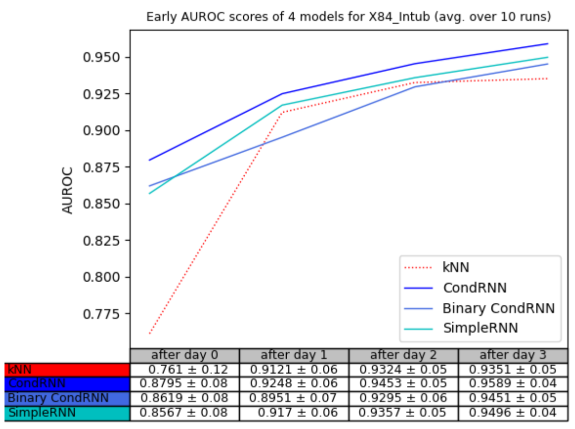
2.6.2. Performance scores of the four predictive models

We now present the resulting step-wise AUROC scores of each predictive model (kNN, RNN, CondRNN and Binary CondRNN) from our NCV process. Due to the random splits of folds as well as the randomly initialized weights of the recurrent neural networks, results vary between each NCV run. To this end, for the Mexican database, we run the NCV ten times and averaged the AUROC scores to provide a more trustworthy comparison. In each run, we fixed a different random seed instead of making it random because the splits should always be the same across experiments; otherwise, our four models are not evaluated in the same conditions, which would potentially disadvantage a model over another. This also allows the reader to reproduce our analyses for verification purpose. For the NCV of the non-Mexican and global databases, we are unfortunately restricted to a single run due to the lack of locally accessible data. The dotted red and cyan lines are respectively associated with the performance of kNN and traditional RNN, our selected baseline methods. The royal blue and the dark blue lines represent the step-wise performance of our proposed conditional RNN variants, respectively Binary CondRNN and CondRNN.



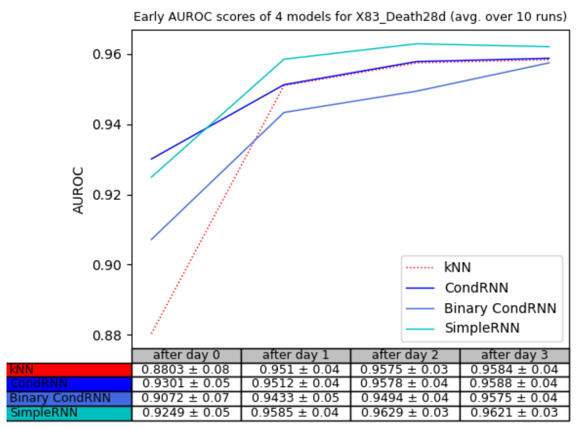
	after day 0	after day 1	after day 2	after day 3
kNN	0.8392 ± 0.09	0.9843 ± 0.02	0.9895 ± 0.02	0.9907 ± 0.02
CondRNN	0.9431 ± 0.05	0.9909 ± 0.01	0.9954 ± 0.01	0.9928 ± 0.02
Binary CondRNN	0.9299 ± 0.07	0.9666 ± 0.04	0.981 ± 0.02	0.9885 ± 0.02
SimpleRNN	0.9244 ± 0.06	0.9852 ± 0.02	0.9906 ± 0.01	0.9898 ± 0.02

(a) Primary outcome



	after day 0	after day 1	after day 2	after day 3
kNN	0.761 ± 0.12	0.9121 ± 0.06	0.9324 ± 0.05	0.9351 ± 0.05
CondRNN	0.8795 ± 0.08	0.9248 ± 0.06	0.9453 ± 0.05	0.9589 ± 0.04
Binary CondRNN	0.8619 ± 0.08	0.8951 ± 0.07	0.9295 ± 0.06	0.9451 ± 0.05
SimpleRNN	0.8567 ± 0.08	0.917 ± 0.06	0.9357 ± 0.05	0.9496 ± 0.04

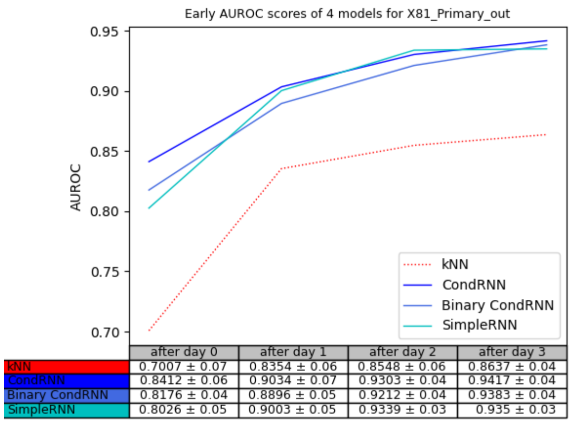
(b) Intubation outcome



	after day 0	after day 1	after day 2	after day 3
kNN	0.8803 ± 0.08	0.951 ± 0.04	0.9575 ± 0.03	0.9584 ± 0.04
CondRNN	0.9301 ± 0.05	0.9512 ± 0.04	0.9578 ± 0.04	0.9588 ± 0.04
Binary CondRNN	0.9072 ± 0.07	0.9433 ± 0.05	0.9494 ± 0.04	0.9575 ± 0.04
SimpleRNN	0.9249 ± 0.05	0.9585 ± 0.04	0.9629 ± 0.03	0.9621 ± 0.03

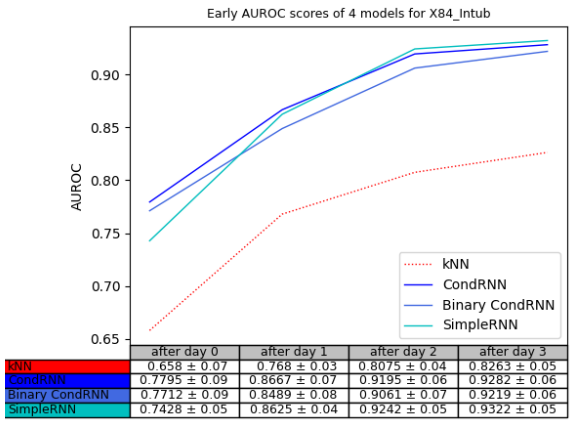
(c) Death outcome

Fig. 2.10. Step-wise performance comparison between four different models on the Mexican database. (a) Each row represents the AUROC scores of a given model after NCV such that the model is trained for the primary outcome and the testing patients' time series are reduced to day 0 (column "after day 0"), days 0–1 (column "after day 1"), days 0–2 (column "after day 2"), and days 0–3 (column "after day 3") values in order to measure early prognosis performance. The above graph plots the content of the table for an easier comparison between the four models: the red dotted line plots the kNN row, the dark blue line plots the CondRNN row, the royal blue line plots the Binary CondRNN row and the cyan line plots the SimpleRNN row. Sub-figures (b) and (c) are similarly built, except that models are trained for the intubation and the death outcomes, respectively. Overall, CondRNN is the best performing model for an early prognosis on the Mexican database.



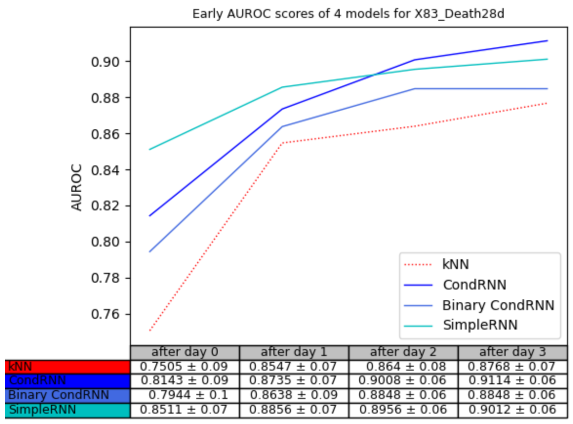
	after day 0	after day 1	after day 2	after day 3
kNN	0.7007 ± 0.07	0.8354 ± 0.06	0.8548 ± 0.06	0.8637 ± 0.04
CondRNN	0.8412 ± 0.06	0.9034 ± 0.07	0.9303 ± 0.04	0.9417 ± 0.04
Binary CondRNN	0.8176 ± 0.04	0.8896 ± 0.05	0.9212 ± 0.04	0.9383 ± 0.04
SimpleRNN	0.8026 ± 0.05	0.9003 ± 0.05	0.9339 ± 0.03	0.935 ± 0.03

(a) Primary outcome



	after day 0	after day 1	after day 2	after day 3
kNN	0.658 ± 0.07	0.768 ± 0.03	0.8075 ± 0.04	0.8263 ± 0.05
CondRNN	0.7795 ± 0.09	0.8667 ± 0.07	0.9195 ± 0.06	0.9282 ± 0.06
Binary CondRNN	0.7712 ± 0.09	0.8489 ± 0.08	0.9061 ± 0.07	0.9219 ± 0.06
SimpleRNN	0.7428 ± 0.05	0.8625 ± 0.04	0.9242 ± 0.05	0.9322 ± 0.05

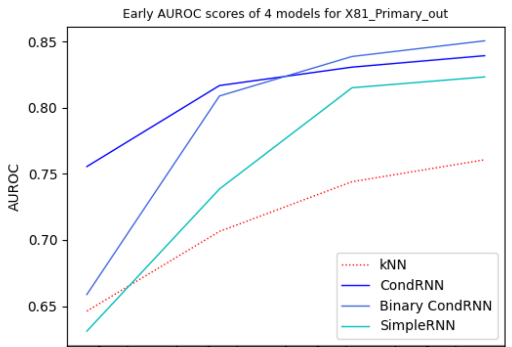
(b) Intubation outcome



	after day 0	after day 1	after day 2	after day 3
kNN	0.7505 ± 0.09	0.8547 ± 0.07	0.864 ± 0.08	0.8768 ± 0.07
CondRNN	0.8143 ± 0.09	0.8735 ± 0.07	0.9008 ± 0.06	0.9114 ± 0.06
Binary CondRNN	0.7944 ± 0.1	0.8638 ± 0.09	0.8848 ± 0.06	0.8848 ± 0.06
SimpleRNN	0.8511 ± 0.07	0.8856 ± 0.07	0.8956 ± 0.06	0.9012 ± 0.06

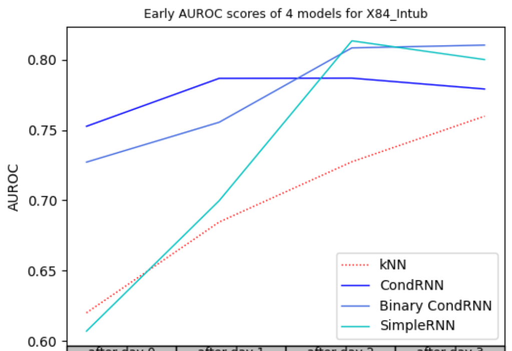
(c) Death outcome

Fig. 2.11. Step-wise performance comparison between four different models on the global database. (a) Each row represents the AUROC scores of a given model after NCV such that the model is trained for the primary outcome and the testing patients' time series are reduced to day 0 (column "after day 0"), days 0–1 (column "after day 1"), days 0–2 (column "after day 2"), and days 0–3 (column "after day 3") values in order to measure early prognosis performance. The above graph plots the content of the table for an easier comparison between the four models: the red dotted line plots the kNN row, the dark blue line plots the CondRNN row, the royal blue line plots the Binary CondRNN row and the cyan line plots the SimpleRNN row. Sub-figures (b) and (c) are similarly built, except that models are trained for the intubation and the death outcomes, respectively. Overall, CondRNN is the best performing model for an early prognosis on the global database.



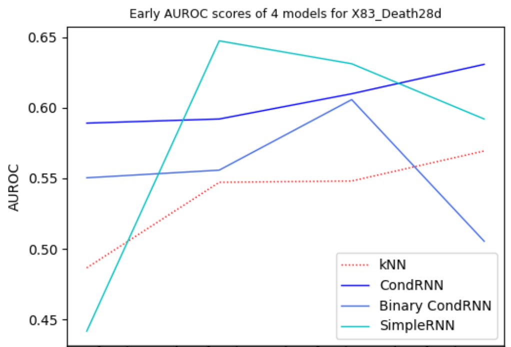
	after day 0	after day 1	after day 2	after day 3
kNN	0.6462 ± 0.13	0.7064 ± 0.1	0.744 ± 0.09	0.7606 ± 0.09
CondRNN	0.7555 ± 0.1	0.8167 ± 0.07	0.8306 ± 0.07	0.8392 ± 0.11
Binary CondRNN	0.6588 ± 0.14	0.8088 ± 0.09	0.8386 ± 0.07	0.8505 ± 0.08
SimpleRNN	0.6311 ± 0.12	0.7385 ± 0.14	0.815 ± 0.1	0.8232 ± 0.09

(a) Primary outcome



	after day 0	after day 1	after day 2	after day 3
kNN	0.62 ± 0.09	0.6845 ± 0.06	0.7274 ± 0.08	0.7598 ± 0.1
CondRNN	0.7526 ± 0.18	0.7867 ± 0.16	0.7868 ± 0.15	0.7791 ± 0.14
Binary CondRNN	0.7271 ± 0.22	0.7555 ± 0.17	0.8083 ± 0.13	0.8103 ± 0.11
SimpleRNN	0.607 ± 0.11	0.6997 ± 0.14	0.8133 ± 0.12	0.8 ± 0.08

(b) Intubation outcome



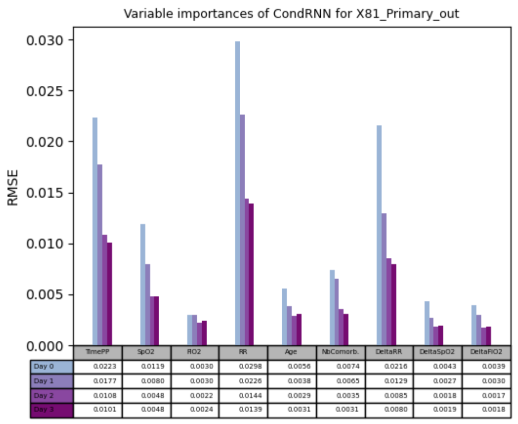
	after day 0	after day 1	after day 2	after day 3
kNN	0.4866 ± 0.19	0.5472 ± 0.16	0.5481 ± 0.15	0.5694 ± 0.15
CondRNN	0.5892 ± 0.31	0.5921 ± 0.2	0.61 ± 0.29	0.6308 ± 0.26
Binary CondRNN	0.5504 ± 0.3	0.5558 ± 0.37	0.6058 ± 0.23	0.5054 ± 0.25
SimpleRNN	0.4417 ± 0.25	0.6475 ± 0.24	0.6312 ± 0.21	0.5921 ± 0.3

(c) Death outcome

Fig. 2.12. Step-wise performance comparison between four different models on the non-Mexican database. (a) Each row represents the AUROC scores of a given model after NCV such that the model is trained for the primary outcome and the testing patients' time series are reduced to day 0 (column "after day 0"), days 0–1 (column "after day 1"), days 0–2 (column "after day 2"), and days 0–3 (column "after day 3") values in order to measure early prognosis performance. The above graph plots the content of the table for an easier comparison between the four models: the red dotted line plots the kNN row, the dark blue line plots the CondRNN row, the royal blue line plots the Binary CondRNN row and the cyan line plots the SimpleRNN row. Sub-figures (b) and (c) are similarly built, except that models are trained for the intubation and the death outcomes, respectively. Although CondRNN seems to remain the best option for early prognosis in the non-Mexican data, high variability is observed in the results, preventing any confident comparison.

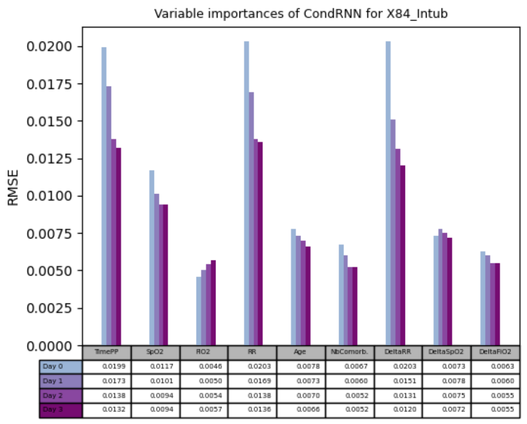
2.6.3. Feature importances in CondRNN

In this section, we present the feature importances in our CondRNN model. We took advantage of the 10 training-test outer splits of our NCV to apply input perturbation on each training set, and the overall variable importances are obtained by averaging the 10 resulting $RMSE$. This saved considerable running time while still providing generalized variable importances by going through multiple training sets. Figures 2.13, 2.14 and 2.15 show our CondRNN's feature importances of the Mexican, global and non-Mexican databases trained separately for each of three outcomes of interest. The mean and standard deviation of the Gaussian noise have been set to 0 and 0.1 respectively. Since the number of input features is $K = 9$ (4 longitudinal features and 5 static features, excluding the gender because it is binary) and $T = 3$, each figure consists of 9 $RMSE$ bar plots subdivided into 4 $RMSE$ bars with respect to step-wise outputs for Day 0, Day 1, Day 2 and Day 3 after enrolment.



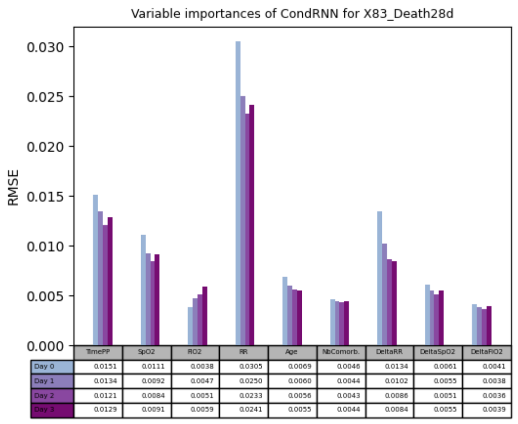
	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0223	0.0119	0.0030	0.0298	0.0056	0.0074	0.0216	0.0043	0.0039
Day 1	0.0177	0.0080	0.0030	0.0226	0.0038	0.0065	0.0129	0.0027	0.0030
Day 2	0.0108	0.0048	0.0022	0.0144	0.0029	0.0035	0.0085	0.0018	0.0017
Day 3	0.0101	0.0048	0.0024	0.0139	0.0031	0.0031	0.0080	0.0019	0.0018

(a) Primary outcome



	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0199	0.0117	0.0046	0.0203	0.0078	0.0067	0.0203	0.0073	0.0063
Day 1	0.0173	0.0101	0.0050	0.0169	0.0073	0.0060	0.0151	0.0078	0.0060
Day 2	0.0138	0.0094	0.0054	0.0138	0.0070	0.0052	0.0131	0.0075	0.0055
Day 3	0.0132	0.0094	0.0057	0.0136	0.0066	0.0052	0.0120	0.0072	0.0055

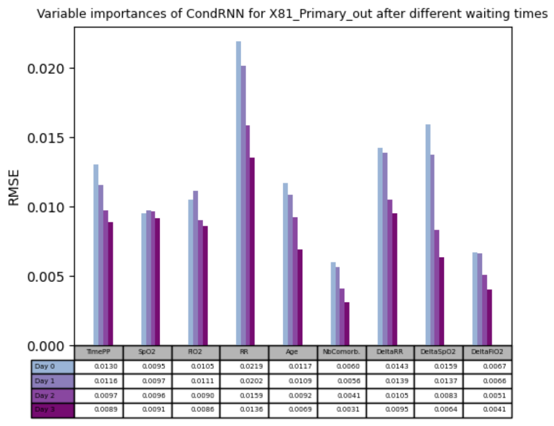
(b) Intubation outcome



	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0151	0.0111	0.0038	0.0305	0.0069	0.0046	0.0134	0.0061	0.0041
Day 1	0.0134	0.0092	0.0047	0.0250	0.0060	0.0044	0.0102	0.0055	0.0038
Day 2	0.0121	0.0084	0.0051	0.0233	0.0056	0.0043	0.0086	0.0051	0.0036
Day 3	0.0129	0.0091	0.0059	0.0241	0.0055	0.0044	0.0084	0.0055	0.0039

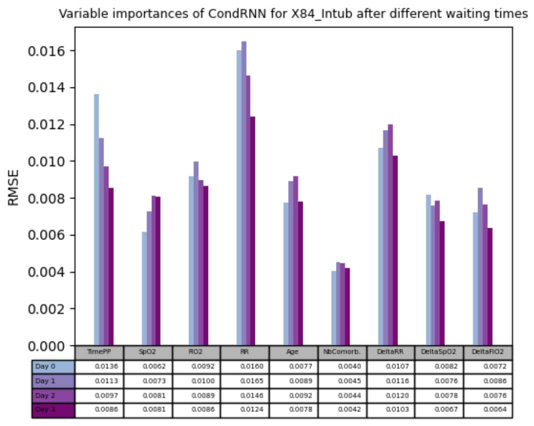
(c) Death outcome

Fig. 2.13. Variable importances of CondRNN trained on the Mexican database for the three different outcomes. (a) Given CondRNN trained for the primary outcome, each group of 4 bar plots (from light to darker colors) refers to the four values of the below column showing the RMSE of a specific feature (in grey) subject to Gaussian noise. Each bar color is associated with the RMSE between the step-wise original and perturbed predictions on the training data set: RMSE of the outputs at Day 0 (first row), Day 1 (second row), Day 2 (third row) or Day 3 (fourth row). (b) and (c) are built similarly, except that CondRNN is trained for the intubation and death outcomes, respectively.



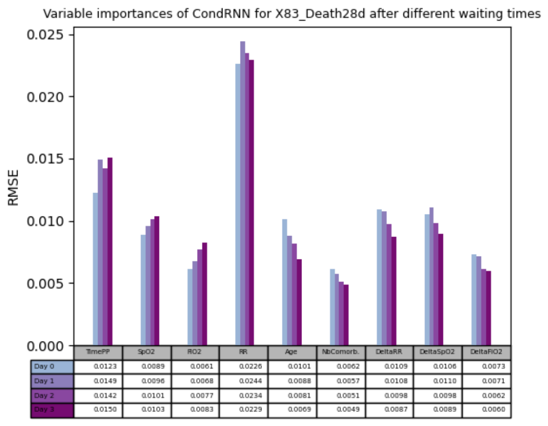
	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0130	0.0095	0.0105	0.0219	0.0117	0.0060	0.0143	0.0159	0.0067
Day 1	0.0116	0.0097	0.0111	0.0202	0.0109	0.0056	0.0139	0.0137	0.0066
Day 2	0.0097	0.0096	0.0090	0.0159	0.0092	0.0041	0.0105	0.0083	0.0051
Day 3	0.0089	0.0091	0.0086	0.0136	0.0069	0.0031	0.0095	0.0064	0.0041

(a) Primary outcome



	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0136	0.0062	0.0092	0.0160	0.0077	0.0040	0.0107	0.0082	0.0072
Day 1	0.0113	0.0073	0.0100	0.0165	0.0089	0.0045	0.0116	0.0076	0.0086
Day 2	0.0097	0.0081	0.0089	0.0146	0.0092	0.0044	0.0120	0.0078	0.0076
Day 3	0.0086	0.0081	0.0086	0.0124	0.0078	0.0042	0.0103	0.0067	0.0064

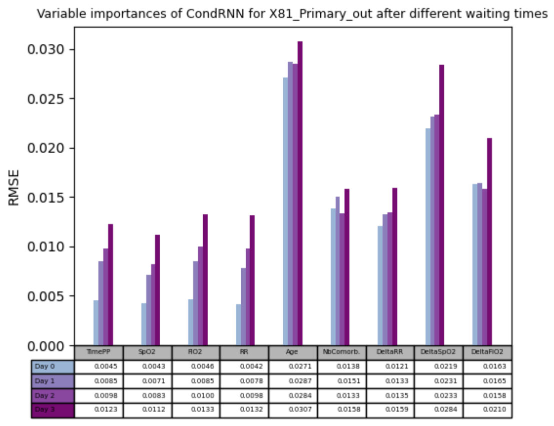
(b) Intubation outcome



	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0123	0.0089	0.0061	0.0226	0.0101	0.0062	0.0109	0.0106	0.0073
Day 1	0.0149	0.0096	0.0068	0.0244	0.0088	0.0057	0.0108	0.0110	0.0071
Day 2	0.0142	0.0101	0.0077	0.0234	0.0081	0.0051	0.0098	0.0098	0.0062
Day 3	0.0150	0.0103	0.0083	0.0229	0.0069	0.0049	0.0087	0.0089	0.0060

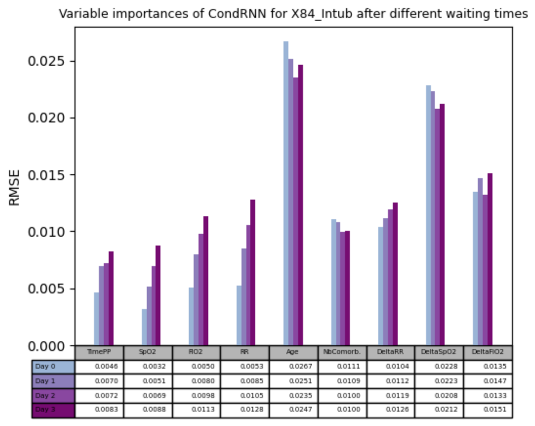
(c) Death outcome

Fig. 2.14. Variable importances of CondRNN trained on the global database for the three different outcomes. (a) Given CondRNN trained for the primary outcome, each group of 4 bar plots (from light to darker colors) refers to the four values of the below column showing the RMSE of a specific feature (in grey) subject to Gaussian noise. Each bar color is associated with the RMSE between the step-wise original and perturbed predictions on the training data set: RMSE of the outputs at Day 0 (first row), Day 1 (second row), Day 2 (third row) or Day 3 (fourth row). (b) and (c) are built similarly, except that CondRNN is trained for the intubation and death outcomes, respectively.



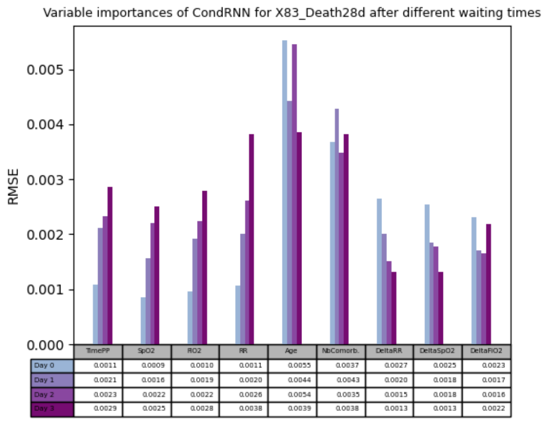
	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0045	0.0043	0.0046	0.0042	0.0271	0.0138	0.0121	0.0219	0.0163
Day 1	0.0085	0.0071	0.0085	0.0078	0.0287	0.0151	0.0133	0.0231	0.0165
Day 2	0.0098	0.0083	0.0100	0.0098	0.0284	0.0133	0.0135	0.0233	0.0158
Day 3	0.0123	0.0112	0.0133	0.0132	0.0307	0.0158	0.0159	0.0284	0.0210

(a) Primary outcome



	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0046	0.0032	0.0050	0.0053	0.0267	0.0111	0.0104	0.0228	0.0135
Day 1	0.0070	0.0051	0.0080	0.0085	0.0251	0.0109	0.0112	0.0223	0.0147
Day 2	0.0072	0.0069	0.0098	0.0105	0.0235	0.0100	0.0119	0.0208	0.0133
Day 3	0.0083	0.0088	0.0113	0.0128	0.0247	0.0100	0.0126	0.0212	0.0151

(b) Intubation outcome



	TimePP	SpO2	FiO2	RR	Age	NbComorb.	DeltaRR	DeltaSpO2	DeltaFiO2
Day 0	0.0011	0.0009	0.0010	0.0011	0.0055	0.0037	0.0027	0.0025	0.0023
Day 1	0.0021	0.0016	0.0019	0.0020	0.0044	0.0043	0.0020	0.0018	0.0017
Day 2	0.0023	0.0022	0.0022	0.0026	0.0054	0.0035	0.0015	0.0018	0.0016
Day 3	0.0029	0.0025	0.0028	0.0038	0.0039	0.0038	0.0013	0.0013	0.0022

(c) Death outcome

Fig. 2.15. Variable importances of CondRNN trained on the non-Mexican database for the three different outcomes. (a) Given CondRNN trained for the primary outcome, each group of 4 bar plots (from light to darker colors) refers to the four values of the below column showing the RMSE of a specific feature (in grey) subject to Gaussian noise. Each bar color is associated with the RMSE between the step-wise original and perturbed predictions on the training data set: RMSE of the outputs at Day 0 (first row), Day 1 (second row), Day 2 (third row) or Day 3 (fourth row). (b) and (c) are built similarly, except that CondRNN is trained for the intubation and death outcomes, respectively.

2.7. Discussion

This section discusses the results in Section 2.6 related to our proposed exploratory and predictive machine learning models (Sections 2.3 and 2.4) in order to demonstrate whether the latter provide clinically interesting insights into patients' APP treatment response, as well as an efficient and confident early prognosis of treatment failure.

2.7.1. Quality of the trajectory visualization with PHATE

The 2D Mexican PHATE embedding is organized in connected clusters forming four big branches allowing easy visualization of changes in terms of the six input variables when looking at the input coloring in parallel (Figure 2.4). In other words, we properly embed the original 6-dimensional state phase into the 2-dimensional Euclidean space in which patients' longitudinal features at time t are living and any patient's motion between t and t' can be explained in terms of the six input features through a simple 2-dimensional investigation (Figure 2.16). The four resulting clusters of trajectories (Figure 2.5) fully benefit from this embedding method. Indeed, since the four representative medoids stand sufficiently apart from each other and their motion follows the branching structure, we can immediately spot four significantly different APP responses within the Mexican population. As for the other embedding algorithms, we do not retrieve the same structural quality. Indeed, the PCA plot (Appendix A.6) does not show any interesting structure but a single big cluster where differences in terms of the six features are very hazily separated. t-SNE (Appendix A.8) manages to create clusters, but the connections between them is not well-defined and we end up with micro-clusters having their own specific structure, preventing from forming a coherent whole with smooth transitions. This may be due to the inherent purpose of its algorithm which only preserves locality [**139**] and is prone to fail our visualization since we are interested in global trends. UMAP (Appendix A.10) should fix it by preserving both local and global distances [**98**], but it results in an isolated cluster that also has its own structure and does not connect with the main cluster. Those structural issues inevitably impact the projected k-medoids clusters (Appendices A.7, A.9 and A.11), where none of the alternative embeddings plot trajectories following uniform patterns that show temporal trends associated with each group. We come to the conclusion

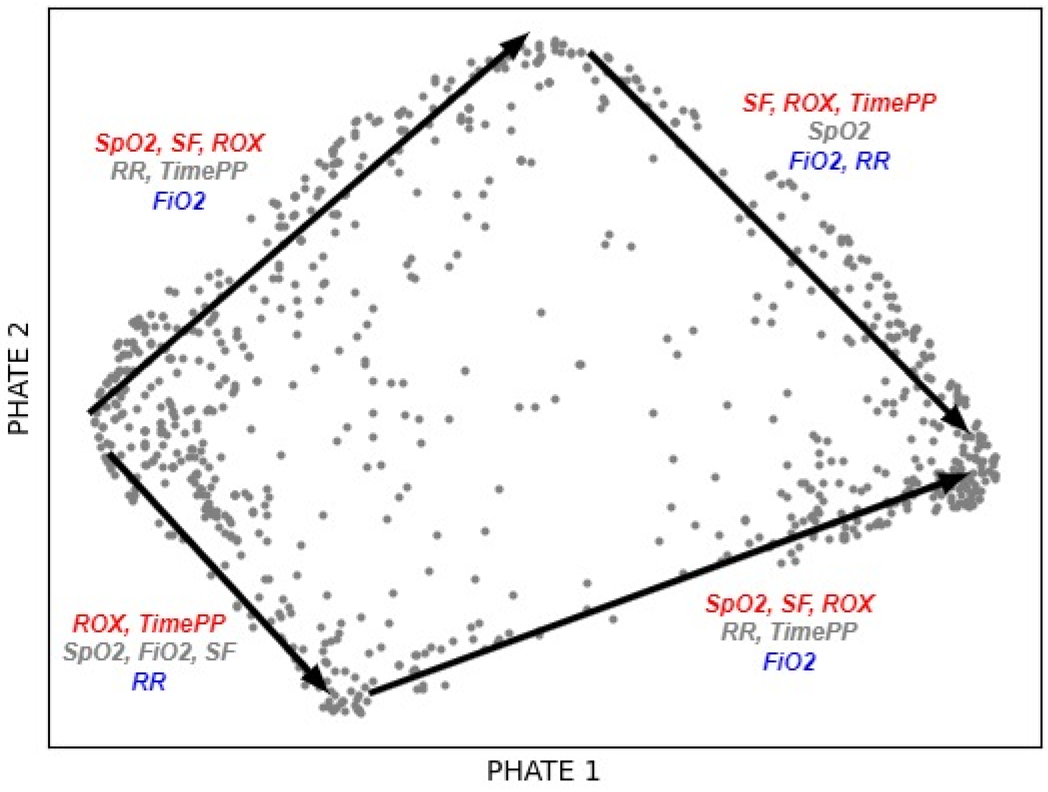


Fig. 2.16. The interpretation of the 2D PHATE embedding with respect to change in the six predictors along the branching structure. Red, grey and blue colors mean increase, stagnation and decrease of the corresponding features with the indicated direction. Progressions in terms of 6 features are embedded in 2 dimensions, a unique PHATE's property allowing an easy visualization of MTS data.

that PHATE is, by far, the most convenient method to visualize different high-dimensional APP responses.

Let us now discuss about the clinical meaning of the drawn trajectories with PHATE. There is a lot of information contained in the drawn trajectories and associated box plots, bar plots and tables, but some observations draw our attention, particularly in the four Mexican trajectories (Figure 2.5). Indeed, there are two pairs of time series clusters of balanced populations with very similar outcomes: clusters 0 and 2 fail the APP treatment with respectively 84.62% and 87.5% (see Table A.14 for a summary of the frequencies within each cluster) of positive primary outcomes (i.e. $Primary\_out$ =1), whereas clusters 1 and 3 are largely successful with APP treatment, with respectively 2.33% and 0%

of positive primary outcomes. At the same time, all the four clusters follow very distinctive progressions (see Figure 2.5). Cluster 0 consists of patients starting with the worst features (low $SpO2$, SF, ROX, and high $FiO2$, RR) that remain stable after enrolment. Conversely, cluster 3 mostly consists of patients starting in a very good condition (high $SpO2$, SF, ROX, and low $FiO2$ and RR) that remain stable after enrolment, even though some boundary patients start in a slightly worse condition from the top-right of the graph to evolve better along the right branch. Although those two clusters are not very insightful in the sense that they relate to typical grouped progressions leading to treatment failure and success, their presence allows us to retrieve what has been observed so far in the former APP study and assert the quality of the visualization. There is much more to say about the in-between clusters 1 and 2. Cluster 2 consists of people in average condition (high $SpO2$, medium $FiO2$, SF, ROX, and low $FiO2$ and RR) that progressively deteriorates towards the "sick" left branch and mostly fail the treatment. In contrast, patients in cluster 1 start in a worse condition than cluster 2 (much worse SF and even a slightly worse ROX), but their progressions along the bottom branch reflect a good response to APP despite their former very bad condition, and the treatment is successful for almost everybody. This difference in treatment response between clusters 1 and 2 is something that, according to I. Pavlov, has not been easily observed with the regressions of the former APP study and justifies the contribution brought by our proposed visualization. We also note that, except a slight increase in ROX, the progression of cluster 1 mostly translates to a gradual increase in $TimePP$ across post-enrolment days. Thus, the positive outcome of this group of patients has probably been uniquely determined by intensifying their APP treatment, which still needs to be clinically explained. Moreover, it is not clear what prevented the clinicians from intensifying APP for the cluster 2, because clusters 1 and 3 endure more and more $TimePP$ during post-enrolment days despite starting in both worse and better conditions than cluster 2. All those interesting interrogations related to the APP protocol and hidden secondary predictors define the main topic the future directions (see Section 2.8).

In the global drawn trajectories (Figure 2.7), we also establish what has been observed in the former APP study, especially the difference in the APP protocol between the Mexican patients and the others, while keeping the interesting Mexican trajectories. Indeed, cluster

1 consists of non-Mexican patients starting in very good condition (high $SpO2$, SF, ROX, and low $FiO2$ and RR) that quickly recover from the disease (10.38% of primary outcome, see Table A.17), whereas cluster 5 mostly consists of Mexican patients (probably from the cluster 3 of the Mexican plot) that also follow the same healthy trend, except that cluster 1 is not subject to a gradual increase in $TimePP$. That is, cluster 1 is the non-Mexican counterpart of cluster 5 and agrees with the former observation that APP is performed very differently in the Mexican and non-Mexican populations, where it is mostly associated with success in the Mexican population and with failure in the non-Mexican population. The non-Mexican cluster 4 follows the exact same trajectory as cluster 1 along the right branch, except that it starts from the bottom of the graph in a worse condition (lower SF and ROX) and stops its route in the middle of the right branch, which naturally reflects a slightly lower success (26% of primary outcome). Clusters 0 and 2 respectively refer to clusters 0 and 1 from the Mexican plot. Cluster 3 stands as the most unclear grouped trajectory, consisting of both Mexican and non-Mexican patients that do not follow a healthy trend and mostly fail the treatment. Since $TimePP$ is mitigated in this cluster, it should probably contain most of the cluster 2 from the Mexican plot, but $TimePP$ still remains higher than the healthy non-Mexican right branch, which does not contradict the observed correlation between high $TimePP$ and failure within the non-Mexican population. To sum up, despite the lack of data access, this global visualization with PHATE provides a compact and insightful macroscopic overview of sub-populations' progressions, and properly highlight huge differences in the APP protocol between the Mexican and non-Mexican cohort.

Finally, the lack of clear definition in the non-Mexican plot (Figure 2.9) prevents any confident interpretation. Indeed, apart from cluster 1 which represents people in originally very condition that greatly overcome the disease (3.33% of positive primary outcome, see Table A.20), the three others consist of mixed endpoints and their trajectories do not follow well-defined patterns along a branching structure. Thus, this population primarily needs further investigation regarding the APP protocol, and, possibly, a refinement of clustering.

2.7.2. Comparison of the four predictive models

In general, regarding the Mexican and global populations, we observe that the AUROC scores increase with the waiting time. This agrees with the clinical intuition that waiting for a while before determining the outcome of a patient provides a better prognosis since a patient's response in terms of longitudinal features during the first days of APP treatment should be a factor of success/failure. At the same time, the variances remain relatively low and decrease over time so we can make observations and comparisons with confidence.

Since the domain experts need a model to be accurate as soon as possible for a fast clinical decision, let us first draw attention to the performances for waiting times up to Day 0 and Day 1 post-enrolment, i.e. the first two time steps of the plots. In the Mexican and global plots, we observe that CondRNN outperforms all the other methods in all cases, except when trained for the death outcome where SimpleRNN does better after Day 1 (Figure 2.10 (c)) or right from enrolment day (Figure 2.11 (c)). When considering longer waiting times, CondRNN's scores are either similar to SimpleRNN or the highest ones, except for the death in the Mexican database where the SimpleRNN remains above (Figure 2.10 (c)). As for the Binary CondRNN, it never reaches the top score but still compete with the other methods despite the drastic binarization, especially in the global population. Finally, our non-binarized RNNs outperform kNN at each time step, especially after Day 0, which demonstrates the powerful use of more complex NN models over a straightforward kNN. Even though Binary CondRNN falls below kNN in some Mexican plots, this does not invalidate our latter conclusion since this performance gap is rather due to the input type than the model architecture.

APP is performed very differently on the non-Mexican population, where it seems to be applied according to the clinician diagnosis of each individual. This obviously compromises the construction of a general rule in terms of the dynamic variables that are directly affected to the ongoing APP protocol, and it reflects on the model performances. Indeed, AUROC scores are overall very low, even below the random classifier for the death outcome, and the extremely high variances do not even allow us to state a proper ranking. Moreover, the expected trend increasing with the waiting time disappears. The most relevant solution is to include additional variables that accurately explains how APP is applied to each patient, so that other dynamic features make more sense together with them.

To sum up, within the Mexican and global population, we have proven the benefit from implementing a trackable prognosis with standard and conditional RNNs over the baseline kNN, as well as the interest in incorporating static features along with dynamic ones by comparing CondRNN with SimpleRNN. Unfortunately, our Binary CondRNN fails to present as a reliable solution due to some lower scores than kNN in the Mexican population, unless a strong clinical importance is given to binarized inputs. This makes CondRNN the best overall candidate for early prognosis in the Mexican and global populations. Regarding the Mexican population, due to the high variability of the results, we cannot establish a fair ranking, and due to significantly lower scores than in the Mexican and global populations, we do not recommend any of the four models for a potential deployment in clinical environment.

2.7.3. Analysis of the important features in CondRNN

For the Mexican cohort (Figure 2.13), the hierarchy remains stable across the three outcomes, with RR, $TimePP$ and $DeltaRR$ as the leading features, followed by $FiO2$. This agrees with the former predictors study on the Mexican population, where $TimePP$ and ROX has been stated as being the most powerful variables [**68**]. Another interesting observation is the decreasing in $RMSE$ with the waiting time without altering the importance hierarchy. This means that the more we wait to deliver a prognosis, the less the resulting score is subject to uncertainty, which is exactly how we want the model to behave. For the non-Mexican cohort (Figure 2.15), due to the poor performance and high variability of the model, it is difficult to interpret the bar plot. One thing is however blindingly obvious: the importance of the dynamic variables is below the importance of the static variables. The only explanation we can provide is directly linked to the different APP protocol between the Mexican and non-Mexican populations. Indeed, when $TimePP$ is pushed to a maximum bearable, as within the Mexican population, $TimePP$ and the other dynamic features that are affected by the proning sessions become a fitness test that better evaluates the prognosis of a patient than other background features, because they directly estimate its physiological ability that may also depends on the age and other background features. APP is performed very differently on the non-Mexican population where APP seems to be performed with respect to each individual, which may justify this reversed scenario. Finally, the global population (Figure 2.14) is less concerned with the degenerated importances of the non-Mexican population because we find good signs of a confident early prognosis of patients subject to APP, such as RR being the most important features in all cases and and a relatively high importance of the other dynamic features. Still, static features are almost as important as the dynamic ones compared to the Mexican database and the hierarchy is much more hazy, raising doubts about the relevance of the main dynamic features and the stability of the model. In other words, we properly face in-between importance plots that coincide with our above-mentioned remarks regarding the Mexican and non-Mexican importance plots.

We conclude that our perturbation analysis builds enough confidence to clinically deploy CondRNN for the Mexican population, but not enough to deploy it for the non-Mexican and global populations.

2.8. Future directions

On the exploratory side, future directions involve going deeper into the explanation of the different trajectory groups; that is, clinically and biologically determining causal effects between some groups' features and trajectories. Since we do not have the necessary domain knowledge to assert the relevance of some differences in terms of features that may appear between those different groups of patients, we need input from domain experts. However, thanks to our detailed descriptions of each group in Appendices A.12, A.13, A.14, A.15, A.16, A.17, A.18, A.19 and A.20, we believe that we provided experts with enough tools to investigate in this direction. In addition, from a more macroscopic point-of-view, the highlighted differences in the APP protocol between the Mexican population and the rest of the cohort should be extensively investigated to determine if they have an influence on the patients' outcomes and head towards a local or global revision of the APP protocol. Last, and not least, we are looking forward to getting local access to the global and non-Mexican databases in order to have more control on them and facilitate exploration tasks that have been remotely performed with difficulty.

On the predictive side, since we were limited by computational resources, global data accessibility as well as the scope of our study, important aspects were not covered in our research. Considering the heavily applied dimension of our work, one of the most interesting subjects to deal with in depth is the deployment of our predictive model. In Section 2.4.6, we simply mentioned how we can set up a direct workable CondRNN for clinical use, and we did not actually provide an appropriate solution. In reality, setting up such a clinical ML software induces other domain complications that are far beyond the scope of our study, such as the integration into existing clinical workflows and infrastructure [**75**] that involves skills from experts in programming and system architecture. There also remains much to be done in making our system more interpretable. Indeed, the added Gaussian noise of our proposed approach in Section 2.4.5 may result in out-of-domain feature values for specific individuals and can lead to explanations that are not reflective of the systematic behavior of the model [**33**]. A lot of alternatives worth trying have been suggested to mitigate those downsides, such as Augmented Feature Occlusion (AFO) [**136**], where features are replaced with samples from the bootstrapped distribution $p(x_i)$ for each feature i instead of white noise perturbations to avoid generating out-of-distribution samples. Moreover, potential validation improvements naturally go along with all the inevitable assumptions of our NCV framework (see Section 2.4.3). For instance, grid search, although being a simple and exhaustive method, is extremely time consuming. Other methods, such as Randomized Search or Bayesian Optimization [**131**], have been proposed to fix this issue. Bergstra and Bengio [**19**] even found that neural networks configured by randomized search are able to determine models that are "as good or better within a small fraction of the computation time" compared to networks configured by pure grid search. Extending computational resources might also be interesting to fine tune parameters with a more detailed parameter grid or to reduce the bias of our NCV results with an increased number of folds. Finally, since missing values have an underlying meaning (see Section 2.4.2), Lipton et al. [**87**] suggest in their clinical study to complement padding imputation with binary indicators of missingness to get a more accurate distinction between imputed and non-imputed data.

Chapter 3

Multiple sclerosis long-term analysis: a more complex visualization case

3.1. Context and motivation

This chapter is another case study [**3**] about multiple sclerosis (MS). Accurately classifying the course of MS into sub-types is very important for communication, prognostication, design and recruitment of clinical trials, and treatment decision-making [**90**]. Physicians widely rely on the evaluation framework proposed by Lublin et al. [**90**] which categorizes MS in two dominant types: Relapsing-Remitting MS (RRMS) and Secondary Progression MS (SPMS). Other targets, such as worsening scores based on overall disease progression over some period of time, have also been used to classify patients' courses. However, none of these methods results in an accurate classification of MS because its course is well-known for being highly varied and unpredictable over time [**51**]. Moreover, any physician's retrospective evaluation is inherently subjective and limited to few longitudinal descriptors. Thus, there is a need to automatically group and visualize 15-dimensional MS patients' follow-up data in order to suggest a new or revised MS classification.

In this study, unlike the previous chapter, the purpose is not to rigorously demonstrate a practical use of ML on MTS medical data from start to finish. Focusing on the exploratory side, we are instead going to provide visual insights into the data by roughly following the same PHATE framework of Chapter 2, and dive into the unfortunate reality of ML in healthcare by observing the negative impact of highly irregular MTS database on our resulting visualizations.

3.2. Description of the database

The data consists of patients subject to a medical follow-up in which many features used to describe the course of MS disease have been recorded during clinical and radiological sessions after the disease onset. The database is split into two parts: the clinical database and the radiological database. The first contains a cohort of about 16 000 clinical recording days with respect to 600 patients whereas the second one is about 6 300 radiological recording days with respect to 1 500 patients. There are 9 clinical features:

- the Expanded Disability Status Scale (EDSS) feature, the most commonly used scale in MS patients. It provides effective and reliable evaluation at every stage of the disease. The scale ranges from 0 to 10 in half unit increments, and its computation is based on the clinical evaluation of 8 functional systems whose scores are provided by the 8 following features;
- the Pyramidal Score feature, which ranges from 0 to 5 in unit increments, evaluates muscle weakness or difficulty moving limbs;
- the Cerebellar Score feature, which ranges from 0 to 5 in unit increments, evaluates the loss of balance, coordination or tremor;
- the BrainStem Score feature, which ranges from 0 to 5 in unit increments, evaluates problems with speech, swallowing and nystagmus (uncontrolled, repetitive eye movements);
- the Sensory Score feature, which ranges from 0 to 6 in unit increments, evaluates superficial senses, vibration sensation and position sense;
- the Bowel Score feature, which ranges from 0 to 6 in unit increments, evaluates urinary retention, urgency and incontinence;
- the Visual Score feature, which ranges from 0 to 6 in unit increments, evaluates problems with sight such as visual acuity, visual field, and optical disc status;
- the Mental Score feature, which ranges from 0 to 5 in unit increments, evaluates problem with thinking and memory, such as depression, euphoria, mentation status and fatigue;
- the Ambulation Score, which ranges from 0 to 9 in unit increments, evaluates mobility following a 500-meter walking test.

Remark 3.2.1. *For more details related to the computation of EDSS from the 8 functional systems' scores, please refer to* [**124**].

The 6 radiological features describe different types of cerebral lesions in terms of their number and localization recorded during MRI sessions:

- the Number of Lesions T2 feature (integer number) counts the number of lesions using the T2 imaging method [**137**].
- the Number of New or Enlarging Lesions feature (integer number) counts new/enlarging lesions with reference to a baseline scan;
- the McDonald Lesions T2 feature, taking a categorical value among $\{\text{“0”}, \text{“1–2”}, \text{“3–8”}, \text{“9+”}\}$, counts the number of lesions using the T2 imaging method. Categories are used instead of numbers to fit the McDonald criteria for MS diagnosis [**95**];
- the McDonald Lesions Infratentorial feature, which takes a categorical value among $\{\text{“0”}, \text{“1+”}\}$, counts the number of lesions located in the infratentorial brain region. Categories are used instead of numbers to fit the McDonald criteria for MS diagnosis;
- the McDonald Lesions Juxtacortical feature, which takes a categorical value among $\{\text{“0”}, \text{“1+”}\}$, counts the number of lesions located in the juxtacortical brain region. Categories are used instead of numbers to fit the McDonald criteria for MS diagnosis;
- the McDonald Lesions Periventricular feature, which takes a categorical value among $\{\text{“0”}, \text{“1–2”}, \text{“3+”}\}$, counts the number of lesions located in the periventricular brain region. Categories are used instead of numbers to fit the McDonald criteria for MS diagnosis.

Along with those time varying features, some MS outcomes commonly used to classify MS severity based on historical clinical and radiological data after disease onset have been associated with each patient:

- the type of MS: either Relapsing-Remitting MS (RRMS) or Secondary Progressive MS (SPMS). According to the official classification provided by Lublin et al. [**90**], RRMS is a type of MS with relapses (symptoms getting worse) followed by recovery, without overall disease worsening. This form can (but not always) progress to

SPMS during which neurological symptoms continue to get worse without clear periods of remission. In that case, SPMS type is assigned to the patient;

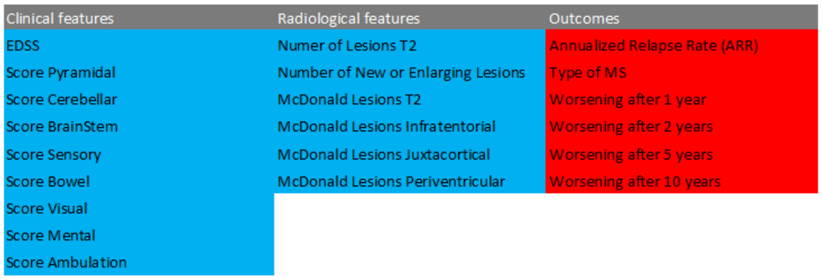
– the Annualized Relapse Rate (ARR). In many clinical trials, the relapse frequency, measured by the ARR, is a primary endpoint, because relapses are meaningful as they reflect how individuals experience fluctuations in symptoms that are central to relapsing MS;

– the worsening after one, two, five and ten years since disease onset. The worsening $\mathcal{W}_i$ after a period of i years is a clinically motivated binary (0/1) worsening score, defined from EDSS as

$$\mathcal{W}_i = \begin{cases} 1 \text{ if } \Delta_{EDSS} \geq 1 \text{ and } EDSS_{t_0} \leq 5.5 \\ 1 \text{ if } \Delta_{EDSS} \geq 0.5 \text{ and } EDSS_{t_0} > 5.5 \\ 0 \text{ otherwise} \end{cases}$$

where $\Delta_{EDSS} = EDSS_{t_0} - EDSS_{t_i}$. In the database, only worsening labels after $i = 1, 2, 5, 10$ years have been included, and we refer to $\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_5$ and $\mathcal{W}_{10}$ as $wors1year$, $wors2years$, $wors5years$ and $wors10years$ respectively throughout this study.

Figure 3.1 summarizes all the input and target variables.



Clinical features	Radiological features	Outcomes
EDSS	Numer of Lesions T2	Annualized Relapse Rate (ARR)
Score Pyramidal	Number of New or Enlarging Lesions	Type of MS
Score Cerebellar	McDonald Lesions T2	Worsening after 1 year
Score BrainStem	McDonald Lesions Infratentorial	Worsening after 2 years
Score Sensory	McDonald Lesions Juxtacortical	Worsening after 5 years
Score Bowel	McDonald Lesions Periventricular	Worsening after 10 years
Score Visual		
Score Mental		
Score Ambulation		

Fig. 3.1. Table of the most relevant input and target features according to domain experts' knowledge. In blue: longitudinal features that are meant to be inputted into the PHATE algorithm. In red: the target features used to discover correlations with groups of patients' long-term MTS.

3.3. Methodology

3.3.1. Data processing

As stated in Section 2.3.1, we want to input as many relevant longitudinal features as possible in our PHATE embedding. This is why we start with all the clinical and radiological features of Figure 3.1 in blue color. Some features are still missing among patients' clinical and radiological recording days. We impute missing clinical features with linear interpolation [**5**] because we have no particular assumption related to their existence and this is the most simple and straightforward way to guess a value with respect to the temporal gap between the two nearest available values. However, this method does not apply to MRI features because most of them are strictly categorical. For instance, McDonald Lesions T2 are categorized as $\{$"0", "1–2", "3–8", "9+"$\}$, and McDonald Lesions Infratentorial simply as $\{$"0", "1+"$\}$. Fortunately, thanks to their natural ordering, the trick is to map each category into its respective integer in $\{0, 1, 2, \ldots\}$, apply linear interpolation and round the imputed value to the nearest integer. This is far from being the smartest imputation method, but this is the best one we found that keeps enough data for now.

Moreover, the whole database is split into two sets because the clinical and radiological recording days do not take place at the same time. Also, patients do not completely intersect the two splits. To make each recording day a combination of both clinical and radiological features, domain experts recommended to build the combined data from the clinical sessions and associate them with the nearest radiological sessions that happened no more than 6 months apart. Clinical sessions without associated radiological exam are removed from the data set.

Finally, time between patients is normalized by setting the origin t_0 to the date of disease onset. In other words, the time t of a patient's recording day corresponds to the number of days between the date of the session and date of disease onset. Thus, t refers to the days since disease onset (ddo), and we also include it into the set of input features in order to accentuate the temporal structure in the PHATE embedding.

To sum up, our data processing results in a similar multi-dimensional time series database $\mathbf{X}$ as in the previous chapter, $\mathbf{X} = \{X_i\}_{i=1}^N$ except that $X_i \subset \mathbb{R}^K$ could be unevenly spaced time series of different lengths. We have a total of 1606 recording days with respect to $N = 197$ patients, and $K = 16$ input features. In matrix notation, the flattened data has the form $\mathbf{X}_{flattened} \in \mathbb{R}^{1606\times 16}$ such that each row is a patient's recording day in terms of the 15 clinical and radiological features as well as the ddo. As usual, each column is Z-transformed before the production of the 2D PHATE embedding.

3.3.2. PHATE trajectory visualization with k-medoids and sliding windows

Since the available data is far more complex than in the previous chapter (larger database, more missing data, longer time periods...) and multiple sclerosis is in itself a challenge, we focus on the visualization of the database to explain and understand long-term outcomes related to MS. More specifically, in terms of both clinical and radiological features, we basically want to discover distinct temporal patterns of groups of patients that correlate with different outcomes. To this end, we are mostly going to follow the same visualization framework as in the APP chapter, including the production of a 2D PHATE embedding for $\mathbf{X}_{flattened}$ and the projection of clusters of trajectories with DTW k-medoids. However, we will quickly figure out that our previous visualization framework does not fit this database. Instead of grouping trajectories in a totally unsupervised way and associating clusters with target features, we suggest to directly make use of patients' targets of interest (worsening, ARR...) to first separate the population in different groups and then try to describe their respective courses over time with the aim of spotting distinct trends. As depicted in Figure 3.2, this is done with a sliding window approach, in which we define a window size $\mathbf{w}$ (in years) and a step size $\mathbf{s} \leq \mathbf{w}$ causing an overlap percentage between windows. Those two parameters generate a sequence of windows $W_0, W_{\mathbf{s}}, W_{2\mathbf{s}}, \ldots$ that average 2D PHATE coordinates of the patients' recording days falling into their respective window time period and plot them along with the former 2D PHATE embedding.

3.3.3. Supervised visualization with RF-PHATE

RF-PHATE [**119**] is a supervised visualization technique inspired by the original unsupervised PHATE [**102**] and Random Forests (RF) [**23**]. The RF-PHATE algorithm is similar to PHATE (Figure 1.8) except that it builds the affinity matrix from random forest proximities [**119**]. The random forest proximity between a pair of instances x and y is obtained

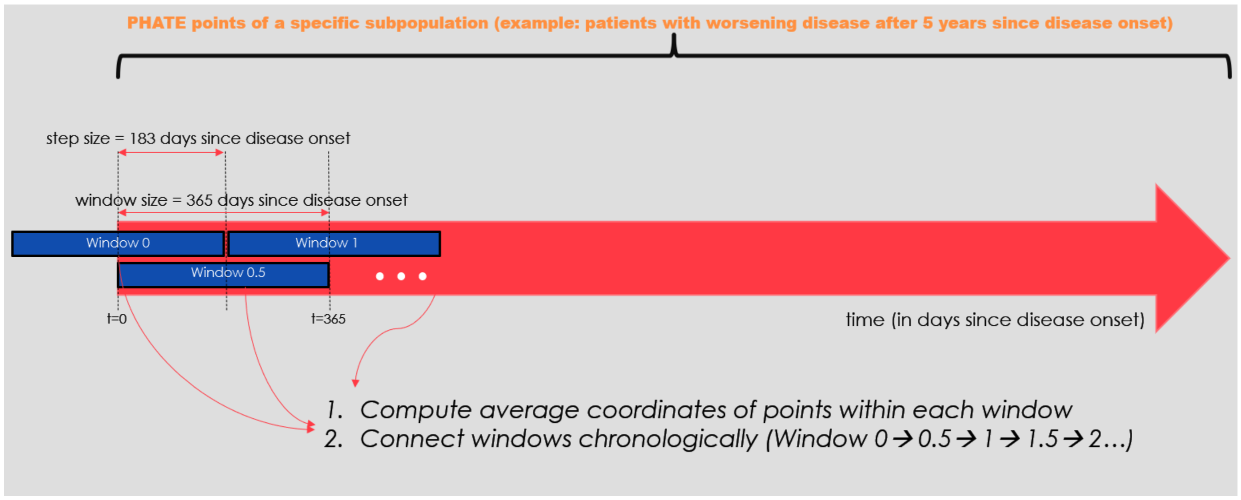

PHATE points of a specific subpopulation (example: patients with worsening disease after 5 years since disease onset)
step size = 183 days since disease onset
window size = 365 days since disease onset
Window 0
Window 1
Window 0.5
t=0
t=365
time (in days since disease onset)
1. Compute average coordinates of points within each window
2. Connect windows chronologically (Window 0 → 0.5 → 1 → 1.5 → 2...)

Fig. 3.2. Scheme of the computation of a group's average trajectory with a sliding window approach. For the purpose of the example, the window and step sizes have been set to $\mathbf{w} = 365$ days since disease onset and $\mathbf{s} = 183$ days since disease onset respectively. The connected sequence of averaged windows $W_0 \to W_{0.5} \to \ldots \to W_5$ corresponds to the global trajectory of the $wors5years = 1$ sub-population up to 5 years after disease onset.

by inputting them into a trained RF and measuring the proportions of trees where x and y share the same terminal node. This approach naturally takes advantage of an outcome of interest, because patients sharing the same label are prone to be similar in term of RF proximity. Since it has also been empirically checked that RF-PHATE retains both the local and global structure of the data while emphasizing the important variables in the visualization [**119**], it should mitigate the above-mentioned negative effects of some categorical inputs without neglecting the structure induced by the other important variables. To this end, we propose to transform $\mathbf{X}_{flattened}$ along with the corresponding $wors5years$ labels using the RF-PHATE algorithm.

3.4. Results

The 2D PHATE embedding colored by inputs and the projected DTW k-medoids with $k \in \{2, 3, 4\}$ are shown in Appendix B.1 and Figure 3.3. Figures 3.4 and 3.5 show average trajectories with sliding windows of the two groups of patients $wors5years = 0, 1$ using PHATE and RF-PHATE embeddings, respectively.

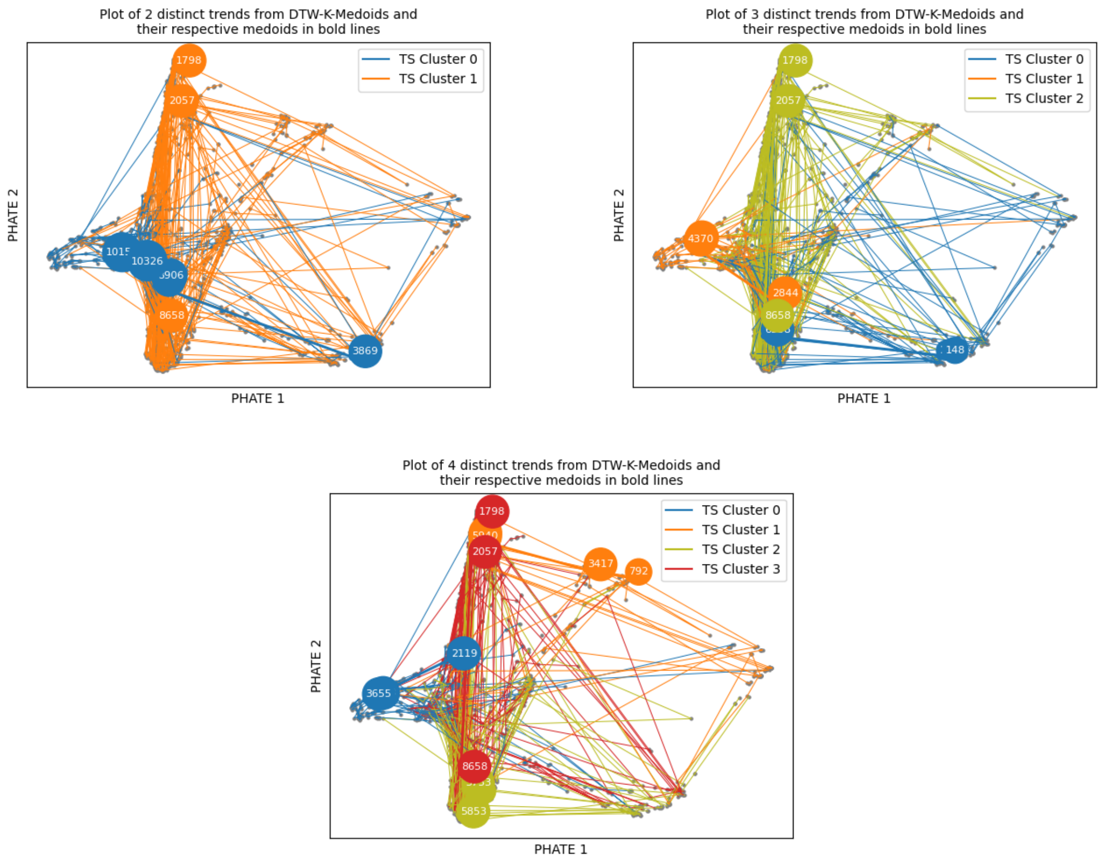


Fig. 3.3. DTW k-medoids clusters of trajectories projected on the 2D MS PHATE embedding with $k = 2$ (top left), $k = 3$ (top right) and $k = 4$ (bottom center). Medoids are in bold lines and labeled according to the ddo of their time steps. None of those attempts result in a satisfactory visualization of temporal trends.

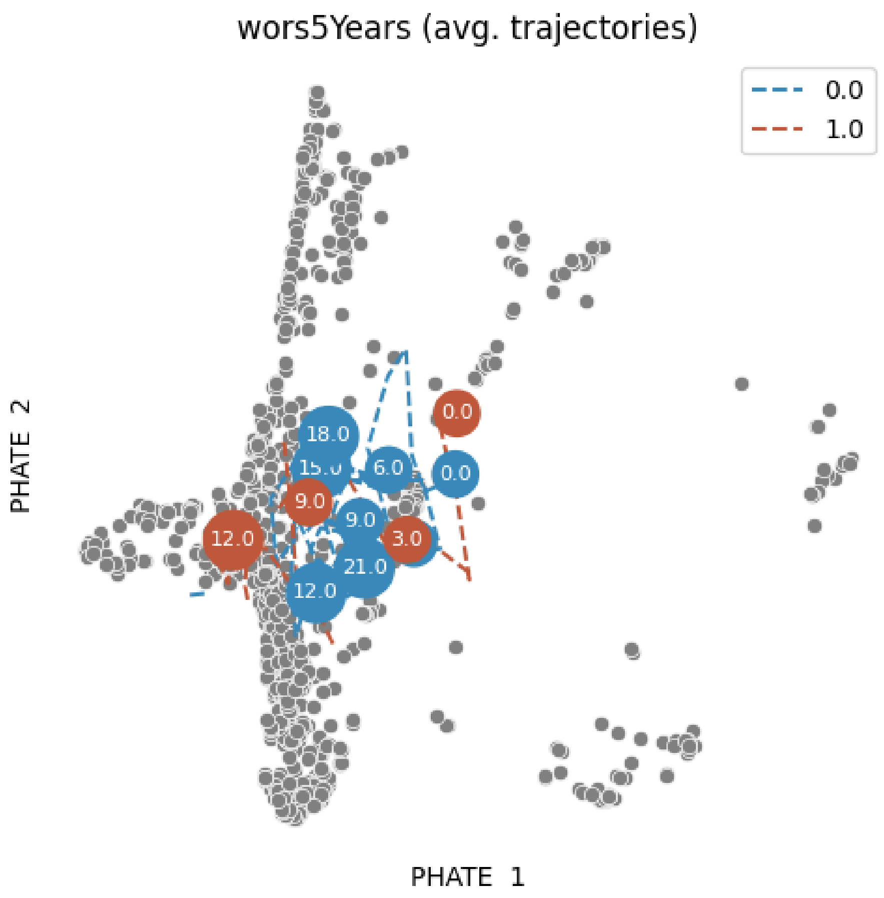


Fig. 3.4. Average PHATE trajectories with sliding windows of the two $wors5years$ groups of patients. The dotted red line plots the average trajectory of the group undergoing disease worsening after 5 years (i.e. $wors5years = 1$). The dotted blue line plots the average trajectory of the group without disease worsening after 5 years (i.e. $wors5years = 0$). Labels represent the time on which each window is centered, in years since disease onset.

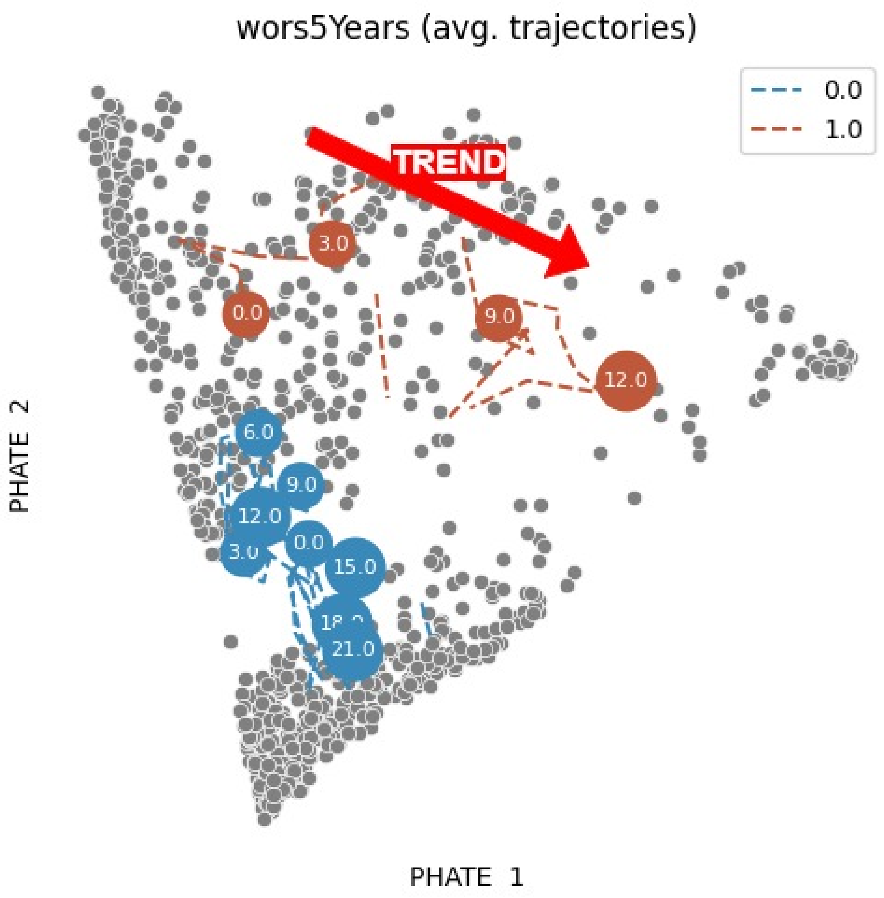


Fig. 3.5. Average RF-PHATE trajectories with sliding windows of the two $wors5years$ groups of patients. The dotted red line plots the average trajectory of the group undergoing disease worsening after 5 years (i.e. $wors5years = 1$). The dotted blue line plots the average trajectory of the group without disease worsening after 5 years (i.e. $wors5years = 0$). Labels represent the time on which each window is centered, in years since disease onset. The big red arrow highlights a temporal trend of the worsening group that was previously undetectable with a standard PHATE plot.

3.5. Discussion

Despite attempts with different k values, there is a strong lack of interpretation regarding the plotted groups of trajectories. Indeed, there is no clear trend following the branching structure of the embedding and clusters tend to mix up. Moreover, medoids are definitely not suitable to reflect long-term groups of trajectories, because they may only consist of very few time steps. Indeed, the longest medoid comprises four time steps and the shortest one only two (Figure 3.3 with $k = 3$, cluster 1). Obviously, we cannot study temporal trends over 5, 10 or even 20 years if we know so few time steps.

The average trajectories with sliding windows are also questionable: they are not really distinguished from each other and no trend in terms of features is easily detectable when looking at the input coloring in parallel (Figure B.1), especially the blue trajectory that looks like a compact block. We believe that one of the main reasons behind those spurious trajectories is the existence of categorical features among the inputs. By its nature, the PHATE algorithm leans towards continuous variables to properly diffuse distances and reflect smooth transitions. If we analyze the input feature coloring, we observe that the top and bottom branches only differ in terms of the binary MRI feature McDonald Lesions Infratentorial; yet, the proportions of $wors5years$ is identical between the two branches (Figure 3.6). This indicates that PHATE overemphasizes on the importance of feature McDonald Lesions Infratentorial that is not much correlated with $wors5Years$. Thus, PHATE spreads class labels artificially, disturbing the visualization of global trends with sliding windows in Figure 3.4.

We observe that RF-PHATE does help keep only the most correlated variables with the target of interest. For instance, McDonald Lesions Infratentorial have no structural consequences (Figure B.2) while the $wors5years$ outcome clearly divides its two labels among the right and bottom branches (Appendix B.3). This reflects a proper elimination of uncorrelated variables, resulting in a more visible temporal trend of the worsening population (Figure 3.5). Similarly to the last chapter, we also provide a visualization of the average group trajectories using the three common unsupervised alternatives: PCA, t-SNE and UMAP (Appendix B.4). In addition, we make use of the supervised counterparts of PCA and UMAP, namely Supervised PCA [**16**] and Supervised UMAP [**50**] (Appendix B.5), as we did with PHATE and RF-PHATE. We observe that none of these alternatives results

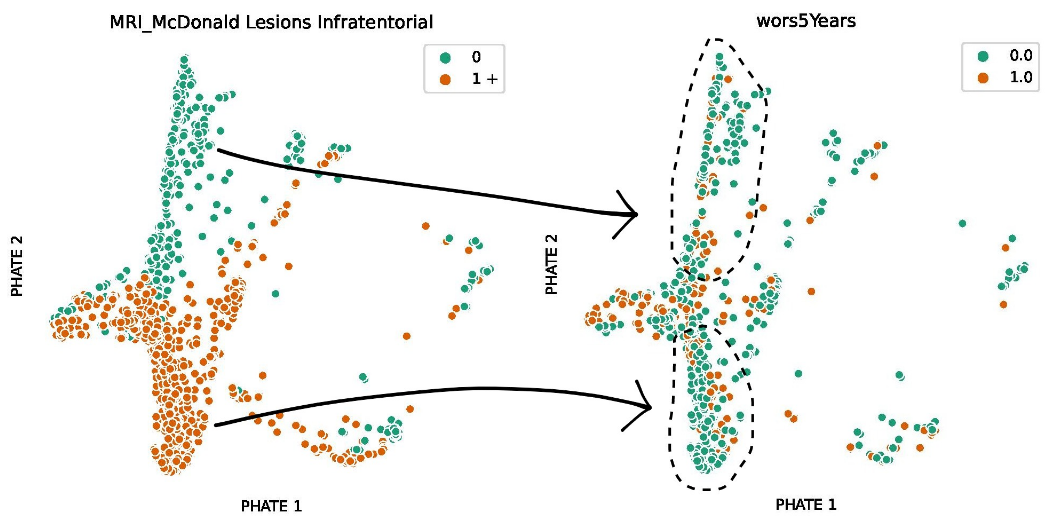


Fig. 3.6. Illustration of the consequences of categorical features on the PHATE branching structure. Two big branches only differ in this category, but the $wors5years$ distribution remains the same within each branch.

in a clear distinction between the worsening and no worsening groups, except Supervised UMAP. However, the latter leans so much towards group separation and compactness that we cannot associate the worsening group with a spatio-temporal pattern. In other words, RF-PHATE remains the most promising approach to visualize trends with respect to sub-populations of interest.

Even though the drawn trajectories along with RF-PHATE give us a hint about a potential interesting trend, they are still subject to high variability because they only rely on spatial averages. Appendix B.6 visually demonstrates this high variability within each $wors5years$ group by plotting individual colored trajectories on the RF-PHATE embedding. In particular, we observe that many worsening patients do not follow the red trend depicted in Figure 3.5. Therefore, we cannot interpret the resulting trends with confidence and need to find a more stable representation of each window.

3.6. Experimental setup

We have experimented within the same setup environment as in Section 2.5, except the addition of the `rf-phate` [**119**] Python package related to the supervised RF-PHATE plots.

3.7. Future directions

A beautiful way to overcome the inherent complexity of our database is to make use of Topological Data Analysis (TDA) (see definition in Appendix B.2.1). TDA presents as a solid future direction thanks to its natural ability to provide a denoised representation of a specific group of patients' trajectories by computing an abstract object, called *persistence diagram*, on each sliding window. Indeed, the fundamental stability theorem [**40**] ensures that the topological representation of the data through persistence diagrams remains the same in the presence of small perturbations. Moreover, the production of a persistence diagram for each window results in sliding windows that are much more representative of a grouped motion. To illustrate, if a window contained a lot of very similar vectors for patient a but one single vector for patient b, which is very likely to happen because sessions are unequally recorded among patients, vectors of patients a would be seen as noisy points and imply topological features of short lifetime. In other words, unlike our above-mentioned averaged trajectories, patients a and b would equally contribute to the production of the window's persistence diagram despite the fact the window is overwhelmed by patient a's recording days. A sketched idea of a future practical use is depicted in Appendix B.2.2.

Conclusion

From our positive results regarding the greater ability of PHATE and conditional neural networks over more traditional methods to respectively visualize sick patients' multivariate progressions and provide them with a confident early prognosis, we have demonstrated the proficiency of recent ML research with MTS data in healthcare. More specifically, on the exploratory side in Chapter 2, we have provided domain experts with a 2-dimensional PHATE visualization of trajectory clusters to improve the understanding of patients' responses to APP treatment in terms of 6 clinical factors simultaneously, and we have shown that other popular embedding methods do not perform as well as PHATE in this task. On the predictive side in Chapter 2, we have empirically proven the benefits of the flexibility of recurrent neural networks to provide real-time likelihood scores related to incoming patients' outcomes based upon both longitudinal and background features by showing that a conditional variant of a RNN classifier outperforms kNN and traditional RNN in the Mexican and global populations. Moreover, input feature perturbations in our conditional RNN mostly retrieved the hierarchy of the most important features that have been clinically established by domain experts for early prognosis. Unfortunately, high heterogeneity within the non-Mexican cohort prevented any conclusive exploratory and predictive results. Finally, we faced the reality on the ground with strong data-related constraints in Chapter 3 confirming that there is no universal formula of a successful ML framework and that it is rather a matter of fitting each specific study case independently. This conclusion originates from a noisy, non-distinguishable visualization of MS patients' trajectories using the same PHATE approach as in Chapter 2. However, a supervised version of this visualization approach, based upon RF-PHATE and sliding windows, gave us a hint about potential multi-factorial MS trends. To sum up, we contributed to the ongoing development of AI-based systems in healthcare relying upon strong scientific foundations that have nothing to do with “black magic”, and which are rather meant to augment physicians instead of replacing the essential physician–patient relationship [**6**]. Still, the significance of our work should be received with modesty: we do not pretend to revolutionize ML or overvalue its benefits, and the bridge between ML and the reality of its initial purpose in healthcare is far from being completed [**89**].

Among exploratory future directions, besides the obvious need to solve data accessibility issues, box plots, bar plots and tables in Appendix A related to the APP visualization in Chapter 2 would make domain experts able to discover potential background factors that could explain the different APP responses depicted by the detected patients' trajectory clusters. Moreover, PHATE highlighted a huge difference in APP treatment between the Mexican population and the rest of the cohort, pointing out further investigation into the explanation and the consequences of this difference in order to suggest an improved APP protocol. Because of the flexibility and complexity of our NCV framework and conditional RNN model, our predictive results are also subject to many future improvements. For instance, the interpretability could be extended by using Augmented Feature Occlusion [**136**] or more advanced methods such as Shapley Additive Explanations [**91**], which have been specifically developed to better reflect feature importances and recently used in some ML medical studies [**109**]. Randomized Search or Bayesian Optimization [**131**] are strong alternatives to Grid Search hyperparameter optimization to find as good or better combinations within a smaller fraction of time. In addition, expert knowledge in programming and system architecture will be required to effectively deploy our predictive model in a clinical environment, an essential step for practical use. Another interesting direction would be to take advantage of PHATE's ability to deal with time-varying data with geometry regularized autoencoders [**43**] in order to simulate PHATE embeddings for new incoming instances and potentially improve RNNs' performance by considering PHATE embeddings as input. Pre-processing could also increase model performance thanks to indicators of missingness proposed by Lipton et al. [**87**]. Ultimately, the mitigated conclusions in Chapter 3 will hopefully lead us to a promising future direction through Persistent Homology which appears to be a more generalizable approach as it naturally lowers the strong data-related constraints and allows a more reliable data analysis. Even though we introduced Topological Data Analysis in Chapter 3 as a circumstantial solution, there is much more to do in this direction, such as a robust and denoised visualization of MS trajectory clusters through persistence landscapes (Appendix B.2.3).

Disclosure of Potential Conflicts of Interest

Parts of this thesis are the result of a personal involvement in existing projects initiated by other researchers whose contributions are cited to the greatest possible extend. Still, some references may be missing due to unpublished work in progress. To ensure full integrity, all my collaborators and I declare no conflict of interest.

References

[1] *Partitioning Around Medoids (Program PAM)*, chapter 2, pages 68–125. John Wiley and Sons, Ltd, 1990.

[2] Towards trustable machine learning. *Nature Biomedical Engineering*, 2(10):709–710, Oct 2018.

[3] P004: Can t cell migratory signature predict ms disease progression? *Multiple Sclerosis Journal*, 26(1_suppl):16–89, 2020.

[4] Amaia Abanda, Usue Mori, and Jose A. Lozano. A review on distance based time series classification. *Data Mining and Knowledge Discovery*, 33(2):378–412, Mar 2019.

[5] Mohd Mustafa Al Bakri Abdullah. Filling missing data using interpolation methods: Study on the effect of fitting distribution. *Key Engineering Materials*, 594-595:889–895, 01 2014.

[6] Abhimanyu S. Ahuja. The impact of artificial intelligence in medicine on the future role of the physician. *PeerJ*, 7, 2019.

[7] Zeynettin Akkus, Alfiia Galimzianova, Assaf Hoogi, Daniel L. Rubin, and Bradley J. Erickson. Deep learning for brain mri segmentation: State of the art and future directions. *Journal of Digital Imaging*, 30(4):449–459, Aug 2017.

[8] Ana Alberdi-Iglesias, Francisco Martín-Rodríguez, Guillermo Ortega Rabbione, Ana I. Rubio-Babiano, María G. Núñez-Toste, Ancor Sanz-García, Carlos del Pozo Vegas, Miguel A. Castro Villamor, José L. Martín-Conty, Cristina Jorge-Soto, and Raúl López-Izquierdo. Role of spo2/fio2 ratio and rox index in predicting early invasive mechanical ventilation in covid-19. a pragmatic, retrospective, multi-center study. *Biomedicines*, 9(8), 2021.

[9] Eskandarinia Alireza, Nazarpour Hadi, Mehdi Teimouri, and Mirkhalegh Ahmadi. Comparison of neural network and k-nearest neighbor methods in daily flow forecasting. *Journal of Applied Sciences*, 10, 11 2010.

[10] Kartik Anand, Ginestra Bianconi, and Simone Severini. Shannon and von neumann entropy of random networks with heterogeneous expected degree. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 83:036109, 03 2011.

[11] Preeti Arora, Deepali, and Shipra Varshney. Analysis of k-means and k-medoids algorithm for big data. *Procedia Computer Science*, 78:507–512, 2016. 1st International Conference on Information Security and Privacy 2015.

[12] Francisco Azuaje, Ian Witten, and Frank E. Witten ih, frank e: Data mining: Practical machine learning tools and techniques. *Biomedical Engineering Online - BIOMED ENG ONLINE*, 5:1–2, 01 2006.

[13] Kevin Bache, Dennis DeCoste, and Padhraic Smyth. Hot swapping for online adaptation of optimization hyperparameters. 12 2014.

[14] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31, 05 2017.

[15] Joseph Barker, David Koeckerling, Nadeesha Lakmal Mudalige, Oluwatobiloba Oyefeso, and Daniel Pan. Awake prone positioning for patients with covid-19. *BMJ*, 376, 2022.

[16] Elnaz Barshan, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, 44(7):1357–1371, 2011.

[17] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. *CoRR*, abs/1206.5533, 2012.

[18] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[19] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(null):281–305, feb 2012.

[20] Daniel Berrar. *Cross-Validation*. 01 2018.

[21] Johannes Blömer, Christiane Lammersen, Melanie Schmidt, and Christian Sohler. *Theoretical Analysis of the k-Means Algorithm – A Survey*, pages 81–116. Springer International Publishing, Cham, 2016.

[22] Gavin J Bowden, Holger R Maier, and Graeme C Dandy. Optimal division of data for neural network models in water resources applications. *Water resources research*, 38(2):2–1, 2002.

[23] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, oct 2001.

[24] Thomas Breuel. The effects of hyperparameters on sgd training of neural networks. 08 2015.

[25] N. Brich, C. Schulz, J. Peter, W. Klingert, M. Schenk, D. Weiskopf, and M. Krone. Visual analytics of multivariate intensive care time series data. *Computer Graphics Forum*, n/a(n/a), 2022.

[26] Rickard Brüel-Gabrielsson, Vignesh Ganapathi-Subramanian, Primoz Skraba, and Leonidas J. Guibas. Topology-aware surface reconstruction for point clouds. *Computer Graphics Forum*, 39(5):197–207, 2020.

[27] Peter Bubenik. Statistical topological data analysis using persistence landscapes. *J. Mach. Learn. Res.*, 16(1):77–102, jan 2015.

[28] Mat Buckland and Mark Collins. *AI Techniques for Game Programming*. Premier Press, 2002.

[29] Kristy Carpenter, David Cohen, Juliet Jarrell, and Xudong Huang. Deep learning and virtual drug screening. *Future Medicinal Chemistry*, 10, 10 2018.

[30] Carmelo Cassisi, Placido Montalto, Marco Aliotta, Andrea Cannata, and Alfredo Pulvirenti. *Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining*. 09 2012.

[31] Gavin C. Cawley and Nicola L.C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *J. Mach. Learn. Res.*, 11:2079–2107, aug 2010.

[32] J.M. Chambers, W.S. Cleveland, B. Kleiner, and P.A. Tukey. *Graphical Methods for Data Analysis*. Chapman & Hall statistics series. Wadsworth International Group, 1983.

[33] Chun-Hao Chang, Elliot Creager, Anna Goldenberg, and David Duvenaud. Explaining Image Classifiers by Counterfactual Generation. *arXiv e-prints*, page arXiv:1807.08024, July 2018.

[34] Peter H. Charlton and Vaidotas Marozas. 12 - wearable photoplethysmography devices. In John Allen and Panicos Kyriacou, editors, *Photoplethysmography*, pages 401–439. Academic Press, 2022.

[35] Robert Chen, Walter F Stewart, Jimeng Sun, Kenney Ng, and Xiaowei Yan. Recurrent neural networks for early detection of heart failure from longitudinal electronic health record data: Implications for temporal modeling with respect to time before diagnosis, data density, data quantity, and data type. *Circ Cardiovasc Qual Outcomes*, 12(10):e005114, October 2019.

[36] KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259, 2014.

[37] Dami Choi, Christopher Shallue, Zachary Nado, Jaehoon Lee, Chris Maddison, and George Dahl. On empirical comparisons of optimizers for deep learning. 10 2019.

[38] Francois Chollet. *Deep Learning with Python*. Manning Publications Co., USA, 1st edition, 2017.

[39] Francois Chollet et al. Keras, 2015.

[40] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete & computational geometry*, 37(1):103–120, 2007.

[41] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006. Special Issue: Diffusion Maps and Wavelets.

[42] Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri. A comprehensive survey and performance analysis of activation functions in deep learning. *CoRR*, abs/2109.14545, 2021.

[43] Andres F. Duque, Sacha Morin, Guy Wolf, and Kevin Moon. Extendable and invertible manifold learning with geometry regularized autoencoders. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, dec 2020.

[44] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 454–463, 2000.

[45] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[46] Cristobal Esteban, Oliver Staeck, Stephan Baier, Yinchong Yang, and Volker Tresp. Predicting clinical events by combining static and dynamic information using recurrent neural networks. pages 93–101, 10 2016.

[47] Michael Fralick, Michael Colacci, Laveena Munshi, Kevin Venus, Lee Fidler, Haseena Hussein, Karen Britto, Rob Fowler, Bruno R Da Costa, Irfan Dhalla, et al. Prone positioning of patients with moderate hypoxaemia due to covid-19: multicentre pragmatic randomised trial (covid-prone). *bmj*, 376, 2022.

[48] Patrizio Frosini. A distance for similarity classes of submanifolds of a euclidean space. *Bulletin of the Australian Mathematical Society*, 42(3):407–415, 1990.

[49] Stepfany Fuentes and Yuvraj S Chowdhury. Fraction of inspired oxygen. In *StatPearls*. StatPearls Publishing, Treasure Island (FL), January 2022.

[50] Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, and Mark Crowley. Uniform manifold approximation and projection (umap) and its variants: Tutorial and survey. *arXiv preprint arXiv:2109.02508*, 2021.

[51] Marvin M Goldenberg. Multiple sclerosis review. *P T*, 37(3):175–184, March 2012.

[52] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016.

[53] John C. Gower. Similarity, dissimilarity, and distance measure. 2005.

[54] Claude Guérin, Richard K Albert, Jeremy Beitler, Luciano Gattinoni, Samir Jaber, John J Marini, Laveena Munshi, Laurent Papazian, Antonio Pesenti, Antoine Vieillard-Baron, et al. Prone position in ards patients: why, when, how and for whom. *Intensive care medicine*, 46(12):2385–2396, 2020.

[55] Claude Guérin, Jean Reignier, Jean-Christophe Richard, Pascal Beuret, Arnaud Gacouin, Thierry Boulain, Emmanuelle Mercier, Michel Badet, Alain Mercat, Olivier Baudin, et al. Prone positioning in severe acute respiratory distress syndrome. *New England Journal of Medicine*, 368(23):2159–2168, 2013.

[56] Jiawei Han, Micheline Kamber, and Jian Pei. 10 - cluster analysis: Basic concepts and methods. In Jiawei Han, Micheline Kamber, and Jian Pei, editors, *Data Mining (Third Edition)*, The Morgan Kaufmann Series in Data Management Systems, pages 443–495. Morgan Kaufmann, Boston, third edition edition, 2012.

[57] J.A. Hanley and Barbara Mcneil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143:29–36, 05 1982.

[58] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[59] Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6(1):96, Jun 2019.

[60] Jeffrey Heer, Michael Bostock, and Vadim Ogievetsky. A tour through the visualization zoo. *ACM Queue*, 8:20, 05 2010.

[61] Mallory Helms, Shaun Ault, Guifen Mao, and Jin Wang. An overview of google brain and its applications. pages 72–75, 03 2018.

[62] Felix Hensel, Michael Moor, and Bastian Rieck. A survey of topological machine learning methods. *Front Artif Intell*, 4:681108, May 2021.

[63] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[64] Tsung-Yu Hsieh, Suhang Wang, Yiwei Sun, and Vasant G. Honavar. Explainable multivariate time series classification: A deep neural network which learns to attend to important variables as well as informative time intervals. *CoRR*, abs/2011.11631, 2020.

[65] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[66] Byungjin Hwang, Ji Hyun Lee, and Duhee Bang. Single-cell rna sequencing technologies and bioinformaticspipelines. *Experimental & Molecular Medicine*, 50(8):1–14, Aug 2018.

[67] Miguel Ibarra-Estrada, María J. Gamero-Rodríguez, Marina García-de Acilu, Oriol Roca, Laura Sandoval-Plascencia, Guadalupe Aguirre-Avalos, Roxana García-Salcido, Sara A. Aguirre-Díaz, David L. Vines, Sara Mirza, Ramandeep Kaur, Tyler Weiss, Claude Guerin, and Jie Li. Lung ultrasound response to awake prone positioning predicts the need for intubation in patients with covid-19 induced acute hypoxemic respiratory failure: an observational study. *Critical Care*, 26(1):189, Jun 2022.

[68] Miguel Ibarra-Estrada, Jie Li, Ivan Pavlov, Yonatan Perez, Oriol Roca, Elsa Tavernier, Bairbre McNicholas, David Vines, Miguel Marín-Rosales, Alexandra Vargas-Obieta, Roxana García-Salcido, Sara A. Aguirre-Díaz, José A. López-Pulgarín, Quetzalcóatl Chávez-Peña, Julio C. Mijangos-Méndez, Guadalupe Aguirre-Avalos, Stephan Ehrmann, and John G. Laffey. Factors for success of awake prone positioning in patients with covid-19-induced acute hypoxemic respiratory failure: analysis of a randomized controlled trial. *Critical Care*, 26(1):84, Mar 2022.

[69] Peter W. Jones, Mauro Maggioni, and Raanan Schul. Manifold parametrizations by eigenfunctions of the laplacian and heat kernels. *Proceedings of the National Academy of Sciences*, 105(6):1803–1808, 2008.

[70] Georgios A. Kaissis, Marcus R. Makowski, Daniel Rückert, and Rickmer F. Braren. Secure, privacy-preserving and federated machine learning in medical imaging. *Nature Machine Intelligence*, 2(6):305–311, Jun 2020.

[71] Jonathan Kanevsky, Jason Corban, Richard Gaster, Ari Kanevsky, Samuel Lin, and Mirko Gilardino. Big data and machine learning in plastic surgery: A new frontier in surgical innovation. *Plastic and reconstructive surgery*, 137:890e–897e, 05 2016.

[72] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. page 10, 01 2014.

[73] Shachar Kaufman, Saharon Rosset, and Claudia Perlich. Leakage in data mining: Formulation, detection, and avoidance. volume 6, pages 556–563, 01 2011.

[74] Pearse A. Keane and Eric J. Topol. With an eye to ai and autonomous diagnosis. *npj Digital Medicine*, 1(1):40, Aug 2018.

[75] Christopher J. Kelly, Alan Karthikesalingam, Mustafa Suleyman, Greg Corrado, and Dominic King. Key challenges for delivering clinical impact with artificial intelligence. *BMC Medicine*, 17(1):195, Oct 2019.

[76] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386, Mar 2005.

[77] Natalia A. Khovanova, Torgyn Shaikhina, and Kajal K. Mallick. Neural networks for analysis of trabecular bone in osteoarthritis. *Bioinspired, Biomimetic and Nanobiomaterials*, 4(1):90–100, 2015.

[78] Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. Unsupervised recurrent neural network grammars. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1105–1117, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[79] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[80] Matthieu Komorowski, Dominic C. Marshall, Justin D. Salciccioli, and Yves Crutain. *Exploratory Data Analysis*, pages 185–203. Springer International Publishing, Cham, 2016.

[81] V. Kotu and B. Deshpande. *Data Science: Concepts and Practice*. Elsevier Science, 2018.

[82] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, Mar 1964.

[83] JB Kruskal and Mark Liberman. The symmetric time-warping problem: From continuous to discrete. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, 01 1983.

[84] J.B. Kruskal and M. Wish. *Multidimensional Scaling*. Sage Publications, 1978.

[85] William H. Kruskal and W. Allen Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952.

[86] Blake Lebaron and Andreas Weigend. A bootstrap evaluation of the effect of data splitting on financial time series. *Neural Networks, IEEE Transactions on*, 9:213 – 220, 02 1998.

[87] Zachary C. Lipton, David C. Kale, and Randall Wetzel. Modeling missing data in clinical time series with rnns, 2016.

[88] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[89] Charles Lu, Ken Chang, Praveer Singh, Stuart Pomerantz, Sean Doyle, Sujay Kakarmath, Christopher Bridge, and Jayashree Kalpathy-Cramer. Deploying clinical machine learning? consider the following... 09 2021.

[90] Fred D Lublin, Stephen C Reingold, Jeffrey A Cohen, Gary R Cutter, Per Soelberg Sørensen, Alan J Thompson, Jerry S Wolinsky, Laura J Balcer, Brenda Banwell, Frederik Barkhof, Bruce Bebo, Jr, Peter A Calabresi, Michel Clanet, Giancarlo Comi, Robert J Fox, Mark S Freedman, Andrew D Goodman, Matilde Inglese, Ludwig Kappos, Bernd C Kieseier, John A Lincoln, Catherine Lubetzki, Aaron E Miller, Xavier Montalban, Paul W O'Connor, John Petkau, Carlo Pozzilli, Richard A Rudick, Maria Pia Sormani, Olaf Stüve, Emmanuelle Waubant, and Chris H Polman. Defining the clinical course of multiple sclerosis: the 2013 revisions. *Neurology*, 83(3):278–286, May 2014.

[91] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874, 2017.

[92] Markus Löning, Anthony Bagnall, Sajaysurya Ganesh, Viktor Kazakov, Jason Lines, and Franz Király. sktime: A unified interface for machine learning with time series. 09 2019.

[93] Markus Löning, Franz Király, Tony Bagnall, Matthew Middlehurst, Sajaysurya Ganesh, George Oastler, Jason Lines, ViktorKaz, Martin Walter, Lukasz Mentel, Stanislav Khrapov, RNKuhns, chrisholder, Leonidas Tsaprounis, Taiwo Owoseni, Patrick Rockenschaub, jesellier, danbartl, eenticott shell, Guzal Bulatova, Ciaran Gilbert, Lovkush, Kejsi Take, Svea Marie Meyer, AidenRushbrooke, Mirae Parker, Patrick Schäfer, oleskiewicz, Yi-Xuan Xu, and Afzal Ansari. alan-turing-institute/sktime: v0.12.1, June 2022.

[94] Vivek Mahato, Martin O'Reilly, and Padraig Cunningham. A comparison of k-nn methods for time series classification and regression. 12 2018.

[95] Vittorio Mantero, Lucia Abate, Roberto Balgera, Loredana La Mantia, and Andrea Salmaggi. Clinical application of 2017 McDonald diagnostic criteria for multiple sclerosis. *J Clin Neurol*, 14(3):387–392, July 2018.

[96] Matteo Mazziotta and Adriano Pareto. Everything you always wanted to know about normalization (but were afraid to ask). *Rivista Italiana di Economia, Demografia e Statistica*, LXXV:41–52, 01 2021.

[97] Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342, 1993.

[98] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018.

[99] Daniel Mesafint and Manjaiah D H. Grid search in hyperparameter optimization of machine learning models for prediction of hiv/aids test results. *International Journal of Computers and Applications*, pages 1–12, 09 2021.

[100] Tomas Mikolov, Armand Joulin, Sumit Chopra, Michael Mathieu, and Marc'Aurelio Ranzato. Learning longer memory in recurrent neural networks, 2014.

[101] Emi Minamitani, Takuma Shiga, Makoto Kashiwagi, and Ippei Obayashi. Topological descriptor of thermal conductivity in amorphous si. *The Journal of Chemical Physics*, 156(24):244502, 2022.

[102] Kevin R. Moon, David van Dijk, Zheng Wang, Scott Gigante, Daniel B. Burkhardt, William S. Chen, Kristina Yim, Antonia van den Elzen, Matthew J. Hirn, Ronald R. Coifman, Natalia B. Ivanova, Guy Wolf, and Smita Krishnaswamy. Visualizing structure and transitions for biological data exploration. *bioRxiv*, 2019.

[103] Boaz Nadler, Stéphane Lafon, Ronald R. Coifman, and Ioannis G. Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In *Advances in Neural Information Processing Systems 18 - Proceedings of the 2005 Conference*, Advances in Neural Information Processing Systems, pages 955–962, December 2005. 2005 Annual Conference on Neural Information Processing Systems, NIPS 2005 ; Conference date: 05-12-2005 Through 08-12-2005.

[104] Boaz Nadler, Stéphane Lafon, Ronald R. Coifman, and Ioannis G. Kevrekidis. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21(1):113–127, 2006. Special Issue: Diffusion Maps and Wavelets.

[105] T. N. Narasimhan. Fourier's heat conduction equation: History, influence, and connections. *Proceedings of the Indian Academy of Sciences - Earth and Planetary Sciences*, 108(3):117–148, Sep 1999.

[106] Amin Nayebi, Sindhu Tipirneni, Brandon Foreman, Jonathan Ratcliff, Chandan K Reddy, and Vignesh Subbian. Recurrent neural network based Time-Series modeling for long-term prognosis following acute traumatic brain injury. *AMIA Annu Symp Proc*, 2021:900–909, February 2022.

[107] Andrei Novikov. PyClustering: Data mining library. *Journal of Open Source Software*, 4(36):1230, apr 2019.

[108] Ippei Obayashi, Takenobu Nakamura, and Yasuaki Hiraoka. Persistent homology analysis for materials research and persistent homology software: Homcloud. *Journal of the Physical Society of Japan*, 91(9):091013, 2022.

[109] Chika Ogami, Yasuhiro Tsuji, Hiroto Seki, Hideaki Kawano, Hideto To, Yoshiaki Matsumoto, and Hiroyuki Hosono. An artificial neural network- pharmacokinetic model and its interpretation using shapley additive explanations. *CPT: pharmacometrics & systems pharmacology*, 10(7):760–768, 2021.

[110] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

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

[112] Joanne Peng, Kuk Lee, and Gary Ingersoll. An introduction to logistic regression analysis and reporting. *Journal of Educational Research - J EDUC RES*, 96:3–14, 09 2002.

[113] Rosalind W. Picard and Edward W. Boyer. Smartwatch biomarkers and the path to clinical use. *Med*, 2(7):797–799, 2021.

[114] Jaime Pizarroso, José Portela, and Antonio Muñoz. Neuralsens: Sensitivity analysis of neural networks. *Journal of Statistical Software*, 102(7):1–36, 2022.

[115] Prasis Poudel, Sang Hyun Bae, and Bongseog Jang. Comparison of different deep learning optimizers for modeling photovoltaic power. 2018.

[116] Lutz Prechelt. Early stopping - but when? 03 2000.

[117] Michael Recht and R Nick Bryan. Artificial intelligence: threat or boon to radiologists? *Journal of the American College of Radiology*, 14(11):1476–1480, 2017.

[118] Philippe Remy. Conditional rnn for keras. `https://github.com/philipperemy/cond_rnn`, 2020.

[119] Jake Rhodes, Adele Cutler, Guy Wolf, and Kevin Moon. Random forest-based diffusion information geometry for supervised visualization and data exploration. pages 331–335, 07 2021.

[120] Vanessa Robins. Towards computing homology from approximations. *Topology Proceedings*, 24, 01 1999.

[121] Oriol Roca, Berta Caralt, Jonathan Messika, Manuel Samper, Benjamin Sztrymf, Gonzalo Hernández, Marina García-de Acilu, Jean-Pierre Frat, Joan R Masclans, and Jean-Damien Ricard. An index combining respiratory rate and oxygenation to predict outcome of nasal high-flow therapy. *American journal of respiratory and critical care medicine*, 199(11):1368–1376, 2019.

[122] Arthur L. Samuel. Some studies in machine learning using the game of checkers. *IBM JOURNAL OF RESEARCH AND DEVELOPMENT*, pages 71–105, 1959.

[123] Patrick Schäfer and Ulf Leser. Multivariate time series classification with WEASEL+MUSE. *CoRR*, abs/1711.11343, 2017.

[124] Sedat Şen. Neurostatus and EDSS calculation with cases. *Noro Psikiyatr Ars*, 55(Suppl 1):S80–S83, 2018.

[125] Torgyn Shaikhina and Natalia A. Khovanova. Handling limited datasets with neural networks in medical applications: A small-data approach. *Artificial Intelligence in Medicine*, 75:51–63, 2017.

[126] Boyu Shen. E-commerce customer segmentation via unsupervised machine learning. In *The 2nd International Conference on Computing and Data Science*, CONF-CDS 2021, New York, NY, USA, 2021. Association for Computing Machinery.

[127] Richard Simon. *Resampling Strategies for Model Assessment and Selection*, pages 173–186. Springer US, Boston, MA, 2007.

[128] Denis Smirnov and Engelbert Mephu Nguifo. Time series classification with recurrent neural networks. 09 2018.

[129] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. *CoRR*, abs/1803.09820, 2018.

[130] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms, 2012.

[131] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms, 2012.

[132] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[133] Jill C Stoltzfus. Logistic regression: a brief primer. *Acad Emerg Med*, 18(10):1099–1104, October 2011.

[134] Romain Tavenard, Johann Faouzi, Gilles Vandewiele, Felix Divo, Guillaume Androz, Chester Holtz, Marie Payne, Roman Yurchak, Marc Rußwurm, Kushal Kolar, and Eli Woods. Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, 21(118):1–6, 2020.

[135] Elsa Tavernier, Bairbre Mcnicholas, Ivan Pavlov, Oriol Roca, Yonatan Perez, John Laffey, Sara Mirza, David Cosgrave, David Vines, J.-P Frat, Stephan Ehrmann, and Jie Li. Awake prone positioning of hypoxaemic patients with covid-19: Protocol for a randomised controlled open-label superiority meta-trial. *BMJ Open*, 10, 11 2020.

[136] Sana Tonekaboni, Shalmali Joshi, Kieran Campbell, David K Duvenaud, and Anna Goldenberg. What went wrong and when? instance-wise feature importance for time-series black-box models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 799–809. Curran Associates, Inc., 2020.

[137] S A Trip and D H Miller. Imaging in multiple sclerosis. *Journal of Neurology, Neurosurgery & Psychiatry*, 76(suppl 3):iii11–iii18, 2005.

[138] Andrew D Trister, Diana S M Buist, and Christoph I Lee. Will machine learning tip the balance in breast cancer screening? *JAMA oncology*, 3(11):1463—1464, November 2017.

[139] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[140] G. van Rossum. Python tutorial. Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, May 1995.

[141] L. Vietoris. Über den höheren zusammenhang kompakter räume und eine klasse von zusammenhangstreuen abbildungen. *Mathematische Annalen*, 97(1):454–472, Dec 1927.

[142] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator, 2015.

[143] Li Wang, Ji Zhu, and Hui Zou. The doubly regularized support vector machine. *Statistica Sinica*, 16:589–615, 04 2006.

[144] Colin Ware. Chapter ten - interacting with visualizations. In Colin Ware, editor, *Information Visualization (Fourth Edition)*, Interactive Technologies, pages 359–392. Morgan Kaufmann, fourth edition edition, 2021.

[145] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.

[146] Paul Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78:1550 – 1560, 11 1990.

[147] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[148] Xindong Wu, Vipin Kumar, Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, G. Mclachlan, Shu Kay Angus Ng, Bing Liu, Philip Yu, Zhi-Hua Zhou, Michael Steinbach, David Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14, 12 2007.

[149] Zhengzheng Xing, Jian Pei, and Philip S Yu. Early classification on time series. *Knowledge and Information Systems*, 31(1):105–127, April 2012.

[150] Kiyoung Yang and Cyrus Shahabi. A pca-based similarity measure for multivariate time series. In *Proceedings of the 2nd ACM International Workshop on Multimedia Databases*, MMDB '04, page 65–74, New York, NY, USA, 2004. Association for Computing Machinery.

[151] Li Yang and Abdallah Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020.

[152] Xue Ying. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168:022022, 02 2019.

[153] SC Yu, X Qi, YH Hu, WJ Zheng, QQ Wang, and HY Yao. [overview of multivariate regression model analysis and application]. *Zhonghua yu fang yi xue za zhi [Chinese journal of preventive medicine]*, 53(3):334—336, March 2019.

[154] Tong Yu and Hong Zhu. Hyper-parameter optimization: A review of algorithms and applications. *CoRR*, abs/2003.05689, 2020.

[155] Afra Zomorodian. Fast construction of the vietoris-rips complex. *Computers & Graphics*, 34(3):263–271, 2010.

Appendix A

APP for COVID-19: complementary information

A.1. Description of each feature

A.1.1. Longitudinal features

- $X20\_ROX\_enrol$: ROX index (SF/RR) at day 0 (D0)
- ROX_D1: ROX index at day 1 (D1)
- ROX_D2: ROX index at day 2 (D2)
- ROX_D3: ROX index at day 3 (D3)
- $X200\_SF\_enrol$: SF ratio at D0
- SF_D1: SF ratio ($SpO2/FiO2$) at D1
- SF_D2: SF ratio at D2
- SF_D3: SF ratio at D3
- $X50\_TimePP0$: total time of all prone sessions with day 0 being time from enrollment to midnight (minutes or hours)
- $X51\_TimePP1$: total time of all prone sessions with day 1 being first full day post enrolment (midnight to midnight) (minute or hours)
- $X52\_TimePP2$: total time of all prone sessions with day 2 being second full day post enrolment (midnight to midnight) (minutes or hours)
- $X53\_TimePP3$: total time of all prone sessions with day 3 being third full day post enrolment (midnight to midnight) (minutes or hours)
- $X12\_SpO2\_enrol$: SpO2 (oxygen saturation measured by pulse oximetry) if available at or around enrolment or prior to first proning session (%)

- $X65\_SpO2\_D1$: SpO2 during morning period (or first supine period if prone in AM) of first day of enrolment (%)
- $X66\_SpO2\_D2$: SpO2 during morning period (or first supine period if prone in AM) of second day of enrolment (%)
- $X67\_SpO2\_D3$: SpO2 during morning period (or first supine period if prone in AM) of third day of enrolment (%)
- $X14\_FiO2\_enrol$: FiO2 (fraction of inspired oxygen) if available at or around enrolment or prior to first proning session (fraction decimal)
- $X68\_FiO2\_D1$: FiO2 during morning period (or first supine period if prone in AM) of first day of enrolment (fraction decimal)
- $X69\_FiO2\_D2$: FiO2 during morning period (or first supine period if prone in AM) of second day of enrolment (fraction decimal)
- $X70\_FiO2\_D3$: FiO2 during morning period (or first supine period if prone in AM) of third day of enrolment (fraction decimal)
- $X18\_RR\_enrol$: RR (respiratory rate) if available at or around enrolment or prior to first proning session (RR/min)
- $X74\_RR\_D1$: RR during morning period (or first supine period if prone in AM) of first day of enrolment (RR/min)
- $X75\_RR\_D2$: RR during morning period (or first supine period if prone in AM) of second day of enrolment (RR/min)
- $X76\_RR\_D3$: RR during morning period (or first supine period if prone in AM) of third day of enrolment (RR/min)

A.1.2. Static features

- $Total\_Time\_PP$: $\sum_{i=0}^{3} X5i\_TimePPi$
- $Mexican\_trial$: patient is from the Mexican trial (0/1)
- $X04\_Gender$: encoded as 0 for Male, 1 for Female
- $X05\_Age$: age at inclusion
- $X08\_BMI$: body mass index
- $X22\_CCD$: chronic cardiac disease: history of HTN, Ischemic heart disease, congestive heart failure

- $X23\_CLD$: patient has known or suspected Chronic Obstructive or restrictive lung Disease (0/1)
- $X24\_DM$: patient has known type 1 or type 2 diabetes mellitus (0/1)
- $X25\_CKD$: patient has known chronic renal failure with a creatinine clearance less than 60 milliliters per minute prior to hospialisation (0/1)
- $X26\_Cancer$: active solid organ or hematological malignancy requiring treatment (0/1)
- $X27\_Obesity$: obesity defined as a BMI>30kg/m2. Provided as mentioned in the patient's file. Precise weight, height missing for some patients (0/1)
- $X28\_CLF$: check if the patient has chronic liver disease with a calculated Child Pugh score$\geq$10 or severe liver disease as per Charlson comorbdity index (0/1)
- $Nb\_Comorbidities$: sum of the 7 above-mentioned co-morbidities
- $X30\_Steroid$: known use of dexamethasone or equivalent steroid for treatment of COVID-19 (0/1)
- $X33d\_Delta\_Per\_PP\_SF$: difference between SF values during and before first PP session
- $X33e\_Delta\_Post\_PP\_SF$: difference between SF values after and before first PP session
- $X44a\_Delta\_Per\_PP\_RR$: difference between RR values during and before first PP session
- $X44b\_Delta\_Post\_PP\_RR$: difference between RR values after and before first PP session
- $X45a\_Delta\_Post\_PP\_SpO2$: difference between $SpO2$ values after and before first PP session
- $X45b\_Delta\_Post\_PP\_FiO2$: difference between $FiO2$ values after and before first PP session
- $X48d\_Delta\_Per\_PP\_ROX$: difference between ROX values during and before first PP session
- $X48e\_Delta\_Post\_PP\_ROX$: difference between ROX values after and before first PP session
- $X49\_Duration\_D0$: duration of the enrolment day (minutes or hours)

Outcomes:

- $X81\_Primary\_out$: death or intubation within 28 days of enrolment (0/1)
- $X83\_Death28d$: death before 28 days post enrolment (0/1)
- $X84\_Intub$: intubation before 28 days post enrolment (0/1)

A.2. Additional plots

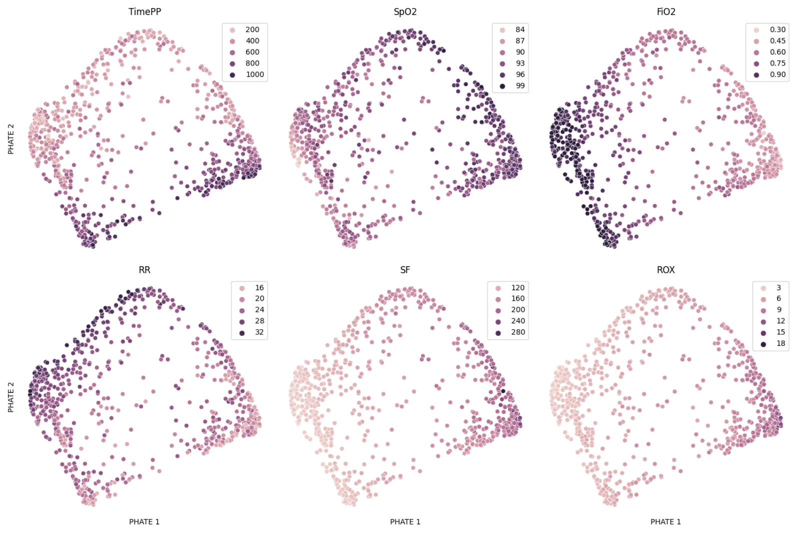
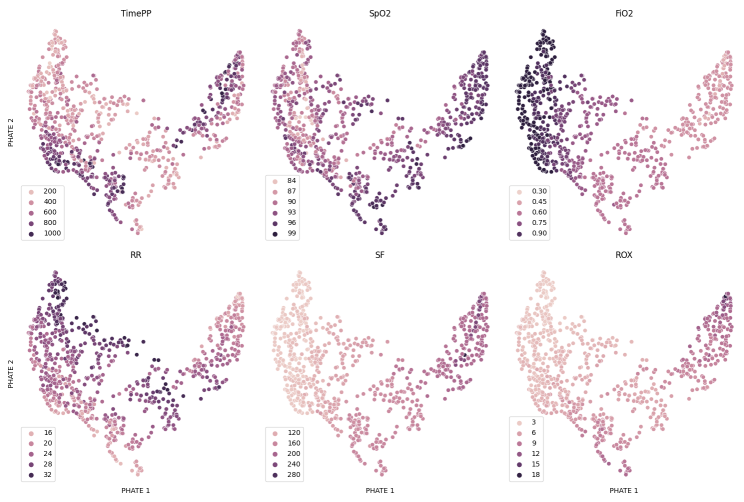
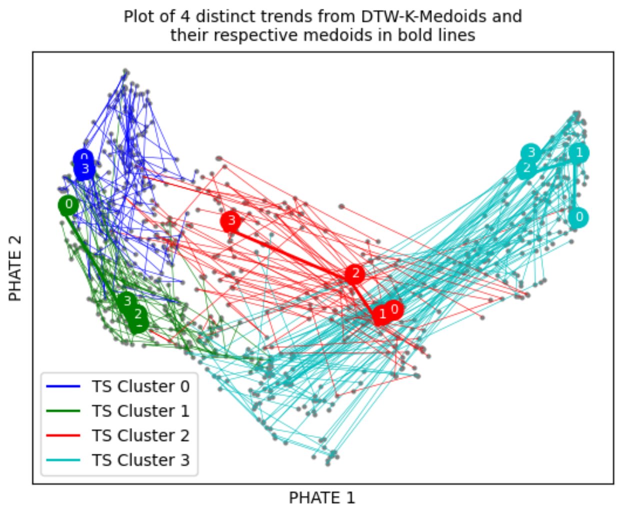


Fig. A.1. 2-dimensional PHATE embedding of the Mexican database following our processing steps in Section 2.3.1 without the forward filling imputation. Each of the six sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1. In comparison with Figure 2.4, the structure of the embedding remains the same, with 810 embedding points with respect to 216 patients in place of 864 embedding points with respect to 216 patients following our data processing steps in Section 2.3.1. This indicates a safe way to make the clustering produce representative medoids while preserving the structure of the data.



(a) 2-dimensional PHATE embedding of the Mexican database following our processing steps in Section 2.3.1 with default PHATE input parameters. Each of the six sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1.



(b) Four main groups of Mexican trajectories detected by 4-medoids on the original 6-dimensional space of patients' time series and projected on the 2-dimensional PHATE embedding (default parameters), with the associated medoids in bold lines.

Fig. A.2. Resulting Mexican plots from our visualization methodology using default PHATE parameters (i.e. $k = 5$, $\alpha = 15$, t set with Von Neumann Entropy, and PHATE log potential). In comparison with Figures 2.4 and 2.5, the PHATE embedding (a) possesses the same overall branching structure, allowing the production of similar visualizations of 4 groups of trajectories (b). Still, in Section 2.6.1, we choose custom PHATE input parameters to avoid the slightly overlapping clusters of trajectories.

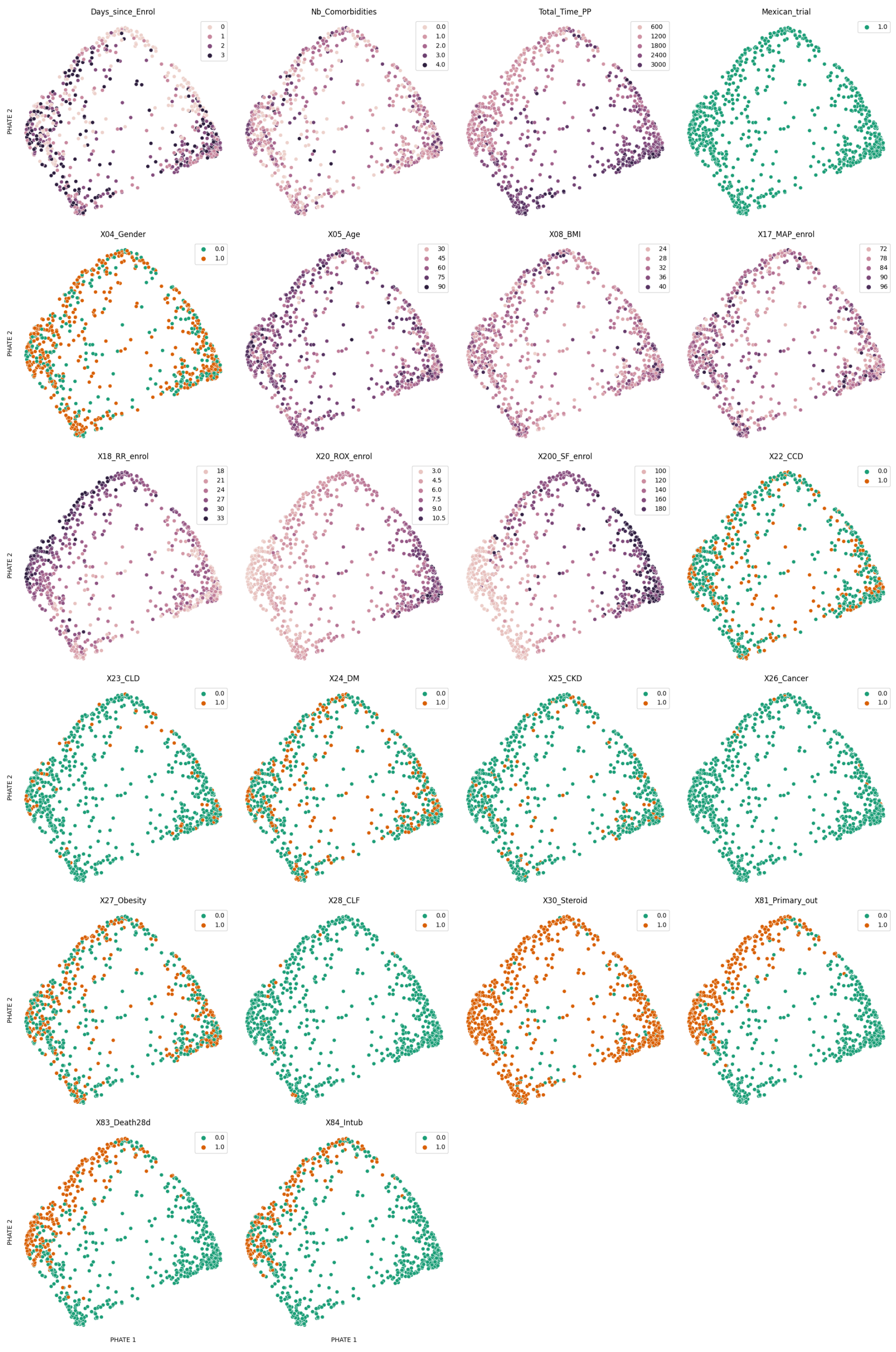


Fig. A.3. 2-dimensional PHATE embedding of the Mexican database following our processing steps in Section 2.3.1. Each of the 22 sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the values of 22 external features, as described in Appendix A.1.2.

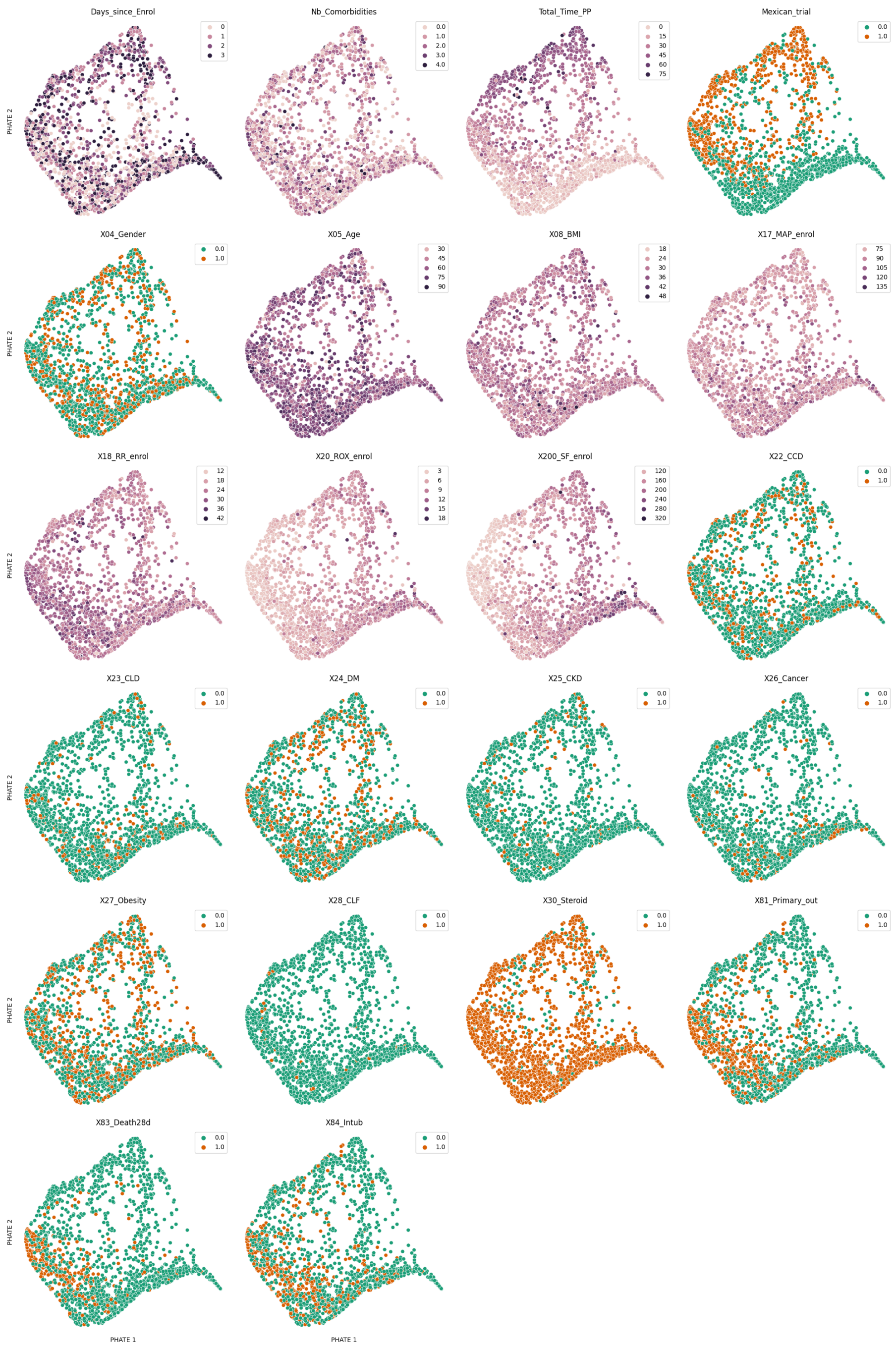


Fig. A.4. 2-dimensional PHATE embedding of the global database following our processing steps in Section 2.3.1. Each of the 22 sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the values of 22 external features, as described in Appendix A.1.2.

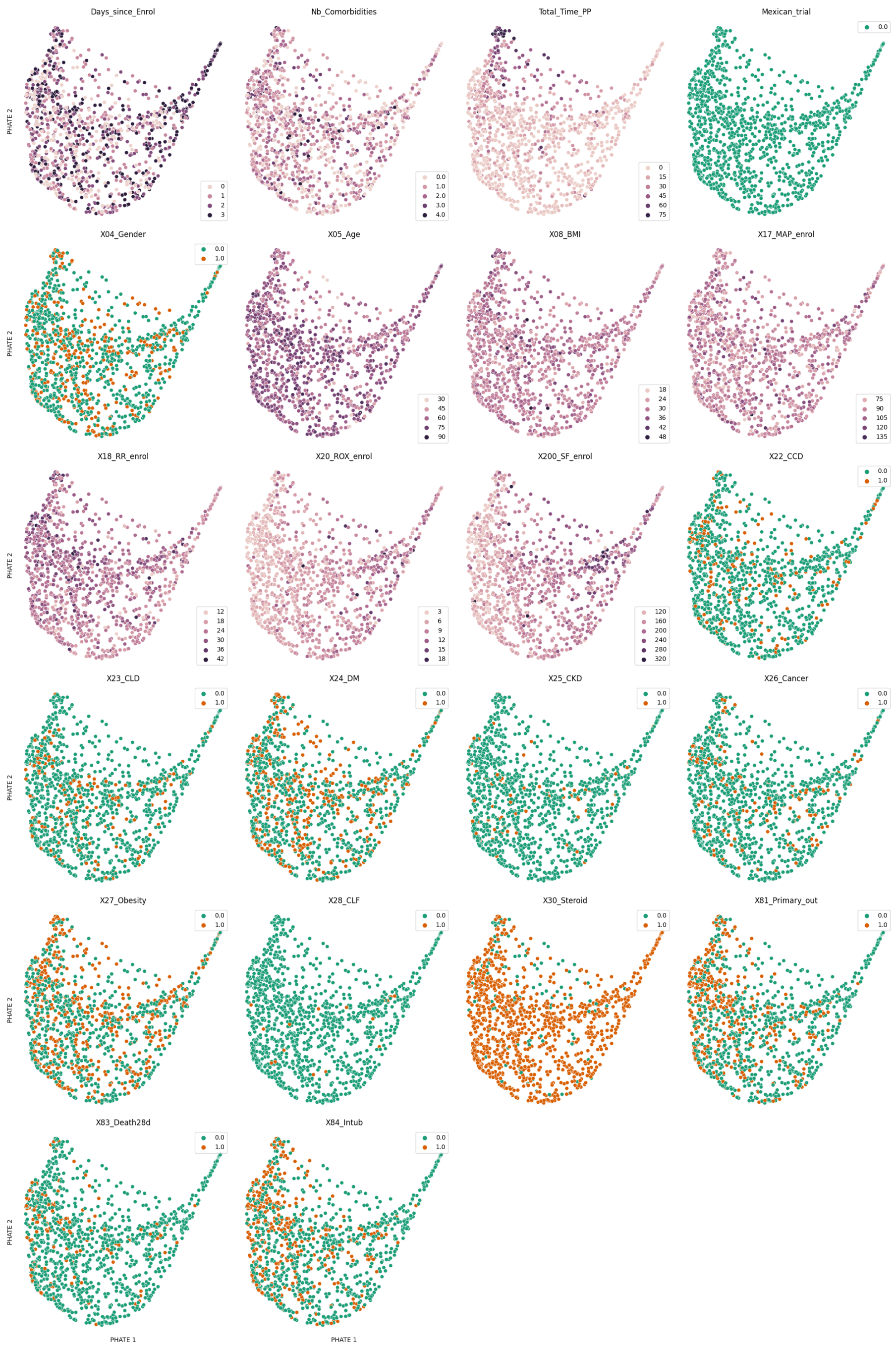


Fig. A.5. 2-dimensional PHATE embedding of the non-Mexican database following our processing steps in Section 2.3.1. Each of the 22 sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the values of 22 external features, as described in Appendix A.1.2.

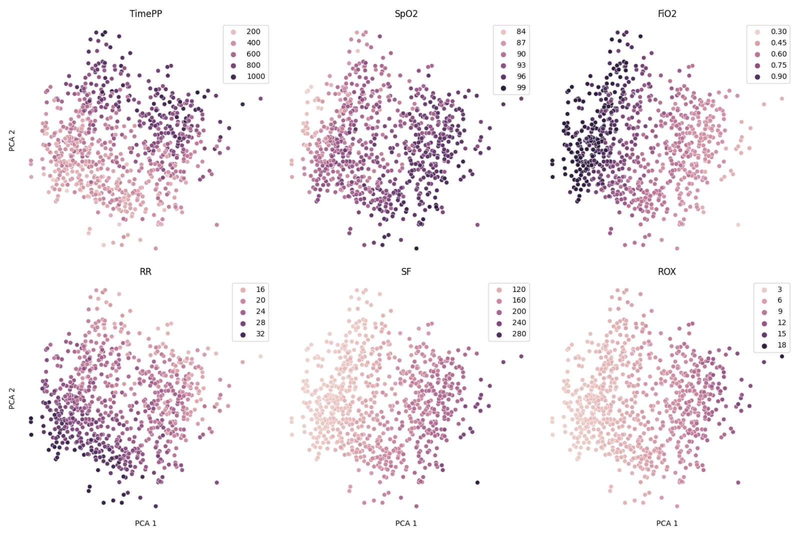


Fig. A.6. 2-dimensional PCA embedding of the Mexican database following our processing steps in Section 2.3.1. Each of the six sub-plots represents the same PCA embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1.

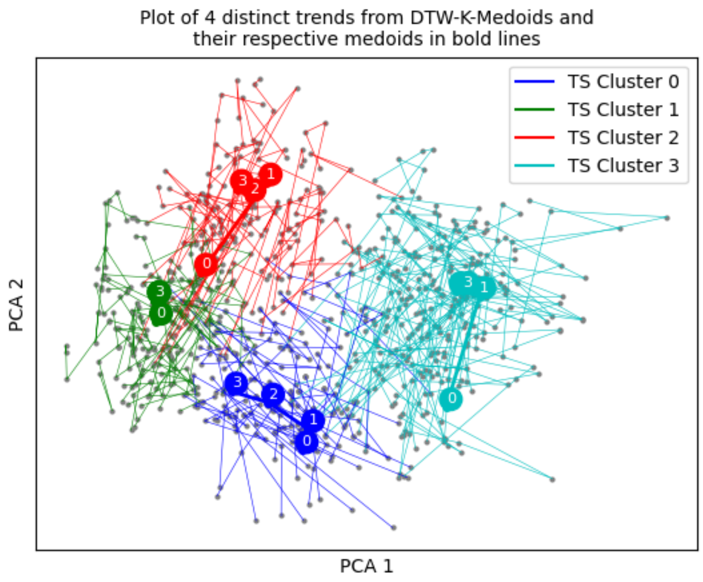


Fig. A.7. Four main groups of Mexican trajectories detected by 4-medoids and projected on the 2-dimensional PCA embedding, with the associated medoids in bold lines.

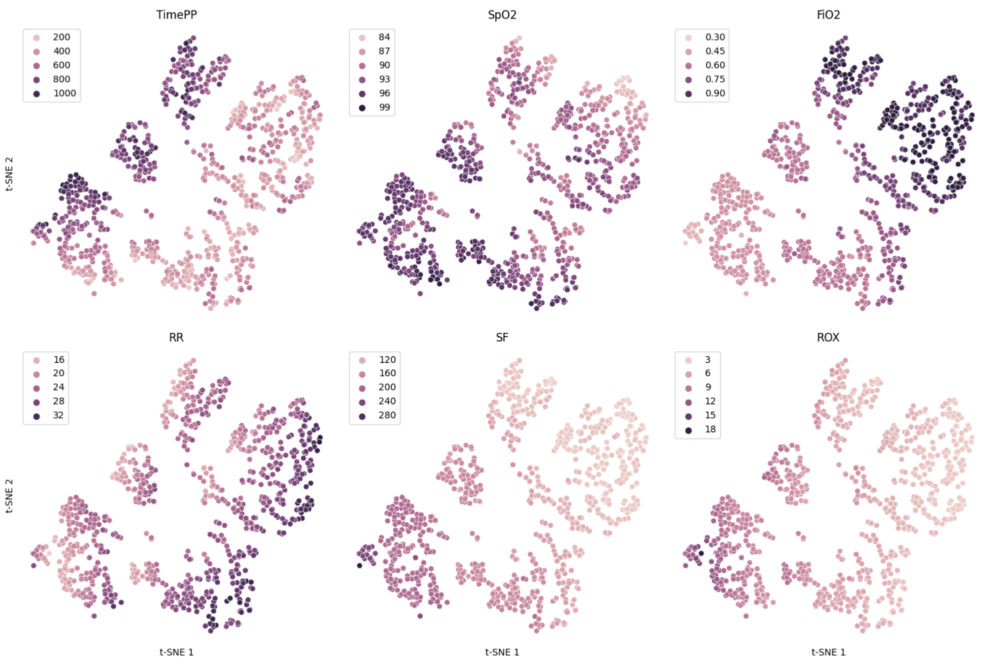


Fig. A.8. 2-dimensional t-SNE embedding of the Mexican database following our processing steps in Section 2.3.1. Each of the six sub-plots represents the same t-SNE embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1.

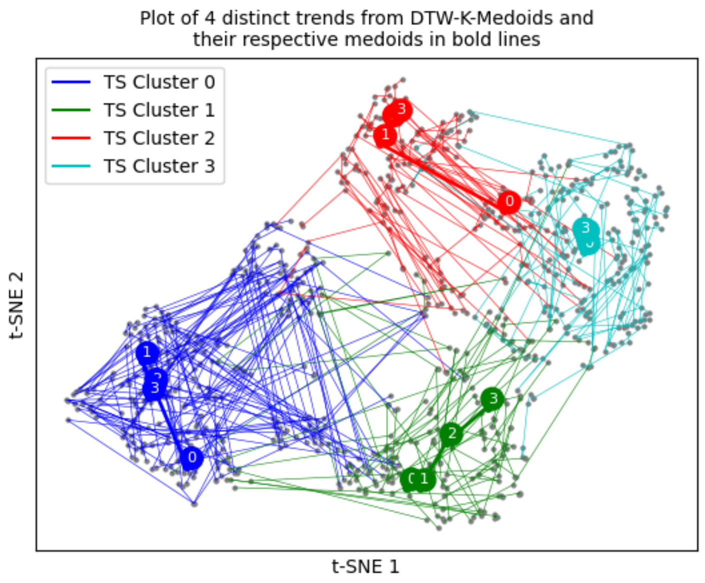


Fig. A.9. Four main groups of Mexican trajectories detected by 4-medoids and projected on the 2-dimensional t-SNE embedding, with the associated medoids in bold lines.

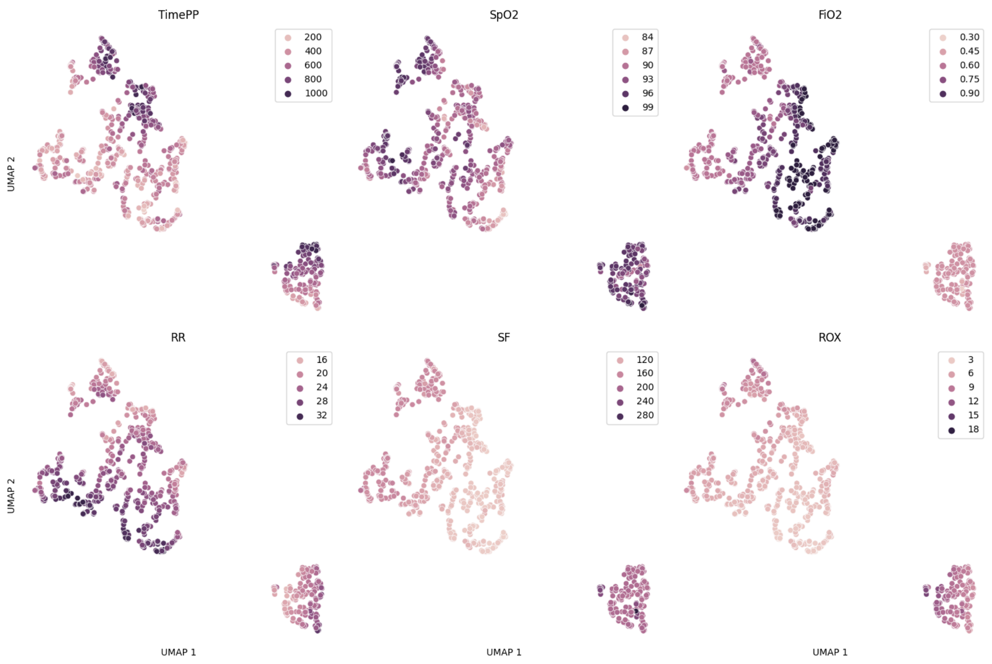


Fig. A.10. 2-dimensional UMAP embedding of the Mexican database following our processing steps in Section 2.3.1. Each of the six sub-plots represents the same UMAP embedding, except that the embedding points are colored differently according to the six main input features as described in Section 2.2 and Appendix A.1.1.

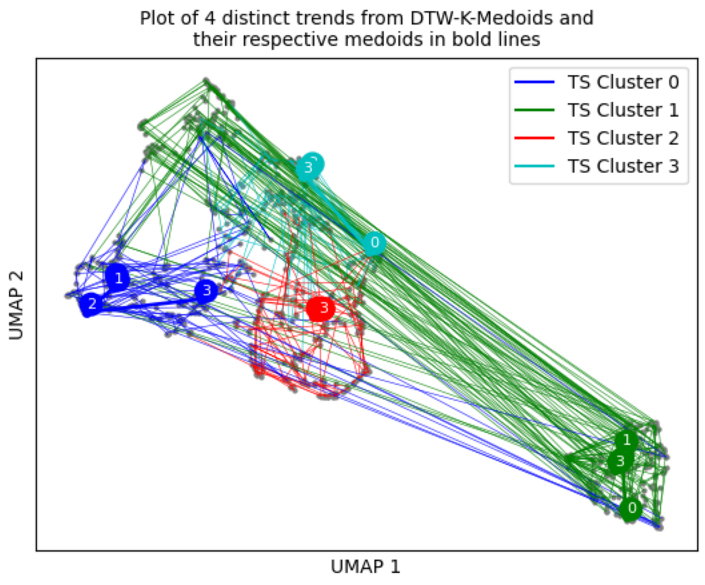


Fig. A.11. Four main groups of Mexican trajectories detected by 4-medoids and projected on the 2-dimensional UMAP embedding, with the associated medoids in bold lines.

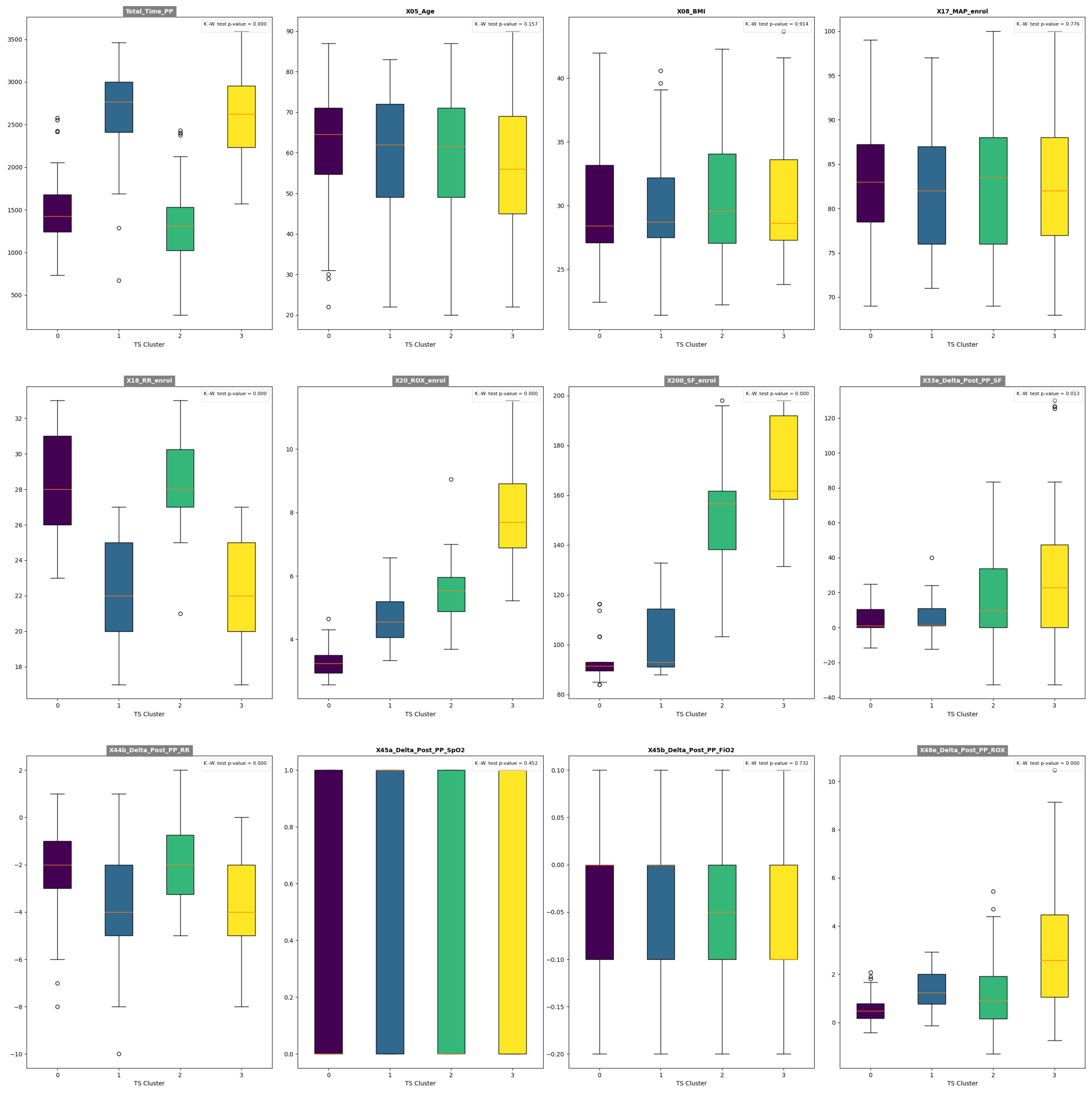


Fig. A.12. Box plots (first, second and third quartiles, whiskers and outliers) of the Mexican intervention group, describing the distribution of some continuous variables described in Appendix A.1 within each of the 4 clusters of trajectories depicted in Figure 2.5. Labels in gray indicates a successful Kruskal-Wallis test, a rank-based nonparametric test that determines if there are statistically significant differences between at least two groups of an independent variable on a continuous or ordinal dependent variable [**85**], with a significance level of 0.05. p-values are reported on the top-right corner of each box plot.

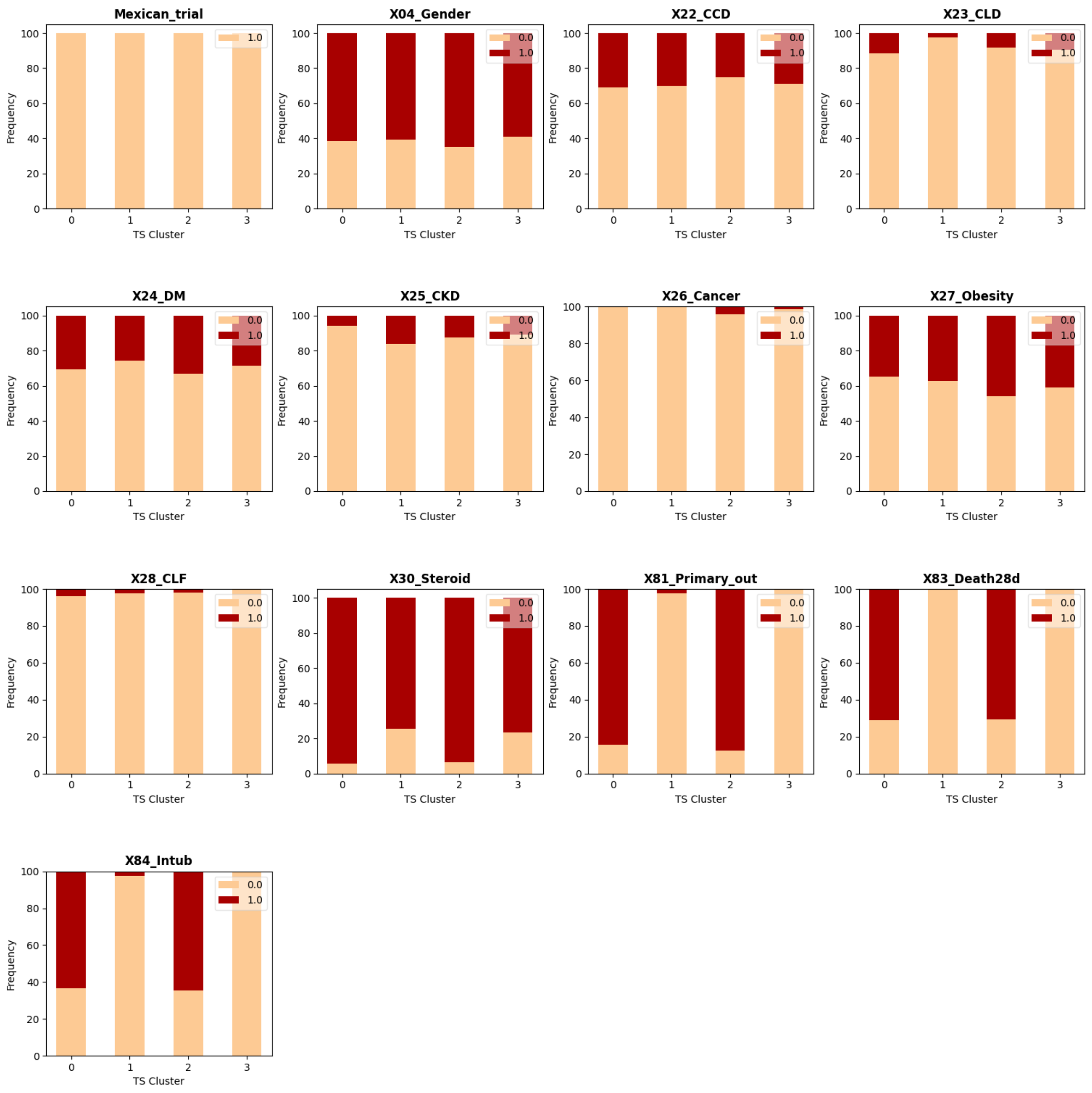


Fig. A.13. Bar plots of the Mexican intervention group, describing the frequency of the outcomes and other categorical static variables described in Appendix A.1 within each of the 4 clusters of trajectories depicted in Figure 2.5.

MEXICAN - APP

	CLUSTER_0	CLUSTER_1	CLUSTER_2	CLUSTER_3
Nb_Patients	52	43	48	73
Total_Time_PP_Median	1426	2765	1312.5	2623
Total_Time_PP_IQR	423.17	534.48	531.78	517.28
Mexican_trial_Perc	100	100	100	100
X04_Female_Perc	61.54	60.47	64.58	58.9
X05_Age_Median	64.5	62	61.5	56
X05_Age_IQR	16.25	23	22	24
X08_BMI_Median	28.4	28.7	29.55	28.6
X08_BMI_IQR	6.1	4.7	7	6.3
X17_MAP_enrol_Mean	83.52	82	82.88	82.41
X17_MAP_enrol_SD	7.44	6.64	7.22	7.61
X18_RR_enrol_Mean	28.25	22.16	28.52	22.18
X18_RR_enrol_SD	2.96	3	2.71	3.28
X20_ROX_enrol_Mean	3.32	4.62	5.46	7.9
X20_ROX_enrol_SD	0.46	0.84	0.98	1.46
X200_SF_enrol_Mean	92.84	100.71	154.39	171.54
X200_SF_enrol_SD	7.88	12.93	22.49	20.08
X22_CCD_Perc	30.77	30.23	25	28.77
X23_CLD_Perc	11.54	2.33	8.33	9.59
X24_DM_Perc	30.77	25.58	33.33	28.77
X25_CKD_Perc	5.77	16.28	12.5	10.96
X26_Cancer_Perc	0	0	4.17	1.37
X27_Obesity_Perc	34.62	37.21	45.83	41.1
X28_CLF_Perc	3.85	2.33	2.08	0
Nb_Comorbidities_Mean	1.17	1.14	1.31	1.21
Nb_Comorbidities_SD	1.04	1.06	1.11	0.96
X281_1+_Comorbidities_Perc	67.31	69.77	70.83	73.97
X282_2+_Comorbidities_Perc	36.54	27.91	43.75	36.99
X283_3+_Comorbidities_Perc	13.46	13.95	12.5	8.22
X284_4+_Comorbidities_Perc	0	2.33	4.17	1.37
X285_5+_Comorbidities_Perc	0	0	0	0
X286_6+_Comorbidities_Perc	0	0	0	0
X287_7+_Comorbidities_Perc	0	0	0	0
X30_Steroid_Perc	94.23	74.42	93.75	76.71
X48_ROX_change_Mean	0.54	1.31	1.19	3.01
X48_ROX_change_SD	0.54	0.82	1.56	2.46
X81_Primary_out_Perc	84.62	2.33	87.5	0
X83_Death28d_Perc	71.15	0	70.83	0
X84_Intub_Perc	63.46	2.33	64.58	0

Fig. A.14. Quantitative description of the Mexican intervention group for each of the 4 clusters of trajectories depicted in Figure 2.5 in terms of features described in Appendix A.1.

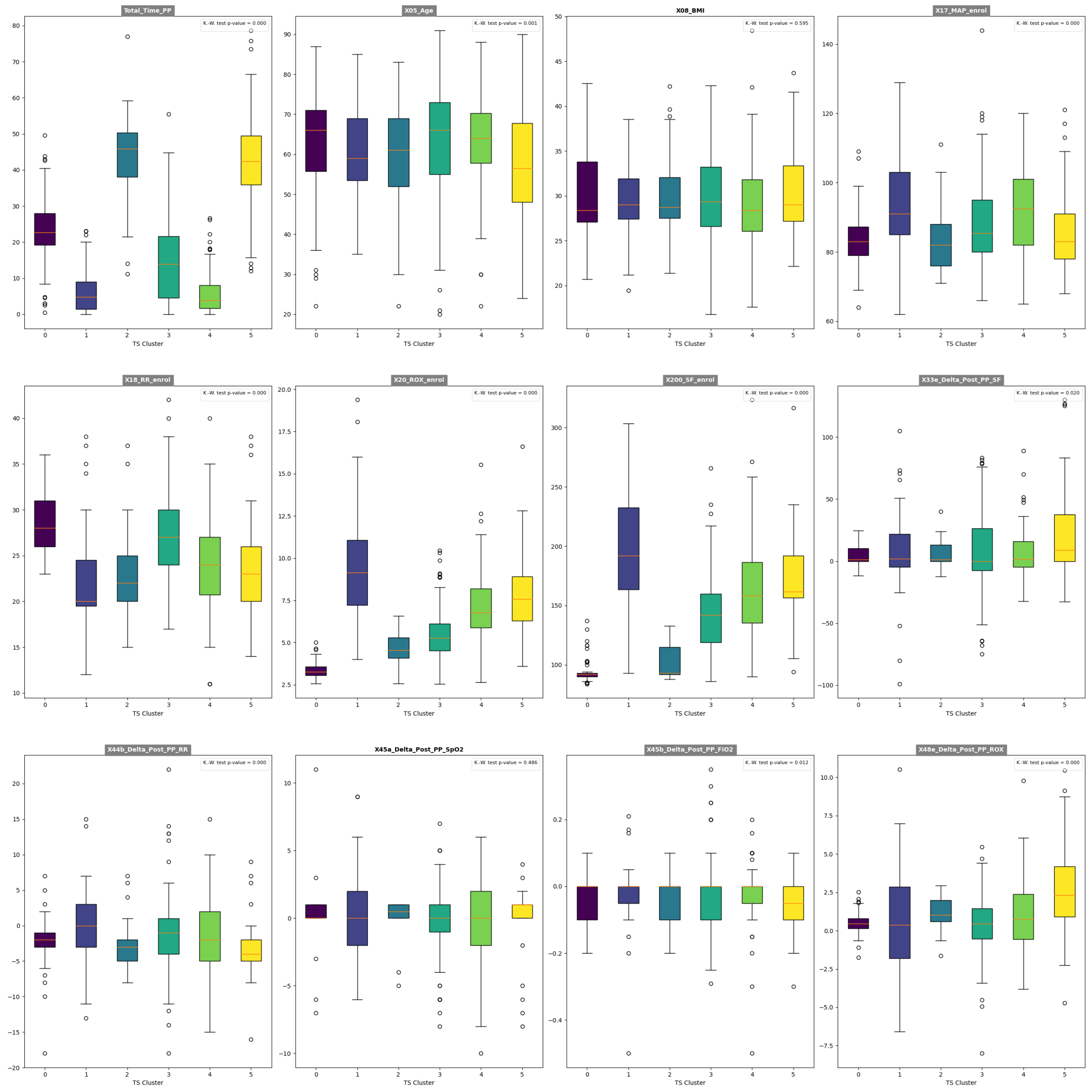


Fig. A.15. Box plots (first, second and third quartiles, whiskers and outliers) of the global intervention group, describing the distribution of some continuous variables described in Appendix A.1 within each of the 6 clusters of trajectories depicted in Figure 2.7. Labels in gray indicates a successful Kruskal-Wallis test, a rank-based nonparametric test that determines if there are statistically significant differences between at least two groups of an independent variable on a continuous or ordinal dependent variable [**85**], with a significance level of 0.05. p-values are reported on the top-right corner of each box plot.

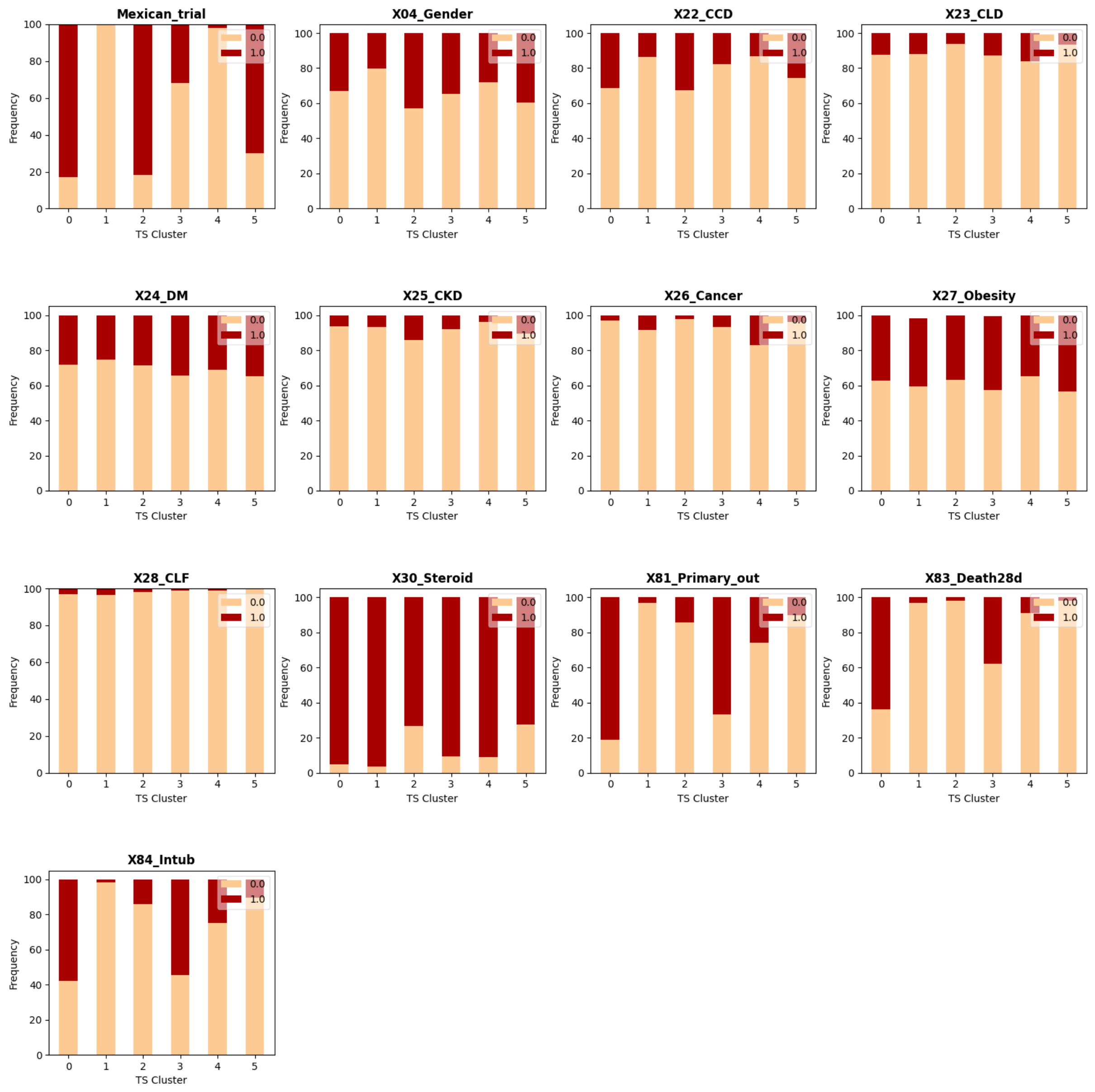


Fig. A.16. Bar plots of the global intervention group, describing the frequency of the outcomes and other categorical static variables described in Appendix A.1 within each of the 6 clusters of trajectories depicted in Figure 2.7.

ALL - APP

	CLUSTER_0	CLUSTER_1	CLUSTER_2	CLUSTER_3	CLUSTER_4	CLUSTER_5
Nb_Patients	64	59	49	148	100	106
Total_Time_PP_Median	22.65	4.83	45.92	13.88	3.82	42.32
Total_Time_PP_IQR	8.81	7.57	12.33	17.16	6.44	13.56
Mexican_trial_Perc	82.81	0	81.63	31.76	2	69.81
X04_Female_Perc	32.81	20.34	42.86	34.46	28	39.62
X05_Age_Median	66	59	61	66	64	56.5
X05_Age_IQR	15.25	15.5	17	18	12.5	19.75
X08_BMI_Median	28.4	29.04	28.71	29.34	28.39	29
X08_BMI_IQR	6.71	4.52	4.54	6.64	5.75	6.16
X17_MAP_enrol_Mean	84.06	93.05	83.67	88.64	91.91	85.17
X17_MAP_enrol_SD	8.39	13.55	9.04	13.11	12.68	10.49
X18_RR_enrol_Mean	28.23	22.22	22.8	26.91	23.89	22.86
X18_RR_enrol_SD	3.1	5.51	4.38	4.63	5.34	4.32
X20_ROX_enrol_Mean	3.38	9.38	4.62	5.48	7.1	7.74
X20_ROX_enrol_SD	0.51	3.13	0.94	1.51	2.16	1.99
X200_SF_enrol_Mean	94.33	196.89	102.22	143.48	161.95	171.24
X200_SF_enrol_SD	10.43	43.81	13.83	31.33	38.17	31.71
X22_CCD_Perc	31.25	13.56	32.65	17.57	13	25.47
X23_CLD_Perc	12.5	11.86	6.12	12.84	16	6.6
X24_DM_Perc	28.12	25.42	28.57	34.46	31	34.91
X25_CKD_Perc	6.25	6.78	14.29	8.11	4	10.38
X26_Cancer_Perc	3.12	8.47	2.04	6.76	17	3.77
X27_Obesity_Perc	37.5	38.98	36.73	41.89	35	43.4
X28_CLF_Perc	3.12	3.39	2.04	1.35	1	0
Nb_Comorbidities_Mean	1.22	1.08	1.22	1.23	1.17	1.25
Nb_Comorbidities_SD	1	1.02	1.12	1	1.04	0.96
X281_1+_Comorbidities_Perc	71.88	64.41	71.43	72.97	70	75.47
X282_2+_Comorbidities_Perc	37.5	33.9	30.61	37.84	35	37.74
X283_3+_Comorbidities_Perc	12.5	8.47	16.33	10.14	8	10.38
X284_4+_Comorbidities_Perc	0	1.69	4.08	2.03	4	0.94
X285_5+_Comorbidities_Perc	0	0	0	0	0	0
X286_6+_Comorbidities_Perc	0	0	0	0	0	0
X287_7+_Comorbidities_Perc	0	0	0	0	0	0
X30_Steroid_Perc	95.31	96.61	73.47	90.54	91	72.64
X48_ROX_change_Mean	0.49	0.57	1.15	0.4	0.95	2.67
X48_ROX_change_SD	0.73	3.5	0.97	1.99	2.27	2.57
X81_Primary_out_Perc	81.25	3.39	14.29	66.89	26	10.38
X83_Death28d_Perc	64.06	3.39	2.04	37.84	9	1.89
X84_Intub_Perc	57.81	1.69	14.29	54.73	25	10.38

Fig. A.17. Quantitative description of the global intervention group for each of the 6 clusters of trajectories depicted in Figure 2.7 in terms of features described in Appendix A.1.

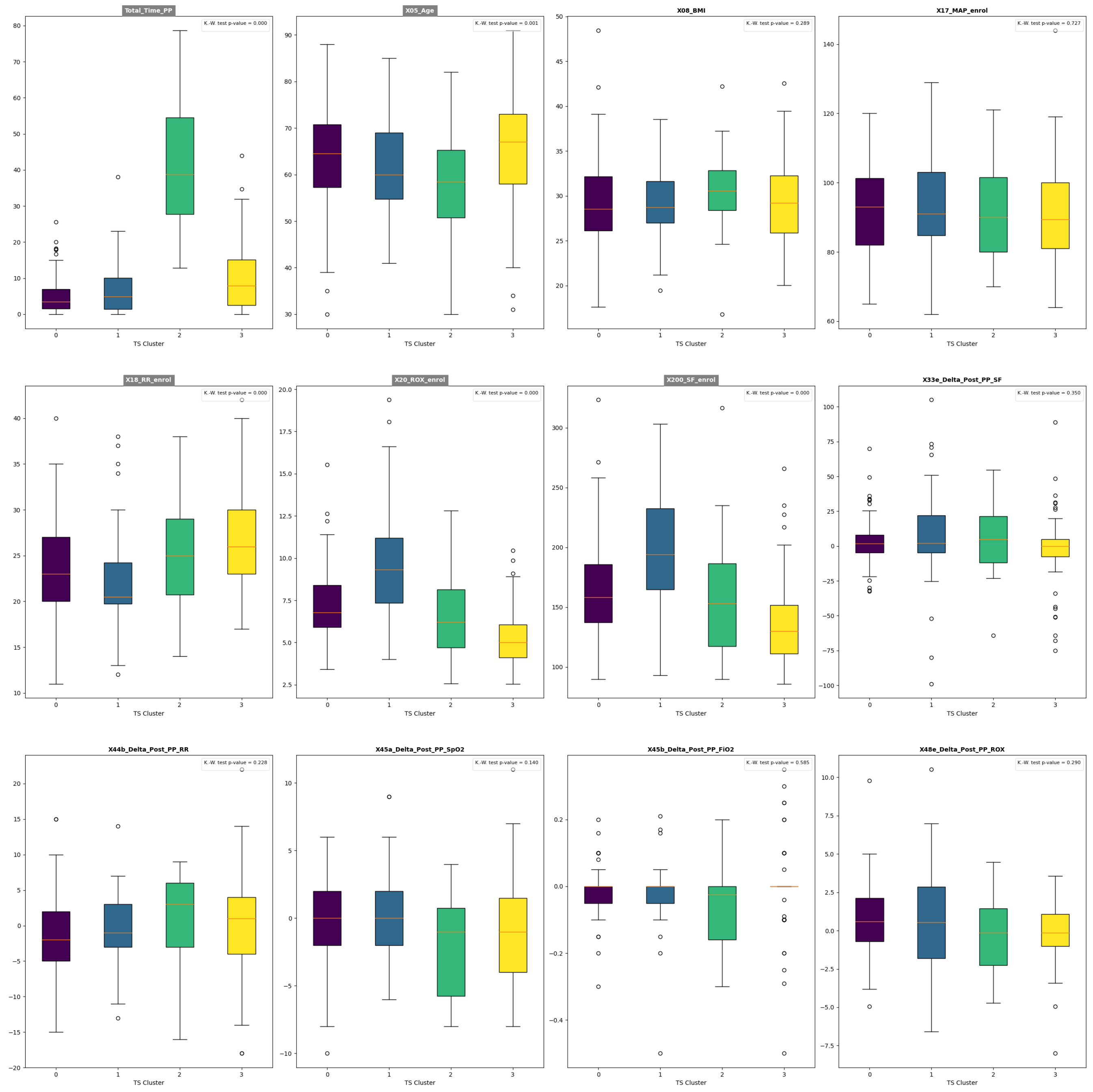


Fig. A.18. Box plots (first, second and third quartiles, whiskers and outliers) of the non-Mexican intervention group, describing the distribution of some continuous variables described in Appendix A.1 within each of the 4 clusters of trajectories depicted in Figure 2.9. Labels in gray indicates a successful Kruskal-Wallis test, a rank-based nonparametric test that determines if there are statistically significant differences between at least two groups of an independent variable on a continuous or ordinal dependent variable [**85**], with a significance level of 0.05. p-values are reported on the top-right corner of each box plot.

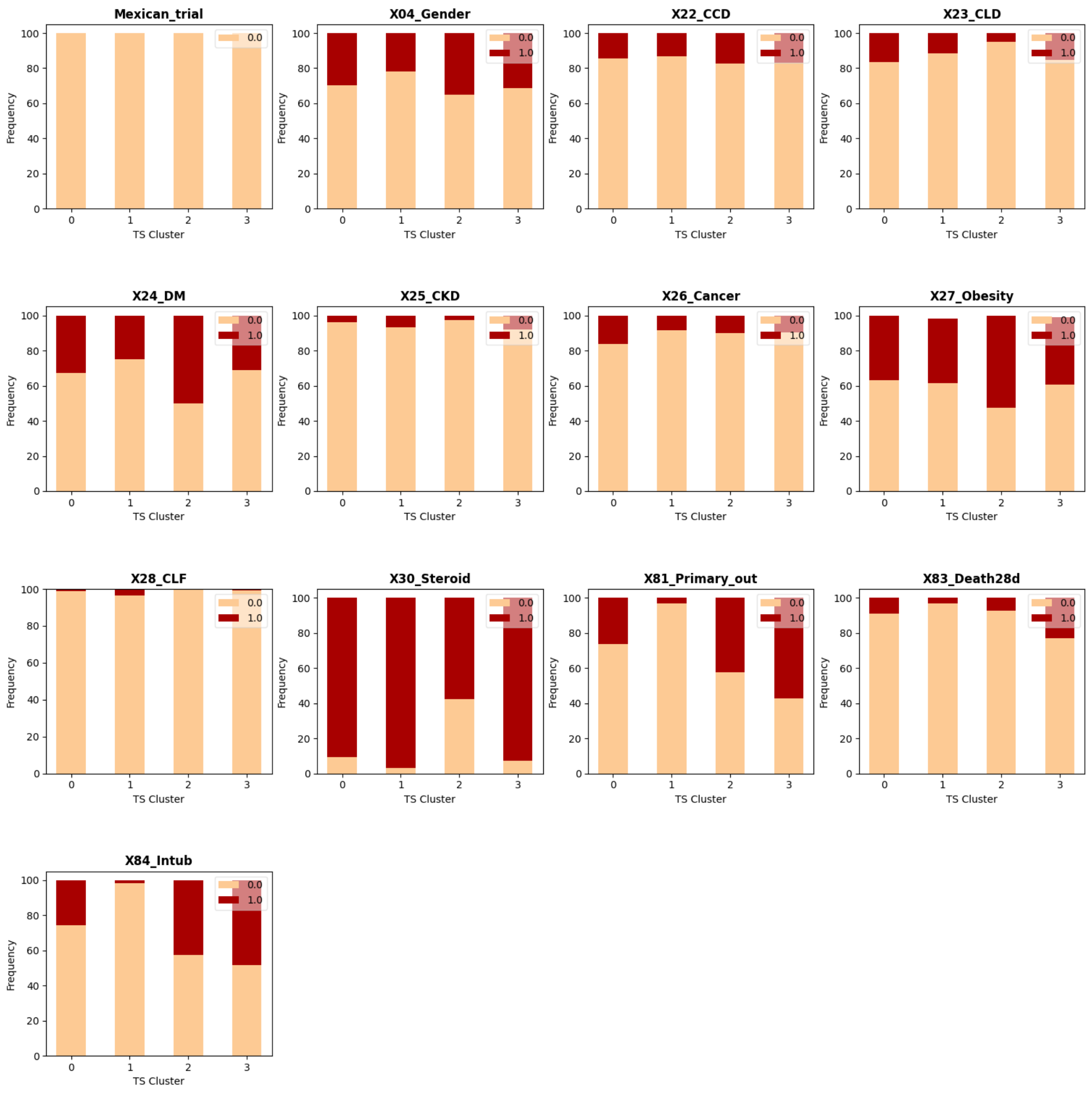


Fig. A.19. Bar plots of the non-Mexican intervention group, describing the frequency of the outcomes and other categorical static variables described in Appendix A.1 within each of the 4 clusters of trajectories depicted in Figure 2.9.

NON-MEXICAN - APP

	CLUSTER_0	CLUSTER_1	CLUSTER_2	CLUSTER_3
Nb_Patients	98	60	40	112
Total_Time_PP_Median	3.5	4.92	38.83	7.88
Total_Time_PP_IQR	5.42	8.64	26.81	12.64
Mexican_trial_Perc	0	0	0	0
X04_Female_Perc	29.59	21.67	35	31.25
X05_Age_Median	64.5	60	58.5	67
X05_Age_IQR	13.5	14.25	14.5	15
X08_BMI_Median	28.51	28.73	30.57	29.21
X08_BMI_IQR	6.01	4.65	4.46	6.36
X17_MAP_enrol_Mean	91.93	93	92.07	90.94
X17_MAP_enrol_SD	12.81	13.53	14.06	13.85
X18_RR_enrol_Mean	23.73	22.13	25.35	26.36
X18_RR_enrol_SD	5.32	5.55	6.07	5.08
X20_ROX_enrol_Mean	7.15	9.54	6.62	5.27
X20_ROX_enrol_SD	2.13	3.24	2.45	1.68
X200_SF_enrol_Mean	162.37	198.63	159.06	133.56
X200_SF_enrol_SD	37.36	43.68	48.74	33.76
X22_CCD_Perc	14.29	13.33	17.5	16.96
X23_CLD_Perc	16.33	11.67	5	15.18
X24_DM_Perc	32.65	25	50	31.25
X25_CKD_Perc	4.08	6.67	2.5	8.04
X26_Cancer_Perc	16.33	8.33	10	9.82
X27_Obesity_Perc	36.73	36.67	52.5	38.39
X28_CLF_Perc	1.02	3.33	0	0.89
Nb_Comorbidities_Mean	1.21	1.05	1.38	1.21
Nb_Comorbidities_SD	1.05	1.03	1	0.94
X281_1+_Comorbidities_Perc	71.43	61.67	82.5	75
X282_2+_Comorbidities_Perc	36.73	33.33	37.5	35.71
X283_3+_Comorbidities_Perc	9.18	8.33	15	8.93
X284_4+_Comorbidities_Perc	4.08	1.67	2.5	0.89
X285_5+_Comorbidities_Perc	0	0	0	0
X286_6+_Comorbidities_Perc	0	0	0	0
X287_7+_Comorbidities_Perc	0	0	0	0
X30_Steroid_Perc	90.82	96.67	57.5	92.86
X48_ROX_change_Mean	0.71	0.73	-0.12	-0.19
X48_ROX_change_SD	2.27	3.38	3.03	1.98
X81_Primary_out_Perc	26.53	3.33	42.5	57.14
X83_Death28d_Perc	9.18	3.33	7.5	23.21
X84_Intub_Perc	25.51	1.67	42.5	48.21

Fig. A.20. Quantitative description of the non-Mexican intervention group for each of the 4 clusters of trajectories depicted in Figure 2.9 in terms of features described in Appendix A.1.

Appendix B

MS long-term visualization: complementary information

B.1. Additional plots

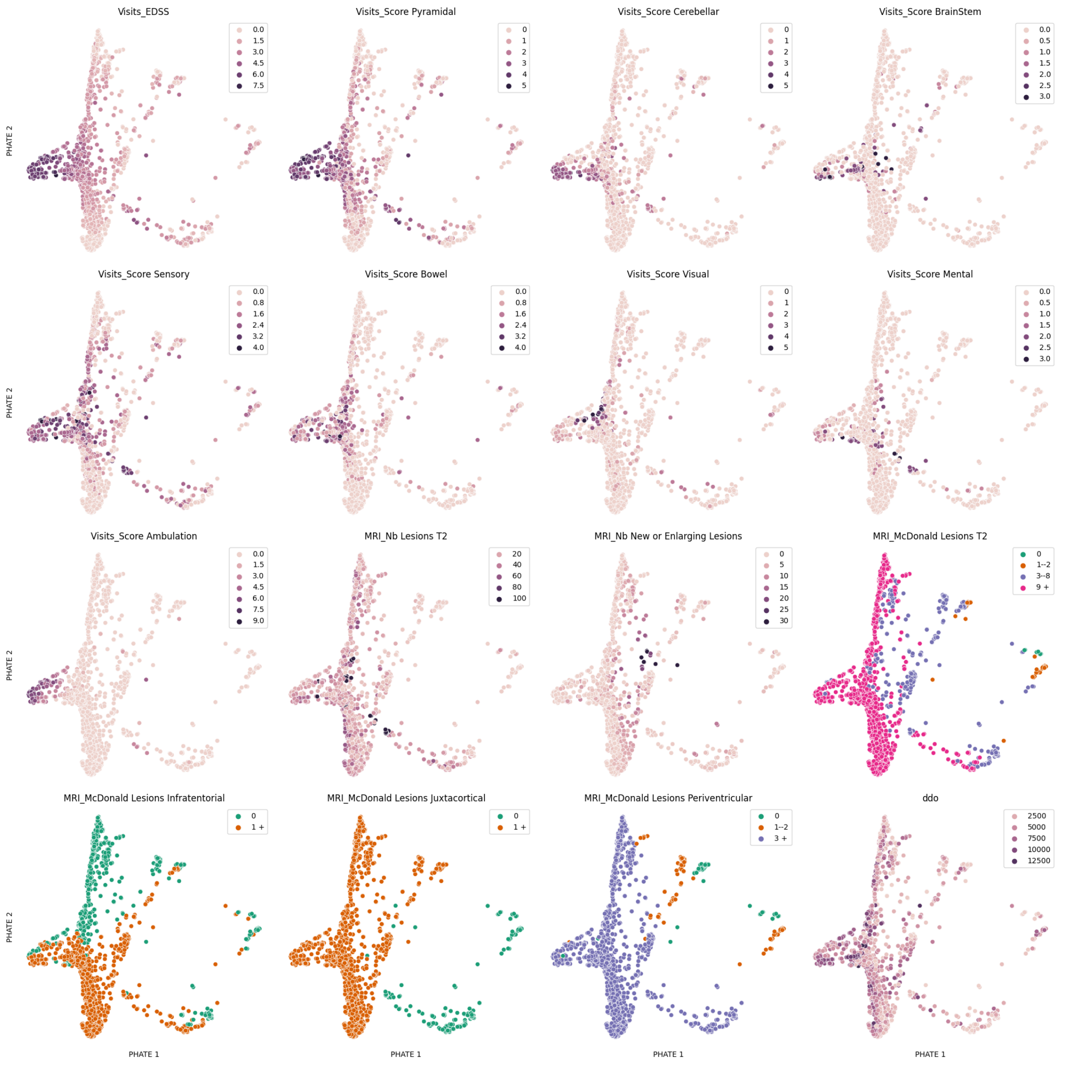


Fig. B.1. 2-dimensional PHATE embedding of the MS database following our processing steps in Section 3.3.1. Each of the 16 sub-plots represents the same PHATE embedding, except that the embedding points are colored differently according to the 16 input features as described in Section 3.2.

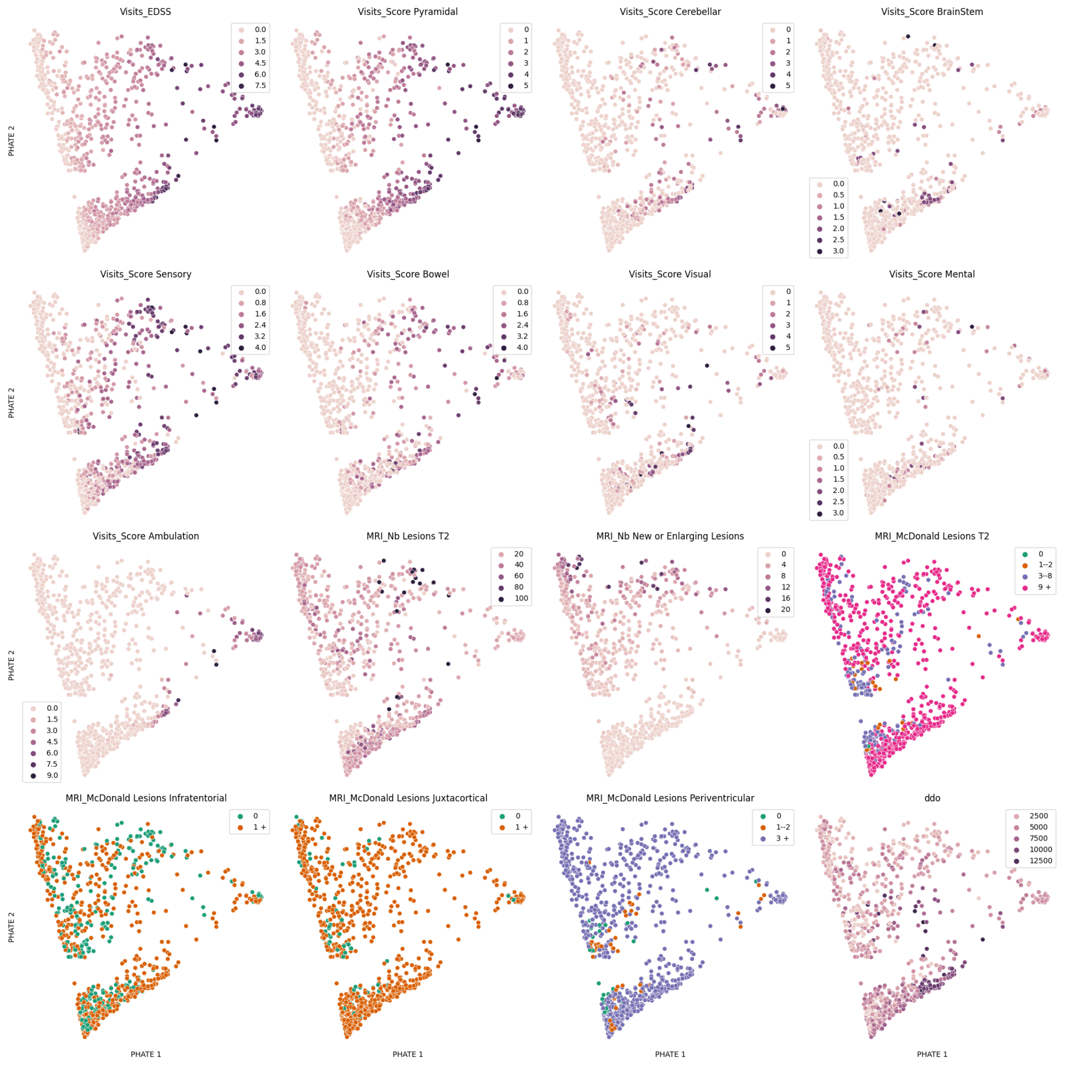


Fig. B.2. 2-dimensional RF-PHATE embedding of the MS database using the target feature $wors5years$ and following our processing steps in Section 3.3.1. Each of the 16 sub-plots represents the same RF-PHATE embedding, except that the embedding points are colored differently according to the 16 input features as described in Section 3.2.

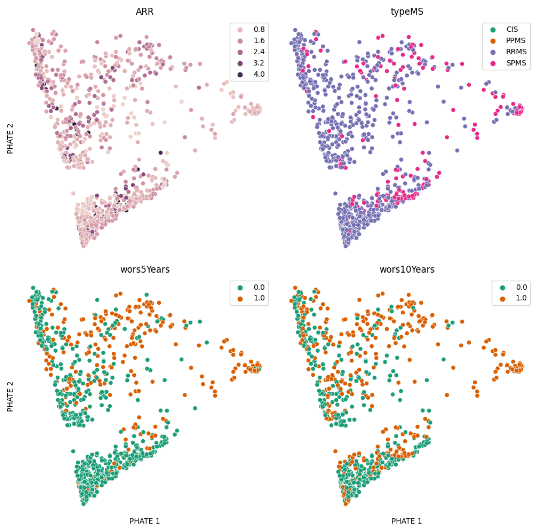


Fig. B.3. 2-dimensional RF-PHATE embedding of the MS database using the target feature $wors5years$ and following our processing steps in Section 3.3.1. Each of the 6 sub-plots represents the same RF-PHATE embedding, except that the embedding points are colored differently according to 6 external features as described in Section 3.2.

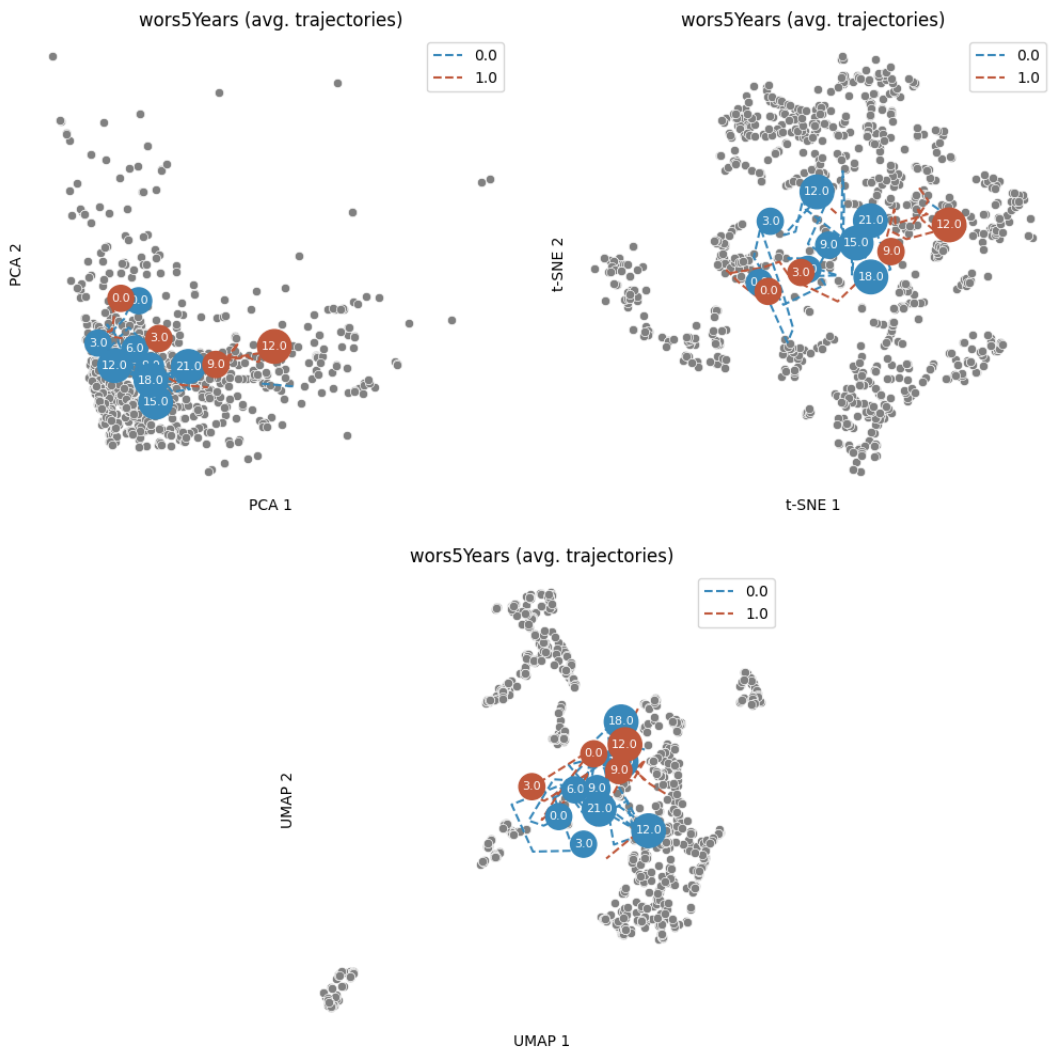
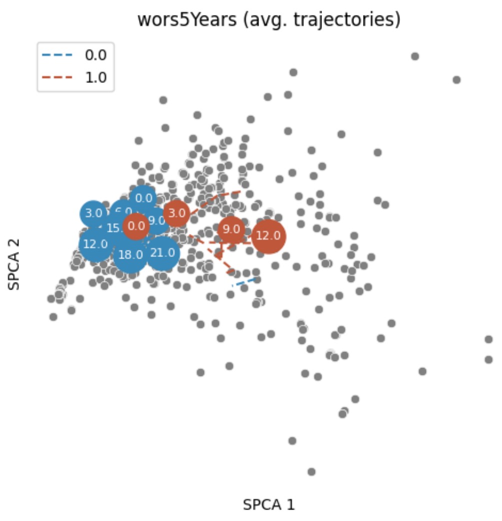


Fig. B.4. Average trajectories with sliding windows of the two $wors5years$ groups of patients using three unsupervised 2-dimensional embeddings: PCA (top left), t-SNE (top right) and UMAP (bottom center). The dotted red line plots the average trajectory of the group undergoing disease worsening after 5 years (i.e. $wors5years = 1$). The dotted blue line plots the average trajectory of the group without disease worsening after 5 years (i.e. $wors5years = 0$). Labels represent the time on which each window is centered, in years since disease onset.



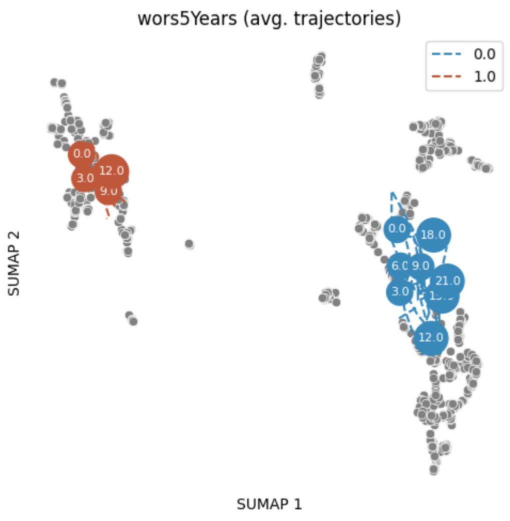


Fig. B.5. Average trajectories with sliding windows of the two $wors5years$ groups of patients using two supervised 2-dimensional embeddings: supervised PCA (top) and supervised UMAP (bottom). The dotted red line plots the average trajectory of the group undergoing disease worsening after 5 years (i.e. $wors5years = 1$). The dotted blue line plots the average trajectory of the group without disease worsening after 5 years (i.e. $wors5years = 0$). Labels represent the time on which each window is centered, in years since disease onset.

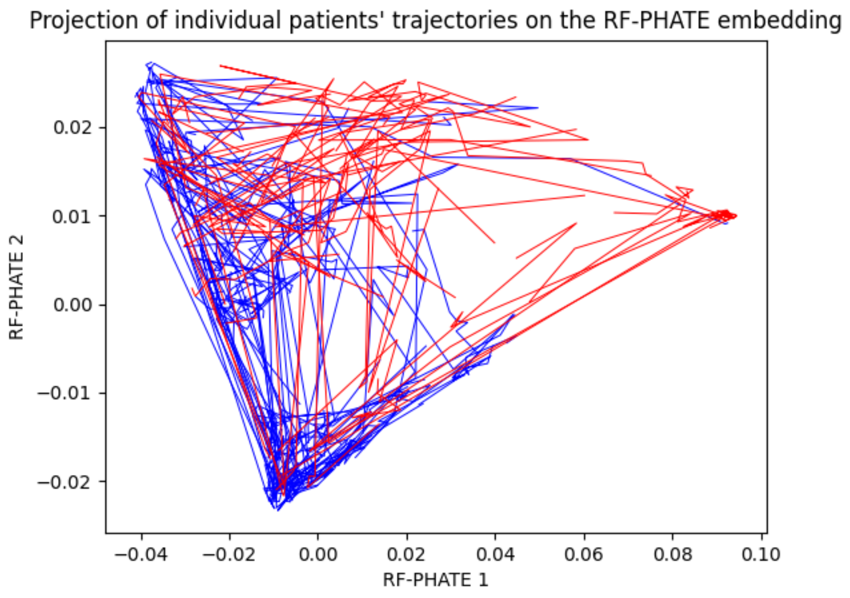


Fig. B.6. 2-dimensional RF-PHATE projection of each individual patient's trajectory. Trajectories in blue refer to patients without MS worsening after 5 years since disease onset (i.e. $wors5years$=0) whereas trajectories in red refer to patients with MS worsening after 5 years since disease onset (i.e. $wors5years$=1)

B.2. Topological Data Analysis

B.2.1. Definition

TDA can be seen as the bridge between the strong theory of algebraic topology and its computational application. Whereas the early stages of TDA date back to the 1990s [**48**][**120**], it was definitely introduced in 2000, when Edelsbrunner, Letscher and Zomorodian [**44**] released a paper which formalized an idea of TDA through the use of Persistent Homology (PH). PH defines topological structures as connected components, holes or rings, and uses them to characterize the shape of data at multiple scales [**108**]. The basic idea of PH is to make use of a filtration function such as the Vietoris–Rips filtration [**141**] which, given a resolution scale parameter r, converts a n-dimensional point cloud data set $\mathbf{X} \subset \mathbb{R}^n$ into a more abstract algebraic structure $\mathbf{X}_r$ called *simplicial complex* on which the above-mentioned features (connected components, holes etc.) are rigorously defined as k-dimensional holes referring to elements of the corresponding k-homology groups (Figure B.7 [**26**]). Finally, the persistence of a k-dimensional hole is defined as its lifetime during the filtration process, that is, how long it lasts between its scale of appearance (or birth) $r = b$ and its scale of disappearance (or death) $r = d$. For each k, the persistence of all the k-dimensional holes are summarized in a persistence diagram D_k.

Remark B.2.1. *To lighten the section, we do not detail the theoretical foundations of PH, including the fundamental definition of the k-homology groups. Instead, we provide an intuitive idea of the process and rather focus on its practical usefulness.*

The steps of the algorithm leading to the production of the k-homology persistence diagrams D_k is performed as follows:

(1) create n-dimensional balls of increasing diameter r with each data point as their centre;
(2) build a Vietoris–Rips filtration: for each value of r, connect with an edge pairs of points that are no further apart than r and add in complete simplices if multiple points are in the same r-neighbourhood;
(3) apply homology to the resulting simplicial complex for all values of r to detect any k-dimensional holes (Figure B.8, (b)-(e)[**108**]);

(4) follow the persistence of any k-dimensional hole as the value of r increases and create persistent diagram. Here, the persistence interval of a hole corresponds to $[b, d)$ for which b and d are the values of r where the hole is born and dies, respectively (Figure B.8, (f) [**108**]).

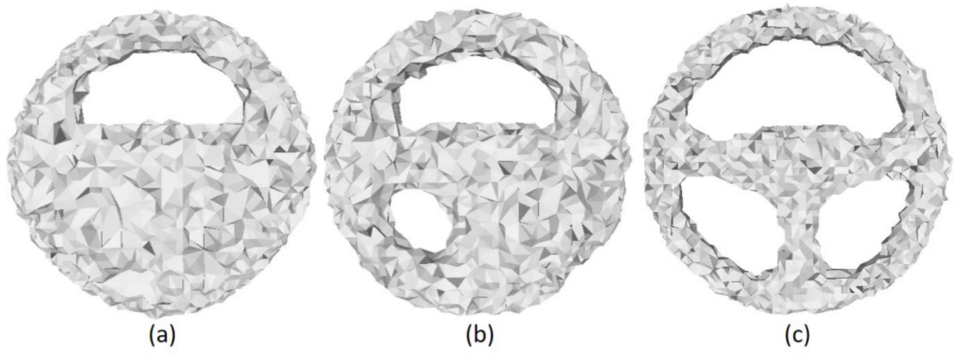


Fig. B.7. Topological representation of a steering wheel through some filtration function that segments the original point cloud at three different resolution scales with an agglomeration of simplexes, a generalized notion of triangles. (a) Largest resolution scale of the filtration in which only 1 hole has been properly reconstructed. (b) Decreasing the resolution results in the appearance of another hole. (c) Finest resolution in which the three typical rings of the steering wheel have been reconstructed. Persistent homology considers all of those scales to detect holes that persist across the filtration, because long-lasting holes are generally associated with the most interesting topological features to retrieve from a point cloud.

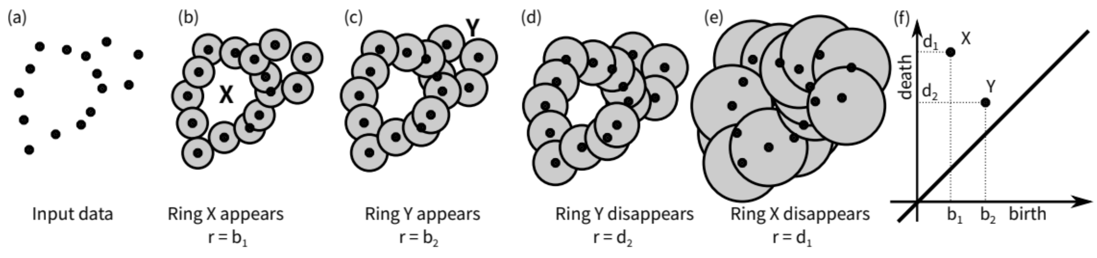


Fig. B.8. Production of the 1-homology group persistence diagram D_1 –informally, loops, or rings– through Vietoris–Rips filtration. (a) Input point cloud data. (b) Point cloud and discs of increasing diameters with $r = b_1$, (c) $r = b_2$, (d) $r = d_2$ and (e) $r = d_1$. (f) D_1.

B.2.2. Sketched idea of the use of TDA

More practically, let us focus on the task of determining distinct temporal patterns for the binary outcome $wors5years$ in terms of all the K available variables, both continuous and categorical. We have two groups of patients' trajectories $\mathbf{G}_0$ and $\mathbf{G}_1$ corresponding to the endpoints $wors5years = 0$ and $wors5years = 1$ respectively. To be consistent with the outcome, we apply a time cutoff of $T \leq 5$ years since disease onset on trajectories such that

$$\mathbf{G}_0 = \{P_1^T, \ldots, P_m^T\}$$
$$\mathbf{G}_1 = \{Q_1^T, \ldots, Q_n^T\}$$

where P_i^T and Q_j^T for $1 \leq i \leq m$ and $1 \leq j \leq n$ are patients' i and j K-dimensional time series up to T years since disease onset. In our previous approach, we described $\mathbf{G}_0$ and $\mathbf{G}_1$ with spatial averages on sliding windows. Here, we keep the sliding windows but suggest to describe $\mathbf{G}_0$ and $\mathbf{G}_1$ thanks to sequences of persistence diagrams encoding the topological structure of each window. In other words, with predefined values for $\mathbf{w}$ and $\mathbf{s} \leq \mathbf{w}$, we first generate the two sequences of windows

$$\mathbf{G}_0^{win} = \{W_0^0, W_{\mathbf{s}}^0, W_{2\mathbf{s}}^0, \ldots, W^0_{\lceil \frac{T}{\mathbf{s}} \rceil \mathbf{s}}\}$$
$$\mathbf{G}_1^{win} = \{W_0^1, W_{\mathbf{s}}^1, W_{2\mathbf{s}}^1, \ldots, W^1_{\lceil \frac{T}{\mathbf{s}} \rceil \mathbf{s}}\}$$

where $W_i^j \subset \mathbb{R}^K$ is the aggregation of all patients' K-dimensional recording days of group j within the time period $[i - \mathbf{w}/2, i + \mathbf{w}/2]$, i.e. a window of size $\mathbf{w}$ centered at time $t = i$ ddo. In this way, we no longer consider individual patients' time series; instead, $\mathbf{G}_j^{win}$ can be seen as an aggregate of the global motion of the group of patients' trajectories $\mathbf{G}_j$. Still, we have no control over the patients' recording days included into W_i^0 and W_i^1 point clouds that contain all the original noise and outliers as they simply take the data set as it is and put it into boxes. In fact, those sliding windows are neither suitable for visualizing group trajectories (see our average trajectories in the previous sections), nor for quantitatively comparing the temporal trends of $\mathbf{G}_0$ and $\mathbf{G}_1$, because we cannot choose a meaningful (dis)similarity measure between the global trajectories $\mathbf{G}_0^{win}$ and $\mathbf{G}_1^{win}$ if the windows W_i^j differ a lot in size and/or in number of patients.

Therefore, we need to provide an alternative representation for each V_i^j and our direction inclines towards TDA. For each window $V_i^j \subset \mathbb{R}^K$, we first construct its distance matrix $\mathbb{M}_i^j$ with a predefined distance $f : \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}^+$, such as the Euclidean distance.

Then, we generate $\mathcal{K}_i^j = \left(D_0(\mathrm{M}_i^j), D_1(\mathrm{M}_i^j)\right)$ where D_k maps a distance matrix to its corresponding persistence diagram of the k-homology class by Vietoris–Rips filtration. Thus, we have managed to describe $\mathbf{G}_0$ and $\mathbf{G}_1$ with the sequences

$$\mathbf{G}_0^{diag} = \{\mathcal{K}_0^0, \mathcal{K}_{\mathrm{s}}^0, \mathcal{K}_{2\mathrm{s}}^0, \ldots, \mathcal{K}^0_{\lceil \frac{T}{\mathrm{s}} \rceil \mathrm{s}}\}$$
$$\mathbf{G}_1^{diag} = \{\mathcal{K}_0^1, \mathcal{K}_{\mathrm{s}}^1, \mathcal{K}_{2\mathrm{s}}^1, \ldots, \mathcal{K}^1_{\lceil \frac{T}{\mathrm{s}} \rceil \mathrm{s}}\}.$$

Technically, $\mathcal{K}_i^j$ is a pair of persistence diagrams: one for the 0-homology class (connected components) and the other for the 1-homology class (loops). However, we can combine those diagrams so that $\mathcal{K}_i^j$ can be seen as a single diagram summarizing the persistence of multiple homology classes. We can also extend the number of homology classes to explore more structure; however, because of lengthy computations of Vietoris–Rips complexes [**155**], it is often a good idea to start with the first homology classes.

As a first step into our MS time series exploration with TDA, we can compare the temporal trends of $\mathbf{G}_0$ and $\mathbf{G}_1$ through their topological sequences $\mathbf{G}_0^{diag}$ and $\mathbf{G}_1^{diag}$ along with an appropriate (dis)similarity measure between sequences of diagrams. Since we already know a robust distance d between persistence diagrams, such as the Bottleneck distance that has been used in the stability theorem [**40**], we can make use of it to extend to sequences of diagrams. For instance, one could define the local distance between two pairs of diagrams $(\mathcal{K}_i^0, \mathcal{K}_i^1)$ as the averaged distance between diagrams of the respective homology classes,

$$\texttt{LocalDist}(\mathcal{K}_i^0, \mathcal{K}_i^1) = \frac{d\left(D_0(\mathrm{M}_i^0), D_0(\mathrm{M}_i^1)\right) + d\left(D_1(\mathrm{M}_i^0), D_1(\mathrm{M}_i^1)\right)}{2},$$

and the global distance between the two sequences as the step-wise average of the local distances over the windows $0 \le i \le \lceil \frac{T}{\mathrm{s}} \rceil$ s:

$$\texttt{GlobalDist}(\mathbf{G}_0^{diag}, \mathbf{G}_1^{diag}) = \frac{1}{\lceil \frac{T}{\mathrm{s}} \rceil + 1} \left(\sum_i \texttt{LocalDist}(\mathcal{K}_i^0, \mathcal{K}_i^1) \right).$$

Of course, the comparison of those two groups of patients is not inherently insightful since there is no interpretation of a single resulting distance. In fact, the interesting part arises when considering a lot of different groups and comparing their pairwise global distances through heat maps, a task that is left for future directions.

B.2.3. Visual representation with persistence landscapes

Instead of describing grouped motions with sequences of persistence diagrams which cannot be easily visualized, there have been novel advancements to directly describe grouped motions with sequences of persistence landscapes [**27**]. Informally, given a homology class k, a persistence landscape is a set of piecewise linear functions containing the amount of topological structure in a persistence diagram D_k, from which metrics quantifying this amount can be derived using L^p-norms. Thus, a sequence of persistence landscapes depicts the evolution of a group of patients' MTS over time with a single UTS that preserves the invariant and stable properties of TDA (Figure B.9)! However, one question suddenly

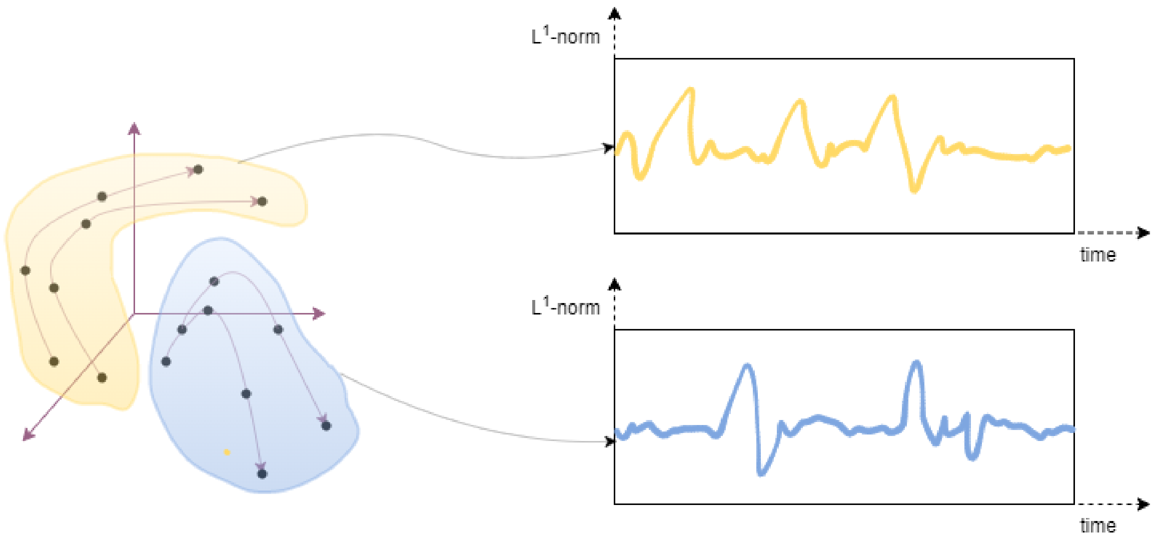


Fig. B.9. Synthetic scheme of the topological signals associated with two groups of patients' MTS. Left: two groups of patients' MTS according to some outcome of interest, such as worsening/no worsening. Right: corresponding topological UTS representations of the two groups, obtained through persistence landscapes on sliding windows for some k-homology class.

comes out: how is a topological UTS clinically insightful? Since some papers observe that topological descriptors can reveal interesting hidden structures that were not previously studied [**101**], we believe that there may exist underlying pattern correlations between the topological UTS and the MS progression of a group of patients. Still, due to the lack of persistence landscape applications among the existing literature, the best way to face those unanswered questions is through future experimentation on our MS data and other healthcare projects.