PROYECTO FIN DE GRADO

TITLE:

Subsequence-based counterfactual explanations for time series

AUTOR/A: Pablo Sanz Rodríguez

TITULACIÓN: Grado en Ingeniería y Sistemas de Datos

DIRECTOR/A: Mario Refoyo López

TUTOR/A: David Luengo García

DEPARTAMENTO: Departamento de Ingeniería Audiovisual y Comunicaciones

VºBº TUTOR/A

Miembros del Tribunal Calificador:

PRESIDENTE/A: Agustín Rodríguez Herrero

TUTOR/A: David Luengo García

SECRETARIO/A: María Luisa López Ibáñez

Fecha de lectura: 30/09/2024

Calificación:

El Secretario/La Secretaria,

Acknowledgements

Primeramente, me gustaría agradecer a mi familia, en especial a mis padres y a mi hermana que son quienes han vivido conmigo el día a día de estos 4 años de carrera y son quienes me han visto en mis mejores y en mis peores momentos, siendo siempre el mejor apoyo que he podido tener.

A mis amigos de toda la vida, quienes han vivido conmigo los años previos a la carrera y los que han vivido conmigo indirectamente este camino por la universidad. Por hacerme con vuestra compañía y amistad la vida un poco más bonita.

A mis amigos de la carrera, mis compañeros de batalla estos últimos años, los cuales han endulzado mi paso por la universidad y de los que he podido aprender y llevarme un poco de ellos conmigo para siempre.

Y por último a David y Mario quienes han puesto su tiempo, sus conocimientos y su dedicación para que yo pudiese ponerle broche final a esta etapa con la realización de este proyecto, que queda como mi granito de arena tanto para la tesis doctoral de Mario como para el área de la Inteligencia Artificial Explicable.

Gracias a todos vosotros por ser partícipes de mi camino.

Resumen

Este trabajo presenta un enfoque innovador dentro del ámbito de la Inteligencia Artificial Explicable (XAI) para la generación de explicaciones contrafácticas basadas en subsecuencias para modelos de clasificación de series temporales. El creciente uso de modelos de *Machine Learning* (ML), en áreas críticas como la medicina, las finanzas y la seguridad plantea la necesidad de métodos que permitan entender y justificar las decisiones tomadas por estos sistemas. Las explicaciones contrafácticas son especialmente útiles para este propósito, ya que permiten identificar los cambios mínimos necesarios en los datos de entrada que habrían conducido a un resultado diferente, facilitando así la interpretación y aumentando la confianza de los usuarios en los modelos de ML.

El objetivo principal de este proyecto es implementar y evaluar el funcionamiento de tres métodos post-hoc para la generación de explicaciones contrafácticas en series temporales: TimeX, Attention-based Counterfactual Explanation for Multivariate Time Series (AB-CF) y and Motif-Guided Time Series Counterfactual Explanations (MG-CF). El método TimeX utiliza un enfoque de optimización iterativa basado en el promedio baricéntrico dinámico (DBA) para ajustar segmentos significativos de las series temporales, mejorando la plausibilidad de las explicaciones generadas. AB-CF se basa en identificación de las subsecuencias críticas y genera explicaciones contrafácticas mediante el intercambio de estas subsecuencias más representativas de la clase de entrada por su *Nearest Unlike Neighbor*, permitiendo modificaciones precisas que alteren la clasificación del modelo. Por último, MG-CF emplea la detección de *motifs,* que son los segmentos más representativos específicos de cada clase de salida, y los intercambia.

Estos tres métodos han sido evaluados utilizando cinco conjuntos de datos del archivo *University of California, Riverside* (UCR) de clasificación de series temporales: CBF, Chinatown, ECG200, Gunpoint y Coffee. Estos *datasets* ofrecen una diversidad de desafíos, lo que ha permitido analizar la efectividad de los métodos en diferentes escenarios. Las métricas de evaluación incluyen proximidad, *sparsity*, plausibilidad, validez y eficiencia. Los resultados de estas métricas muestran que AB-CF proporciona explicaciones altamente interpretables, al conseguir unos excelentes valores de validez, una *sparsity* moderada y un tiempo de ejecución muy bajos, mientras que TimeX sorprende con una validez muy baja y con tiempos de ejecución altísimos. MG-CF, por su parte, fluctúa con valores bajos como altos dependiendo del conjunto de datos evaluado y con unos tiempos de cálculo que facilitan enormemente su uso.

Por último, se toma como conclusión la significativa aportación del proyecto al campo de XAI mediante la comparación de diferentes metodologías que mejoran la interpretabilidad de los modelos de series temporales. Estas técnicas no solo incrementan la transparencia en la toma de decisiones automatizadas, sino que también potencian la adopción y confianza de sistemas de ML en sectores donde la explicabilidad es crucial.

Abstract

This paper presents an innovative approach within the field of Explainable Artificial Intelligence (XAI) for the generation of subsequence-based counterfactual explanations for time series classification models. The increasing use of Machine Learning (ML) models in critical areas such as medicine, finance and security raises the need for methods to understand and justify the decisions made by these systems. Counterfactual explanations are particularly useful for this purpose, as they allow identifying the minimum necessary changes in the input data that would have led to a different outcome, thus facilitating interpretation and increasing users' confidence in ML models.

The main objective of this project is to implement and evaluate the performance of three post-hoc methods for generating counterfactual explanations in time series: TimeX, Attention-based Counterfactual Explanation for Multivariate Time Series (AB-CF) and Motif-Guided Time Series Counterfactual Explanations (MG-CF). The TimeX method uses an iterative optimisation approach based on dynamic barycentric averaging (DBA) to fit significant segments of the time series, improving the plausibility of the generated explanations. AB-CF is based on the identification of critical sub-sequences and generates counterfactual explanations by exchanging these most representative subsequences of the input class for their Nearest Unlike Neighbor, allowing precise modifications that alter the classification of the model. Finally, MG-CF employs the detection of motifs, which are the most representative segments specific to each output class, and swaps them.

These three methods have been evaluated using five datasets from the University of California, Riverside (UCR) time series classification archive: CBF, Chinatown, ECG200, Gunpoint and Coffee. These datasets offer a diversity of challenges, allowing the effectiveness of the methods to be analysed in different scenarios. The evaluation metrics include proximity, sparsity, plausibility, validity and efficiency. The results of these metrics show that AB-CF provides highly interpretable explanations by achieving excellent validity values, moderate sparsity and a very low runtime, while TimeX surprises with very low validity and very high runtimes. MG-CF, on the other hand, fluctuates with both low and high values depending on the dataset evaluated and with calculation times that make it very easy to use.

Finally, we conclude that the project makes a significant contribution to the field of XAI by comparing different methodologies that improve the interpretability of time series models. These techniques not only increase transparency in automated decision making, but also boost the adoption and confidence of ML systems in sectors where explainability is crucial.

List of figures

List of tables

List of abbreviations

AB-CF - Attention-based Counterfactual Explanation for Multivariate Time Series

AI - Artificial Intelligence

AUC - Area Under the Curve

CBF - Cylinder-Bell-Funnel

CNN - Convolutional Neural Network

DBA - Dynamic Barycenter Average

DTW - Dynamic Time Warping

FN - False Negative

FP - False Positive

IF_OS - Isolation Forest Outlier Score

LIME - Local Interpretable Model-Agnostic Explanations

MG-CF - Motif-Guided Time Series Counterfactual Explanations

ML - Machine Learning

MRML - Multiple Linear Regression Model

NUN - Nearest Unlike Neighbor

PCA - Principal Component Analysis

ReLU - Rectified Linear Unit

RNN - Recurrent Neural Network

ROC - Receiver Operating Characteristic

SHAP - SHapley Additive exPlanations

SVM - Support Vector Machine

TN - True Negative

TP - True Positive

XAI - Explainable Artificial Intelligence

Table of Contents

1. Introduction

Artificial Intelligence (AI) has emerged as a powerful tool in a variety of fields, revolutionizing the way we process information and make decisions [1]. By employing algorithms and computational models, AI has the ability to analyze large amounts of data and find complex patterns that can be used to make predictions and automated decisions.

However, in many cases the functioning of AI models can be opaque and difficult for humans to understand. This is where explainable AI (XAI) comes in, a branch of artificial intelligence that focuses on making selected models interpretable. This approach is essential in domains where the decisions made by the models are of great importance, where transparency and interpretability for the user are crucial in making informed decisions [2].

Within the framework of explainable AI, one of the key concepts is the use of counterfactual explanations [3], which will be the basis of the project to be developed. While traditional explanations can provide answers as to why a decision was made, counterfactual explanations go further by offering concrete suggestions as to how a different classification outcome could have been achieved. These explanations, by identifying specific changes in characteristics or input variables, provide guidance on actions that could have led to an alternative conclusion.

More specifically, the project will focus on counterfactual explainable methods for time series [4], and its main goal will be on the implementation of the following three post-hoc explanatory AI algorithms: TimeX [5], AB-CF [6] and MG-CF [7]. These 3 methods focus on sub-sequences within the total time series, trying to find the minimum change in these small sections so that the classification result is the opposite of the one initially obtained.

1.1 Objectives

The main objective of the project is the implementation of three current methods of counterfactual explanations and the evaluation of their results through different metrics, in order to provide understandable and useful interpretations for time series of different kinds.

For the development of this main objective, a number of specific project objectives have been defined, which will help to fulfil the main goal. The sub-objectives of the project are as follows:

- Implementation of the XAI methods using Python, as well as the necessary metrics for the evaluation of their performance.
- Optimization of the implemented code.
- Comparison of the different metrics and drawing conclusions about the quality of the implemented and evaluated methods.

1.2 Design specifications and restrictions

The project must satisfy the following specifications and design constraints:

- The black box models to be explained must maintain an adequate level of accuracy in their classifications, so that the use of the explainability methods makes sense.
- The training and inference time of the models to be explained must be reasonable, depending on the input signals.
- Explanations developed by explainability methods should be clear and consistent with the conclusions drawn in their respective publications.
- The proper functioning of the implemented methods should be checked with several UCR datasets [8].
- The models have to be implemented using the machine learning model building library Tensorflow [9].

1.3 Proposed working methodology

For the development of the project, a cyclical methodological approach involving several iterative stages will be used. First, a phase of research and reading of articles related to the methods to be implemented will be carried out.

Subsequently, we will proceed with the implementation of the explanatory AI algorithms using the Python programming language and the Visual Studio Code development environment. During this stage, the theoretical concepts will be translated into functional code, ensuring consistency with the project objectives and the established specifications.

Once the code has been developed, a debugging phase will be carried out, using unit tests to verify the accuracy and performance of the models. This will involve the identification and correction of possible errors or inconsistencies in the code, thus ensuring the reliability of the results obtained.

Finally, a comparison of the results obtained with the results published in the respective articles will be carried out. If the results match the expectations and conclusions of the existing documentation, the following explanatory method will be implemented. However, if discrepancies are found between the results obtained and the published results, we will return to the phase of research and reading of documents to identify possible failures or areas for improvement in the developed algorithms.

1.4 Breakdown of tasks and timetable

The development of the project lasted approx. 5 months with full-time dedication. Requiring 360 hours, with the following breakdown of tasks:

- Task 1: Initial Tasks

a) Introduction to the context of Explainable AI and specifically to counterfactual methods.

b) Deepening in the different Python libraries to be used for the development of the models.

- Task 2: Implementation of the counterfactual explanation algorithms.

a) Coding of the methods.
b) Testing the code.

- Task 3: Analysis and interpretation of results

a) Performance comparison between the developed methods.
b) Drawing conclusions and discussion of results.

- Task 4: Documentation and drafting of the final report

a) Writing of the introduction, methodology and results obtained.
b) Reviewing and editing the report.
c) Preparing the presentation for the defense of the dissertation.

In Table I, the estimated duration of the tasks mentioned above is shown.

Name	Start Date	Duration (days)	Finish Date	Hours
Task 1a	21/02/2024	49	10/04/2024	25
Task 1b	20/03/2024	26	15/04/2024	35
Task 2a	04/03/2024	82	25/05/2024	100
Task 2b	25/04/2024	56	20/06/2024	60
Task 3a	20/06/2024	11	01/07/2024	30
Task 3b	02/07/2024	14	16/07/2024	30
Task 4a	21/02/2024	140	10/07/2024	50
Task 4b	16/07/2024	10	26/07/2024	15
Task 4c	23/07/2024	7	30/07/2024	15
Total				360

Table I. Tasks duration estimation

Also, Figure I shows the GANT diagram of the above tasks, showing the duration of each task over time.

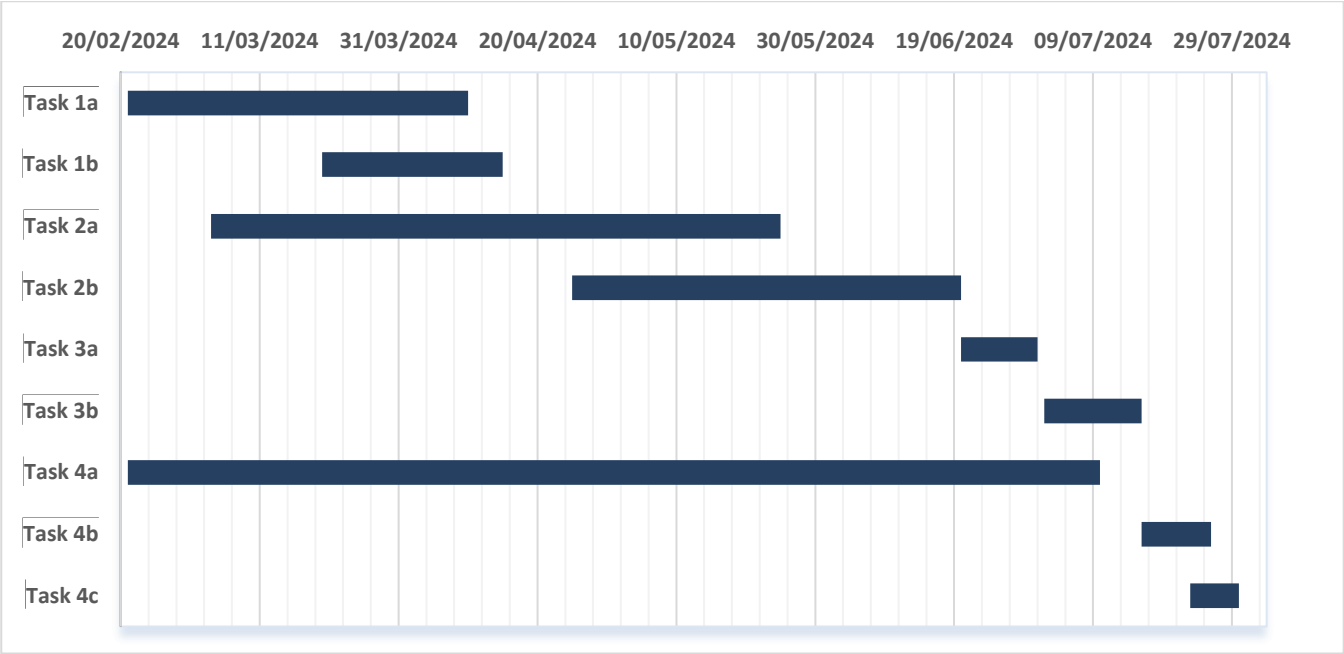

Figure I. Tasks GANT Diagram

1.5 Rest of the report structure

The rest of the report is organized as follows. The second chapter explores the field of Machine Learning and its various typologies, starting with a general introduction before delving into the most recognised methods in the field. An exhaustive analysis is made of the specific types and methods of Machine Learning that will be used in the project, such as classification, convolutional neural networks (CNN) and anomaly detection.

In the third chapter, after having dealt with Machine Learning, we go deeper into Explanaible Artificial Intelligence (XAI), its different approaches and, in particular, counterfactual explanations, since the models developed generate this type of ad-hoc explanations for black box models.

The fourth chapter provides a detailed description of the theoretical basis and implementation of the three XAI models that have been developed for the project: Attention-based Counterfactual Explanation for Multivariate Time Series (AB-CF) [6], Motif-Guided Time Series Counterfactual Explanations (MG-CF) [7] and Counterfactual Explanations using Barycenters (TimeX) [5].

In the fifth chapter, an economic analysis of the project is presented, evaluating the costs from two perspectives: material and personal, concluding with the calculation of the estimated total cost of the project.

The last chapter examines the impact of this project in different areas of society, highlighting and valorizing its contribution to various sectors. In addition, it highlights the relationship of the project with several Sustainable Development Goals (SDGs) of the UN 2030 Agenda, which adds additional value to its realization.

2. Machine Learning

2.1 Introduction

Throughout history, humans have always sought ways to improve their processes and increase efficiency in their daily activities. From the invention of primitive tools to the industrial revolution and the digital age, every advance has been driven by the desire to optimise tasks and a better understanding and adaptation to the environment. In this context, predicting and classifying events have been essential decision-making skills, allowing societies to anticipate changes and adapt their strategies accordingly.

In the current era, characterised by exponential growth in data generation, these skills have become more critical than ever. The ability to analyse large volumes of data and extract useful information from them is fundamental to remain competitive in an endless number of fields. In this context, Machine Learning (ML) has emerged as a revolutionary technology, enabling machines to learn and make data-driven decisions.

Machine Learning is a sub-discipline of artificial intelligence (AI) that focuses on the development of algorithms and models that allow computers to learn from data. Unlike systems explicitly programmed to perform specific tasks, machine learning systems identify patterns and relationships in data, adapting their behaviour to improve their performance on future tasks [10].

Within Machine Learning there are 3 types of models depending on how they learn from data: supervised learning, unsupervised learning and reinforcement learning. In the following, we briefly describe each one.

- **Supervised Learning**

Supervised learning is the most common type of ML nowadays and occurs when a labelled dataset is available. This approach involves training a model using training and test data with labels, i.e. with predicted values assigned. In these cases, the main objective for the model is to be able to relate the features of the labelled data given to it, so that when given only those features it is able to correctly predict the response.

Figure II shows an example of this type of machine learning (ML), where various geometric shapes are labeled (rectangle, triangle, etc.). All this labeled data is fed into a supervised ML model, which learns to identify each shape. After this learning process, new data is passed to our supervised ML model, this time without labels because we want it to predict the labels for the new shapes (a triangle and a circle in this case) shown to it. This new data is known as test data. The model can make these predictions because it has learned how to distinguish the different geometric shapes it encountered during the training process.

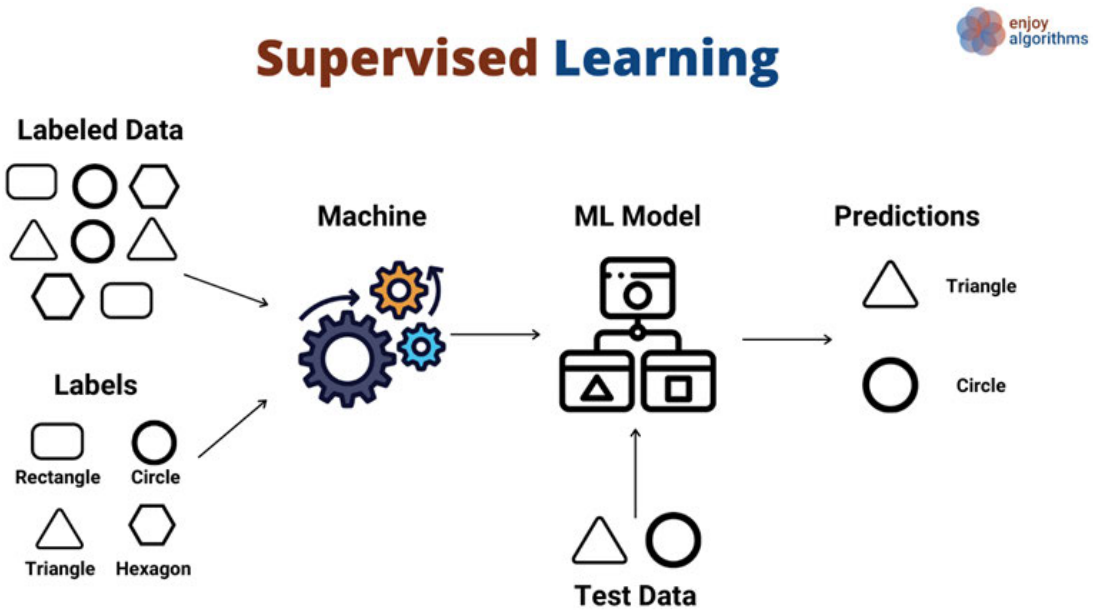

Figure II. Supervised Learning Overview [11]

- **Unsupervised Learning**

In unsupervised learning the data is not labelled, i.e. no desired output is provided. The goal now is to discover hidden patterns or structures in the data. This type of learning is useful for exploring the data and finding unknown relationships. Figure III shows a clustering problem, one of the types of unsupervised learning which will be show in Section 2.3.2. Once again, we have our geometric shapes data, but this time it is unlabeled and now our ML model is able to group together the different geometric shapes.

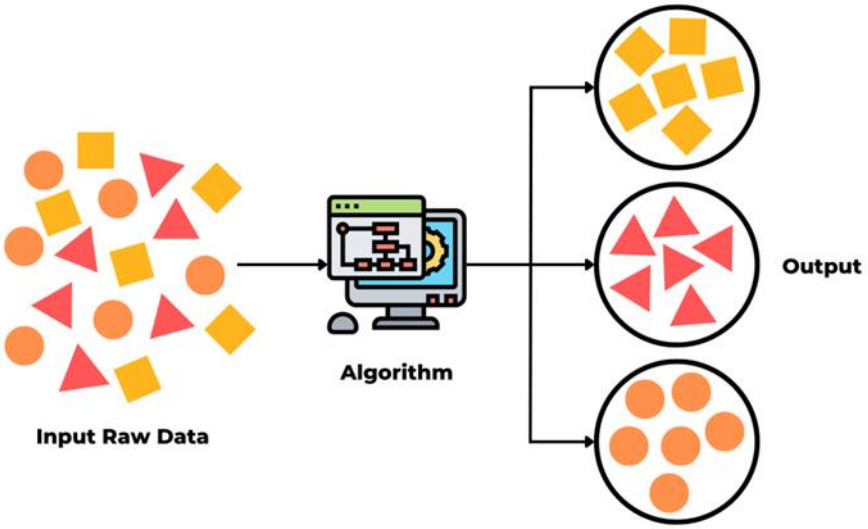

Figure III. Unsupervised Learning Example: Clustering [12]

A clear example of this type of machine learning is the detection of fraud in credit card transactions. In this case, the model learns what your shopping patterns are, such as your shopping hours or the type of purchases you make according to the sector, and the model

tries to identify anomalous or unusual patterns in transactions, which could indicate fraud [13].

- **Reinforcement Learning**

Reinforcement learning is inspired by behavioral psychology and is used to train agents to make sequential decisions. The agent, who is the action-taker, learns through interaction with an environment, receiving rewards or penalties based on its actions and the states it reaches [14].

A classic example from literature is the robot in a maze, as we can see in Figure IV. The more beneficial the robot's movements are in getting out of the maze, the higher rewards it will get, thus learning which movements and states conduct to achieving the final goal.

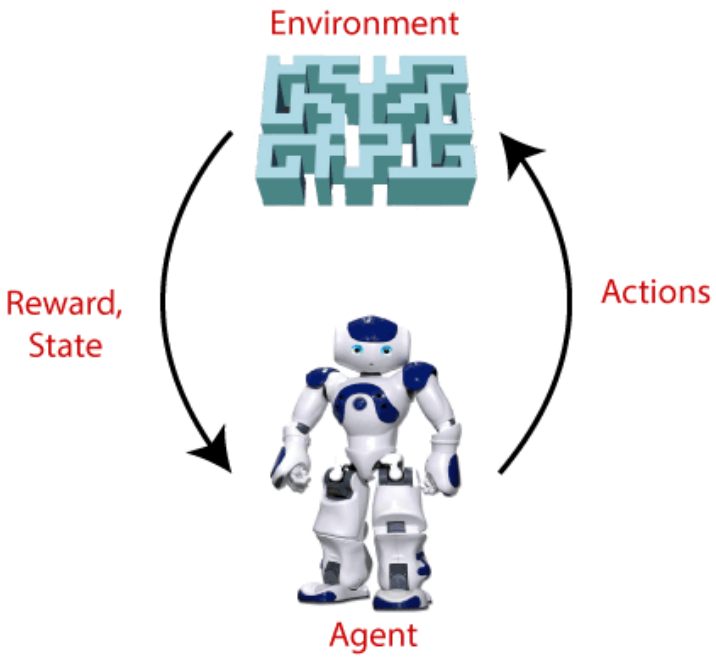

Figure IV. Reinforcement Learning Example: Robot-Maze Problem [15]

2.2 Supervised Learning

Let us take a deeper dive into supervised learning and the methods that constitute this type of machine learning.

2.2.1 Regression

Regression refers to any statistical technique that attempts to estimate or predict a continuous variable. It involves finding a mathematical function that relates one or more independent variables to the dependent variable; this function to be developed can be linear or non-linear.

Regression can be divided into several categories, but the most common ones are linear regression and non-linear regression. Here we will only describe linear regression.

Simple Linear Regression

Linear regression is one of the simplest and most widely used forms of regression. It is based on the assumption that there is a linear relationship between the independent variables and the dependent variable. In its simplest form, linear regression can be represented by the equation:

$$y = \theta_1 + \theta_2 x + \varepsilon$$

where:

θ_1 is the intercept

θ_2 is the slope of the regression

x is the independent variable

ε is the assumed error between the dependent and independent variable

Figure V shows a linear regression example, where we can see how it is trying to fit a dataset and all the variables explained before.

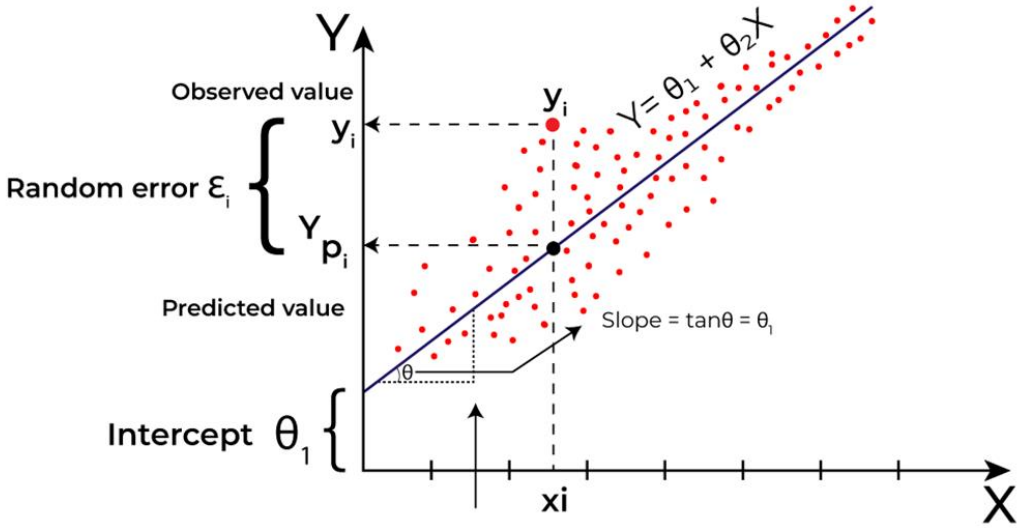

Figure V. Simple Linear Regression [16]

Multiple Linear Regression Model (MLRM)

Linear regression can be extended to include multiple independent variables, resulting in multiple linear regression. The model's equation is now expressed as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \varepsilon$$

where p is the number of independent variables. In this model, each coefficient β_i represents the contribution of the corresponding independent variable x_i to the prediction of y.

Linear Regression Model Fitting

The process of fitting a multiple linear regression model (MLRM) typically involves finding the values of $\beta_0, \beta_1, \dots, \beta_p$ that minimise the sum of squares of the errors (differences between observed and predicted values). This method is known as the least squares method and its error or cost function is defined as:

$$J(\beta_0, \beta_1, \dots, \beta_p) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where n is the number of observations, y_i are the actual values of the dependent variable and $\hat{y}_i$ are the values predicted by the model. A closed form solution exists for this problem, so training the MLRM can be done fast and efficiently [17].

2.2.2 Classification

Classification is a fundamental task in the field of machine learning, where the goal is to assign a label or category to new observations based on a previously labelled training data set. Unlike regression, where a continuous value is predicted, classification predicts a discrete (categorical) label. Classification applications range from spam detection in emails to disease diagnosis in medical imaging [18].

Classification Basics

In a classification problem, there is an input data set X and an output label set Y. The goal is to build a model $f : X \to Y$ that can predict the correct label for any new observation x. Depending on the number of labels, classification problems are divided into [19]:

Binary Classification: there are only two possible classes. A classic example of this type of classification is spam detection, as we can see in Figure VI, where we only have the possibility that the e-mail arriving in the inbox is either spam or not [20].

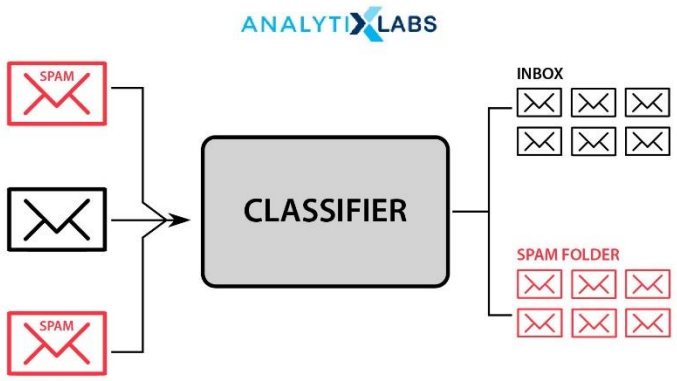

Figure VI. Example of binary classification: Spam detection [21]

Multi-class classification: There are more than two possible classes. An example of this type of classification that has been widely studied in the literature is that of irises, where images of this flower have to be classified according to whether they are iris setosa, iris versicolor or iris virginica through the length and the width of the sepal and the petal of the flowers (Figure VII).

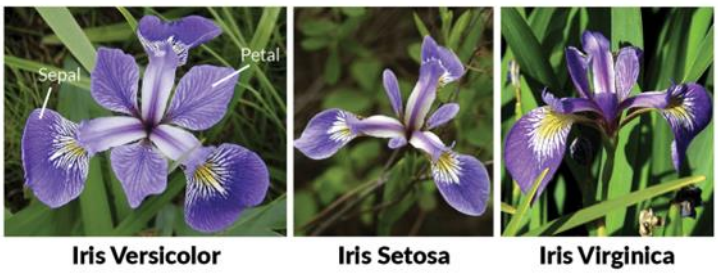

Figure VII. Example of multi-class classification: Iris classification [22]

Multi-Label Classification: Each instance can belong to multiple classes simultaneously. An example is image tagging, where an image can have multiple tags. Figure VIII shows an example of this type of classification: in the image shown the classifier is able to distinguish the different contents found on the image.

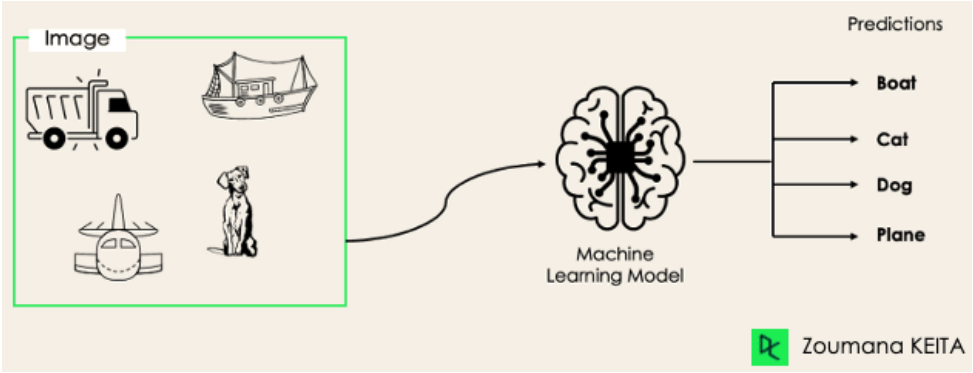

Figure VIII. Multi-Label Classification Example [19]

Classification Model Evaluation

Evaluating the performance of classification models is crucial to ensure that they provide accurate and useful predictions. Some of the most common metrics include the following.

Confusion Matrix

A confusion matrix is a kxk matrix (where k is the number of classes) that summarises model predictions and compares the predictions with their actual values.

It represents:

- True Positives (TP): those cases where the model has got the predicted class right. In a case of cat and dog image classification, and taking dogs as the positive class, this would be the number of times the model predicted that an image was a dog and it was really a dog.

- True Negatives (TN): those cases where the model was correct that it was not the positive class. Continuing with the example, these would be the cases where the model has predicted cat and it was really a cat.

- False Positives (FP): those cases where the model was not correct that it was the positive class. Here the model predicted a dog and it was a cat.

- False Negatives (FN): those cases where the model was not correct that it was the negative class. In the latter case it predicted a cat and it was a dog.

Figure IX shows a representation of a confusion matrix, how it is conformed by the predicted class of the model and the true class of the prediction.

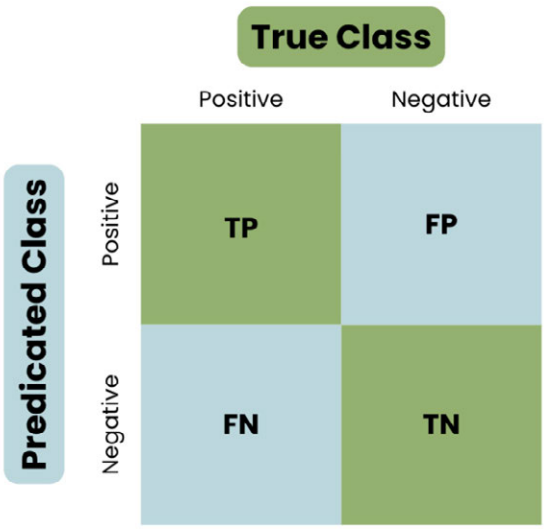

Figure IX. Confusion Matrix [23]

Accuracy, Precision, Recall & F1-Score:

From the confusion matrix, many performance measures can be computed. Some of the most commonly used are the following.

Accuracy: Ratio of correct predictions to total predictions.

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

Precision: Proportion of true positives among positive predictions

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

Recall/True Positive Rate (TPR): Proportion of true positives among the actual positive cases

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

False Positive Rate (FPR): Proportion of false positives among the actual negative cases

$$FPR = \frac{FP}{FP + TN}$$

F1-Score: The harmonic mean of precision and recall.

$$F1 - Score = 2 \times \frac{Precisión \times Recall}{Precisión + Recall}$$

Note that the value of all these measures raises from 0 to 1

ROC Curve & AUC

ROC Curve: Represents the relationship between the true positive rate and the false positive rate for different classification thresholds.

AUC (Area Under the Curve): A measure of a model's ability to distinguish between classes. A higher value indicates better performance.

Figure X shows how the ROC Curve and the AUC looks graphically. The ROC Curve is represented by a red line and the AUC by the grey area in the figure below.

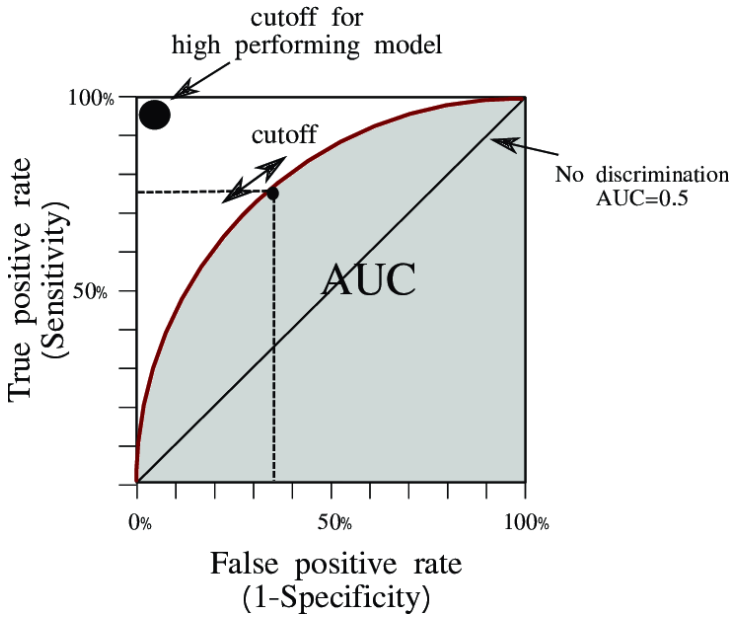

Figure X. ROC Curve & AUC [24]

2.3 Unsupervised Learning

From here onwards, we will examine unsupervised learning and some methods that constitute this type of Machine Learning.

2.3.1 Anomaly Detection

Anomaly detection is used to identify patterns in data that are inconsistent with expected or normal behaviour. These anomalies, also known as outliers, can be indicative of rare and potentially critical events, such as financial fraud, industrial system failures, health problems and cyber-attacks. Two of the most important anomaly detection models are described below.

Isolation Forest

Isolation Forest is an unsupervised learning method for anomaly detection based on the Random Forest model seen in Section 2.2.2 [25]. Instead of building a profile of normal behaviour, like many other methods, Isolation Forest explicitly isolates anomalies. The underlying principle is that anomalies are few and far between, which makes them easier to isolate. The 2 phases of Isolation Forest's functioning are:

- Forest Construction: Several Isolation Trees are constructed. Each tree is constructed by randomly selecting a feature of the data and then randomly selecting a split value between the maximum and minimum value of that feature. This process is repeated until each data point is isolated at a leaf node or until a predetermined maximum depth is reached.

- Anomaly Score Calculation: The depth of the leaf in which a data point is located is a measure of how many splits it took to isolate it. Anomalies tend to isolate quickly and therefore have shallower depths. The anomaly score is calculated as a function of the average depth of isolation across all trees.

Figure XI shows an example of Isolation Trees and how they classify their leaves as anomalies, potential anomalies or normal instances

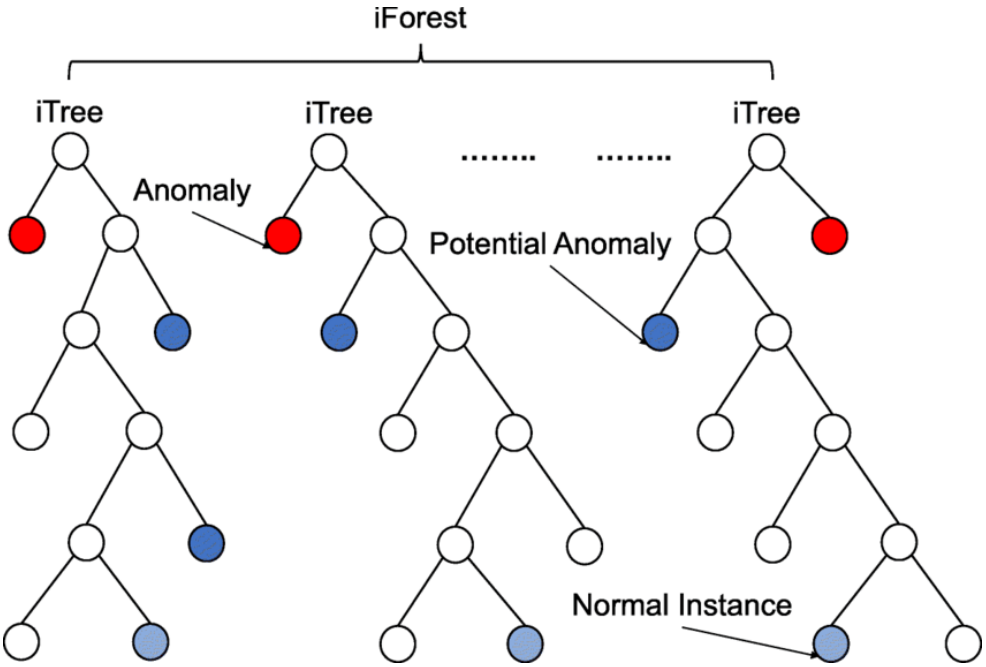

Figure XI. Isolation Trees [26]

One-Class SVM

One-Class SVMs (Support Vector Machines) are a variant of SVMs, reviewed later in Section 2.4.2, designed specifically for anomaly detection. This model tries to learn a function that captures the region where the data density is highest, so that data outside of this region are considered anomalies. The 3 phases of One-Class SVM functioning are:

- Training: The One-Class SVM learns a hyperplane in a higher dimensional than the original feature space that separates most of the data from the source. The goal is to keep most of the training data within a dense region around the hyperplane.

- Prediction: New instances are classified as normal if they are in the same dense region as the training data and as anomalous if they are outside this region.

- Anomaly Score: The One-Class SVM assigns a score to each data point based on its distance from the decision hyperplane. Negative scores indicate points within the normal region, while positive scores indicate anomalies.

In Figure XII an example of a hyperplane calculated by a One-Class SVM is shown, where we can also see how it classifies data depending on their position in that hyperplane.

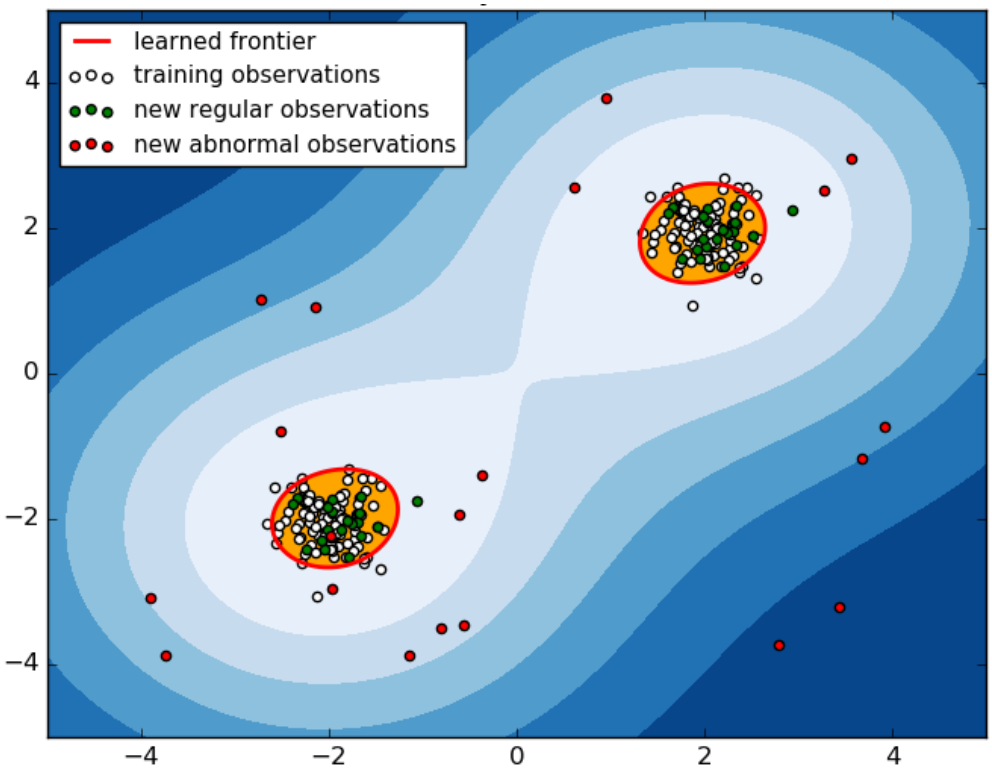

Figure XII. One-Class SVM for anomaly detection using a non-linear separation function [27]

2.3.2 Clustering

Clustering is a technique that consists of dividing a set of data into groups (clusters) so that objects within each group are more similar to each other than to objects in other groups. This similarity can be defined in different ways, depending on the specific problem and the clustering algorithm used.

K-Means

One of the best known and most widely used clustering algorithms is the K-Means algorithm [28], which is simple to implement and can be highly effective in many applications. It is an iterative method that groups a set of data into a predefined number of K clusters (groups). Each cluster is represented by its centroid, which is the average of the data points in that cluster. The objective of the algorithm is to minimise the sum of the squared distances between each data point and the centroid of the cluster to which it belongs. The K-means algorithm follows the next four phases in order to divide the data into clusters:

- Initialisation: Select K initial points as centroids. These points can be selected randomly or can be selected with a method such as K-Means++, that improves the quality of the initial clusters. [29]

- Cluster Assignment: Assigns each data point to the nearest centroid. This assignment is done by calculating the distance (usually the Euclidean distance) between each data point and each of the centroids, and assigning the point to the cluster of the nearest centroid.

- Centroid Update: Calculates the new centroid for each cluster as the average of all data points assigned to that cluster. The centroid is recalculated to better reflect the centre of the cluster points.

- Repeat: Repeats the assignment and update steps until the centroids do not change significantly between consecutive iterations, or until a maximum number of iterations is reached.

Figure XIII shows an example where we can see all this K-Means four phases explained graphically.

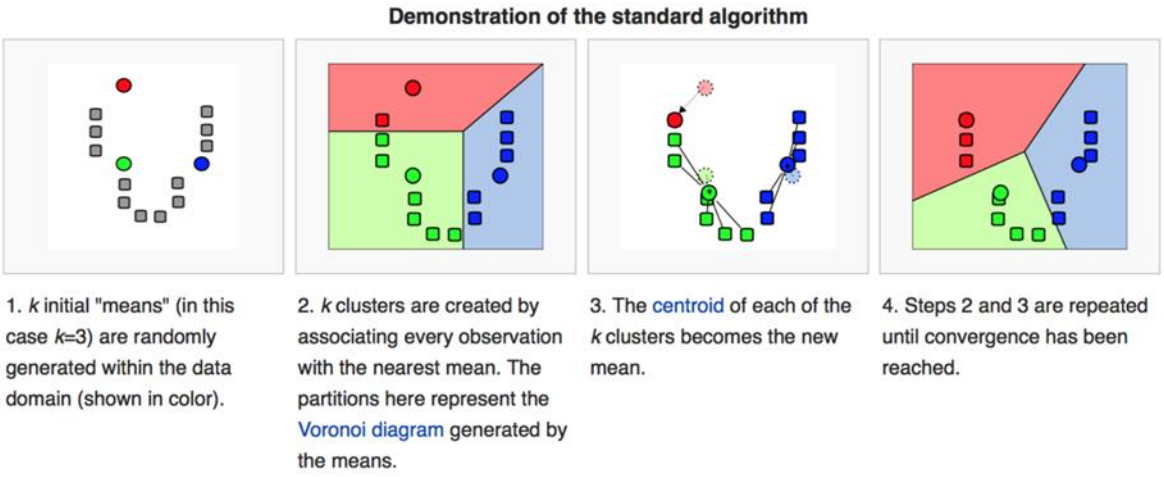

Figure XIII. K-Means clustering phases [30]

2.3.3 Dimensionality Reduction

Dimensionality reduction is a technique that is used to reduce the number of input random variables under consideration (features) by extracting the most important ones. The main goal of dimensionality reduction is to simplify data, to achieve model performance improvement and to reduce computational cost, all while preserving as much information as possible.

Principal Component Analysis (PCA) is a statistical technique that transforms the original features into a new set of uncorrelated variables called principal components [31]. The first principal components capture most of the variability in the data.

PCA finds the directions (components) that maximise the variance of the data. It then projects the original data into this new component space. The number of principal components selected determines the degree of dimensionality reduction. Figure XIV shows an example of the three principal components found by PCA

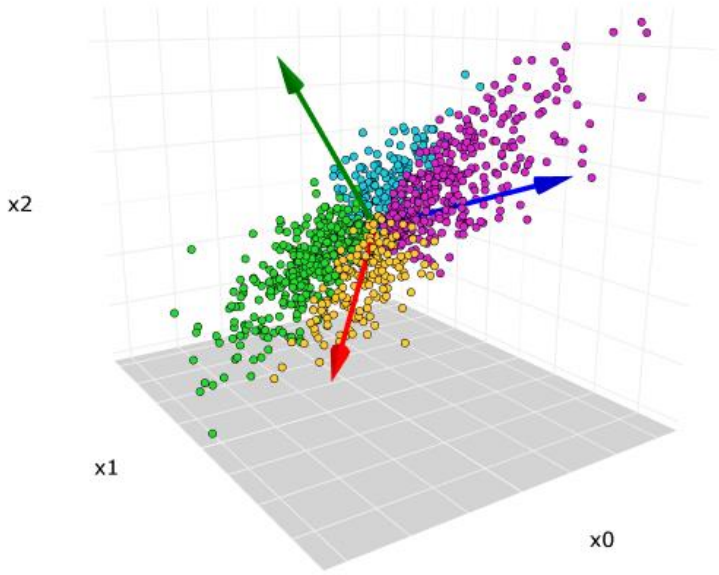

Figure XIV. PCA example: Data points and principal components [32]

2.4 Shallow classification methods

Due to the importance of supervised classification models in the project, some of the most important models in the literature are developed below.

2.4.1 Logistic Regression

Unlike linear regression, which predicts a continuous value, logistic regression predicts the probability that an observation belongs to one of two possible classes [33]. This is based on the logistic regression equation, which is based on the sigmoid function, that maps any real value to a value between 0 and 1:

$$P(Y = 1 \mid x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p)}}$$

Where $P(Y = 1 \mid x)$ represents the probability that the dependent variable y is 1, given the input x, and we can see that the model is based on an MRLM that appears in the exponential. This model is fitted by maximising the likelihood, rather than minimising the sum of squares of the errors.

Figure XV shows a logistic regression example, where we can see the estimated probability of passing an exam according to the hours studied by the students and the final result of the students represented as black dots.

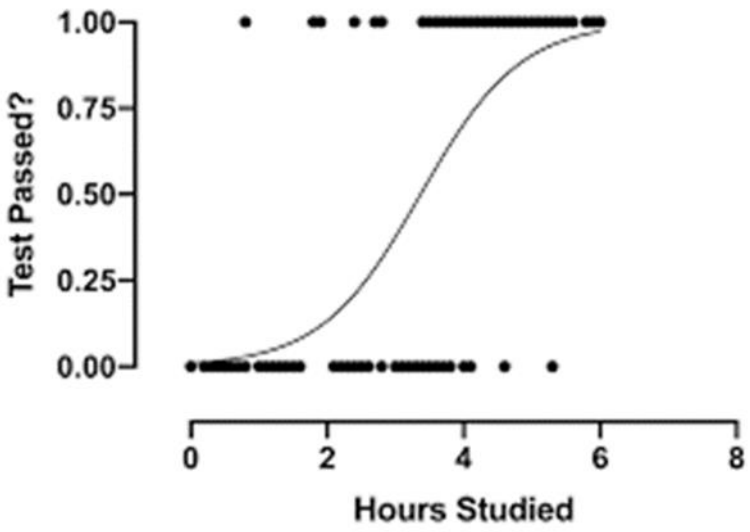

Figure XV. Logistic Regression Example: Test Success vs. Hours Studied [34]

2.4.2 Support Vector Machines (SVM)

The main goal of this algorithm is to find the hyperplane that best separates two classes in a high-dimensional feature space that maximises the margin between the different classes [35]. This margin is the closest distance from the hyperplane to any point in the two classes.

In linearly separable problems, there are multiple hyperplanes that can separate the classes. An SVM searches for the optimal hyperplane that maximises the margin, i.e. the minimum distance between the hyperplane and the closest data points of any class. These closest points are called support vectors.

In situations where classes are not linearly separable, a kernel function can be used to transform the original data into a higher dimensional space, where classes are more likely to be linearly separable.

In Figure XVI an example of an SVM classification problem is shown, where the mentioned two classes division by the hyperplane marked in red can be seen.

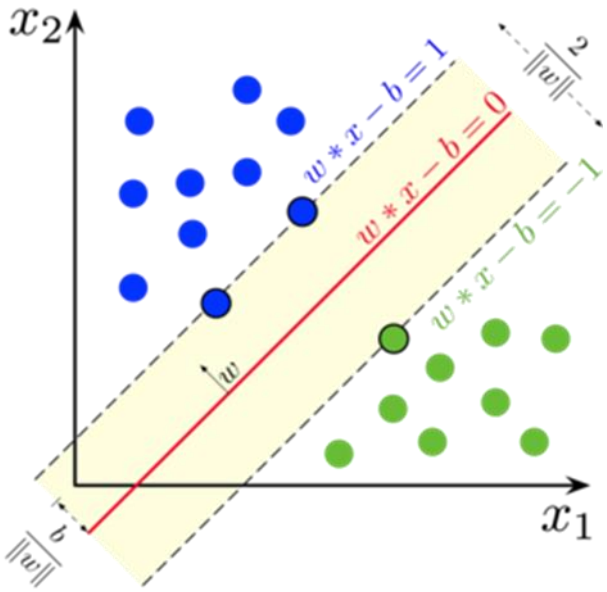

Figure XVI. Support Vector Machine for a linear binary separable problem [36]

2.4.3 Random Forest

Random Forest is an ensemble method that is constructed from multiple decision trees, and its effectiveness lies in combining multiple individual predictions to improve accuracy and reduce the risk of overfitting [37].

In Figure XVII an example of a Random Forest is shown, where all the branches and leaves that make it up are displayed.

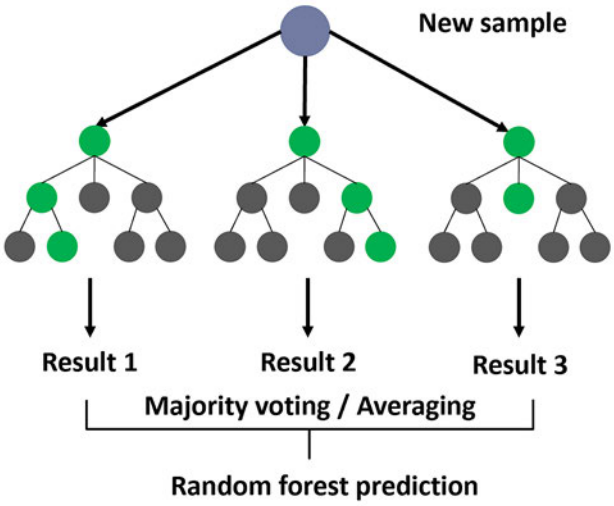

Figure XVII. Random Forest [38]

Model Construction

Random Forest model development has the following four steps:

- 1. Decision Trees: A decision tree is a predictive model that uses a set of decision rules derived from the input data to make predictions. Each node in the tree represents a feature of the data, each branch a decision rule and each leaf an output or prediction.

- 2. Tree Ensemble: In Random Forest, an ensemble of decision trees is constructed. Each tree is trained with a different sample from the original dataset, obtained by a process that involves randomly selecting with replacement a part of the dataset to train each tree.

- 3. Random Feature Selection: To add more diversity and reduce the correlation between trees, each split in the tree nodes considers only a random subset of features instead of all available features. This not only improves the robustness of the model, but also helps to handle datasets with many features.

- 4. Aggregation: For classification problems, each individual tree in the forest provides a prediction. The final prediction of the Random Forest model is obtained by majority voting among all individual tree predictions.

2.5 Deep Learning for classification

2.5.1 Neural Networks

Neural networks are a fundamental class of ML algorithms inspired by the structure and functioning of the human brain [39]. They are used to model and solve complex problems that are difficult to tackle with traditional techniques, such as pattern recognition or the classification algorithms described in Section 2.4.

Architecture of Neural Networks

A neural network is composed of several concatenated layers. Three types of layers can be distinguished:

- Input Layer: This layer receives the features of the input data. The number of neurons in this layer corresponds to the number of features in the data.

- Hidden Layers: These layers process the inputs using weighted combinations and non-linear activation functions. Deep neural networks have multiple hidden layers, allowing them to capture more abstract and complex features of the data.

- Output Layer: Produces the final prediction. The number of neurons in this layer depends on the type of problem: a single neuron for regression problems, several neurons for multi-class classification problems.

Figure XVIII shows how all these types of layers which a NN could have through graphical example.

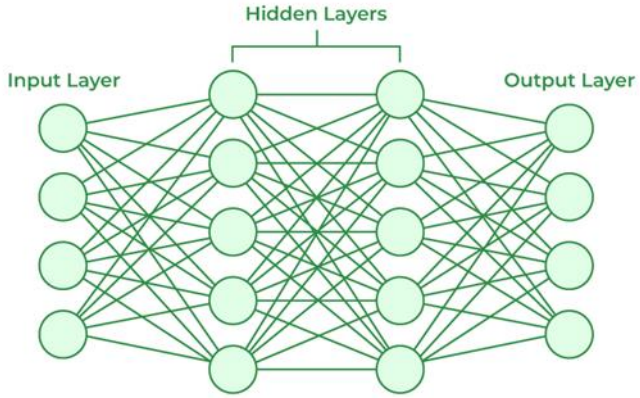

Figure XVIII. Neural Network architecture [40]

Artificial Neuron

A neural network is made up of basic units called neurons or nodes. Each neuron in one layer is connected to neurons in the next layer by weighted connections. These connections are adjusted during the training process to minimise the error in the model predictions.

As shown in Figure XIX, each neuron receives a set of inputs, which are the output values of the neurons in the previous layer, and combines them using a weighted sum. An activation function is applied to this sum to introduce non-linearities into the model, allowing the network to learn complex representations of the data. Common activation functions include the sigmoid, the hyperbolic tangent (tanh) and the Rectified Linear Unit (ReLU) function [41].

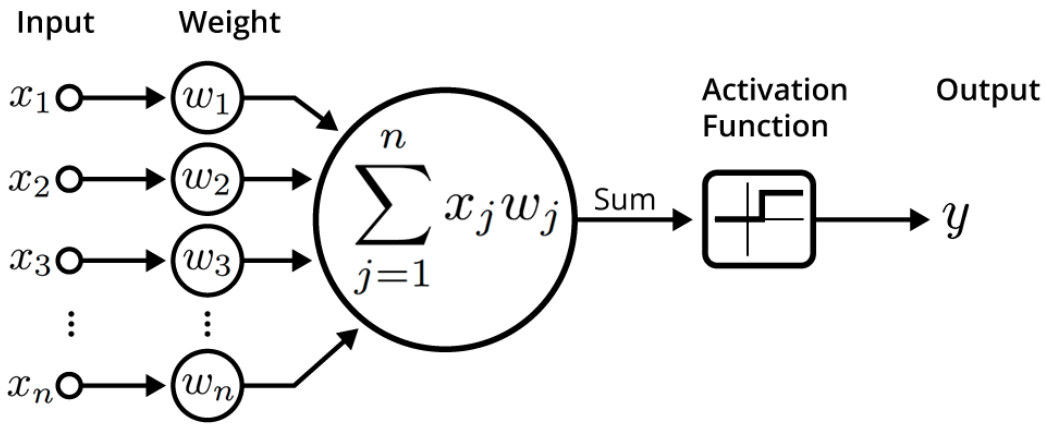

Figure XIX. Artificial neuron output calculation [42]

2.5.2 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are a class of deep neural networks specifically designed to process and analyse data with a grid structure, such as images [43]. Their architecture uses convolutional layers that allow them to extract local and hierarchical features, which makes them extremely effective for computer vision tasks, among others.

A CNN is composed of several layers of different types, each of which has a specific function in processing the input data. The main layers in a CNN are convolutional layers, pooling layers and fully connected layers.

1. Convolutional Layers:

The convolutional layer is the cornerstone of a CNN. It applies a set of filters (kernels) on the input to produce feature maps. Each filter acts as a local feature detector, capable of capturing patterns such as edges, textures and shapes.

- Filters: These are arrays of weights that are trained during learning. Each filter moves (or convolves) along the input, making dot products between the filter and sections of the input. This process highlights local image features, such as edges and textures.

- Stride and Padding: Stride determines the pitch of the filter during convolution. Padding is the process of adding zeros around the edges of the input to control the size of the feature map (e.g. to maintain the dimensionality of the input).

2. Pooling layers:

The pooling layer reduces the dimensionality of feature maps through sampling operations, such as max pooling or average pooling. This reduces the number of parameters and computational cost, and helps to control overfitting.

- Max Pooling: Selects the maximum value in a defined region of the feature map.

- Average Pooling: Calculates the average value in a defined region.

3. Activation functions:

ReLU (Rectified Linear Unit): In CNNs, it is typical to use de ReLU activation function, which introduces non-linearity to the model, thus allowing more complex functions to be learned. The widespread use of ReLU in CNNs is due to the fact that it allows for a simple gradient calculation and because it has no saturation.

The ReLU function is $f(x) = \max(0, x)$.

4. Fully Connected Layers:

These layers are at the end of the network and act like a traditional neural network, where each neuron is connected to all neurons in the previous layer. Their function is to combine the extracted features to make the final prediction.

Figure XX shows an example of a number recognition CNN where all the types of layers mentioned before can be seen. Firstly, this architecture has a combination of a 5x5 convolutional layer and a 2x2 max-pooling which is repeated two times, then it has a fully connected layer with a ReLU activation and finally another fully connected layer to obtain the final predictions of the CNN.

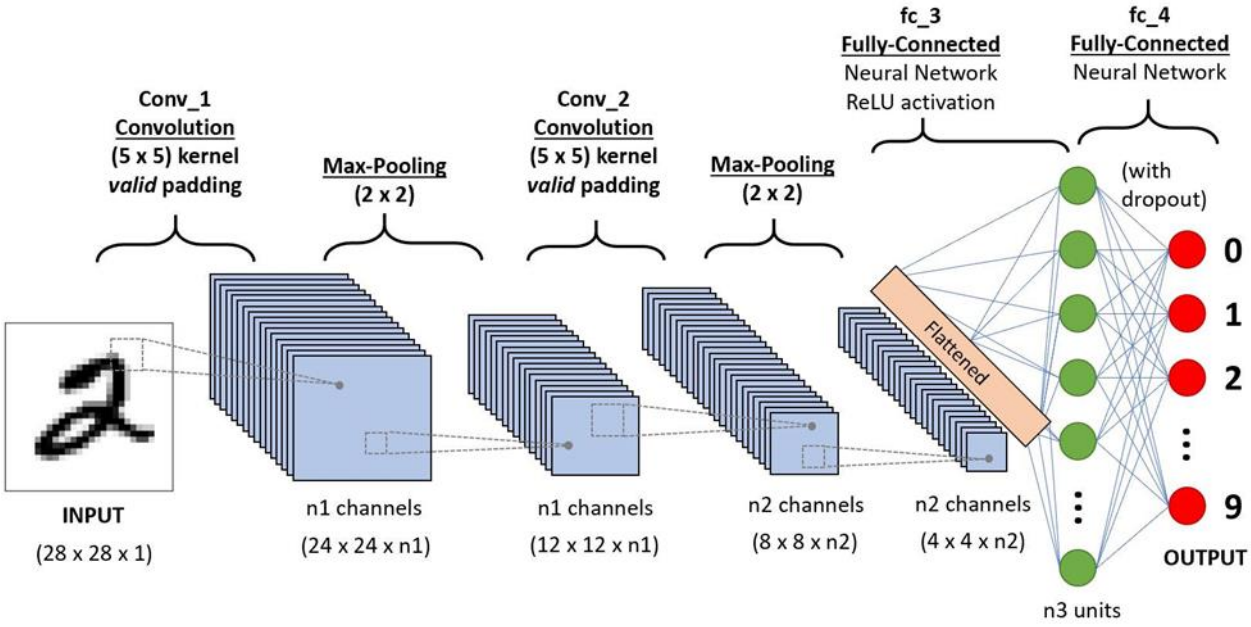

Figure XX. Convolutional Neural Network architecture [44]

Learning Mechanism

Depending on the direction of the learning mechanisms of a CNN, we can have the following two types of propagations:

- Forward Propagation: The input data passes through all layers of the network, and each layer applies its operations (convolution, pooling, activation) to transform the data to obtain a final output.

- Backward Propagation: Uses the downward gradient and backpropagation algorithm to adjust the filter weights and connections. The loss function calculates the error between the prediction and the actual label, and the gradients are used to update the weights in the direction that minimises the error.

Figure XXI illustrates this forward-backward propagation mechanism.

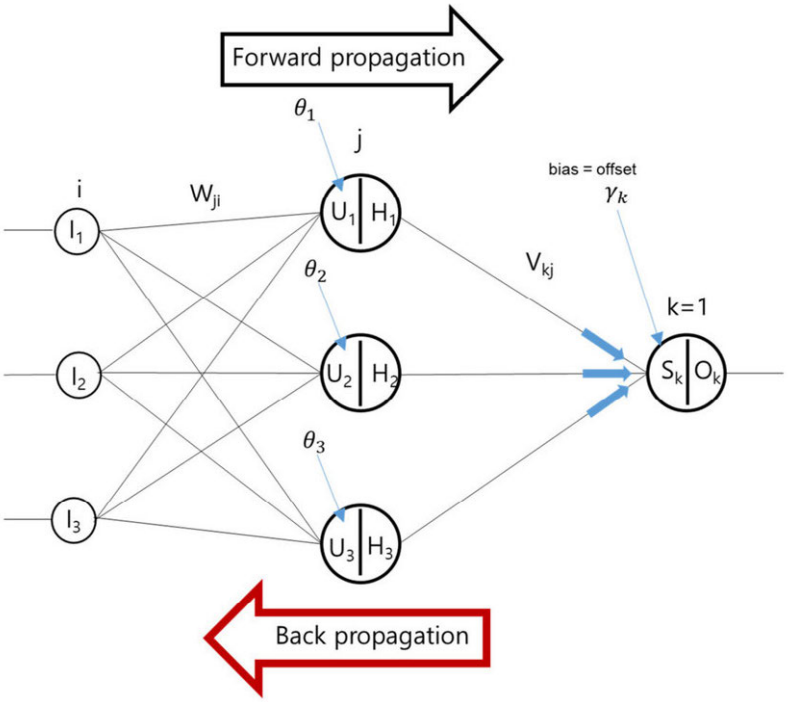

Figure XXI. Forward and back propagation in a neural network [45]

2.5.3 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) are a class of neural networks designed to handle sequential data, such as time series, text and signals [46]. Unlike traditional neural networks, which assume that inputs are independent of each other, RNNs have recurrent connections that allow information to persist throughout the sequence. This feature makes them particularly suitable for tasks where the context and the order of inputs are important.

In an RNN, each neuron receives not only the input from the previous layer, but also its own output from the previous time step. This is achieved by introducing a recurrent connection that feeds the neuron's activation in the previous time step as input to the current time step. The equation describing the state of a neuron in an RNN is as follows:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$

where:

h_t is the hidden state at time t

x_t is the input at time t

W_{hh} is the recurrent weight matrix

W_{xh} is the input weight matrix

b_h is the bias

σ is the activation function

In Figure XXII we can see the connections of an RNN, highlighting their typical connection mentioned before, which remembers the output of the previous time step.

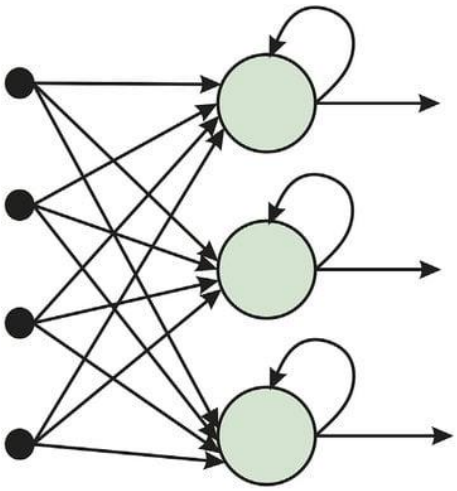

Figure XXII. RNN recurrent connection [47]

2.6 Chapter Discussion

This chapter offers a brief review of Machine Learning (ML) and its various types and methodologies. The discussion begins by defining ML as a subset of Artificial Intelligence (AI) that enables systems to learn from data, identify patterns, and make decisions with minimal human intervention. It also outlines the different types of ML, emphasizing their unique capabilities and relevance to real-world problems.

In the ML field, models can be classified into three primary types: **Supervised Learning**, which relies on labeled data to train algorithms for specific tasks; **Unsupervised Learning**, which discovers hidden patterns in unlabeled data; and **Reinforcement Learning**, where algorithms learn optimal behaviors through trial and error in dynamic environments.

Classification is highlighted as a crucial task within the realm of supervised learning. This chapter discusses various classification techniques used to categorize data into predefined classes. By training algorithms on labeled datasets, the system can accurately classify new, unseen data, demonstrating the practical utility of ML in automating decision-making processes.

A key focus is also placed on anomaly detection, a critical aspect of unsupervised learning. This technique is vital for identifying rare or unusual patterns in data. The chapter explains how different anomaly detection models such as One Class SVM or Isolation Forest are implemented, and highlights their importance in multiple types of tasks.

Furthermore, the chapter delves into Convolutional Neural Networks (CNNs), a specialized form of neural networks designed for processing structured grid-like data, such as images. The architecture of CNNs, including convolutional layers, pooling layers, and fully connected layers, is discussed in detail, illustrating how these components work together to extract features from input data and improve classification accuracy.

In conclusion, this chapter underscores the versatility and power of ML techniques, particularly in classification, anomaly detection tasks and neural networks such as CNN. The overall conclusion of the chapter highlights the continuous evolution of AI, with various types of methods and applications, and its capacity for solving complex problems across various domains.

3. Explainable Artificial Intelligence

The search for constant improvement of the different processes that encompass humans is something that human beings pursue and will pursue until the end of their existence, as mentioned in Section 1. The search of why things happen is not left behind and likewise reaches the field of machine learning. Although most ML models may work with great effectiveness, they are commonly categorized as black-boxes. That is, they provide an output without giving back an explanation of how they arrived at that final conclusion. However, in many cases it is necessary to know more about the process that the model performs in order to draw its conclusions, and this is where explainability comes in.

In the context of AI, explainability, known as XAI (eXplainable Artificial Intelligence), refers to the ability of artificial intelligence systems to provide understandable interpretations or explanations of their decisions and predictions. It is not just about showing the result, but about breaking down and clarifying the reasons behind each prediction. This is especially important in deep learning models and other advanced techniques, which, while powerful in terms of accuracy and predictive ability, are often inherently opaque and difficult for humans to understand.

The importance of explainability is multifaceted. First, there is the need to build **trust** in AI models. When systems are used in critical areas such as medicine, justice or finance, trust in automated decisions is critical. End users, such as doctors, judges or financial analysts, need to understand and trust the recommendations provided by these models. Explainability allows these professionals to validate the system's decisions, which is crucial for their acceptance and continued use.

In addition, explainability is vital for **fairness**. AI models, when trained on large amounts of data, can pick up and amplify biases present in the training data, which could result in unfair or discriminatory decisions. The ability to explain how a model reached a conclusion allows these biases to be identified and the model to be adjusted to make fairer and more equitable decisions.

Another key issue is regulatory compliance. In many jurisdictions, laws and regulations are beginning to require AI systems to be transparent and able to justify their decisions. This is especially relevant in contexts such as the European Union, where regulations such as the GDPR require that individuals have the right to receive explanations of automated decisions that affect them [48].

In addition, explainability has an **educational and learning value**. It allows developers and data scientists to better understand how their models are working, identify areas for improvement and develop more accurate and efficient versions. It also facilitates knowledge transfer between different disciplines, allowing experts in areas such as biology, medicine or economics to use machine learning models without needing to be experts in the underlying technique.

Explainability is not just a desirable addition to AI systems; it is an imperative to ensure that these systems operate in an ethical, fair and compliant manner. By making models transparent and understandable, XAI helps build AI systems that are not only powerful, but also accountable and reliable.

3.1 Types of XAI

A series of classifications XAI methods according to [2]:

- Model agnostic / Model specific: If the explanations can only be obtained for a specific type of model, it is a model specific XAI method. If, on the other hand, the explanations can be obtained for any trained model, it is a model agnostic XAI method.

- Global / Local: If the method provides explanations for individual predictions, it is a local XAI method. On the other hand, if it tries to explain the model as a whole, not for specific cases, it is a global XAI method. Such methods that attempt to explain the entire model are very difficult to achieve in practice [2].

- Ad-hoc or intrinsic / post-hoc: If the model is explainable due to its structure, it is an ad-hoc or intrinsic interpretable model, such as decision trees or linear regression. On the other hand, if the method is applied to a trained black-box model, it is a post-hoc XAI method.

3.2 Common approaches for XAI

The following section describe the two methods of explanatory artificial intelligence which form the basis of the field. Both of these techniques are classified as agnostic, post-hoc local explanation methods. That is, they work for any given trained model and provide explanations for every individual prediction. These techniques are SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations).

3.2.1 Local Interpretable Model-agnostic Explanations

Local Interpretable Model-agnostic Explanations (LIME) is an explainability technique that has gained popularity due to its ability to provide understandable explanations of the predictions by locally approximating the black box model using simple and interpretable surrogate models [49].

The first step in LIME is to generate a set of perturbed data from the input instance of interest. These perturbations are created by slightly modifying the original features. The idea is to explore how small variations in the features affect the model prediction.

Once the perturbations are generated, this new data is fed back to the original model to obtain its predictions. The resulting data set, consisting of the perturbations and their corresponding predictions, forms the basis for fitting a local interpretable model.

The next step is to fit an intrinsically interpretable model (e.g. a linear regression or a decision tree). In most cases, LIME uses a weighted linear regression for this purpose.

The disturbances are weighted according to their proximity to the original instance by using a proximity weight function. This function assigns a higher weight to disturbances that are closer to the original instance, allowing the local approximation previously mentioned.

Once the interpretable model is fitted, explanations are derived directly from the model coefficients. Since they represent the importance of each feature in the model's prediction. A high coefficient indicates that the corresponding feature has a large influence on the prediction.

In Figure XXIII we can see the explanation given by LIME to a specific prediction of a diabetes classifier. First, we see the result of the prediction, with 72% chance the patient is diabetic and with a 28% chance is not. In the middle plot, we see features values that are more important to decide for each of the classes, for example being between 30 and 40 years old is something relevant to the “no diabetes” class. Finally, we see the instances features.

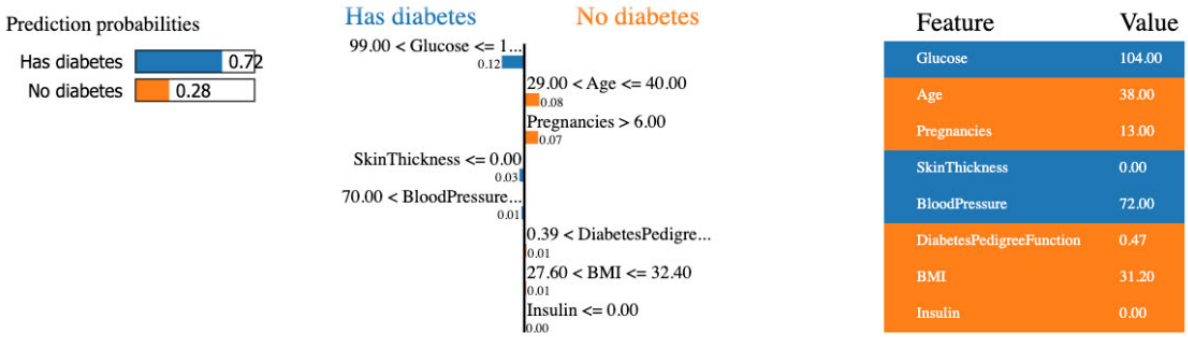

Figure XXIII. LIME explanation for diabetes detection example [50]

3.2.2 SHapley Additive exPlanations

SHapley Additive exPlanations (SHAP) is a technique based on cooperative game theory [51]. The objective of SHAP is to estimate the Shapley value, a value that distributes payoffs fairly among all players in a cooperative game. SHAP provides a consistent and unified way of explaining predictions, and can be applied to any machine learning model, offering both local and global explanations.

In the context of machine learning, the ‘players’ are the features of the model, and the ‘payoff’ is the difference between the prediction and the average prediction in a reference set. In other

words, SHAP fairly distribute the prediction deviation to the average output between all the features. If a feature contributed more to such deviation, then SHAP will assign more importance to that feature.

For a specific instance, Shapley values are estimated for each feature, providing a clear measure of the contribution of every feature to the prediction. The model prediction is decomposed into a sum of individual feature contributions. In Figure XXIV we see an example of this type of application. We can see how marital status type 4 is the feature that contributes the most to the final result. The sum of all feature contributions explains the final result which is 0.99 in relation to the mean of all predictions which is 0.79.

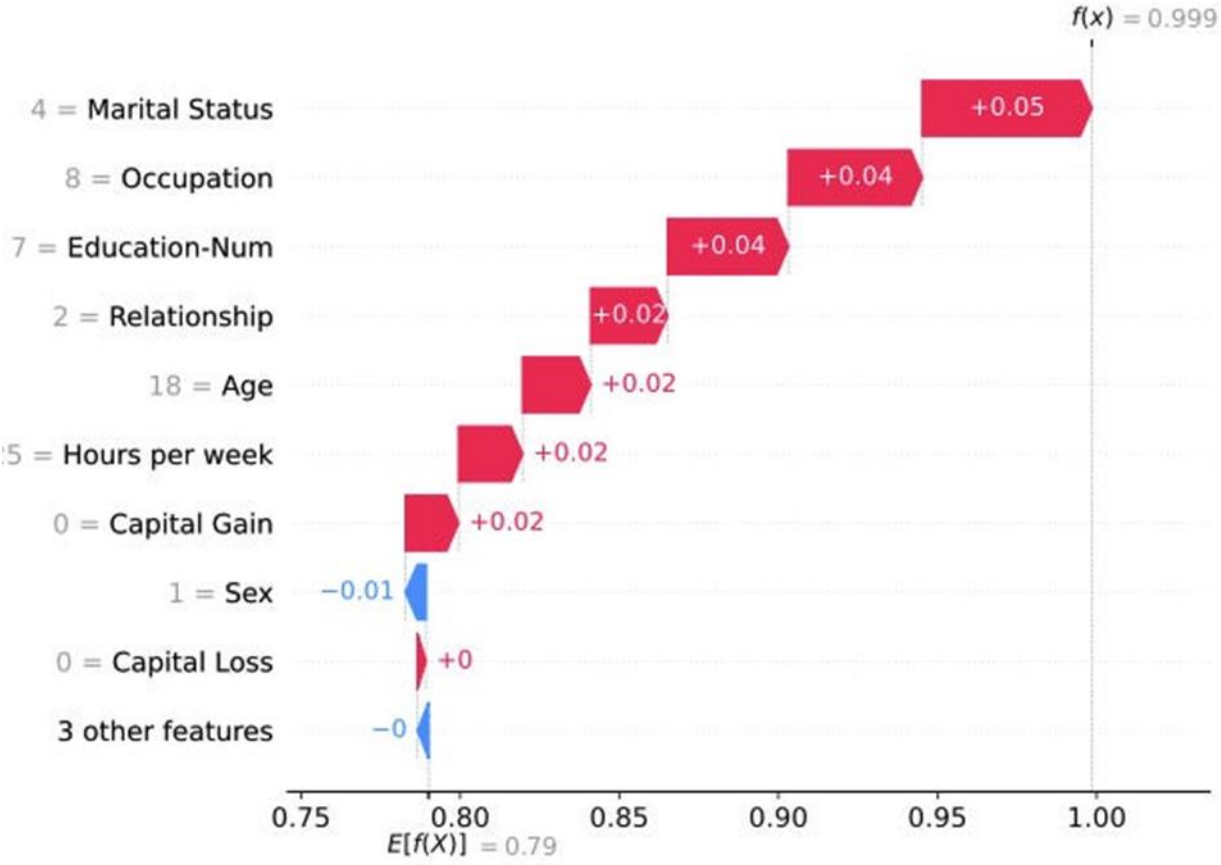

Figure XXIV. SHAP specific instance example [52]

To get an overview of the importance of features from a global perspective, the absolute values of the Shapley values can be averaged across all instances of the dataset. This allows to identify which features have a greater impact on the model output globally.

In addition, it is common to visualize the overall contribution of features by means of a graph showing the individual contribution of each data item for each feature, thus providing an overall summary of the various contributions observed within a dataset.

These two examples can be seen in Figure XXV below: on the left, the mean of the absolute values of the Shapley values is presented, while on the right, a diagram showing all Shapley values is displayed.

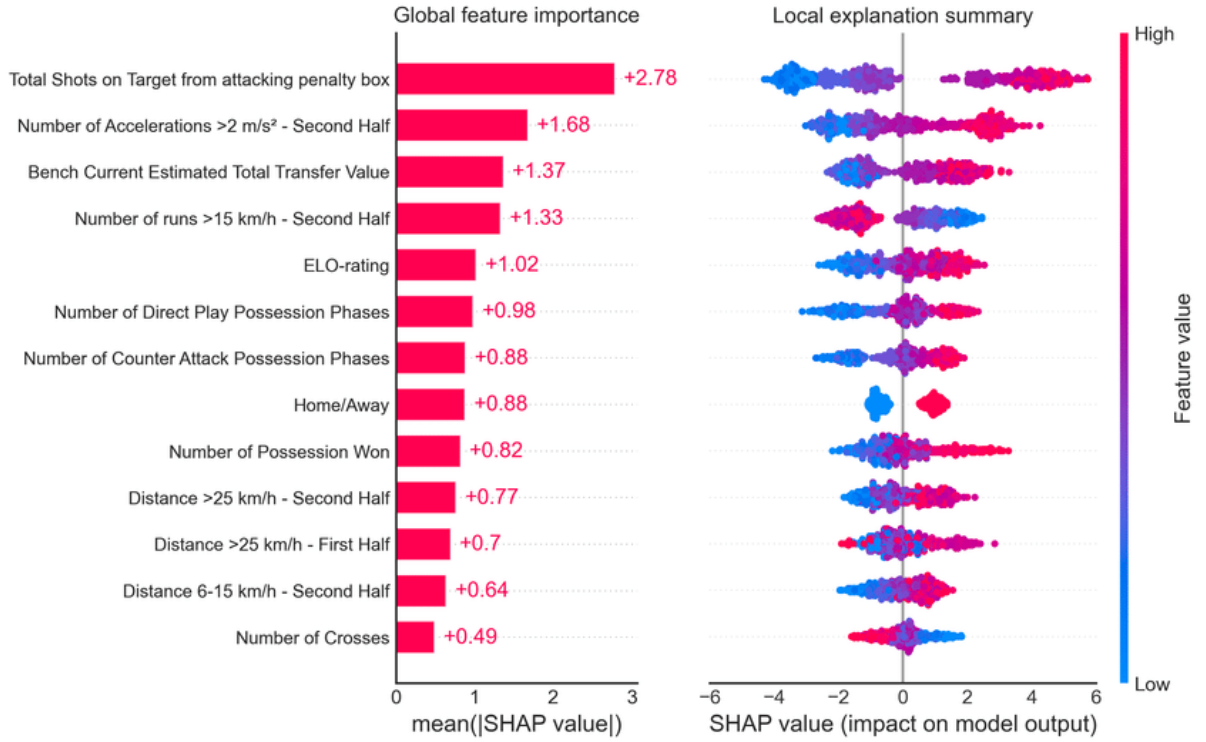

Figure XXV. SHAP characteristics importance example [53]

3.3 Counterfactual Explanations

Within the multiple options that exist in the area of XAI, counterfactual explanations are a great tool that can help people to understand different black box models. These methods of explainability are born from the exploration of different possible realities that could change the final outcome of the original situation [2].

Let's take an example to clarify this new concept: 'If I had taken the car, I would not have been late for the event'; given the situation of being late for an event because of having gone by another slower means of transport, changes are sought in the causes that have provided this, looking for a different result, which would be not to be late.

In its application in the area of machine learning, counterfactual explanations are based on the search for changes in the minimum possible input data, so that the classification that has been provided changes [2]. Let us consider an applied example:

We own a vehicle that we would like to sell for more than 10,000 €, it has the characteristics displayed in Table II. In this table we see in white those characteristics of the car that can be

modified and in grey we see those attributes that cannot be modified and will not be taken into account. The table also shows the prediction of a black-box model, confirming that the car could not be worth more than 10.000€

Horsepower	Color	Length	Infotainment?	Hubcaps?	Financing available?	Price > 10k €?
135 cv	White	4.397 mm	No	Yes	No	No

Table II. Car example input data

This is where the counterfactual explanations come in. Since our car cannot be sold for more than 10k €, we ask ourselves: What are the minimum modifications that our car has to undergo before we can sell it for more than 10.000 €? The counterfactual explanation method answers that by painting the car black and allowing financing we will be able to sell our vehicle for more than 10.000 € (Table III).

Horsepower	Color	Length	Infotainment?	Hubcaps?	Financing available?	Price > 10k €?
135 cv	Black	4.397 mm	No	Yes	Yes	Yes

Table III. Car example counterfactual

In the specific case of the project, we are going to apply counterfactual methods to time series classification models. We provide a graphical example of a counterfactual explanation and a possible interpretation of it.

Imagine that the following time series, shown in Figure XXVI, is an electrocardiogram of a patient that goes to the hospital. This hospital is a very advanced one that uses machine learning classification methods for diagnosis, to evaluate if the heart of the patients is healthy or not, and in this case, the ML model says that it is an unhealthy heart.

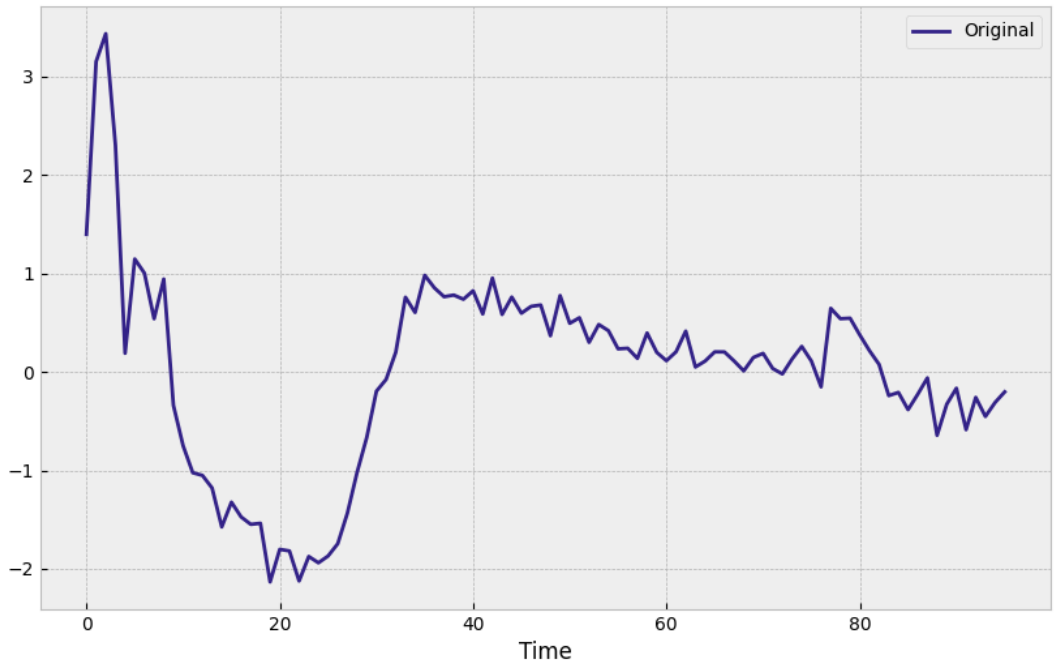

Figure XXVI. ECG200 dataset 10th time series

As described in the first page of this chapter, the need for explainability in methods of this style and in such critical situations makes the chief cardiologist of the hospital request the use of explainability methods and this is where counterfactual explanations come in with the following result (Figure XXVII)

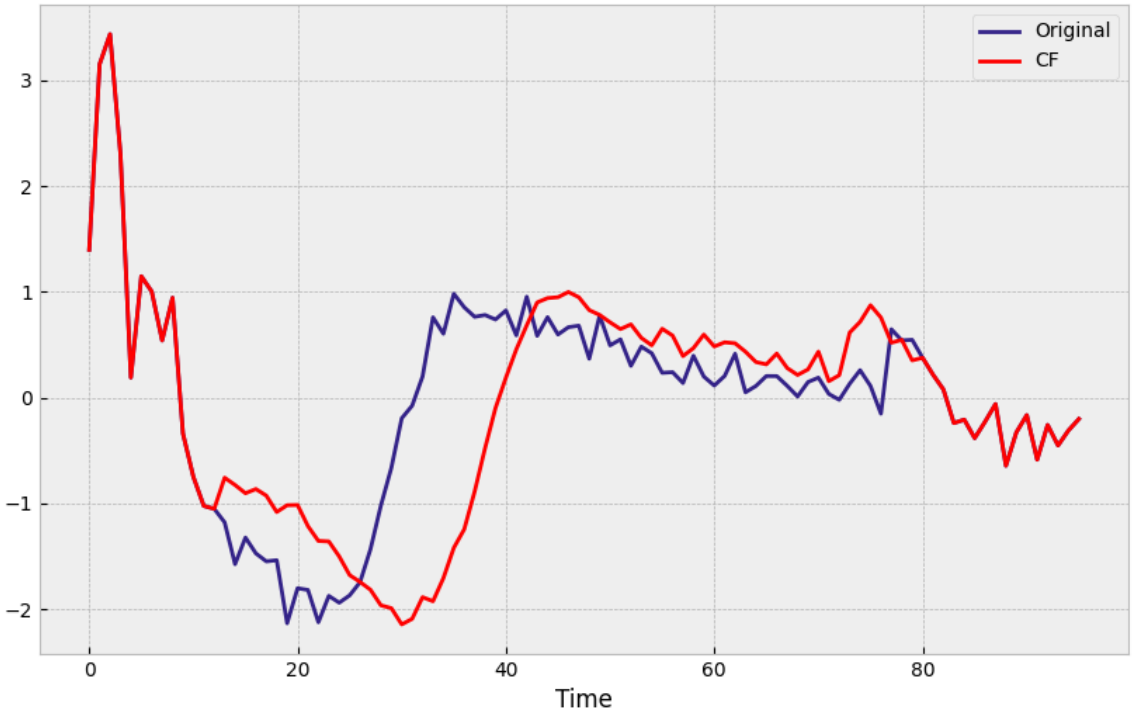

Figure XXVII. Counterfactual explanation for ECG200 dataset 10th time series generated with MG-CF

The cardiologist can determine, through the counterfactual explanation, that the classifier has taken more into account certain parts of the time series to make the decision, in this case it could realise that if the time series were shifted by 10 seconds it would have been classified in the opposite class. The expert would understand the importance of these shift because they would be the minimum change to be made in the input data to change the label of the prediction, indicating a high importance in those parts of the time series.

But the generation of these counterfactual explanations applied to time series, being completely different from all other methods of explainability, has a series of requirements that must be met, which are the following [54] :

- Closeness or proximity: the counterfactual explanation generated must be as similar as possible to the input. To measure this characteristic, the L1 norm (also known as Manhattan distance) and the L2 norm (or Euclidean distance) are common metrics in counterfactual explanations. Here are their formulas:

$$L1(x, x') = \sum_{i=1}^{N} |x_i - x'_i|$$

$$L2(x, x') = \sqrt{\sum_{i=1}^{N} (x_i - x'_i)^2}$$

where:

x_i is the original time series

x'_i is the generated counterfactual

N is the number of points of the original time series

- Sparsity: the changes generated on the initial instance should be as few as possible, since the greater the number of characteristics changed in the explanation, the more difficult to understand for the people who are going to use it. To measure sparsity, we use the following formula:

$$sparsity = \frac{\sum_{i=0}^{N} g(x_{cf_i}, x_i)}{N}$$

$$g(x, y) = \begin{cases} 1, & if\ x \neq y \\ 0, & otherwise \end{cases}$$

where:

x_{cf_i} is the counterfactual value at time instant i

N is the number of points of the original time series

- Plausibility: the generation of counterfactual explanations must follow the distribution of the input data; it cannot be an outlier. To measure it, it is typical to use Outlier Detection techniques as seen in Section 2.3.1 (autoencoders, Isolation Forest or One Class SVM) to enforce and measure the plausibility of the counterfactual explanations

- Validity: The generated counterfactuals must, when passed through the trained classifier, be predicted as the opposite class.

- Efficiency: The generation of the counterfactual explanation must be generated as quickly as possible.

All of the above requirements are general requirements, which must happen in all methods of counterfactual explanations. The following requirement is a specific requirement of some methods that arises from the attempt to adapt counterfactuals to the particularities of the time series. The importance of taking into account the temporal correlation of the time series and the possibility that these time series have a high dimensionality has led to the literature to opt strongly for contiguity in the counterfactuals and to the development of methods that group the modifications in subsequences, such as some of the XAI methods implemented in the project that will be seen in Section 4.

- Contiguity: The perturbations produced in the counterfactual explanations must respect the notion of waveform data in the time series, not simply changing single points.

3.4 Chapter Discussion

This chapter covers Explainable Artificial Intelligence (XAI), an emerging field that seeks to make artificial intelligence models, often considered as 'black boxes', more understandable to humans. The need to explain the decisions and predictions made by these models stems from several factors: generating confidence in AI models, detecting biases, compliance with regulations such as GDPR or the development of knowledge and improvement of these models.

Different classifications that can be given to different XAI models according to their characteristics are also presented and the concept of counterfactual explanations is introduced. Counterfactuals are particularly useful, because they allow different hypothetical scenarios to be explored to understand how the models' decisions might have changed if certain inputs had been altered. For example, if a patient is not diagnosed with a disease, a counterfactual explanation could show what minimal changes in their symptoms would have led to a positive diagnosis.

Finally, the two most popular XAI approaches, LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive exPlanations), are also described. Regarding these methods, LIME generates local explanations by approximating the prediction of the black-box model by fitting a simple an intrinsically interpretable model around the specific instance to be explained. SHAP, on the other hand, uses game theory to assign importance to each feature.

4. Implemented XAI Methods

This chapter presents three methods developed to enhance the interpretability of multivariate time series classification models within the scope of explainable artificial intelligence (XAI). Each method adopts different methodologies to unravel the complexities of AI decision-making, offering insights that make these processes more transparent and understandable.

The first method, “Attention-based Counterfactual Explanation for Multivariate Time Series” (AB-CF) [6], introduces changes in form of subsequences to the input instances for the generation of counterfactuals based on which segments the classifier considers to be most important, changing the most representative ones to Nearest Unlike Neighbors segments.

The second, “Motif-Guided Time Series Counterfactual Explanations” (MG-CF) [7], employs a motif-based approach, guiding the generation of counterfactual explanations through the identification and modification of recurrent patterns, also known as “motifs” or “shapelets”, typical of each output class within the time series data.

The final method, “TimeX” [5], adds to the loss function developed by Wachter et al. [55] a new term based on the Dynamic Barycenter Averaging (DBA) which gives the counterfactual explanations a higher plausibility as well as restricting the changes to a single subsequence to ensure contiguity.

Together, these models contribute to the growing field of explainable AI by offering distinct yet complementary approaches to making AI systems more interpretable, particularly in the context of complex time series data.

4.1 Attention-based Counterfactual Explanation for Multivariate Time Series

The method of counterfactual explanations is developed for univariate time series [6]; it is based on the search for representative segments of the classes and swap them for the opposite classes segments in order to change the prediction of the classifier. Next, the operation of this method will be detailed.

The first step of AB-CF is the search for the k most important non-contiguous segments in the dataset. For this purpose, a sliding window is used to obtain the aforementioned subsequences. These subsequences will be passed to a pre-trained model. As these subsequences are smaller, they have to be filled with zeros through a padding technique, so that they have the same size as the training data and can be passed to the classifier.

As a result of passing each sub-sequence to the classifier, we obtain a vector of probabilities of that segment, which indicates probability of each of the classes of our dataset. To find out which segment provides us with most information for each class, the Shannon entropy is used.

This dictates that the closer the entropy value is to 0, the more confident the model is abouts its prediction, and the more relevant this segment is to the output class.

$$Entropy(X) = -\sum_{c=1}^{|n|} p_c \log_2(p_c)$$

where:

n is the number of classes

c is the segment class

p_c is the class probability of c

Taking this into account, the k segments with the lowest entropy are chosen and these are the ones that are replaced by their Nearest Unlike Neighbor (NUN).

The Nearest Unlike Neighbor of a given instance T_q of class C is the nearest instance T'_{NUN} to it of the opposite class, as we can see in Figure XXVIII. Therefore, such sequences are exchanged for NUNs in order to match the original label while maintaining plausibility [56].

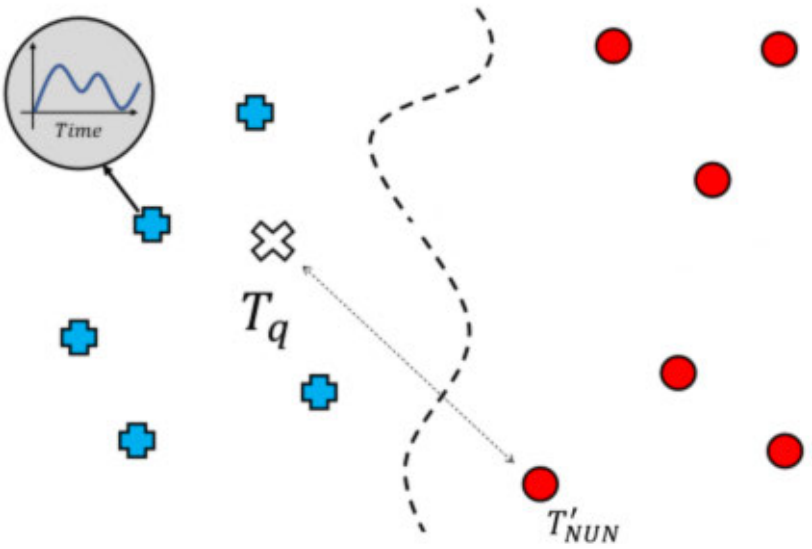

Figure XXVIII. NUN selection example adapted from [56]

Figure XXIX shows the general functioning step-by-step of the ABCF model explained above.

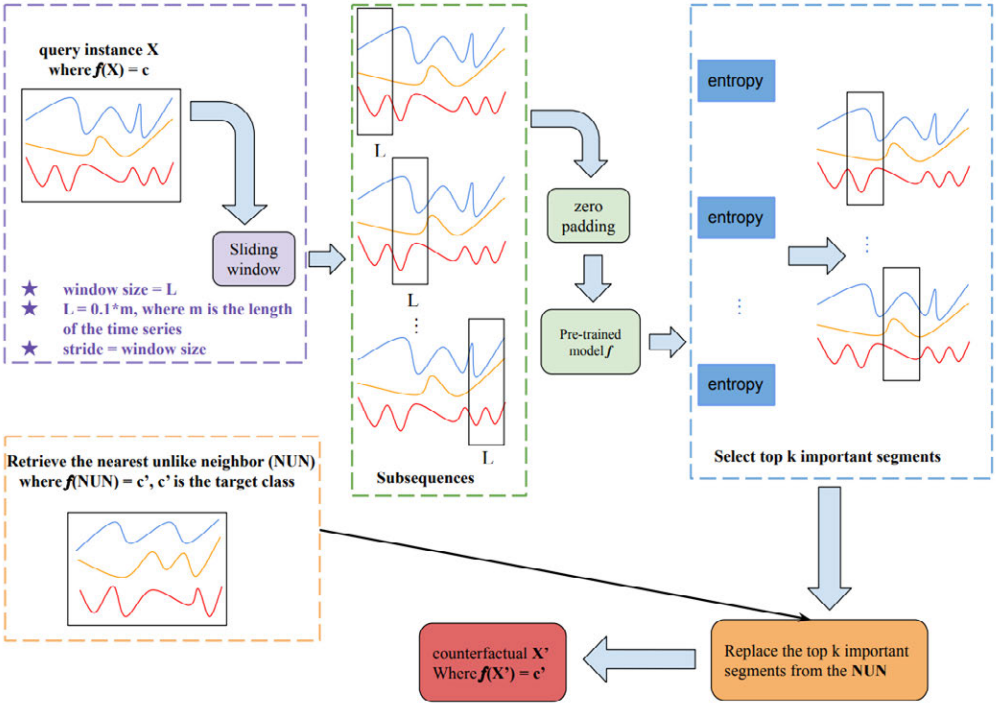

Figure XXIX. AB-CF General Overview [6]

4.2 Motif-Guided Time Series Counterfactual Explanations

The MG-CF method has two main steps [7]: motive mining and the generation of counterfactual explanations. A general outline of the MG-CF process of creating counterfactual explanations is shown below, followed by a step-by-step explanation.

Motif's Extraction

Firstly, what is exactly a motif? Motifs are sub-sequences of time series that optimally represent a specific class of a dataset [7]. Now we can see how the algorithm proceeds:

1. Candidate Generation**:** Subsequences are generated, through the Shapelet Transform algorithm, of a specific length for each time series belonging to the training dataset.

2. Distance Measurement**:** For each, its Euclidean distance to the rest of the of the time series is calculated.

3. Discriminating Power: The Shannon entropy is used to measure the representability of a motif for a class. This is measured for all segments and a list generated with the most representative motifs according to the entropy.

Shannon entropy dictates that the closer the entropy value is to 0 the more information that segment has, so it is more decisive when it comes to making a prediction.

$$Entropy(X) = -\sum_{c=1}^{|n|} p_c \log_2(p_c)$$

where:

n is the number of classes

c is the segment class

p_c is the class probability of c

In Figure XXX we can see the pseudocode of this section of Motif Guided. We can see the generation of candidates *generateCandidates*, the measurement of distances *findDistances* and the measurement of the power of the candidates *assessCandidate*.

Algorithm 1 Motifs mining

Input: Training set samples **T** with labeled binary classes C = [0, 1]
Output: Extracted motifs for each class

```
Motifs = ∅
N = length(T[0])                          ▷ Number of time series samples
m = length(T[0][0])                       ▷ Length of time series
for T_i ← T_1 to T_N do
    Motifs ← ∅
    for l in [0.3m, 0.5m, 0.7m] do
        W_{i,l} ← generateCandidates(T_i, l)
        for all subsequences S in W_{i,l} do
            D_S ← findDistances(S, W_{i,l})
            quality ← assessCandidate(S, D_S)
            Motifs.add(i,start_idx, end_idx, S, quality) ▷ The index of time series,
the start idx and end idx of motifs will be stored
        end for
    end for
    sortByQuality(Motifs)
end for
return [[i_0, start_idx_0, start_idx_0], [i_1, start_idx_1, start_idx_1]]   ▷
return the index information for motifs of different classes
```

Figure XXX. MG-CF motif mining code [7]

Generation of Counterfactual Explanations:

Once motifs of each class have been identified, the MG-CF method generates counterfactual explanations by replacing the parts of the time series corresponding to the motifs with motifs of the opposite class, while keeping the remaining parts of the time series intact. As in the previous figure, pseudocode of second part of the MG-CF model is shown in Figure XXXI.

Algorithm 2 CF generation

Input: Test set samples *samples*, pretrainined time series classifier f, extracted motifs for each class from Motif mining algorithm

Output: CF, Counterfactual explanation for each sample in test set

```
preds = f.predict(samples)        ▷ Applying the test data on f to get the prediction results
targets= the opposite label of the preds        ▷ Get the target labels based on the prediction results
for sample ← samples do
    if target(sample) == 0 then
        i, start_idx, end_idx = [i_0, start_idx_0, end_idx_0]  ▷ extracted index information of the motif from class 0
    else
        i, start_idx, end_idx = [i_1, start_idx_1, end_idx_1]  ▷ extracted index information of the motif from class 1
    end if
    sample[start_idx, end_idx] = T[i][start_idx, end_idx]
    cf_sample = sample
    CF.append(cf_sample)
end for
return CF
```

Figure XXXI. MG-CF Counterfactual Generation Code [7]

Figure XXXII shows in a very simple way the previously explained steps of the Motif Guided model.

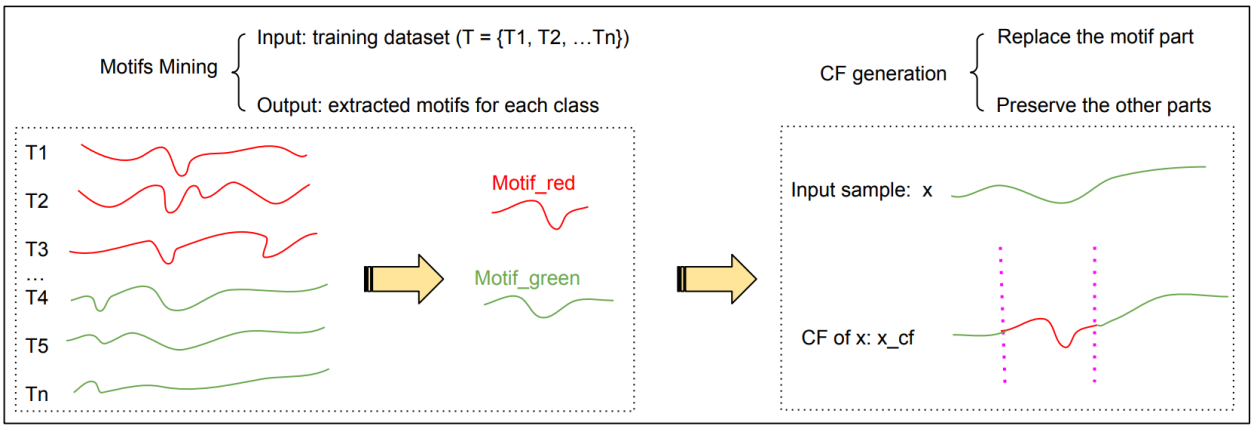

Figure XXXII. Motif Guided General Overview [7]

4.3 TimeX

TimeX [5] is designed to generate counterfactual explanations for time series with an iterative method based on the formulation of Wachter et al. [55]:

- **Initial Prediction:**

TimeX starts by predicting the labels of the training dataset using the pre-trained model to be explained. After prediction, the training dataset is divided according to the labels assigned by the model.

- **Most Relevant Segment Search and Shape Optimization:**

To generate a counterfactual explanation of a time series, the TimeX optimizer takes the time series and the DBA (Dynamic Barycenter Average) of the opposite class of that time series as we will we see later on. The DBA is computed using the Dynamic Time Warping (DTW) algorithm to find the optimal alignment among all the time series in a class and then averaging the aligned series. TimeX uses a saliency map to generate a feature attribution vector, identifying the most significant segment of the time series to perturb. Initially, the size of the segment is 1% of the length of the time series.

The shape of most relevant segment is optimized following the desired properties of the counterfactual encoded in the new loss function.

- **Iteration and Evaluation:**

At the end of each learning iteration, the generated counterfactual explanation is evaluated for validity using the pre-trained model. If the counterfactual is not valid, because it does not change the initial label or does not reach a set probability threshold, the changed segment size is considered to be too small to achieve validity.

As the changed segment size in the time series is not large enough, an iterative process starts where the most significant segment of the new size according to the saliency map is chosen and the loss term is applied to it until a size is reached where the prediction probability of the target class exceeds a threshold or the classification decision is changed. We can see all this procedure summarized in Figure XXXIII.

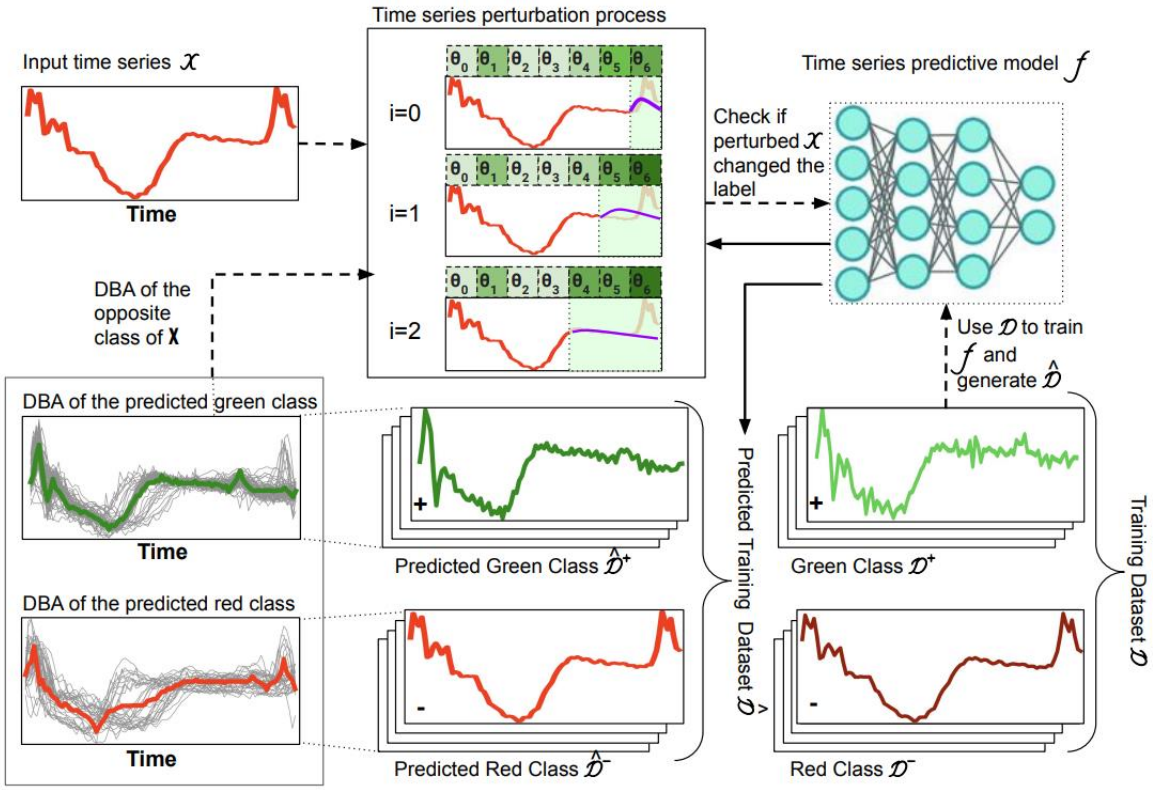

Figure XXXIII. TimeX General Overview [5]

Loss Function and Metrics

The optimisation of the step 2 is guided by a loss function based on the one seen in the Wachter et al. [55]:

$$\arg\min max\ \lambda\left(f\left(\widetilde{TS}\right)-\tilde{y}\right)^2 + d(TS_i, TS),$$

$$\tilde{y} \neq y_{TS}$$

We can break down this loss function as follows:

1. Prediction Loss:

Maximises the change in model output for the counterfactual to be classified in the target class.

$$L_{pred} = \left(f\left(\widetilde{TS}\right)-\tilde{y}\right)^2$$

2. L1 distance:

Minimises the L1 distance between the original time series and the counterfactual explanation, ensuring that the counterfactual is as close as possible to the original input.

$$L_{L1} = d(TS_i, TS)$$

To this initial loss function by Wachter et al. a new loss term is added. This is where the DBA explained above comes in. This inclusion addresses the interpretability and contiguity desirable properties.

3. DBA Loss:

Minimises the distance between the counterfactual explanation to be generated and the DBA of the desired class, promoting the plausibility of the counterfactual making it similar to the DBA.

$$L_{dba} = \theta \| TS - dba_{\tilde{y}} \|$$

In the end we are left with a loss term that takes into account the 3 previous losses:

$$L = L_{L1} + L_{pred} + L_{dba}$$

4.4 Chapter Discussion

This chapter presents the three methods developed in the project. Each method adopts different approaches to the development of counterfactual explanations to help resolve complexities of AI decision making, offering insights that make these processes more transparent and understandable.

Attention-based Counterfactual Explanation for Multivariate Time Series (AB-CF): This method trades the most relevant segments for the Nearest Unlike Neighbor (NUN) segments.

Motif-Guided Time Series Counterfactual Explanations (MG-CF): This method is based on the identification and substitution of typical subsequences typical of each output class within the time series data.

TimeX: This method incorporates a new term based on dynamic barycenter averaging (DBA) into the loss function developed by Wachter et al. [55] which gives counterfactual explanations greater plausibility and restricts changes to a single subsequence to ensure contiguity.

5. Experiments and results

This section presents the results obtained by evaluating three customized XAI methods for time series classification: Attention-based Counterfactual Explanation for Multivariate Time Series (AB-CF), Motif-Guided Time Series Counterfactual Explanations (MG-CF), and TimeX applied to five datasets belonging to the University of California, Riverside UCR Time Series Classification Archive [8]: CBF, Chinatown, ECG200, Gunpoint and Coffee. Each of these datasets represents unique challenges in the field of time series classification, allowing for a diverse and robust assessment of the performance of the methods.

The validation metrics used in this section have been described previously in Section 3.3, so only the characteristics of the datasets used are briefly described below.

- CBF (Cylinder-Bell-Funnel): The time series in CBF are generated using three mathematical functions representing different geometric shapes (cylinder, bell and funnel). Each time series has a length of 128 points, and the set is composed of 930 instances (30 training and 900 test). The distinctive feature of CBF is that the patterns are very clear and distinct from each other, which allows the models' ability to detect subtle differences to be analysed.

- Chinatown: This dataset is composed of time series obtained from security camera sensors located in the Chinatown neighborhood, which is in Melbourne, Australia, focused on detecting movement patterns and anomalous behaviours in the urban environment. This dataset contains 345 instances with a length of 24, with 20 instances used for training and 325 for testing.

- ECG200: This dataset contains electrocardiogram signals, where the objective is to classify between two types of heartbeats: normal and arrhythmia. With 200 instances in total, with 100 used for training and another 100 for testing, and time series of 96 points in length, this dataset is useful for evaluating the accuracy of models in detecting subtle and clinically relevant physiological patterns.

- Gunpoint: Originating from studies of human movement, the Gunpoint dataset seeks to classify two types of gestures: when a person points a gun at a target and when they do not. This dataset contains 200 instances, 50 for training and 150 for testing, with a serial length of 150 points, and represents an interesting challenge for models due to the natural variability of human movement.

- Coffee: This dataset comes from the spectral analysis of different types of green coffee. It contains 56 instances (28 training and 28 test) with a serial length of 286 points. The spectral signals allow distinguishing between coffee varieties on the basis of their chemical and physical characteristics.

By applying the Motif-Guided, AB-CF and TimeX methods to these five datasets, with their various characteristics mentioned above, a picture of their performance in a wide range of scenarios is expected to be obtained.

5.1 Outlier Detection Experiments

In order to measure the plausibility of the described methods in Section 5.2Implemented, two of the models mentioned in Section 3.3 have been chosen to check whether the generation of counterfactuals of the 3 implemented XAI methods respects the distribution of the input data. The models implemented are Isolation Forest and One Class SVM.

5.1.1 Isolation Forest

To find the Isolation Forest (IF) model that best fits each dataset, we used the `GridSearchCV` [57] tool, which allowed to find the best combination of hyperparameters for our model. This tool works by performing an exhaustive search in a user-defined parameter space, and with its use it has been possible to generate twenty-four combinations of hyperparameters. In these combinations, different values of the following parameters have been alternated [58]:

- **n_estimators**: The number of base estimators in the ensemble. For this hyperparameter the values 100, 150 and 200 have been used to iterate in the `GridSearchCV` method.

- **max_samples**: The number of samples to draw from X to train each base estimator. In this case only 0.5 and 1.0 have been selected for use.

- **contamination**: The amount of contamination of the data set, i.e. the proportion of outliers in the data set. Used when fitting to define the threshold on the scores of the samples. For this third hyperparameter also only two values have been selected, which are 0.01 and 0.05.

- **bootstrap**: For this hyperparameter only two cases could be selected which are as follows: If True, individual trees are fit on random subsets of the training data sampled with replacement. If False, sampling without replacement is performed.

In the following Table IV, the resulting combinations of hyperparameters can be seen. Each combination has an assigned identifier that will be used later to ease the evaluation of their performances.

combination	n_estimators	max_samples	contamination	bootstrap
h1	100	0.5	0.01	True
h2	100	0.5	0.01	False
h3	100	0.5	0.05	True
h4	100	0.5	0.05	False
h5	100	1.0	0.01	True
h6	100	1.0	0.01	False
h7	100	1.0	0.05	True
h8	100	1.0	0.05	False
h9	150	0.5	0.01	True
h10	150	0.5	0.01	False
h11	150	0.5	0.05	True
h12	150	0.5	0.05	False
h13	150	1.0	0.01	True
h14	150	1.0	0.01	False
h15	150	1.0	0.05	True
h16	150	1.0	0.05	False
h17	200	0.5	0.01	True
h18	200	0.5	0.01	False
h19	200	0.5	0.05	True
h20	200	0.5	0.05	False
h21	200	1.0	0.01	True
h22	200	1.0	0.01	False
h23	200	1.0	0.05	True
h24	200	1.0	0.05	False

Table IV. Combinations of hyperparameters for the Isolation Forest Model

Once the different combinations of hyperparameters had been generated, a metric was sought visualise in a simple way which combination best fits each data set. This is where the Silhouette Score [59] comes into play. This metric is calculated using the formula $\frac{(b-a)}{\max(a,b)}$, where a is the average distance between a point and the other points within its cluster, and b is the average distance to the nearest cluster to which it does not belong. This formula provides a range of values from -1 to 1, with values close to 1 indicating a good cluster assignment, and values close to -1 indicating an incorrect assignment.

It should be noted that there have been two cases for IF where all labels in the classification model prediction for test data had the same value. Due to the nature of the metric used and the absence of a second cluster, in these cases it is impossible to derive a value. It has therefore been decided to assign NaN value indicating that the above-mentioned situation has occurred.

Table V shows the results of the aforementioned metrics for each hyperparameter combination and dataset. The best performers for each dataset are:

- CBF: Combination of hyperparameters h22 (0.234966)
- Chinatown: Combination of hyperparameters h6 (0.505248)
- ECG200: Combination of hyperparameters h19 (0.344670)
- Gunpoint: Combination of hyperparameters **h1** y h17 (0.401708)
- Coffee: Combination of hyperparameters **h2** y h17 (0.407126)

In cases where the same value of Silhouette Score is obtained as in Gunpoint and Coffee, the selection criterion is to take the simplest model, for these cases is h1 and h2 respectively, as they have a lower number of estimators.

combination	CBF	Chinatown	ECG200	Gunpoint	Coffee
h1	0.223222	0.301666	0.327424	**0.401708**	0.311211
h2	0.194052	0.490164	0.315879	0.389055	**0.407126**
h3	0.175816	0.377113	0.310765	0.394840	0.368580
h4	0.197272	0.380249	0.340249	0.384927	0.336947
h5	0.228811	0.490164	0.330858	0.392154	0.368580
h6	0.224892	**0.505248**	0.289987	0.392154	0.368580
h7	0.142709	0.490164	0.336727	0.385951	0.333798
h8	0.193651	0.490164	0.336727	0.389668	0.368580
h9	0.220716	0.490164	0.300233	0.381128	0.368580
h10	0.230034	NaN	0.283441	0.392154	0.368580
h11	0.203367	0.363351	0.319620	0.385951	0.368580
h12	0.150853	0.490164	0.307822	0.384927	0.319249
h13	0.219858	0.490164	0.286282	0.392154	0.368580
h14	0.233282	NaN	0.286282	0.401708	0.368580
h15	0.231847	0.490164	0.323391	0.384927	0.336947
h16	0.194095	0.490164	0.329121	0.382873	0.333798
h17	0.222366	NaN	0.289827	0.401708	0.407126
h18	0.223433	NaN	0.331238	0.392154	0.368580
h19	0.143494	0.490164	**0.344670**	0.385951	0.336947
h20	0.190065	0.445319	0.314349	0.385951	0.319249
h21	0.206794	NaN	0.326626	0.381128	0.317093
h22	**0.234966**	0.490164	0.274479	0.392154	0.368580
h23	0.184403	0.490164	0.336727	0.384927	0.368580
h24	0.168683	0.474494	0.323391	0.389668	0.368580

Table V. Silhouette score for each combination of hyperparameters and datasets for Isolation Forest

5.1.2 One Class SVM

As for IF, for One Class SVM an 18 hyperparameter combinations has been generated to search of the best fit for each dataset. This iteration has also been done through `GridSearchCV`. In this case, the following three hyperparameters of the model have been used [60]:

- nu: An upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. For this parameter 3 values have been selected: 0.1, 0.5 and 0.9.

- kernel: Specifies the kernel type to be used in the algorithm. Three kernels have been selected for the generation of different hyperparameters: linear, rbf and poly.

- gamma: Kernel coefficient for 'rbf', 'poly' and 'sigmoid'. Two cases have been selected for this hyperparameter: If 'auto', uses $\frac{1}{n\_features}$ and if 'scale' which is the default value, it uses $\frac{1}{n_{features} \times X.var()}$ as value of gamma

The iteration through the aforementioned hyperparameters values has resulted in the combinations shown in Table VI. Again, each combination has an identifier associated to help with the presentation of the performance results.

combination	nu	kernel	gamma
h1	0.1	linear	scale
h2	0.1	linear	auto
h3	0.1	rbf	scale
h4	0.1	rbf	auto
h5	0.1	poly	scale
h6	0.1	poly	auto
h7	0.5	linear	scale
h8	0.5	linear	auto
h9	0.5	rbf	scale
h10	0.5	rbf	auto
h11	0.5	poly	scale
h12	0.5	poly	auto
h13	0.9	linear	scale
h14	0.9	linear	auto
h15	0.9	rbf	scale
h16	0.9	rbf	auto
h17	0.9	poly	scale
h18	0.9	poly	auto

Table VI. Combination of hyperparameters for the One Class SVM model

For the OC-SVM model, the Silhouette Score [59] was also used to assess the best fit for each dataset. As explained above, this metric is calculated with the formula $\frac{(b-a)}{\max(a,b)}$, where a is the average distance within the cluster and b the distance to the nearest cluster. Values close to 1 indicate good cluster separation, while those close to -1 reflect an incorrect assignment.

Similarly to IF, there are cases where the labels provided by the classifier are all the same. In those situations, the silhouette score cannot be calculated and a NaN value is assigned.

Table VII**¡Error! No se encuentra el origen de la referencia.** shows the results of the above metric, for each hyperparameter and for each dataset, obtaining that the best results for each dataset are:

- CBF: Combination of hyperparameters h1 y **h2** (0.090563)
- Chinatown: Combination of hyperparameters h13 y **h14** (0.474494)
- ECG200: Combination of hyperparameters h3 y **h4** (0.224865)
- Gunpoint: Combination of hyperparameters h5 y **h6** (0.274067)
- Coffee: Combination of hyperparameters h1, **h2**, h3, h4, h5, y h6 (0.263254)

As with IF, for One Class SVM there are several cases where the best Silhouette Score value for a dataset is repeated. In these cases, the simplest hyperparameter combination is selected, which is highlighted in bold. These results indicate that that the gamma parameter does not affect the quality of the model as it fails to change the metric value.

combination	CBF	chinatown	ECG200	gunpoint	coffee
h1	0.090563	0.145459	0.256362	0.259865	0.263254
h2	0.090563	0.145459	0.256362	0.259865	0.263254
h3	0.037846	0.262872	0.224865	0.267360	0.263254
h4	0.039636	NaN	0.224865	0.267360	0.263254
h5	-0.049749	0.145459	0.198796	0.274067	0.263254
h6	-0.049749	0.145459	0.198796	0.274067	0.263254
h7	0.036213	0.409985	0.156363	0.160745	-0.020563
h8	0.036213	0.409985	0.156363	0.160745	-0.020563
h9	0.010539	0.111704	0.120067	0.170606	-0.020563
h10	0.012042	NaN	0.120067	0.170606	-0.020563
h11	-0.049292	0.409064	0.120193	0.150479	-0.020563
h12	-0.049292	0.409064	0.120193	0.150479	-0.020563
h13	-0.099362	0.474494	-0.157377	-0.051304	-0.157066
h14	-0.099362	0.474494	-0.157377	-0.051304	-0.157066
h15	-0.116696	-0.067589	-0.166634	-0.122312	-0.157066
h16	-0.116696	NaN	-0.166634	-0.122312	-0.157066
h17	-0.111790	0.474494	-0.145391	-0.130399	-0.157066
h18	-0.111790	0.474494	-0.145391	-0.130399	-0.157066

Table VII. Silhouette score for each combination of hyperparameters and datasets for One Class SVM

5.2 Implemented counterfactual explanation methods comparison

The following is a comparison of the results of the three implemented methods, AB-CF, MG-CF and TimeX, seen for each of the aforementioned datasets used.

CBF

The AB-CF method affects approximately half of the time series, with a sparsity value of 0.52, which could imply a somewhat sparse but manageable explanation. The distances L1 and L2 are notable, at 48.87 and 7.51, respectively, indicating that the changes it makes are not minimal but are distributed in a way that does not drastically alter the series.

Furthermore, AB-CF stands out with 100% validity, which means that all generated counterfactuals are labelled by the classification model with a different label than the original instance.

As for its performance in outlier detection models, the Isolation Forest models consider most of the counterfactuals generated to be anomalous compared to the training data. In the case of the OC-SVM it considers it to a lesser extent resulting in a 0.63.

As for the number of substrings affected, on average 2.41 substrings need to be changed by their respective NUNs and the execution time of the method is relatively low, with 1.38 seconds per counterfactual, making this method a viable option considering its computational time.

On the other hand, is the most concise method for this dataset, with a sparsity of 0.3, meaning that modifications affect a part of the series, providing localised explanations. However, its L1 distance is 40.65, the highest of the three methods, and its L2 distance is the highest at 8.03, suggesting that the changes made have a larger impact.

However, its validity rate is relatively low, only 41%, which means that less than half of the generated explanations have reached the mentioned label change.

As for the outlier detection models, something similar happens as in ABCF. For the autoencoder and the IF the generated counterfactuals are more anomalous than the training data, for the SVM rather less than the other ones. And finally in terms of time, MG-CF is the fastest of the three methods, with an extremely low execution time.

Finally, TimeX is the method that requires the most changes, with an average of 128, and affects the entire time series, with a sparsity of 1.0. Although it has the smallest distances L1 (32.0) and L2 (4.23), the fact that it modifies the whole series to achieve this can make intuitive interpretation of the explanations difficult. As a main feature to highlight, its validity rate is the lowest, with only 17%, which implies that most of the generated explanations are not considered by the classification model as different than the original instance. TimeX is also the

slowest method, with a running time of 134.1 seconds, which makes it significantly less efficient than the other two.

method	sparsity	L1	L2	valid	IF_OS	SVM_OS	subsequences	times
ab_cf	0.52 ± 0.19	48.87 ± 18.73	7.51 ± 2.11	1.0 ± 0.0	1.09 ± 0.05	0.63 ± 0.14	2.41 ± 0.77	1.38 ± 0.59
mg_cf	0.3 ± 0.0	40.65 ± 9.13	8.03 ± 1.87	0.41 ± 0.49	1.12 ± 0.05	0.59 ± 0.2	1.0 ± 0.0	0.0 ± 0.0
TimeX	1.0 ± 0.0	32.0 ± 5.36	4.23 ± 0.37	0.17 ± 0.38	1.05 ± 0.04	0.61 ± 0.16	1.0 ± 0.0	134.1 ± 31.08

Table VIII. CBF metrics

Chinatown

The AB-CF sparsity of 0.38 that suggests that it affects just over a third of the time series, implying a localised and relatively concise explanation. In terms of distances, AB-CF records values of 956.16 for L1 and 430.97 for L2, reflecting that the changes made are very significant.

Furthermore, it has perfect validity, with 100%, ensuring that all explanations are considered by the classification method with the opposite label to the original time series.

In the anomaly detection models, with an IF_OS of 1.12 and SVM_OS of 0.69, indicating that it considers the counterfactuals generated in relation to the training data to be anomalous to a greater extent for the IF and to a lesser extent for the OC-SVM.

AB-CF changes 1.1 relevant subsequences on average, which implies an explanation-centred approach and, finally its execution time is 1.08 seconds, which makes it efficient in terms of processing.

On the other hand, MG-CF stands out for its sparsity that is 0.29, which means that it affects less than 30% of the time series, providing very localised explanations. However, although it requires fewer changes, the distances L1 (1017.59) and L2 (554.22) are still higher than those of AB-CF, suggesting that the changes made have an even greater impact in terms of magnitude.

But this is affected by the validity rate of 73%, which means that not all explanations are able to change the label.

Finally, TimeX does not provide results on this dataset, as it fails to generate any counterfactual that the classification model considers to have been relabelled with respect to the original time series.

method	sparsity	L1	L2	valid	IF_OS	SVM_OS	subsequences	times
ab_cf	0.38 ± 0.07	956.16 ± 287.53	430.97 ± 123.65	1.0 ± 0.0	1.12 ± 0.06	0.69 ± 0.1	1.1 ± 0.31	1.08 ± 0.14
mg_cf	0.29 ± 0.01	1017.59 ± 145.21	554.22 ± 82.92	0.73 ± 0.45	1.15 ± 0.04	0.7 ± 0.1	1.03 ± 0.18	0.0 ± 0.0
TimeX	nan ± nan	nan ± nan	nan ± nan	0.0 ± 0.0	1.1 ± 0.06	0.69 ± 0.1	nan ± nan	78.2 ± 21.39

Table IX. Chinatown metrics

ECG200

AB-CF methods requires an average 59% of changes w.r.t. the total length of the time series changes to modify the prediction, which puts it ahead of MG-CF and TimeX in terms of the number of modifications needed. The L1 and L2 distances are 23.0 and 3.77, respectively, indicating that the modifications are relatively moderate compared to the other methods.

. Furthermore, this previous result becomes more valuable when we know that AB-CF achieves 94% validity, which ensures that the vast majority of its explanations are considered with the opposite label as the original time series they come from.

In the outlier detection models, AB-CF presents high results in the comparisons with the training data, which indicates that the counterfactuals generated in relation to these data are considered. AB-CF detects on average 1.61 relevant subsequences, which implies a detailed approach to explanations. The execution time is 1.46 seconds, which makes it a relatively efficient method in terms of processing.

MG-CF, on the other hand, requires a higher number of changes, with 67 on average, and its sparsity of 0.7, which means that it affects a larger percentage of the time series changed compared to AB-CF. Although it needs more changes, the L1 (41.51) and L2 (6.71) distances are higher, indicating that the modifications made have a greater impact on the series.

However, the validity rate is significantly lower, only 32%, which implies that many of the explanations generated are not as good as they do not have the opposite label to the original instance. Additionally, execution time of MG-CF is instantaneous, making it the fastest method despite its low validity.

Lastly, TimeX is the method that requires the most changes, with 96 on average, and affects the entire time series, as indicated by its sparsity of 1.0. Despite making the highest number of changes, its L1 (16.59) and L2 (2.93) distances are the lowest, indicating that its modifications have a smaller impact. In this case it stands out that its validity rate is the lowest, with only 16%, indicating that only this small percentage are the counterfactuals that come to be considered with the opposite label by the classification model, which largely invalidates the method.

method	sparsity	L1	L2	valid	IF_OS	SVM_OS	subsequences	times
ab_cf	0.59 ± 0.31	23.0 ± 14.87	3.77 ± 1.85	0.94 ± 0.24	1.25 ± 0.09	0.89 ± 0.08	1.61 ± 0.78	1.46 ± 0.55
mg_cf	0.7 ± 0.0	41.51 ± 10.48	6.71 ± 1.49	0.32 ± 0.47	1.27 ± 0.12	0.89 ± 0.09	1.0 ± 0.0	0.0 ± 0.0
TimeX	1.0 ± 0.0	16.59 ± 4.7	2.93 ± 0.51	0.16 ± 0.37	1.18 ± 0.1	1.01 ± 0.09	1.0 ± 0.0	217.77 ± 41.06

Table X. ECG200 metrics

Gunpoint

AB-CF makes an average of 43% of changes, which places it in an intermediate position between the other two methods in terms of percentage number of modifications. In terms of distances, it has an L1 value of 17.58 and an L2 value of 2.92, indicating that the modifications made are moderate in magnitude.

The probability of belonging to the desired label is perfect at 100%, making it a very reliable method.

In the outlier detection models, AB-CF results in IF_OS of 1.12 and SVM_OS of 0.92. This means that for both IF and OC-SVM, the generated counterfactuals are anomalous compared to the training data. In addition, its execution time is 1.11 seconds, which makes it efficient in terms of processing.

MG-CF, on the other hand, requires an average of 50% changes, which makes it the method that needs the most modifications after TimeX. The distances L1 (20.71) and L2 (3.35) are larger than those of AB-CF, indicating that modifications have a greater impact than this other method. In this case, the validity rate is extremely low, only 3%, which means that almost all the generated explanations are not correct, as they are not considered by the classification model of the class contrary to the original instance.

Finally, TimeX requires the largest number of changes, with an average of 150, and affects the entire time series, as indicated by its sparsity of 1.0. Its L1 (28.48) and L2 (3.93) distances are the highest, reflecting that modifications have a considerable impact relative to the other models. Again, its validity rate is low, only 21%, which means that many of the explanations would not be usable as they are not labelled with the opposite class to the original one. This method detects only one relevant sub-sequence, and its execution time is considerably high at 343.85 seconds, making it by far the least time-efficient method.

method	sparsity	L1	L2	valid	IF_OS	SVM_OS	subsequences	times
ab_cf	0.43 ± 0.27	17.58 ± 14.58	2.92 ± 1.86	1.0 ± 0.0	1.12 ± 0.06	0.92 ± 0.05	1.98 ± 0.52	1.11 ± 0.45
mg_cf	0.5 ± 0.0	20.71 ± 22.02	3.35 ± 2.75	0.03 ± 0.18	1.13 ± 0.07	0.88 ± 0.06	1.0 ± 0.0	0.0 ± 0.0
TimeX	1.0 ± 0.0	28.48 ± 5.05	3.93 ± 0.43	0.21 ± 0.41	1.1 ± 0.07	0.85 ± 0.08	1.0 ± 0.0	343.85 ± 41.3

Table XI. Gunpoint metrics

Coffee

AB_CF requires an average of 56% of changes to modify the prediction, making it a moderate adjustment method compared to the other two. The L1 and L2 distances are 13.43 and 1.38, respectively, suggesting that the modifications made are relatively small in magnitude.

Furthermore, AB_CF has a validity of 100%, indicating that all generated explanations are considered to be of the opposite class of the original instance they come from.

Regarding its performance in the outlier detection models, it is considered that the generated counterfactuals are anomalous in relation to the training data. As in previous datasets, AB_CF execution time is reasonably efficient with 1.35 seconds.

MG-CF, on the other hand, requires considerably fewer changes, 30%, making it the most efficient method in terms of modifications. The distances L1 (14.12) and L2 (2.02) are slightly larger than those of AB_CF, suggesting a somewhat larger impact on modifications, but over a shorter time span.

However, the validity rate of only 54% suggest that as many of the explanations are not labelled correctly. Its instantaneous execution time makes it the fastest method, although its low validity is a clear disadvantage.

TimeX is the method that requires the highest number of changes, with 286 on average, and affects the entire time series (sparsity of 1.0). Despite making more changes, the L1 (10.93) and L2 (1.1) distances are the lowest, indicating that the changes have a relatively minor impact in terms of magnitude. Again, its validity is very low, only 4%, indicating that for almost no time series that is entered a counterfactual is generated that has the opposite label. Also again, its running time is considerably high, at 276.07 seconds, making it by far the least time efficient method.

method	sparsity	L1	L2	valid	IF_OS	SVM_OS	subsequences	times
ab_cf	0.56 ± 0.16	13.43 ± 4.56	1.38 ± 0.36	1.0 ± 0.0	1.13 ± 0.07	1.0 ± 0.01	1.54 ± 0.74	1.35 ± 0.24
mg_cf	0.3 ± 0.0	14.12 ± 2.08	2.02 ± 0.27	0.54 ± 0.51	1.14 ± 0.06	1.01 ± 0.02	1.0 ± 0.0	0.0 ± 0.0
TimeX	1.0 ± nan	10.93 ± nan	1.1 ± nan	0.04 ± 0.19	1.09 ± 0.07	0.99 ± 0.01	1.0 ± nan	276.07 ± 8.16

Table XII. Coffee metrics

5.3 Discussion

Figure XXXIV shows a visual comparison of some examples of counterfactual explanations generated by the XAI methods implemented for each dataset. The figure shows the behaviours of the methods through representation where the original instances in blue and the changes generated by each model in red.

The MGCF method is based on changing segments of a predetermined length by the “motif” or “shapelet” that provides the most information of the opposite class for that segment. This generates a single change, of different lengths for each case, in the counterfactual explanation. Despite this, the validity metric of this method has not been specifically high, with fluctuations ranging from extremely low values such as 0.03 to more acceptable values such as 0.73.

On the contrary, ABCF is based on the substitution of the k most important non-continuous segments of the input data by their respective Nearest Unlike Neighbor. This is done until the label of the counterfactual is contrary to the time series of the input data, providing high validity, or no more non-continuous changes can be made. Figure XXXIV shows the different segments that have been changed by the NUNs. It can be seen how in some cases only one change has been necessary and in others the model has needed more than one to change the label.

Finally, the TimeX method, both the results seen in the previous section and in this one confirm that the results have not been as expected. In the case of the Chinatown dataset, as we have also seen in the previous section, it does not generate changes and for the other 3 remaining datasets it may seem to generate acceptable counterfactuals, but as we have seen in the previous section, most of these counterfactuals do not manage to change the label of the prediction in relation to the input time series.

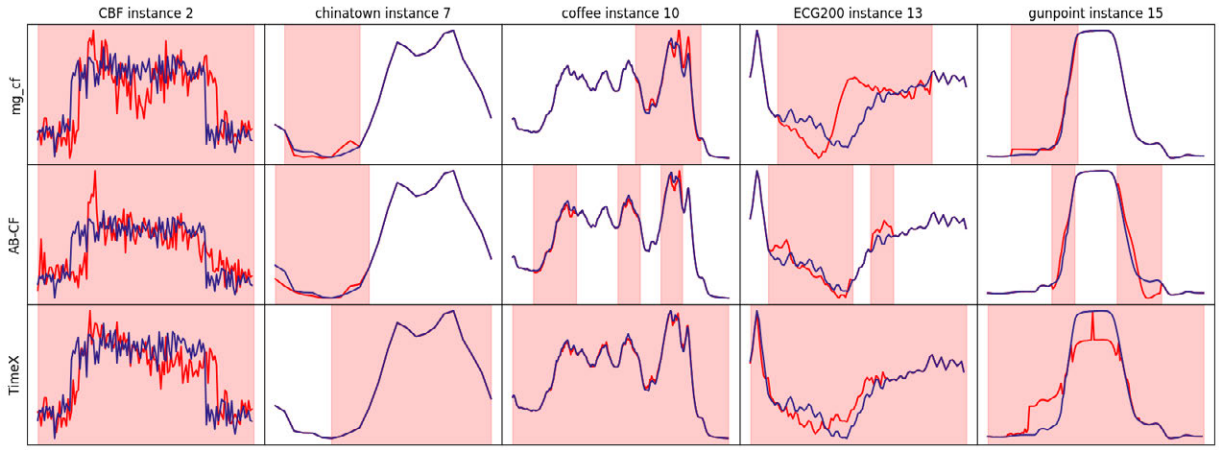

Figure XXXIV. Visual comparison of the counterfactuals generated by the implemented XAI methods.

6. Budget

In this section the budget related to the project is presented. The total cost is divided into the cost related to the materials used for the development of the project and the cost of the people who have participated in the project. Finally, the sum of the total cost of the project is displayed at the end.

Material costs

In Table XIII below, the material costs are detailed.

ITEM	PRICE
Windows 11 Home	145,00 €
Microsoft Office 365 License	69,00 €
IDE Visual Studio Code	0,00 €
Python packages needed	0,00 €
UCR database	0,00 €
Laptop Hewlett-Packard EliteBook 640 G9 12th Gen Intel® Core™ i7-1265U 1,8 GHz 16 GB RAM	1.545,63 €
HP M22f 21.5" LED IPS FullHD 75Hz FreeSync Monitor	119,00 €
Project binding	175,00 €
TOTAL	**2.053,63 €**

Table XIII. Material costs of the project

Personal costs

In the personal costs of the project, summarised in Table XIV, are included:

- 360 hours of student project work as estimated at the beginning of the project, giving the student the position of Junior Data Scientist.
- 36 hours of work as the 10 % of the total estimated project hours associated to the tutor and the project manager, with positions of Tenured Full Professor and Senior Data Scientist respectively.

ITEM	PRICE / YEAR	DURATION (HOURS)	SUBTOTAL
Junior Data Scientist	25.000,00 €	360	4.687,50 €
Tenured Full Professor	65.000,00 €	36	1.218,75 €
Senior Data Scientist	50.000,00 €	36	937,50 €
		TOTAL	**6.843,75 €**

Table XIV. Personal costs of the project

The salaries associated with these positions have been taken from Glassdoor [61] as the average salaries for each case.

Total cost

Finally, the material costs plus the personnel costs are added together, plus a profit from the sale of the project of 7% of the above-mentioned costs to give the final cost of the project (Table XV).

ITEM	PRICE
Material costs	2.053,63 €
Personal costs	6.843,75 €
Benefits	622,82 €
TOTAL	**9.520,20 €**

Table XV. Total costs of the project

7. Impacts of the project

This project achieves, through the development of different models of Explainable Artificial Intelligence, a great impact on different areas of society that can be aligned with the Sustainable Development Goals (SDGs).

Social Impact

The impact of this project on people is the most notorious of all, since, as developed in Chapter Explainable Artificial Intelligence3, XAI is born from the need of human beings to know the reasons why things happen.

Therefore, this project contributes enormously to society's confidence in the models that are developed thanks to its improvement in the transparency and the reliability of AI and increases their use in countless sectors.

Technological Impact

Due to the nature of the XAI methods, the technological contribution of the project is very broad. This is because the explanatory models developed are model agnostic, as explained in Section 3.1 , meaning that the generation of counterfactuals can be developed for any trained model, which allows numerous applications for time series.

Contribution to Sustainable Development Goals (SDG)

In addition to the impacts already mentioned, this project can be clearly aligned with the Sustainable Development Goals (SDGs) of the United Nations (UN) Agenda 2030. [62]

SDG 3: Good Health and Well-being. The project is closely related to SDG 3, because the health area is one where the work on transparency in AI models is mostly needed, since it is crucial to know where decisions come from. This strengthens the capacity for early warning, risk reduction and health risk management, as indicated in sub-goal 3.D [63].

SDG 9: Industry, Innovation and Infrastructure. This SDG is also clearly aligned with the project, as it promotes the increase of scientific research and the improvement of the technological capacity of industrial sectors, including promoting innovation as indicated in SDG sub-goal 9.5 [64].

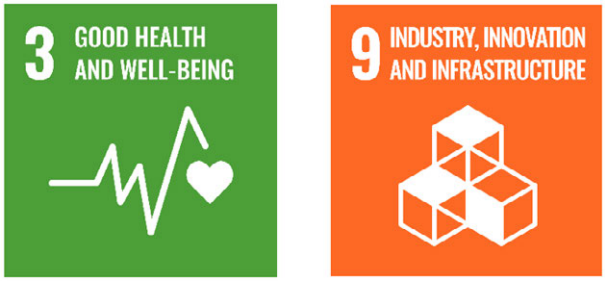

Figure XXXV. SDGs aligned with the project [65]

8. Conclusions and future work

8.1 Conclusions

As a personal conclusion of the project, highlights the growing need for Explainable Artificial Intelligence (XAI) as the use of machine learning (ML) expands day by day. It is essential to develop methods to understand and justify automated decisions, promoting user confidence and the ethical use of these systems in critical areas. This experience has reinforced the importance of continued research in both ML and XAI to ensure in this case more transparent and reliable models for society.

Regarding the methods evaluated, the conclusions are as follows:

- The AB-CF method stood out for offering highly interpretable explanations as it managed to obtain excellent results in vitally important metrics such as validity and proximity, generating clearer and more useful explanations, and also achieving these good results with very affordable calculation times.

- TimeX proved to be inefficient, requiring numerous changes in the time series (96 to 286) and extremely high run times (134.1 to 343.85 seconds), with an incredibly low validity rate (0% to 21%), which limits its practical utility in real-time applications.

- MG-CF, on the other hand, requires fewer modifications (6.96 to 85) and is significantly faster, but its validity rate varies widely (3% to 73%). Although it is more efficient in execution time, the high variability in its effectiveness and the larger distances indicate that, although it may be useful in some contexts, its applicability is limited.

8.2 Future works

The completion of this project leaves the door open to different possible future works that can be carried out to find improvements in the results obtained and developed in the project. Possible improvements are as follows:

- For the Motif-Guided method, it is suggested to work on optimising the ‘validity’ metric, which is not particularly high. This is due to the classification model used, which is more robust than the one used in the original article, where better results are obtained in this metric.

- In the case of TimeX, there are two key aspects to improve: execution times and validity. Currently, the generation of a counterfactual takes around 7 minutes, which has generated calculation sessions of more than 24 hours. In addition, the validity of the method has been zero for the Chinatown dataset and very low for the other datasets. These problems do not seem to be the result of the adaptation carried out,

but were already present in the notebook associated with the original article, where equally poor results were observed.

- Once these issues have been resolved, it would be advisable to explore different combinations of hyperparameters for TimeX. Performing a GridSearchCV, similar to the one applied in Isolation Forest and OneClass SVM, would allow finding the optimal combination for this model.

- Finally, it is suggested to use more datasets to evaluate the behaviour of the different methods to get a broader view of their performance in different scenarios.

9. References

[1] K. P. Murphy, Probabilistic Machine Learning: An Introduction, MIT Press, 2022.

[2] C. Molnar, "Interpretable Machine Learning: A Guide for Making Black Box Models Explainable", Munich, 2022.

[3] S. Verma, Counterfactual Explanations and Algorithmic Recourses for Machine Learning, Seattle: arXiv, 2022.

[4] T. Rojat, "Explainable Artificial Intelligence (XAI) on TimeSeries Data," *arXiv,* 2021.

[5] S. F. Boubrahimi and S. M. Hamdi, "On the Mining of Time Series Data Counterfactual Explanations using Baricenters," in *CIKM '22: Proceedings of the 31st ACM International Conference on Information & Knowledge*, Atlanta, 2022.

[6] P. Li, O. Bahri, S. F. Boubrahimi and S. M. Hamdi, "Attention-based Counterfactual Explanation for Multivariate Time Series," in *International Conference on Big Data Analytics and Knowledge Discovery*, 2023.

[7] P. Li, "Motif-Guided Time Series Counterfactual Explanations," *arXiv,* 2023.

[8] R. University of California, "UCR Time Series Classification Archive," 7 Marzo 2024. [Online]. Available: https://www.cs.ucr.edu/~eamonn/time_series_data/.

[9] "TensorFlow," [Online]. Available: https://www.tensorflow.org/?hl=es-419. [Accessed 7 Marzo 2024].

[10] T. Mitchell, Machine Learning, Mc-Graw Hill, 1997.

[11] N. Gupta, "A guide to Supervised Learning," 15 April 2024. [Online]. Available: https://medium.com/@ngneha090/a-guide-to-supervised-learning-f2ddf1018ee0.

[12] Eastgate Software, "What is Unsupervised Learning," 26 January 2024. [Online]. Available: https://eastgate-software.com/what-is-unsupervised-learning/.

[13] Z. Hong, "Fraud Detection with Machine Learning: Identifying Suspicious Patterns in Financial Transactions," Medium, 20 April 2024. [Online]. Available: https://medium.com/@zhonghong9998/fraud-detection-with-machine-learning-identifying-suspicious-patterns-in-financial-transactions-8558f3f1e22a.

[14] R. Sutton and A. Barto, Reinforcement Learning: An Introduction, Cambridge: The MIT Press, 2014.

[15] Javatpoint, "Reinforcement Learning," [Online]. Available: https://www.javatpoint.com/reinforcement-learning.

[16] Geeks for Geeks, "Linear Regression in Machine Learning," 20 March 2024. [Online]. Available: https://www.geeksforgeeks.org/ml-linear-regression/.

[17] D. Montgomery and G. Runger, Applied Statistics and Probability for Engineers, John Wiley & Sons, 2018.

[18] M. Sarvandani, "Top 20 applications of classification in machine learning," Medium, 9 June 2023. [Online]. Available: https://medium.com/@mohamadhasan.sarvandani/top-applications-of-classification-in-machine-learning-e7b4351f64eb.

[19] Z. Keita, "Clasificación en Machine Learning: Introducción," Datacamp, March 2024. [Online]. Available: https://www.datacamp.com/es/blog/classification-machine-learning.

[20] S. K. Trivedi, "A study of machine learning classifiers for spam detection," in *4th International Symposium on Computational and Business Intelligence (ISCBI)*, Olten, Switzerland, 2016.

[21] Analytix Labs, "What is Classification Algorithm in Machine Learning? With Examples," 25 January 2023. [Online]. Available: https://www.analytixlabs.co.in/blog/classification-in-machine-learning/.

[22] 1938, "Metodos Supervisados," [Online]. Available: https://1938.com.es/supervisados.

[23] N. A. Ahmed, "What is A Confusion Matrix in Machine Learning? The Model Evaluation Tool Explained," November 2023. [Online]. Available: https://www.datacamp.com/tutorial/what-is-a-confusion-matrix-in-machine-learning.

[24] M. M. Rodríguez-Hernández, R. E. Pruneda and J. M. Rodríguez Diaz, "Statistical Analysis of the Evolutive Effects of Language Development in the Resolution of Mathematical Problems in Primary School Education," *Mathematics,* 2021.

[25] F. T. Liu, K. M. Ting and Z.-H. Zhou, "Isolation Forest," in *Eighth IEEE International Conference on Data Mining*, Pisa, 2008.

[26] Y. Regaya, F. Fadli and A. Amira, "Point-Denoise: Unsupervised outlier detection for 3D point clouds enhancement," *Multimedia Tools and Applications,* 2021.

[27] scikit-learn, "One-class SVM with non-linear kernel (RBF)," [Online]. Available: https://scikit-learn.org/0.17/auto_examples/svm/plot_oneclass.html.

[28] X. Jin and J. Han, "K-Means Clustering," in *Encyclopedia of Machine Learning*, Boston, MA, Springer, 2011, pp. 563-564.

[29] S. Khosla, "Algoritmo ML | K-means++," Geeks for Geeks, [Online]. Available: https://www.geeksforgeeks.org/ml-k-means-algorithm/.

[30] N. Landman, H. Pang and C. Williams, "k-Means Clustering," [Online]. Available: https://brilliant.org/wiki/k-means-clustering/.

[31] I. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences,* vol. 374, 2016.

[32] C. Cheng, "Principal Component Analysis (PCA) Explained Visually with Zero Math," 3 February 2022. [Online]. Available: https://towardsdatascience.com/principal-component-analysis-pca-explained-visually-with-zero-math-1cbf392b9e7d.

[33] C.-Y. J. Peng, K. L. Lee and G. Ingersoll, "An Introduction to Logistic Regression Analysis and Reporting," *The Journal of Educational Research,* pp. 3-14, 2010.

[34] GraphPad, "How simple logistic regression differs from simple linear regression," [Online]. Available: https://www.graphpad.com/guides/prism/latest/curve-fitting/reg_simple_logistic_and_linear_difference.htm.

[35] M. Hearst, S. Dumais, E. Osuna, J. Platt and B. Schölkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications,* vol. 13, no. 4, pp. 18-28, 1998.

[36] Wikipedia, "Support Vector Machine," [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine.

[37] L. Breiman, "Random Forests," in *Machine Learning*, Springer Netherlands, 2001, pp. 5-32.

[38] D. R. Yehoshua, "Random Forests," 25 March 2023. [Online]. Available: https://medium.com/@roiyeho/random-forests-98892261dc49.

[39] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[40] Geeks for Geeks, "Artificial Neural Networks and its Applications," 02 June 2023. [Online]. Available: https://www.geeksforgeeks.org/artificial-neural-networks-and-its-applications/.

[41] V. Nair and G. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *ICML'10: Proceedings of the 27th International Conference on International Conference on Machine Learning*, Haifa, 2010.

[42] N. McCullum, "Deep Learning Neural Networks Explained in Plain English," 28 June 2020. [Online]. Available: https://www.freecodecamp.org/news/deep-learning-neural-networks-explained-in-plain-english/.

[43] Y. LeCun, Y. Bengio and G. Hinton, in "Deep learning", Nature, 2015, pp. 436-444.

[44] P. Ratan, "What is the convolutional network architecture," 06 November 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/.

[45] K. Kim, W. Kim, J. Seo and Y. Jeong, "Prediction of Concrete Fragments Amount and Travel Distance under Impact Loading Using Deep Neural Network and Gradient Boosting Method," *Materials,* 2022.

[46] I. Goodfellow, Y. Bengio and A. Courville, "Sequence Modeling: Recurrent and Recursive Nets.," in *Deep Learning*, MIT Press, 2016.

[47] P. Antoniadis, "Recurrrent Neural Networks," 18 March 2024. [Online]. Available: https://www.baeldung.com/cs/recurrent-neural-networks.

[48] European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data," [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj.

[49] M. T. Ribeiro, S. Singh and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," *arXiv,* 2016.

[50] Z. Keita, "eXplainable AI (XAI): LIME & SHAP, Two Great Candidates to Help You Explain Your Machine Learning Models," Medium, [Online]. Available: https://towardsdatascience.com/explainable-ai-xai-lime-shap-two-great-candidates-to-help-you-explain-your-machine-learning-a95536a46c4e.

[51] S. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," *arXiv,* 2017.

[52] C. Molnar, "Why SHAP needs to be estimated," Mindful Modeler, [Online]. Available: https://mindfulmodeler.substack.com/p/why-shap-needs-to-be-estimated.

[53] Y. Geurkink, J. Boone, S. Verstockt and J. Bourgois, "Machine Learning-Based Identification of the Strongest Predictive Variables of Winning and Losing in Belgian Professional Soccer," *Applied Sciences,* 2021.

[54] M. Refoyo and D. Luengo, "Sub-SpaCE: Subsequence-Based Sparse Counterfactual Explanations for Time Series Classification Problems," in *XAI*, 2024.

[55] S. Wachter, B. Mittelstadt and C. Russell, "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR," *arXiv,* 2017.

[56] E. Delaney, D. Greene and M. Keane, "Instance-based Counterfactual Explanations for Time Series Classification," *arXiv,* 2020.

[57] scikit-learn, "GridSearchCV," scikit-learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[58] scikit-learn, "IsolationForest," scikit-learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html.

[59] scikit-learn, "silhouette_score," scikit-learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html.

[60] scikit-learn, "OneClassSVM," scikit-learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html.

[61] "Glassdoor," [Online]. Available: https://www.glassdoor.es/.

[62] United Nations, "United Nations Sustainable Development Goals," [Online]. Available: https://www.un.org/sustainabledevelopment/.

[63] United Nations, "Health and Well-being SDG," [Online]. Available: https://www.un.org/sustainabledevelopment/health/.

[64] United Nations, "Industry SDG," [Online]. Available: https://www.un.org/sustainabledevelopment/infrastructure-industrialization/.

[65] United Nations, "SDG Communications Material," [Online]. Available: https://www.un.org/sustainabledevelopment/news/communications-material/.

[66] Y. Huang, "GitHub Reporsitory," [Online]. Available: https://github.com/yuxiaohuang/research/tree/master/gwu/accepted/sam_2021.