Escuela Politécnica Superior

21
22

Master thesis

Interpretable Machine Learning for Intrapartum Fetal Hypoxia Detection

Aprendizaje Automático Interpretable en la Detección de Hipoxia Fetal Intraparto

Carmen Plaza Seco

Escuela Politécnica Superior
Universidad Autónoma de Madrid
C\Francisco Tomás y Valiente nº 11

UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Master as Research and Innovation on Computational Intelligence and Interactive Systems

MASTER THESIS

Interpretable Machine Learning for Intrapartum Fetal Hypoxia Detection
Aprendizaje Automático Interpretable en la Detección de Hipoxia Fetal Intraparto

Author: Carmen Plaza Seco
Advisor: Carlos María Alaíz Gudín and Ángela Fernández Pascual
Ponente: José Ramón Dorronsoro Ibero

September 2021

Carmen Plaza Seco
Interpretable Machine Learning for Intrapartum Fetal Hypoxia Detection

Aprendizaje Automático Interpretable en la Detección de Hipoxia Fetal Intraparto

Carmen Plaza Seco
C\␣Francisco␣Tomás␣y␣Valiente␣Nº 11␣

PRINTED␣IN␣SPAIN␣

Nothing in life should be feared, only understood.
Now is the time to understand more, to fear less.

Marie Curie

Agradecimientos

En primer lugar a mis tutores, Ángela Fernández Pascual y Carlos María Alaíz Gudín, por haber confiado en mí y haber contado conmigo. Dar las gracias también a mis compañeros de proyecto Isabel Gallego Llorente y David López Ramos. Gracias por vuestro tiempo y por acompañarme en este proceso. Gracias al proyecto DAHFI de la Comunidad de Madrid y la Universidad Autónoma de Madrid (referencia: SI1/PJI/2019-00528) donde se enmarca este trabajo.

Al grupo de Aprendizaje Automático (GAA) y al Centro de Computación Científica (CCC) de la UAM, por el uso de las instalaciones.

Por último, dar las gracias a mis compañeros y amigos, pero sobre todo a mi familia por motivarme y apoyarme siempre en cada paso.

Resumen

Actualmente, el aprendizaje automático (AA) se ha convertido en una herramienta muy utilizada en distintos campos debido a su gran capacidad para aprender a resolver problemas de forma automática y para analizar grandes cantidades de datos de forma eficiente. De hecho, en los últimos años, se han llegado a resolver problemas del mundo real con muy buenos resultados con métodos de AA. Sin embargo, incluso para los expertos en el ámbito de AA, a veces sus resultados son difíciles de interpretar debido a que los modelos actúan como cajas negras. Esto puede hacer que estos modelos pierdan gran parte de su fuerza, sobre todo en el ámbito clínico, donde la interpretabilidad es fundamental para ser aplicados en la práctica real. Por esta razón, el aprendizaje automático intrepretable está en continuo crecimiento.

Existen numerosos problemas clínicos en los que es posible hacer uso de métodos de AA para ayudar al personal sanitario. En concreto, este Trabajo de Fin de Máster se centra en la detección de hipoxia fetal intraparto, ya que es de gran importancia preservar el bienestar de los fetos durante el embarazo y durante el parto para evitar posibles daños.

Para ello, en primer lugar, se han estudiado los patrones más utilizados en el ámbito clínico para detectar sufrimiento fetal. Después, se han estudiado y entrenado tanto modelos interpretables por sí mismos como modelos más complejos para resolver el problema. En concreto, modelos lineales, modelos basados en árboles y en métodos de núcleos. Además, para estos últimos, se han utilizado técnicas de intrepretabilidad externas, como LIME y SHAP, para poder entender su funcionamiento. De esta forma, ha sido posible estudiar cuáles son las características que los modelos utilizan para resolver el problema y analizar si son similares a las utilizadas en el ámbito médico, es decir, si los modelos actuán con sentido clínico.

En este documento se presentan las distintas fases desarrolladas a lo largo del trabajo. A través de la aproximación realizada, se ha visto que es posible dar interpretabilidad a los modelos de AA y entender cómo y por qué el modelo realiza las predicciones. El método propuesto proporciona un primer estudio positivo y los alentadores resultados obtenidos en las tareas de clasificación demuestran el interés y la viabilidad de este enfoque para detectar la hipoxia fetal intraparto a través de este camino.

Palabras clave

Aprendizaje automático interpretable, Interpretabilidad, Sufrimiento fetal, Detección de hipoxia fetal intraparto

Abstract

Nowadays, Machine Learning (ML) has become a widely used tool in different fields due to its great capacity to learn to solve problems automatically and to analyze large amounts of data efficiently. In fact, in recent years, real-world problems have been solved with very good results using ML methods. However, even for experts in the ML field, sometimes their results are difficult to interpret because the models act as black boxes. This can cause these models to lose much of their power, especially in the clinical field, where interpretability is essential to be applied in real-world practice. For this reason, interpretable machine learning is continuously growing.

There are many clinical problems where it is possible to make use of ML methods to help healthcare staff. In particular, this Master Thesis focuses on the detection of intrapartum fetal hypoxia, since it is of great importance to preserve the well-being of fetuses during pregnancy and during delivery to avoid possible damages.

For this purpose, first of all, we have studied the most commonly used patterns in the clinical field to detect fetal distress. Then, we have studied and trained both interpretable models by definition and more complex models to solve the problem. Specifically, linear models, tree-based models and kernel-based models. In addition, for the later ones, external interpretability techniques, such as LIME and SHAP, have been used to learn about their performance. In this way, it has been possible to study which are the features that the models use to solve the problem and to analyze if they are similar to those used in the medical field, that is, if the models act with clinical sense.

This document presents the different phases developed throughout this work. By the approach adopted, it has been shown that it is possible to give interpretability to the ML models and to understand how and why the model makes the predictions. The proposed method provides a first positive study and the encouraging results obtained in the classification tasks demonstrate the interest and feasibility of this approach to detect intrapartum fetal hypoxia by this pathway.

Keywords

Interpretable machine learning, Interpretability, Fetal distress, Intrapartum fetal hypoxia detection

Table of Contents

Lists

List of algorithms

List of codes

List of equations

List of figures

1 Introduction

This first chapter of the master thesis describes the main points of this work. Firstly, the motivation that has led to the development of the project is detailed, as well as the objectives that have been set up. Finally, in the last part, the structure of the document is described.

1.1 Motivation

Machine Learning (ML) has become a very useful tool in recent years in our society due to its ability to learn how to solve problems automatically and to analyze large amounts of data efficiently. This has led to this discipline acquiring a relevant role in healthcare innovation in many applications, such as medical diagnosis assistance in a fast and accurate way or the reduction of diagnosis time [1]. Despite its great benefits in the clinical field, in many cases this technology results in black box models [2] which are difficult to interpret for healthcare staff. For this reason, Interpretable Machine Learning (IML) is currently on the rise, since it attempts to alleviate this drawback.

In particular, this project addresses the clinical problem of detecting intrapartum fetal hypoxia, when the fetus can suffer irreversible damages during the pregnancy and the delivery. Therefore, it is of great importance to try to preserve the well-being of the fetus detecting quickly and accurately the possible fetal distress.

Taken all these facts into account, this work focuses on detecting intrapartum fetal hypoxia through linear ML models, which are interpretable by definition, and also with more complex models in which interpretability is lost. Thanks to different interpretability tools, it will be possible to explain the operation of these ML models experimentally and to relate them to the medical practice.

1.2 Objectives

As mentioned above, the main goal is to detect intrapartum fetal distress with IML in order to explain how the models work and relate them to the real clinical area. To this end, the specific objectives are:

Introducing the clinical fundamentals of the problem Intrapartum fetal hypoxia context is explained, as well as the clinical features which define this pathology and the values that, currently, are assumed to determine that a fetus suffers hypoxia. Some previous works are also reviewed and the necessity of interpretability in this field is shown.

Studying ML models Model reviewing will be performed, covering from classical and easily interpretable models to more complex ones, where interpretability is lost. Particularly, the models studied are Logistic Regression (LR), Decision Tree (DT), Random Forest (RF) and Support Vector Machine (SVM).

Signal preprocessing and correlation feature extraction Interpolate missing values of the signals in the Cardiotocography (CTG) database and then, extract representative features, such as correlation, to obtain a dataset to train the models.

IML on fetal hypoxia detection After knowing the problem and analyzing the techniques, ML models will be built to detect fetal distress through a series of experiments. The performance and reliability of the models will be evaluated and two of the most widely used techniques in IML, Local Interpretable Model-agnostic Explanations (LIME) and SHapley Additive ex-Planations (SHAP), will be applied to understand the more complex models. In this way, it will be possible to provide interpretable knowledge and to test if ML models are based on the same principles as models based on expert knowledge. Thus, we will see whether the operation of ML models is related to the rules used by international guidelines for detecting fetal hypoxia during the delivery.

1.3 Document structure

Following the objectives above, the structure of this work is organized as follows:

Chapter 1 **Introduction** This chapter shows the reasons that motivate the development of this project, the proposed objectives and the structure.

Chapter 2 **Clinical fundamentals** Here a brief description of the basic clinical concepts are explained. Also, some background in the detection of intrapartum fetal hypoxia is given.

Chapter 3 **Interpretable machine learning** Theoretical explanation of the methods used in this work are shown in this chapter. Both simple and complex models are presented, as well as the interpretability tools.

Chapter 4 **Experiments** In this chapter the experiments performed and the steps followed during them are detailed, explaining and analyzing the corresponding results of each model and tool.

Chapter 5 **Conclusions and future work** In this chapter the conclusions obtained are presented and new lines of research are proposed.

2 Clinical fundamentals

This second chapter provides the necessary knowledge to understand and follow the problem tackled in this project. Basic clinical background associated to the intrapartum fetal hypoxia is detailed in order to subsequently relate the operation of the ML methods to these concepts of the real clinical field. In this way, it will later be possible to compare their working to the clinical practise.

2.1 Fetal distress

Fetal distress produces a quickly decompensation of the fetus due to intermittent compression of the umbilical cord or by a progressive reduction of uteroplacental circulation because of continuous Uterine Contractions (UC). When the fetus undergoes hypoxic stress during the delivery, a series of physiological responses occurs to compensate the stress and to prevent hypoxic-ischemic damage. The myocardium of the heart is protected as well as the brain and adrenal glands, i.e., the central and essential organs are protected at the expense of non-essential organs. Generally, the progressive increase in basal Fetal Heart Rate (FHR) is caused by the release of adrenaline and noradrenaline. However, when decompensation occurs, the blood supply through the carotid arteries may be reduced causing acidosis in the brain, which results in loss of FHR variability. This loss of FHR variability is characterized by a progressive reduction in the basal fetal heart rate culminating in terminal bradycardia.

During pregnancy, it is of great importance to preserve the well-being of the fetus despite the possible injuries that may happen. In order to monitor fetal health during pregnancy and delivery, cardiotocography (CTG) is used, which monitors FHR and UC. Therefore, its main purpose is to monitor fetal comfort and allow early detection of fetal distress [3]. The occurrence of unbalanced Deceleration (DCC) and Acceleration (ACC) patterns over the FHR are some of the features of CTG that reflect the physiological compensatory mechanism [4] which, as mentioned above, occurs to prevent hypoxic-ischemic damage.

When the fetus is born, there are some ways to know if the baby has suffered fetal hypoxia. The umbilical cord pH value is the most common pointer used as a classification criterion between healthy fetuses and pathological ones, although it is sometimes combined with the base deficit in the extracellular fluid (BDecf), the partial pressure of carbon dioxide in arterial blood (PCO_2) and the result of the Apgar test (appearance, pulse, grimace, activity, and respiration) [4]. The umbilical cord pH range can take any value between 6.55 (acidic) and 7.50 (basic), where the most acidic values indicate that the fetus has suffered fetal distress and the more basic ones that the fetus has had a healthy delivery.

However, the key is to know if the fetus is suffering from fetal hypoxia before it is born in order to react accordingly. When the fetus is in distress it is essential to recognize certain patterns in the CTG recording with the aim of acting quickly and appropriately to prevent the fetus from hypoxia. Traditionally, four basic signal characteristics are crucial for clinical assessment when interpreting a CTG recording: basal fetal heart rate (baseline), variability of FHR, and ACC and DCC patterns. However, the detection of these basic morphological features is based on an abstract visual inspection that may depend on the clinician personal experience and emotional state [5].

Despite the fact that CTG was at first developed as a screening device to anticipate fetal hypoxia, it is suspected that its positive predictive strength for intrapartum fetal hypoxia is low [6]. Also, as it will be discussed below, international guidelines have been created to characterize blends of features to help to anticipate intrapartum fetal hypoxia. However, sometimes, these guidelines may not detect that the fetus has begun labor already compromised or that the fetus has begun to suffer during the labor, so the false-positive rate of CTG is slightly high [6]. These facts are still under study [7], but the main point is that when fetal distress signs are missed, the fetus may suffer irreversible damages such as chronic hypoxia, arrhythmias, or brain damage, among others, so it is important to improve its detection.

As it can be seen, there are some unsolved downsides in the detection of fetal hypoxia that clearly open the way to ML techniques to help healthcare staff in the detection of this pathology thanks to the ability of this technology to capture patterns in the data.

2.2 FIGO guideline

Currently, there are different international rules or agreements, such as Swedish (SWE)-09 [8], International Federation of Gynecology and Obstetrics (FIGO)-15 [9] and SWE-17 [10], which propose different criteria for deciding, based on CTG records, whether or not a fetus is suffering from fetal hypoxia in real clinical situations. In particular, FIGO rules are internationally accepted by the obstetric community and, currently, they are used by doctors when analyzing CTGs in real time. These rules are based on the analysis of the baseline, DCC and ACC patterns, UC and variability of the FHR signal, described as follows [9]:

Baseline fetal heart rate: Mean of the FHR over a 10 minutes period. It can be classified into three types:

- **Normal:** Value between 110 and 160 beats per minute (bpm).

Tachycardia: Value above 160 bpm for more than 10 minutes.

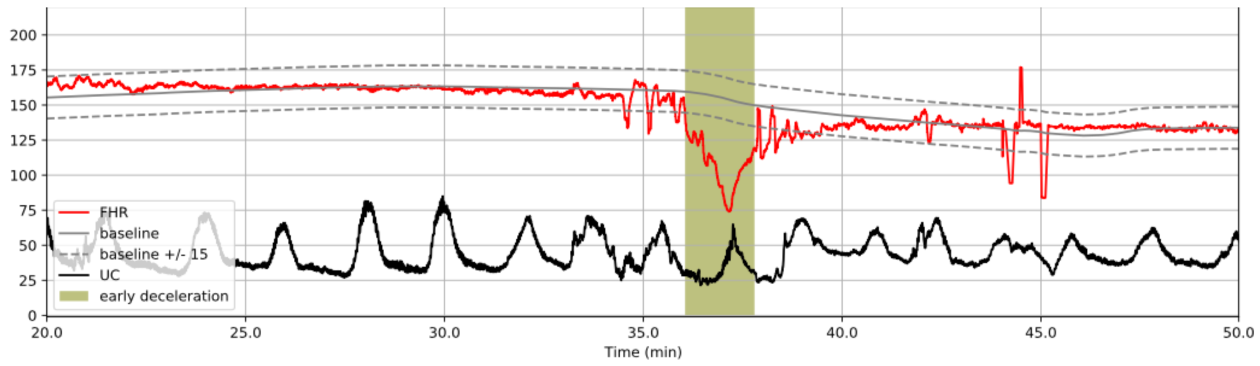
Bradycardia: Value below 110 bpm for more than 10 minutes. Sometimes, in healthy fetuses, values between 90 and 110 bpm may occur and they are not considered pathological.

Contractions: Gradual bell-shaped increases in the signal of uterine activity followed by roughly symmetrical decreases, with a duration of 45–120 seconds. During normal labor, the amplitude of contractions increases from an average of 30 mm Hg (pressure measurement) at the onset of labor to 50–80 mm Hg in the first and second stage.

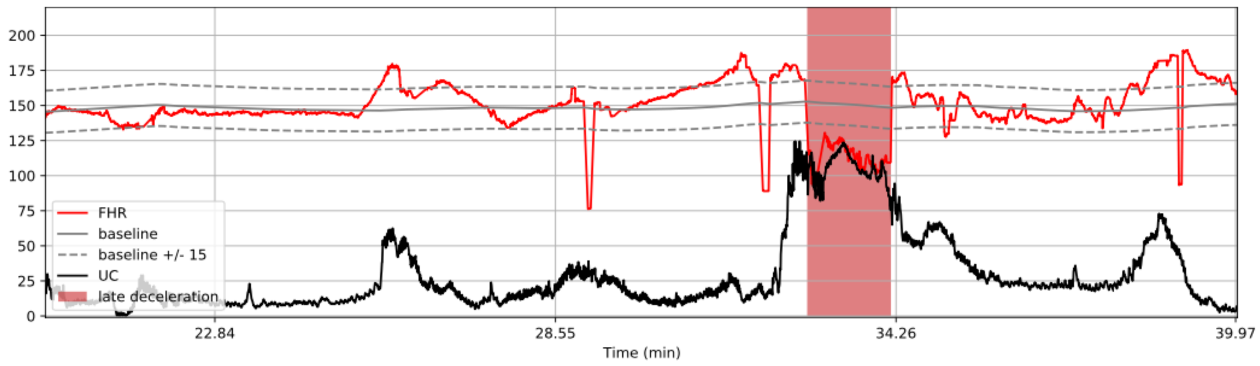
Acceleration (ACC): Abrupt increase in FHR (from onset to peak in less than 30 seconds) of more than 15 bpm amplitude and lasting more than 15 seconds but less than 10 minutes.

Deceleration (DCC): A decrease in FHR below baseline with an amplitude larger than 15 bpm for more than 15 seconds. It can be categorized into four types:

Early: The DCC patterns decrease and return to baseline gradually (within 30 seconds or more). As it can be shown in Figure 2.1(a), the DCC patterns coincide with contractions and preserve variability within the contraction. These patterns are not indicative of hypoxia.



(a) Early deceleration pattern.



(b) Late deceleration pattern.

Figure 2.1: Early and late deceleration patterns in CTG.

Variable: These DCC are 'V' shaped and show a rapid fall (in less than 30 seconds) followed by a rapid retrieval to baseline.

Late: As it can be seen in Figure 2.1(b), it is similar in form to the early DCC, but begins 20 seconds after the onset of contraction. This pattern of correlation between the two signals has been shown to be a clear indicator of intrapartum fetal hypoxia.

Prolonged: Those lasting more than 3 minutes.

Variability: Oscillation of FHR, it corresponds to the bandwidth in a minute segment (the difference between the maximum and the minimum in that minute). Standard Deviation (std) is sometimes used as a proxy summary of the FHR signal. Some of the most common types of variability are the following ones:

Normal: Bandwidth of 5–25 bpm.

Reduced: Bandwidth below 5 bpm for more than 50 minutes at baseline, or more than 3 minutes during DCC.

Increased: Bandwidth exceeding 25 bpm for more than 30 minutes.

Sinusoidal pattern: Smooth and regular undulation with an amplitude of 5–15 bpm and a frequency of 3–5 cycles in 1 minute.

Pseudo-sinusoidal pattern: Pattern similar to the sinusoidal pattern, but with a more angled and beaked shape. Its duration rarely exceeds 30 minutes and is usually preceded and continued with common recording.

Once known the traditional features for the detection of fetal hypoxia, FIGO-15 [9] rules are detailed in Table 2.1. The criteria for the classification of healthy, suspicious and pathological fetuses according to the values of the characteristics seen above can be observed. Therefore, as it can be seen, the most important traditional signs to recognize the pathological fetuses are: low baseline (FHR in bradycardia), changes in variability and DCC patterns. These features are going to be taken into account in order to check later if ML models consider them as important as they are according to the experts.

	Normal CTG	**Suspicious CTG**	**Pathological CTG**
Baseline	$110 - 160$ bpm	Lacking at least one of normal characteristics, but with no pathological features	< 100 bpm
Variability	$5 - 25$ bpm		Reduced/incrceased variability; sinusoidal pattern
Decelerations	No repetitive decelerations		Repetitive, late or prolonged decelerations for > 30 min (or > 20 min if reduced variability); one deceleration > 5 min
Interpretation	No hypoxia/acidosis	Low probability of hypoxia/acidosis	High probability of hypoxia/acidosis

Table 2.1: FIGO-15 classification system [9].

2.3 Related work

In recent years, multiple studies have been done in this field in order to try to detect intrapartum fetal hypoxia, such as [11] where new physiological CTG features are explored with the aim of allowing the clinicians to recognize a fetus that may be suffering from fetal hypoxia.

Moreover, several works have been recently published in the line addressed by this master thesis, such as [5], where an attempt has been made to detect fetal hypoxia by performing a time-frequency analysis based on spectrogram images. In this study it was demonstrated that adding new features such as contrast, correlation, energy or homogeneity allows to improve the performance of classifiers like Least Square Support Vector Machine (LS-SVM). Other more recent works, such as [12], use a new computer-aided diagnostic model where a combination of conventional features, Common Spatial Patterns (CSP) and ML techniques such as Artificial Neural Network (ANN), SVM and k-Nearest Neighbors (k-NN) algorithms were used to solve the problem. Again, adding new features to the conventional set was found to improve the performance of fetal hypoxia detection.

On the other hand, correlation studies between FHR and UC have shown the relationship between them, such as in [13] where dynamic of couplings is studied. This work showed that these signals have a strong link, in particular, a relation between the RR interval of the beats-heart (time between consecutive R waves of the electrocardiogram signal) and UC. Based on this fact and the correlation values obtained, the recordings were grouped with high precision into normal and pathological cases.

Finally, a detail to take into consideration is the umbilical cordon pH value used to separate healthy from pathological fetuses in other studies in this field. There is no general agreement on the threshold, so the separation criterion is at the researcher discretion or depending on the amount of data, among others. In general, when the pH takes values of 7 or lower, the fetus is considered pathological and even with cerebral palsy. Similarly, when the pH is around 7.10 - 7.20 it is considered pathological and, from 7.20 or higher, it is considered a healthy fetus. For this reason, many studies, such as [14] use a separation threshold of 7.15. In [15] can be seen a review of pH threshold criteria comparing different papers.

In summary, in recent years there has been a growing interest in this field and numerous works have been done to introduce new features to detect intrapartum fetal hypoxia and, also, to try to improve the performance of the classifiers as much as possible. However, less attention is paid to the interpretability of these models, i.e. how they work, why they are making these predictions or which features are really important, which is essential for clinicians.

3 Interpretable machine learning

In this third chapter, an explanation of each of the algorithms used in this work is given. First, the importance of interpretability in the ML field is highlighted and the interpretable models by definition used in this project are explained. Next, the complex models used are shown and, finally, two interpretability techniques are illustrated.

3.1 Importance of interpretability

In Machine Learning (ML), two main types of tasks are often distinguished: supervised and unsupervised. The goal of supervised ML is the inference of a function from labeled training data. The two main supervised problems are classification and regression, which result in categorical and continuous outputs, respectively [2]. In this work, supervised ML techniques are going to allow the construction of different models to solve the intrapartum fetal hypoxia problem.

In recent years, ML has assumed an important role in healthcare innovation and this is one of the reasons why IML is on the rise. ML systems developed in medical field must be easily scalable, accurate, robust and stable. These models have proven to be a powerful tools to assist clinicians in their daily clinical routine to produce a more consistent and faster detection [1]. However, many do not provide the physiological basis that allow to understand the results, i.e., their interpretation. On many occasions, the performance of the ML models is excellent, but also they are black boxes where it is impossible to know how the model works. This is not enough to solve real-world tasks and therefore, the interpretability becomes an essential part of the ML development. It is true that sometimes, such as in movie recommendation systems, the interpretation of the models may not be so relevant because they work in a low risk environment where the model decision does not cause serious consequences. However, in most cases and specially in the clinical field, it is necessary to know how the algorithms work to learn more about the problem and about the data [16]. In addition, interpretability allows to know why a prediction was wrong, to know the cause of the error and even to understand how to fix the system if it fails for certain patterns.

When it is possible to interpret a model and know the explanation of its decisions, it is possible to check if it meets the following properties, which are desirable for an ML model as it is explained in [2]:

Fairness: '*Ensuring that predictions are unbiased and do not implicitly or explicitly discriminate against underrepresented groups.*'

Privacy: '*Ensuring that sensitive information in the data is protected.*'

Reliability or Robustness: '*Ensuring that small changes in the input do not lead to large changes in the prediction.*'

Causality: '*Check that only causal relationships are picked up.*'

Trust: '*It is easier to trust a system that explains its decisions compared to a black box.*'

In this way, understanding how the models work, it can be ensured that the ML models developed are robust and, therefore, provide reliable predictions for solving real-world problems. This can facilitate the inclusion of this technology in the daily routine of professionals from different fields.

3.2 Interpretable models

As it has already been mentioned, there are models that are interpretable by definition, i.e. do not require external methods to be able to interpret their operation. This offers great upsides when it comes to understand how a model is solving a given task, which is of high relevance in the medical field. In this section we show two supervised IML models that are easy to interpret and understand: Logistic Regression (LR) and Decision Tree (DT) models.

3.2.1 Logistic Regression models

LR is a linear parametric method used to solve classification problems. This model allows to estimate the label of each sample as a combination of different independent variables or predictors by multiplying each of them by its corresponding weight. These weights must be learned during the training phase.

Thus, the construction of an LR model consists in finding the most appropriate weights of the predictors to generate a boundary to separate the classes in the most efficient way. The output of the LR function is not binary, it is the probability that each sample belongs to a class [17]. In these circumstances it is necessary to set a threshold to define the observation as being of one class or another. In general, this threshold is 0.5 (it can be another), so if the value exceeds this limit, the sample will be of class 1 and otherwise of class 0. The coefficients or weights of the model are found by optimizing the cost function, which allows to obtain a model with the best results:

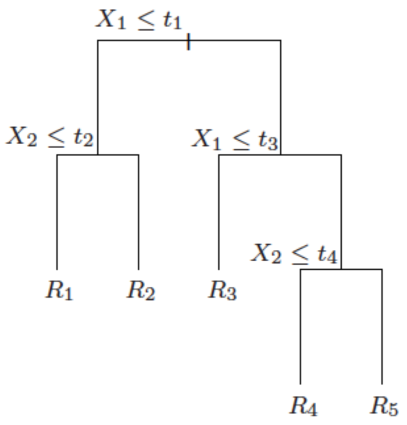
$$\text{minimize} \quad -\sum_{i=1}^{N}\big(y_i\log(\sigma(\mathbf{w}^T x_i)) + (1-y_i)\log(1-\sigma(\mathbf{w}^T x_i))\big) + C\sum_{j=1}^{M}|w_j|, \tag{3.1}$$

where $\mathbf{w} = [w_1, w_2, ..., w_M]$ is the weight vector to be determined and $\sigma(\mathbf{w}^T x_i)$ is the probability output, $\hat{y}_i$. These coefficients describe the size and the direction of the relationship between the predictors and the response variable [18]. As it can be seen in the last term of Equation 3.1, we can add a regularization term in the cost function which prevents the model from being over-fitted, allowing better generalization. In this case, Lasso regularization method has been considered, where C is the regularization coefficient (balance between complexity and accuracy) [18]. This penalty parameter controls the regularization for the error term and the higher this value is, the greater the regularization of the cost function. Thus, an optimal value of it should be sought for the model.

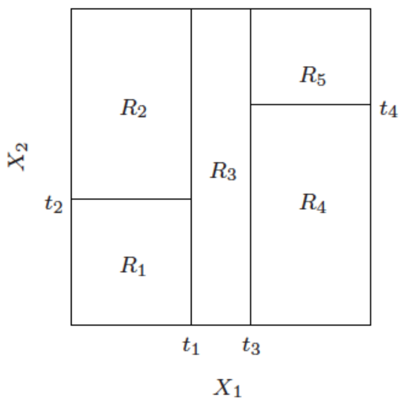
As mentioned above, it should be noted that this algorithm will allow us to estimate the importance of the features directly through the value of the weights associated with each of the variables. Therefore, this algorithm has a great advantage in relation to other ML methods, which is really relevant in the medical practise.

3.2.2 Decision Tree models

DT model is a nonlinear and nonparametric supervised ML algorithm, which consists in making successive decisions to finally predict the class of the observations [19]. Simple decision rules learned from training observations are used to predict the value of the output variable. In each decision, this algorithm separates the set of instances through a condition given by a certain threshold of a particular variable, which can be represented as a hierarchical tree, as it is shown in Figure 3.1(a). Therefore, tree-based methods allow to divide the space into rectangles, as it is present in Figure 3.1(b), where it is easier to observe the interactions between groups of samples and the predicted class.



(a) Decision tree outline.



(b) Space separation from decision tree algorithm.

Figure 3.1: Outline of operation of the DT algorithm obtained from [17].

The main elements that form a decision tree are: root node, branches, intermediate nodes and terminal or leaf nodes. The process of constructing a decision tree begins at the root node with the whole dataset. First, the feature that provides the best purity index (i.e. high probability of obtaining two samples of the same class in a node) is selected, which is going to take a role of great importance as it is the first feature to split all the data into two subsets with a greater number of observations of one class or another. In other words, to divide the root node into two intermediate homogeneous nodes, the predictor variable that provides the highest purity index for each descendant node will be selected. At each split, smaller and smaller sets of cases are obtained until, for example, the maximum tree depth established or the minimum number of samples to split a node is reached. When the tree stops splitting, the leaf or terminal nodes are reached, where the final class to which each set of observations belongs is indicated.

Homogeneity or purity of the node can be measure with different criteria, such as the Gini index, the cross-entropy or the classification error, but in this work the Gini index will be used:

$$G_n = \sum_{k=1}^{K} \widehat{p}_{nk}(1 - \widehat{p}_{nk}),$$

where k are the possible values that the label can take and $\widehat{p}_{nk}$ is the proportion of observations within the node n that belong to the class k. When $\widehat{p}_{nk}$ have values close to 0 or 1, it means that the node will contain samples mostly of a single class. Therefore, the lower the value of the Gini index (G_n), the higher the purity of the node. Moreover, Gini index evaluates the frequency with which a randomly chosen observation from the samples of a node would have been mislabeled by that n node [19].

One of the main advantages of this model is its easy interpretability and similarity with the mental process performed by a clinical specialist to make decisions. In addition, this algorithm requires little data preparation and provides great information about the importance of the variables, since the closer the variables are to the root node, the more discriminating they are. However, there are also some clearly downsides that are necessary to take into account when using this model, such as the lack of softness in the predictions, its high variance or its tendency to overfitting, among others [17]. For this reason, it will be necessary to carefully select different hyperparameters of the model, such as the maximum depth of the tree and the minimum number of samples to split a node, described as follows:

Maximum depth: maximum number of divisions that a single branch can undergo.

Minimum number of samples: the node will only be split if at least the minimum number of samples remains in both branches.

The lower the maximum depth and the higher the minimum number of samples, the smaller the tree size. In both cases, the generalization and the interpretability of the model is increased, although it implies a lower hit rate.

3.3 Complex models

Sometimes, simple and interpretable models fall short to solve real-world problems, and so more complicated ones must be used to solve these tasks. This section shows Random Forest (RF) and Support Vector Machine (SVM) algorithms, which cannot be interpreted by themselves, so it will be necessary to use external tools to understand how they work.

3.3.1 Random Forest

Decision trees allow to capture complex relationships between data and have a low bias error. In contrast, they generally tend to have a very high variance error. However, the variance could be reduced by creating different decision trees (forming a forest) and then averaging them, as it can be seen in Figure 3.2. It can be observed that the final prediction of a sample is obtained from the individual predictions of each decision tree that makes up the forest. Therefore, the main idea of the RF algorithm is based on building a set of decision trees with a very low bias error (most accurate possible predictions) and uncorrelated among them (as diverse as possible) to average them and get a final prediction model with low bias and low variance [20].

As it is said, Random Forest (RF) is a prediction algorithm based on the aggregation of different decision trees, in which each tree is constructed independently using a resample with or without replacement from the initial dataset. In this way, each individual tree makes a prediction and the final result is chosen by majority-voting in the case of classification tasks and averaging in the case of regression ones [18]. In the construction of each tree, p predictors are also randomly selected from the total set of variables. Thus, reducing the features to a random subset, it is achieved that each tree is as different as possible, and, therefore, the correlation between them is as low as possible, obtaining uncorrelated predictors with the aim of reducing overfitting and improving the generalization performance [20].

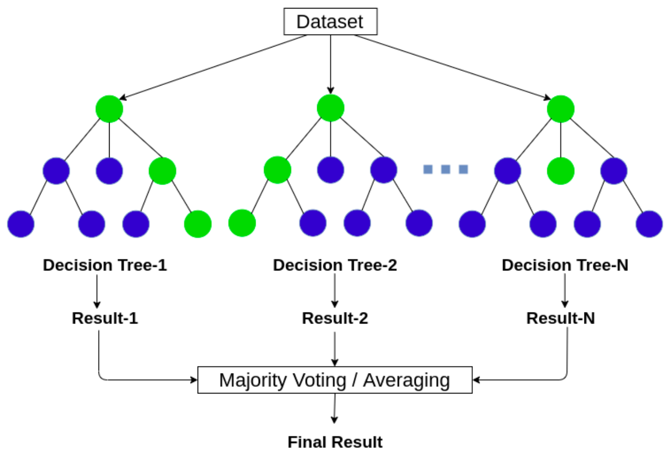


Figure 3.2: Outline of operation of the RF algorithm obtained from [21].

As for the main hyperparameters of the RF algorithm, it has the same as those seen in the previous section of DT, i.e., the minimum number of samples to split a node and the maximum depth of the tree. Also, it is necessary to select the number of decision trees that will form the final model (forest), i.e., the number of estimators. The higher the number of trees created, the higher the computational cost and the lower the interpretability.

One of the most used measures to give interpretability to this model and even to make feature selection is the feature importance. To obtain it, in each split of each tree, the improvement of the splitting criterion is the measure of importance of the splitting variable, and it is accumulated in all the trees of the forest separately for each variable. Finally, it is worth mentioning that, as it will be seen in Chapter 4, there are some specific interpretability tools, such as *proximity plots* that, sometimes, allow to give some interpretability to RF.

3.3.2 Support Vector Machines (SVM)

SVM models have become one of the most prominent supervised learning methods for solving regression, classification or even domain detection problems. It is worth mentioning that this algorithm is able to reduce the overfitting problems present in other classifiers [22].

The idea behind this algorithm for two-class problems is to optimally separate a set of data represented in the space through a hyperplane. To separate two classes of data there can be many possible hyperplanes, but SVM tries to find the one that is at the greatest distance between the data of both classes for maximizing the generalization capability of the model, as it can be seen in Figure 3.3.

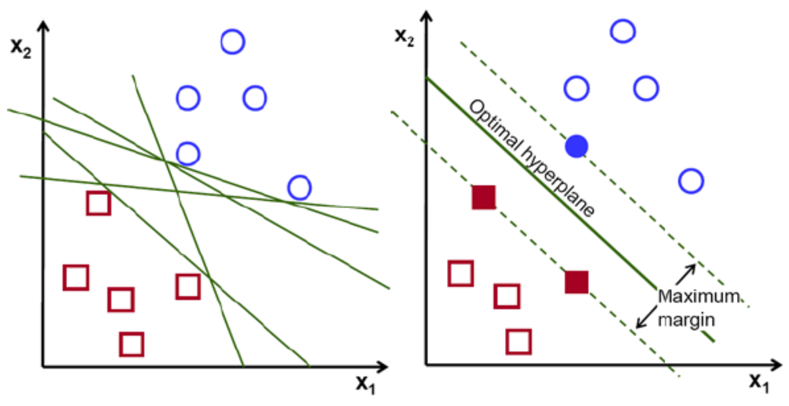


Figure 3.3: Possible separation hyperplanes in a dataset versus the optimal separation hyperplane of the SVM algorithm. Obtained from this online source.

The output of the classifier takes values $y_i \in \{1, -1\}$ depending on the class to which the i-th case belongs and the separation plane is defined as follows:

$$\begin{aligned} \mathbf{w}^T + b &\geq 1 - \zeta_i \quad \text{when } y_i = 1, \\ \mathbf{w}^T + b &\leq 1 + \zeta_i \quad \text{when } y_i = -1, \\ \zeta_i &\geq 0, \end{aligned} \tag{3.2}$$

where the parameter ζ_i allows to introduce some slack (sometimes called loss) in the errors made by the decision maker. The use of this parameter makes the classifier a soft margin classifier and, when no errors are allowed in the classification, the classifier is called a hard margin classifier. The constraints to which the hyperplane's Equation 3.2 is subject can be combined as follows:

$$\begin{aligned} y_i(\mathbf{w}^T + b) &\geq 1 - \zeta_i, \\ \zeta_i &\geq 0. \end{aligned}$$

In the case where the data are linearly separable, the separation boundary is a linear hyperplane. The margin between this hyperplane and any sample from the training set is maximized under the constraint of penalizing the classification error by the following optimization problem:

$$\begin{aligned} \underset{\mathbf{w},b,\zeta}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\zeta_i, \\ \text{subject to} \quad & y_i(\mathbf{w}^T + b) \geq 1 - \zeta_i, \\ & \zeta_i \geq 0, \end{aligned}$$

where C is a regularization parameter to be selected by the user in order to increase o decrease in a controlled way the generalization of the classifier [17]. Among the whole dataset, this algorithm only considers important some samples to build the classifier and maximize the margin, the ones known as Support Vectors (SV). This is an advantage when we want to store the models because the whole training set is not needed.

Due to the limitations of linear learning machines, if the dataset is non-linearly separable, the use of kernel functions is necessary to provide a solution to this problem. Thanks to the nonlinear transformation performed by these kernel functions, the data is projected into a higher dimensional space where the separation of the samples is facilitated by the design of a linear decision maker in the new feature space. Some popular kernel functions are the polynomial or Gaussian and, in order to have a good performance, in the case of the Gaussian kernel, it is necessary to choose the kernel width properly.

As mentioned above, this method can be used in regression approaches. The main differences are that in regression tasks the hyperplane is going to be the line that will help us to predict the continuous output value and, secondly, that the error is going to be fitted within a certain tolerance, epsilon (ϵ), which means the width insensitivity of the boundary tube.

3.4 Model-agnostic interpretable methods

As it has already seen, one way to understand the performance of the models is to use only interpretable models, such as LR or DT. However, they usually have the main disadvantage of losing predictive performance compared to more complex ML models [2]. For this reason, general interpretable techniques have been developed.

Model-agnostic interpretable methods allow to give generalized explanations of ML model predictions, i.e., they have great advantages in terms of model, explanations and representation flexibility. Through these techniques, researchers are free to employ any ML model regardless of whether they are complex and non-interpretable. Usually, when it comes to solving a problem in the real world, not only one algorithm is evaluated, which makes these tools play an important role due to their ability to explain different models independently. Finally, it should be noted that these techniques are not limited to a particular form of explanation, which allows the type of interpretation to be adapted to each model, i.e., in some tasks a linear explanation may be useful, and in others a graph of feature importance is more descriptive [2]. Therefore, the way these algorithms work allows us to explain the predictions of one particular model.

3.4.1 Local Interpretable Model-agnostic Explanations (LIME)

The main objective of LIME is to use a locally interpretable model, which is faithful to the original classifier, in order to explain individual instances [23]. Thus, particular explanations can be obtained and the reasons for the predictions can be understood, which is important for assessing the confidence of the model.

The original model is a black box, but individual cases can be studied by locally applying an auxiliary linear and interpretable model around it, as can be seen in Figure 3.4. In this way, LIME can explain the predictions of any classifier or regressor reliably. The decision function of the nonlinear model, f, is shown with the blue-pink background and the red bold cross is the specific case to be explained, x [23]. To apply the local linear model, g, we take the neighboring cases of x and weight them by their proximity, π_x. With this set of samples and their predictions obtained through their evaluation with f function, we optimize:

$$\zeta(x) = L(f, g, \pi_x) + \Omega(g). \tag{3.3}$$

We locally approximate a complex model, f, around a case, x, and its neighbors using an interpretable model $g \in G$, where G is a class of potentially interpretable models, such as those seen in Section 3.4. Thus, $L(f, g, \pi_x)$ is a measure of how unfaithful g is in approximating f at the location defined by π_x. The linear model is fitted to this region, but not necessarily globally. The second part of

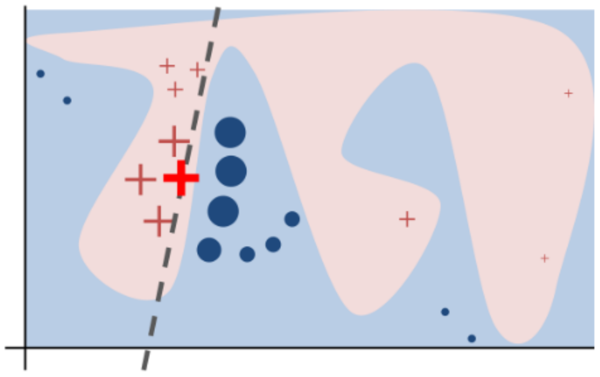


Figure 3.4: Drawing of LIME operation obtained from [23]. The blue-pink background represents the nonlinear model (f), the bold red cross is the specific case to be explained (x), and the dashed black line corresponds to the local linear interpretable model trained over the instance x and its neighbors (its locality is represented by the size of the symbols, the closer the bigger).

the Equation 3.3, $\Omega(g)$, is the complexity factor of the explainable model $g \in G$. For example, in trees it can be the depth of the tree and in linear models the number of non-zero weights. It can be constant or variable, but it is important to constrain it. Therefore, the goal is minimizing $L(f, g, \pi_x)$ and, at the same time, making $\Omega(g)$ low enough to be interpretable.

In summary, LIME has great advantages as it allows to evaluate the confidence of the model and to interpret the predictions on individual cases, which resembles medical diagnosis in the real world. This tool can be used for different applications, such as selecting the features that the model considers relevant or being able to choose a model among a set of models with very similar metrics depending on which features are used to make the predictions. Thanks to the explanations provided by LIME, it is often possible for the medical staff to select a model with slightly worse metrics, but using features that are important in the clinical field. Without LIME this would not be possible and perhaps a model with better performance, but less clinical sense, would be selected. However, if the model is highly nonlinear LIME may not be able to explain individual predictions well. Furthermore, another drawback of LIME is that features that are locally relevant may not necessarily be globally relevant.

3.4.2 SHapley Additive exPlanations (SHAP)

As LIME, SHAP was created to alleviate the interpretability drawback in black box models. The main goal of this method is to try to maintain a balance between the interpretability of the models and their performance [24].

SHAP allows us to explain the predictions by means of explanations based on game theory and SHAP values. It is necessary to know the concepts of coalition game and SHAP values to understand how SHAP works. A coalition game is defined by the number of players and v, which is a function that assigns a reward to each combination of players [25]. SHAP values indicate what the contribution of

Combination	**Alice (A)**	**Bob (B)**	**Charlie (C)**
A, B, C	$v(A) - v(\varnothing)$	$v(A,B) - v(A)$	$v(A,B,C) - v(A,B)$
A, C, B	$v(A) - v(\varnothing)$	$v(A,C,B) - v(A,C)$	$v(A,C) - v(A)$
B, A, C	$v(B,A) - v(B)$	$v(B) - v(\varnothing)$	$v(B,A,C) - v(B,A)$
B, C, A	$v(B,C,A) - v(B,C)$	$v(B) - v(\varnothing)$	$v(B,C) - v(B)$
C, A, B	$v(C,A) - v(C)$	$v(C,A,B) - v(C,A)$	$v(C) - v(\varnothing)$
C, B, A	$v(C,B,A) - v(C,B)$	$v(C,B) - v(C)$	$v(C) - v(\varnothing)$
Mean	Final SHAP value of A (ϕ_A)	Final SHAP value of B (ϕ_B)	Final SHAP value of C (ϕ_C)

Table 3.1: Toy example to explain the calculation of SHAP values.

each player is in a particular process or application. A toy example might be used to see how much each person has contributed to one project, as shown in Table 3.1. To obtain the SHAP values it is necessary to calculate first the marginal contribution of each participant in each of the different situations. For example, in the first line of Table 3.1 the result of the project has been A,B,C. If we want to know the marginal contribution of B in this result, we will need to know first what is the result of only A working on the project, $v(A)$, and then we have to add B to see what is the result when both work together, $v(A, B)$. The difference between $v(A)$ and $v(A, B)$ is the contribution of B. In this way, the contribution of each participant in each case would be calculated. Finally, the average of the marginal contributions of each participant in each situation is calculated and these means will be the final SHAP value of each participant, which will indicate globally how each participant has contributed. In another example, working with a database with different patterns defined by some variables or predictors, it is possible to calculate how much each variable contributes to the predictions of the model. The contribution of each variable differs depending on how many predictors are included and in what order. It can be calculated for each variable as many times as the number of patterns, but, finally, they are averaged to calculate the overall contribution of each variable.

Therefore, the SHAP values allow us to calculate the effect of each predictor individually on the output target, which makes SHAP an additive feature method [24]:

$$g(z') = \phi_0 + \sum_{i=1}^{N} \phi_i z'_i,$$

where ϕ_0 indicates the average prediction of the model if no feature is introduced, N is the number of features considered, ϕ_i are the SHAP values and $z' \in \{0,1\}^N$ is the coalition vector, which means the presence or absence of a particular feature [24].

Each SHAP value, ϕ_i, corresponds to the importance of feature i in those predictions containing that feature. Therefore, when calculating the SHAP values in a given model, M, it is necessary to evaluate it in all feature subsets, $S \subseteq M_i$, in the presence and absence of i and compare the results:

$$\phi_i(M) = \sum_{S \subseteq N_i \setminus i} \frac{|S|!(N_i - |S| - 1)!}{N_i!}(M(S \cup i) - M(S)), \tag{3.4}$$

where the average of the values of all possible subsets is performed in order to obtain the SHAP value for that feature. In Equation 3.4, $M(S \cup i) - M(S)$ refers to the difference in the model results in the presence or absence of a particular variable.

Finally, it should be emphasized that this method is still under development for ML applications and in spite of its documentation is still incomplete [26], its interpretation is so meaningful because it offers many ways to explain the operation of a model.

4 Experiments

This chapter presents the experiments carried out and the results obtained. Beforehand, the used database is described and once the data is known, a correlation study has been done. For the signal preprocessing and the experiments, Python programming language has been used. In the experiments, the interpretability of the models is examined to see if information about the clinical problem is obtained. First, simple interpretable models are proven and, then, more complex ones. Finally, different interpretable techniques are used in order to understand how these models work.

4.1 CTG Database and preprocessing

Intrapartum CTG database from [15] has been used in this work. This database was collected at the obstetrics ward of the University Hospital in Brno, Czech Republic (April 2010 - August 2012). It consists of 552 intrapartum recordings which were stocked in electronic format, at a frequency sample of 4 Hz, and were chosen from 9164 intrapartum recordings with clinical and technical contemplations. All recordings consist of two signals: FHR and UC, which are at least 30 minutes in length and at most one hour and a half. Most of the records (everyone except 46 cesarean sections) are, intentionally, from vaginal deliveries. All of them have accessible biochemical markers, and also some more general clinical features like the mother age or the type of pregnancy, among others. Full description of the dataset and the criteria to select the recordings are presented in [15].

In Figure 4.1 it can be seen an example from the CTG database, in particular the FHR and UC from subject 1014. As it can be noticed, there are several missing values, marked as zeros because this value is not a possible value for FHR in this context. That is why it has been necessary to implement an interpolation routine, which, in this case, consists of a cubic spline interpolation. In particular, `PchipInterpolator` [1] from `scipy` library has been selected to create the interpolation routine applied to all the recordings. This tool uses monotonic cubic splines to find the value of new points [27] and it has been selected due to the excellent fitting to the points and, also, because its calculation is not excessively complex.

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.PchipInterpolator.html

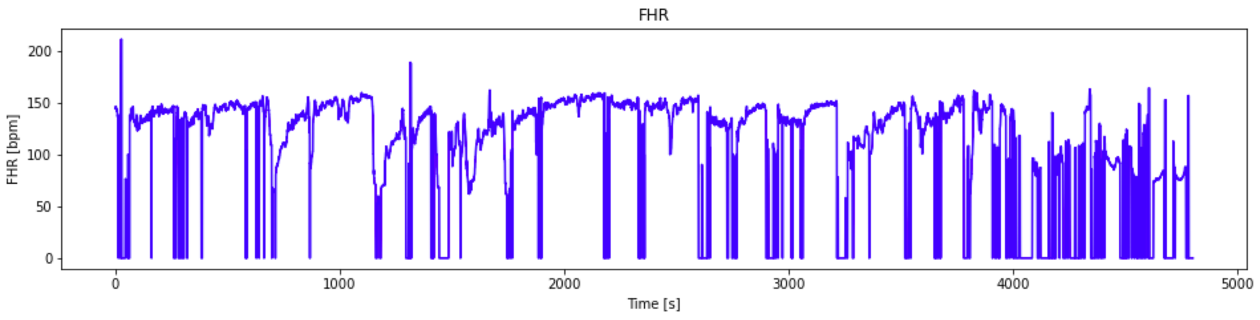


Figure 4.1: Original FHR and UC signals of subject 1014 from CTG database.

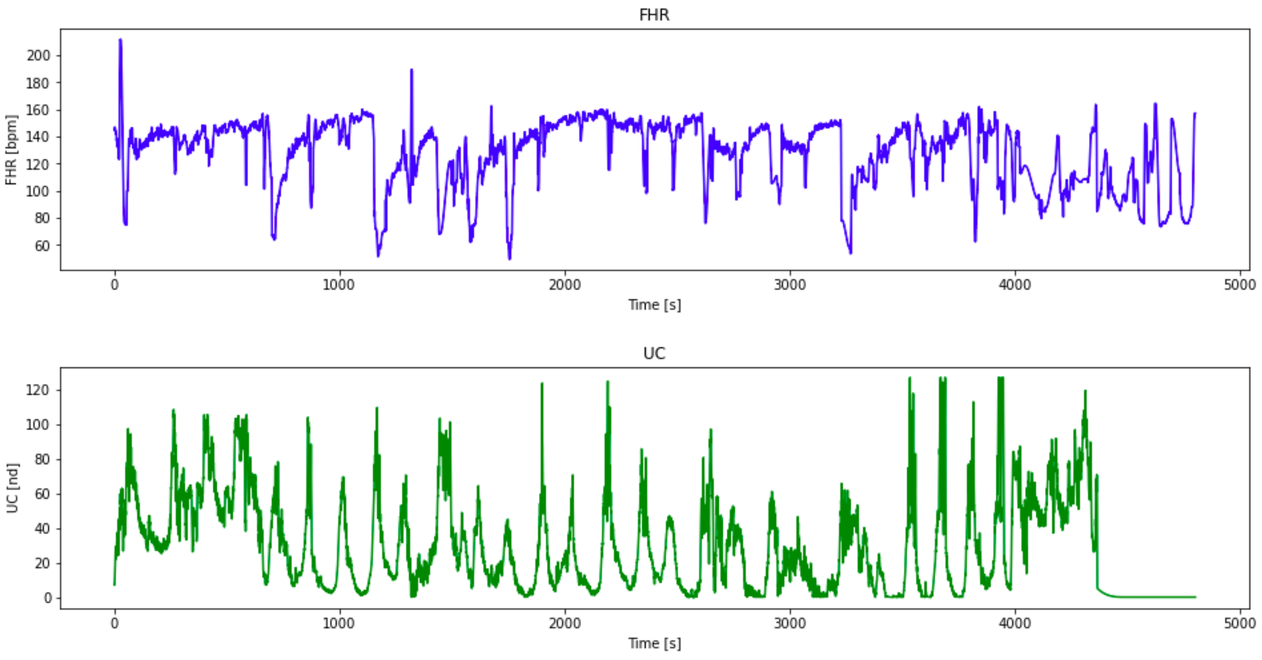


Figure 4.2: Interpolated FHR and UC signals of subject 1014 from CTG database.

In spite of obtaining a good interpolation in general, as it can be seen in Figure 4.2, no interpolation was performed at the missing ends, because it does not returns values within the common range of FHR and UC. In the case of missing values at the extremes, this part of the record has been deleted because there is not significant information in it.

4.2 Methodology

In the development of this project, information from FHR and UC signals is going to be used in order to extract features that allow to detect the intrapartum fetal hypoxia. Also, to split the data in two classes, only one biochemical marker is going to be the key to label the fetuses, the umbilical artery pH values. In this work, the selected separation threshold is 7.20, so when the pH value is lower or equal to 7.20 the subject is considered as pathological and when it is higher the subject is considered healthy.

As mentioned above, all code has been developed in Python programming language due to its adaptability, and its libraries suited for ML, such as `scikit-learn`. In this section, firstly, the feature extraction from the signals is explained and secondly, the model implementation details are shown.

4.2.1 Feature extraction

In this section, it is shown how to process the interpolated signals seen in Section 4.1 in order to extract informative features to train the models.

First, a traditional set of features has been extracted to describe the FHR signal in a powerful way. As Table 4.1 shows, this set consists of 3 morphological features and 4 linear time-domain ones. The morphological features selected are the baseline, and the number of acceleration (ACC) and deceleration (DCC) patterns, and the linear characteristics are the mean and the std of the signal, the Short Term Variability (STV) and the Long Term Variability (LTV), which are usually some of the main predictors to determine fetal well-being in clinical practice, as it has been shown in Chapter 2. To extract this set of features, we followed the procedure shown in [5] which is inspired by FIGO guidelines [9].

However, these features formed a little dataset to train the models. For this reason, as it will be seen below, more features are included in the training set. Based on FIGO criterion [9], correlation between FHR and UC has been demonstrated to be a significant characteristic to suspect fetal hypoxia in the clinical field. Correlation is the linear relationship between two variables or sequences and gives information about the similarity between them [28].

That is why, additionally to traditional features from [5], this characteristic has been extracted in this project. As it can be seen in Figure 4.3, the correlation obtained between FHR and UC signals is normalized, where 1 and -1 indicate that there is high correlation (positive or negative, respectively)

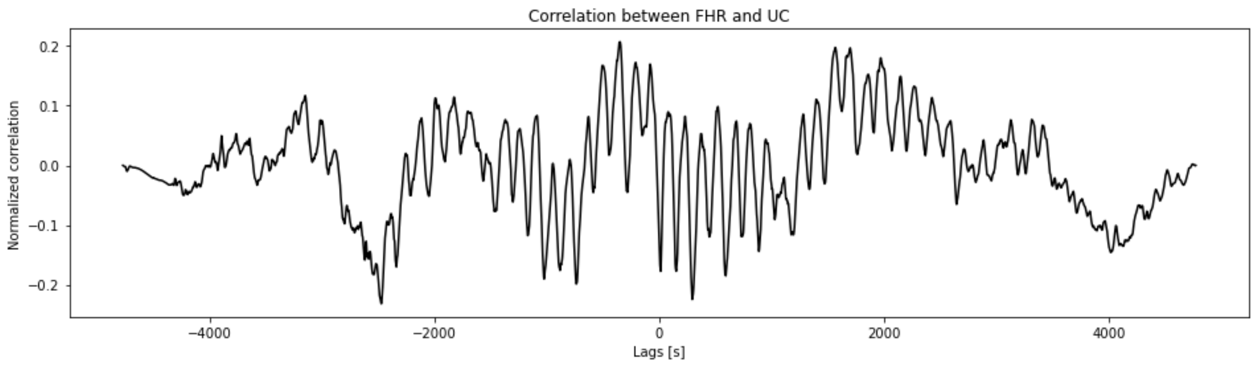


Figure 4.3: Normalized correlation between FHR and UC of subject 1014.

Extended set	**Traditional set**	**Morphological features**	**baseline**	Mean of FHR values within the band: mean $\pm$ (std + $\frac{\text{std}}{\sqrt{\text{nº observations}}}$)
			ACC	Number of times that the signal is above the baseline more than 15 bpm for more than 15 seconds
			DCC	Number of times that the signal is below the baseline more than 15 bpm for more than 15 seconds
		Linear time-domain features	**mean**	Arithmetic average of the whole FHR signal
			std	Standard deviation of the whole FHR signal
			STV	Sum of the difference between '*average values of 2.5-s blocks in the FHR signal*' [5] divided by the total number of minutes of the signal
			LTV	'*Sum of the difference between the maximum and minimum values of the one-minute blocks of the FHR signal*' [5] divided by the total number of minutes of the signal
		Correlation features	**Maximum correlation**	Value of maximum correlation between FHR and UC
			Lag of maximum correlation	Time frame in which the maximum correlation value occurs
			Minimum correlation	Value of minimum correlation between FHR and UC
			Lag of minimum correlation	Time frame in which the minimum correlation value occurs

Table 4.1: Sets of features.

between the signals and 0 indicates that the signals are uncorrelated. As it can be observed in this example, generally the correlation reaches 20%-30% in some sections and, although it may not seem like much, since it is a real and complex problem, this data provides a great deal of information. However, the whole correlation curve has not been used because it is difficult to handle with the complete signal, so only 4 characteristics have been selected from it: the maximum and minimum correlation and the time lags at which each of them occurs. In summary, to train the models, 3 morphological and 4 linear time-domain features from [5] and these 4 extracted correlation features are going to be used, as they are described in Table 4.1.

4.2.2 Model definition

For each algorithm seen in Chapter 3, a two-step pipeline has been built: standardization of the data and model specification. For the first one, the data has been standardized with mean (μ) 0 and deviation (σ) 1, and for the second step, `scikit-learn` tools are been used.

In Code A.1 it is possible to consult the implementation followed to build the models. In each of them, as external cross-validation, results of 10 models have been averaged. To do this, a loop was performed and for each iteration a different random partition is generated. These partitions have been done with 75% of the data for training and the remaining 25% for testing purposes. In this way, the differences between the scores are more robust and make them depend as little as possible on the patterns intended for testing. For obtaining the best classification model with the optimal set of hyperparameters, an internal 5-fold cross-validation was performed in a stratified way. The scoring used in the parameter search is the balanced accuracy, but later, the Area Under the ROC Curve (ROC-AUC) score is also obtained to compare the models. This later score represents the behavior of the sensitivity and specificity of the model and allows to evaluate the ability of the model to classify between the two classes. The closer this value is to 1, the less error the model will have in classifying. As it has seen in Sections 3.2 and 3.3, the compared models in this project are Logistic Regression (LR), Decision Tree (DT), Random Forest (RF) and Support Vector Machine (SVM). The search grids of the different parameters for each algorithm can be seen in Table 4.2.

After building and training the models, two of the most commonly used techniques in the ML interpretation models, LIME and SHAP, will be applied to provide interpretable knowledge about them. Something to keep in mind is that, as seen in Section 3.4, these interpretability techniques are used to explain one particular model. Therefore, the model with the intermediate score was selected among the 10 built models to be interpreted.

Finally, as it will be seen in the results, the main score obtained to evaluate and compare the performance of each algorithm is the ROC-AUC. It should be remarked that the main goal of this master thesis is not to try to obtain the best possible performance, but to be able to interpret and know the decision rules of each algorithm.

Hyperparameter	Range	Description	Models
C	$C \in \{10^i : i \in [-3, 5]\}$	Regularization parameter for the error term	LR, SVM
Gamma	$\gamma \in \{10^i : i \in [-6, 3]\}$	Kernel width	SVM
Maximum depth	`max_depth` $\in \{10^i : i \in [1, 30]\}$	Maximum tree depth	DT, RF
Minimum samples leaf	`min_samples_leaf` $\in \{10^i : i \in [10, 80]\}$	Minimum number of samples per leaf on the tree	DT, RF

Table 4.2: Search grids of the different parameters for each algorithm.

4.3 Results

Following the previous methodology, the two sets of features shown in Table 4.1 will be used. As it can be seen, the traditional set consists of a total of 7 features (morphological and time-domain) and the second set consists of a total of 11 features (morphological, time-domain and correlation-based). These two sets of features will be compared for the different models indicated in the previous section. Table 4.3 compares the obtained results of the mean ROC-AUC score showing also the deviation (across the 10 models). Although for LR the results worsen a bit when including the correlation features, in general taking into account the correlation improves or ties the results and, as previously mentioned, these characteristics have a strong clinical meaning. Because of these reasons, we have done the majority of our experiments using the extended dataset. In the following sections, we will tackle the interpretation of the results seen in Table 4.3 and we will be able to know for each model which characteristics are more relevant to make decisions and, therefore, to determine the class prediction for each subject.

The implementation of all models has been done with tools from `scikit-learn` and the models predefined by this library, as it can be seen in more detail in Appendix B.

Finally, it is necessary to emphasize that, as it can be seen in Table 4.3, the results are not very satisfactory (it is a low ROC-AUC), so there is a lot of scope for future work to improve these results and try more powerful approaches.

	Traditional set	**Extended set**
Logistic Regression	0.669 ± 0.028	0.657 ± 0.035
Decision Tree	0.621 ± 0.030	0.624 ± 0.041
Random Forest	0.645 ± 0.022	0.663 ± 0.027
SVC	0.657 ± 0.026	0.656 ± 0.027

Table 4.3: General ROC-AUC results of traning models with two different sets of features. Mean over the 10 models and its deviation.

4.3.1 Logistic Regression

In this case, LR model has been built using the ℓ_1 norm (Lasso) of the weights as a regularizer, and to cope with the unbalance of the dataset, the classes are weighted inversely proportionally to their frequency. Moreover, as it is explained in Table 4.2, for LR model we have optimized the regularization parameter, C, in order to get a better performance. Table 4.3 shows that the results obtained with this model were among the best and, as it has seen in Section 3.2.1, one of the great advantages of this model is its easy interpretability due to the fact that it is a linear model and directly provides the coefficients or weights associated to each feature. This makes it easy to decide if the features considered relevant by the model make clinical sense.

Once the model has been built and trained, these coefficients have been extracted, as it can be observed in Figure 4.4. It can be noted that the greatest significance is obtained by the mean, baseline and LTV features (see Table 4.1). It is true that the baseline is an important feature to detect intrapartum fetal hypoxia because when it is low it is a clear sign of fetal distress, but the interpretation of the coefficients of this model falls short of what is needed. In addition, as already suspected by the results obtained from the ROC-AUC score, LR does not consider the correlation characteristics to be relevant, which reduces the clinical sense of this model. Since this is a linear model with regularization, perhaps this could be due to the fact that the hyperparameter search has not been too exhaustive, which will have to be fixed in future works.

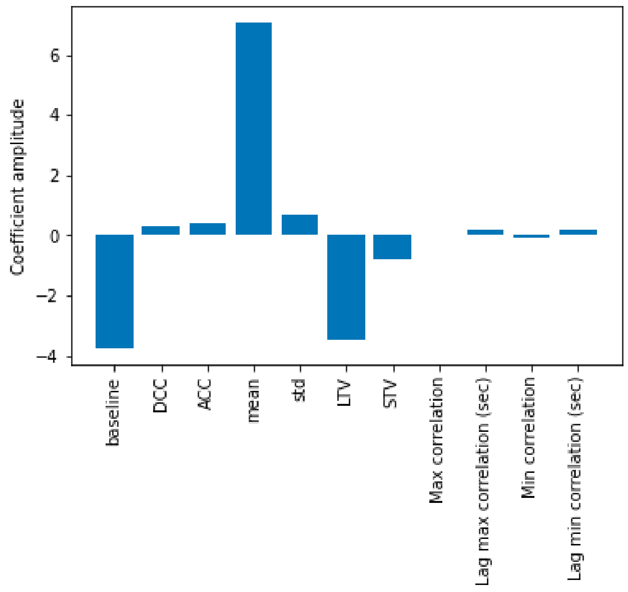


Figure 4.4: Feature importance from Logistic Regression model.

4.3.2 Decision Tree Classifier

For this model, class balancing has also been used and, as it has been seen in Table 4.2, the maximum depth of the tree and the minimum number of samples to split at a node have been optimized.

Models based on decision trees are the simplest models to be interpreted by medical staff because of their logical reasoning, which makes the decisions closely related to the medical reasoning, as it can be seen in Figure 4.5, where one of the resulting trees is depicted. The sample separation criteria used by the model can be seen in each branch, as well as the node impurity. As it can be observed, when the condition is true the branch always goes to the left and when it is false to the right. On the other hand, the impurity can be seen in the *value* list, where the first value refers to the probability that the cases in the node belong to the healthy class and the probability on the right to the pathological one.

In addition, as it can be noted in Figure 4.5, this model considers std and correlation-related features to be relevant. Results from Table 4.3 showed that the correlation features bring significant information to the problem and improved the predictions. As FIGO rules indicate, both the variability of the FHR and the relationship between the decreases in FHR and increases in UC are very relevant when detecting pathological cases, so it makes clinical sense that the features considered by the DT model are determinant.

To sum things up, we have first started with two models, LR and DT, that are directly interpretable and allow to understand how the model makes decisions according to feature weights and Gini impurity criterion, respectively. As it has been seen, the characteristics that these models consider relevant are quite different, so more complex classifiers are going to be studied. As it will be seen below, in order to be able to interpret these following models, it is necessary to use other extra tools, like LIME.

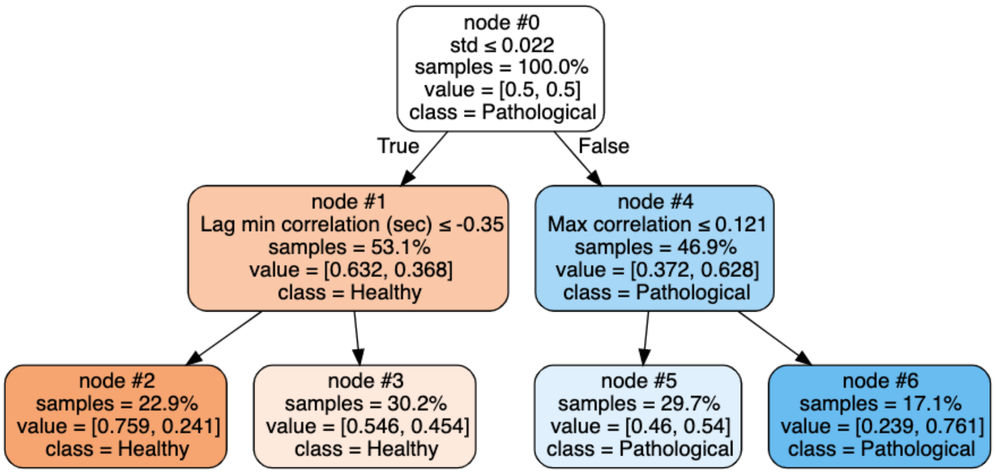


Figure 4.5: Display of a Decision Tree model.

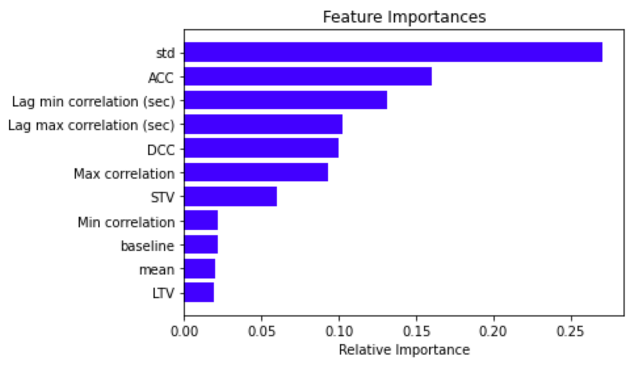
4.3.3 Random Forest Classifier

As the theory explained in Section 3.3.1, this model uses a set of trees to form the forest and, for each of them, the Gini impurity criterion is used to split the nodes. That is why this model is more complex and the majority voting from the 99 trees used in this work complicates its interpretability. As before in DT, the maximum depth of the tree and the minimum number of samples to split at a node have been optimized (see Table 4.2).

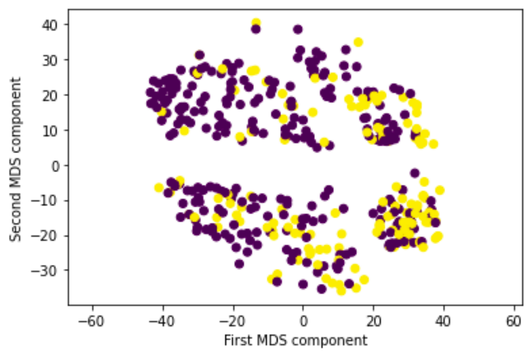
In a first approximation to interpret this model, the feature importance is used. As it was explained in Section 3.3.1, the importance of each variable is calculated as the mean of impurity decrease accumulation in each tree. Figure 4.6(a) shows the feature importance obtained from our trained RF model and, as it is noted, the greatest importance is attached to the std and ACC features, which complies with FIGO standards.

In another attempt to interpret this model, *proximity plot* graph has been used. This graph tries to give an approximation of which observations the model considers to be close together within the RF. Firstly, to obtain the graph it is necessary to perform the proximity matrix between the samples, which is a $N \times N$ matrix, where N is the number of instances, and proximities between them are calculated for each pair of observations in the following way: if two cases occupy the same terminal node in a tree, their proximity increases by one (see Code A.2). Then, it is necessary to apply a dimensionality reduction to two dimensions with multidimensional scaling [29] in order to visualize which cases are close together for the RF classifier [17]. In Figure 4.6(b) it can be observed the *proximity plot* of our trained RF classifier. Cases that are in pure regions are at the extremes, while cases closer to the center are at the decision boundaries, as it is explained in [17]: '*This is not surprising when we consider the construction of the proximity matrices. Neighboring points in pure regions will often end up sharing a bucket, since when a terminal node is pure, it is no longer split by a random forest tree-growing algorithm. On the other hand, pairs of points that are close but belong to different classes will sometimes share a terminal node, but not always.*'. However, it is often difficult to get a good interpretation with this method, especially in real problems where the cases may be very similar, which casts doubt on its usefulness. As it can be seen in Figure 4.6(b), the separation of classes is not straightforward in our case.

Therefore, LIME has been applied in order to understand how the model works on a particular case. Figure 4.7 shows the output layout provided by LIME, where it can be seen how confident the model is to the prediction it has made over this example, what are the decision criteria for each feature (decision thresholds) and which features have encourage one class over another. As it can be observed in Figure 4.7, the most relevant features for this subject are std, Lag min correlation and maximum correlation. Again, just like the reasoning done in the DT model, this indicates that the correlation characteristics add value to the problem and to the performance of the model. In the same way, the reasoning behind this algorithm makes clinical sense.

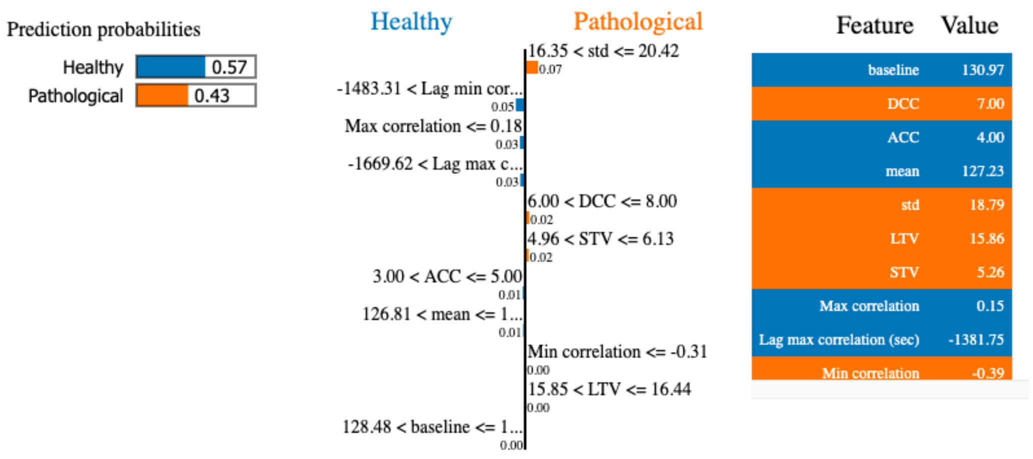


(a) Feature importance.



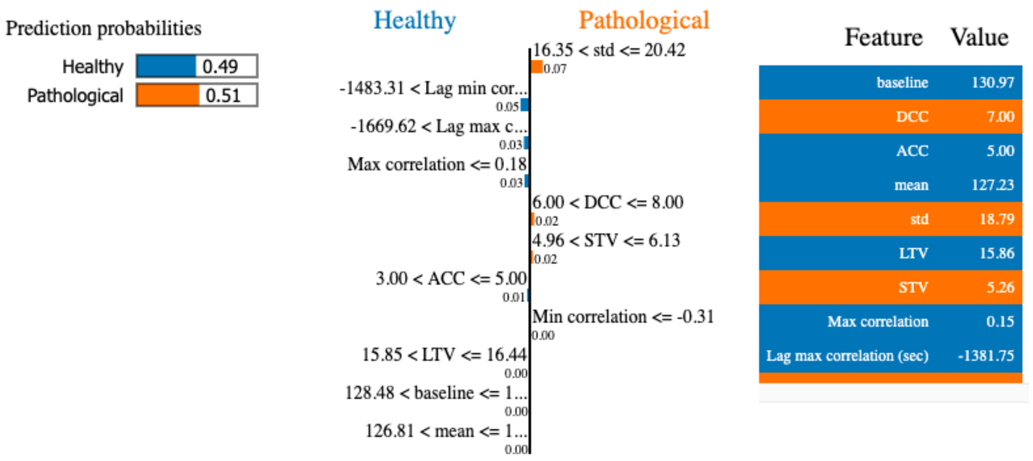
(b) Proximity plot.

Figure 4.6: Feature importance and proximity plot for Random Forest classifier.



Feature	Value
baseline	130.97
DCC	7.00
ACC	4.00
mean	127.23
std	18.79
LTV	15.86
STV	5.26
Max correlation	0.15
Lag max correlation (sec)	-1381.75
Min correlation	-0.39

Figure 4.7: Interpretability of Random Forest classifier with LIME over the extended features set.



Feature	Value
baseline	130.97
DCC	7.00
ACC	5.00
mean	127.23
std	18.79
LTV	15.86
STV	5.26
Max correlation	0.15
Lag max correlation (sec)	-1381.75

Figure 4.8: Linear check of Random Forest classifier with LIME (ACC $= 5$) over the extended features set.

On the other hand, to check whether the model is really using these rules to make the predictions, a linear check has been performed, i.e., we are going to change, for example, the value of the ACC number (abrupt increase in FHR) because according to FIGO the variability of these patterns is relevant for the detection of fetal hypoxia and in the feature importance graph has been shown to be relevant. In this way, as seen in Figure 4.8, we could check how the predictions change. As it can be seen, if there had been one more ACC, the model would have considered this fetus as pathological. This allows to see not only which features the model considers important, but also how the predictions change with small perturbations in the value of the features.

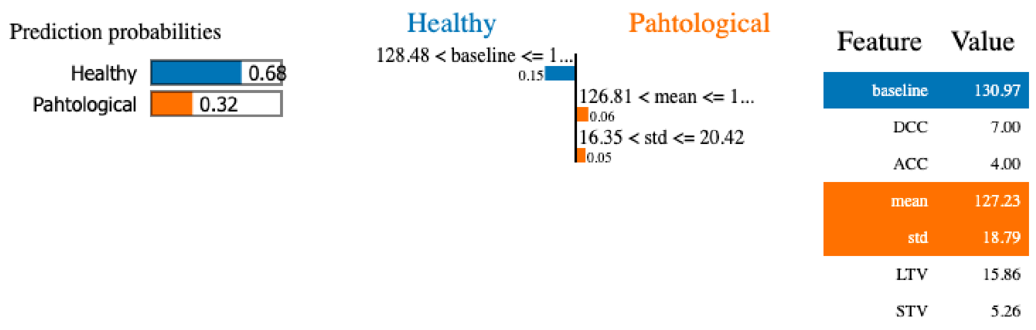
4.3.4 Support Vector Classifier

For this model, as discussed in the explanation of Section 3.3.2, it is important to determine quite well some parameters. In this case, a Gaussian kernel with Radial Basis Function (RBF) has been used where the kernel width, γ, and the regularization parameter, C, have been optimized, as it has been pointed in Table 4.2.

In this model it is not possible to obtain a graph of the feature importance directly so, as for the previous classifier, LIME is used to interpret individual examples of predictions made by this model. In this approach, we wanted to observe not only the importance of the features such as correlation-related ones, which have already been seen to add value to the problem, but also to observe the behavior of the model when working with the traditional feature set (see Table 4.1).

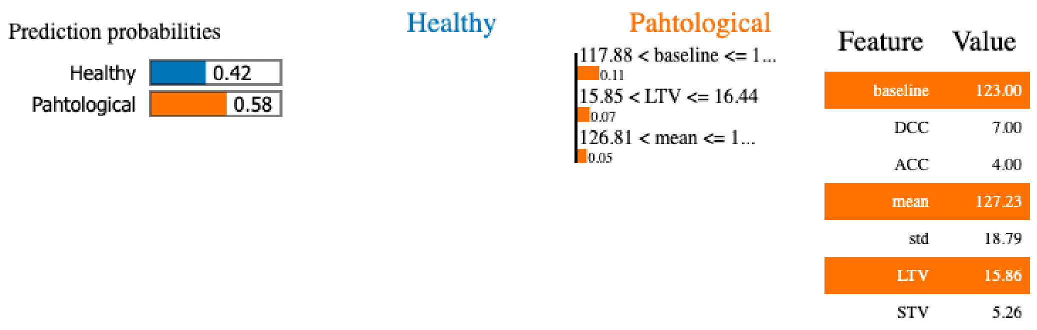
In the example shown in Figure 4.9, we have chosen the same fetus as in RF classifier in order to make comparisons. When the baseline is lowered from 130 to 123, the probability of hypoxia is greatly increased, which is in agreement with FIGO guidelines [9]. Despite this, it could be observed that this model has a strong dependence on the baseline feature when working with the traditional set. When performing the linear check and slightly reducing the value of this feature, the pathological class is excessively triggered, i.e., the model classifies the instances basically according to the value of this feature. Then, as it can be observed in see Figure 4.10, we introduce the correlation-based features over the same subject and we perform the same linear check. When reducing, as before, the baseline from 130 to 123, it is seen that the baseline loses importance and it is necessary to modify also other features, such as the maximum correlation or DCC patterns to change the model output, which makes more clinical sense.

Finally, it is worth mentioning that the characteristics considered relevant by this model are quite different from the features considered by tree-based models, since the baseline or the LTV features were not important and now for Support Vector Classifier (SVC) they are. However, others, such as maximum correlation or DCC patterns, continue to have great relevance, which gives medical sense to the operation of the models.



Feature	Value
baseline	130.97
DCC	7.00
ACC	4.00
mean	127.23
std	18.79
LTV	15.86
STV	5.26

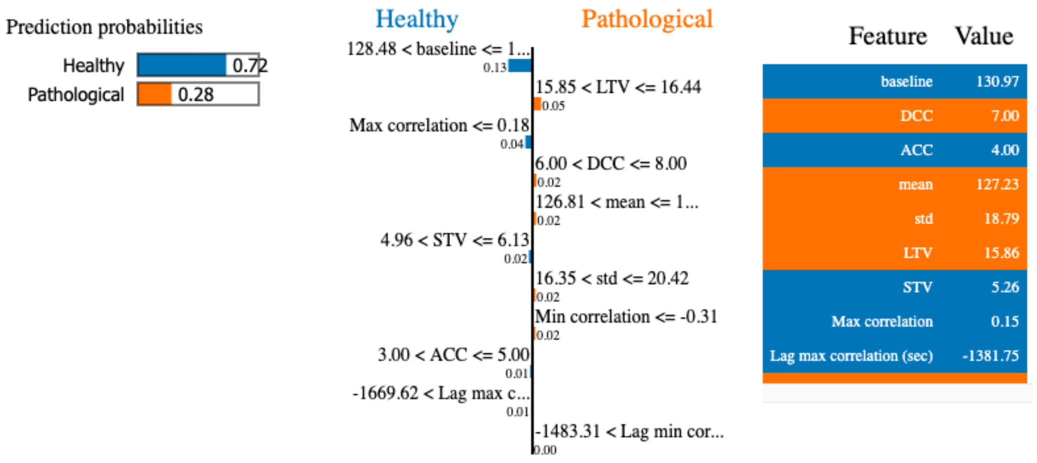
(a) Interpretability of SVC over the traditional feature set.



Feature	Value
baseline	123.00
DCC	7.00
ACC	4.00
mean	127.23
std	18.79
LTV	15.86
STV	5.26

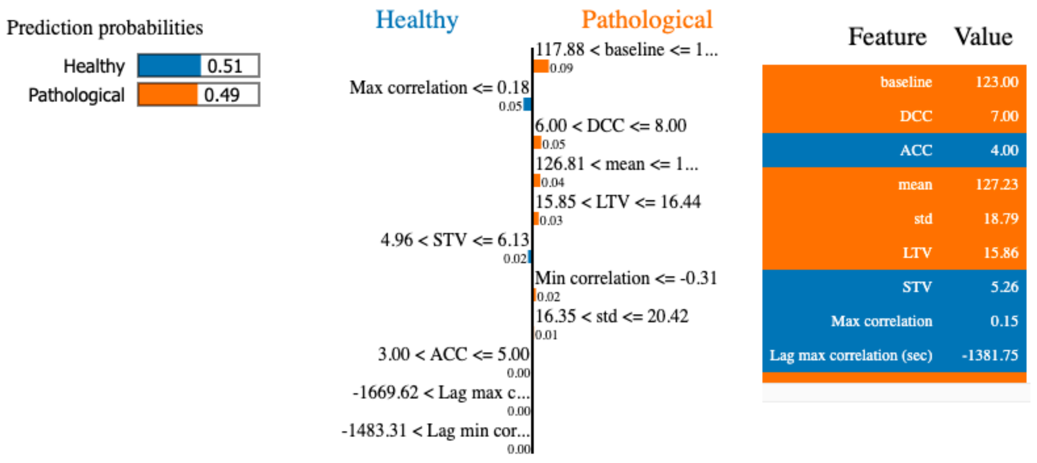
(b) Linear check of SVC (baseline = 123).

Figure 4.9: Interpretability of Support Vector classifier with LIME and linear check over the traditional feature set.



Feature	Value
baseline	130.97
DCC	7.00
ACC	4.00
mean	127.23
std	18.79
LTV	15.86
STV	5.26
Max correlation	0.15
Lag max correlation (sec)	-1381.75

(a) Interpretability of SVC over the the extended feature set.



Feature	Value
baseline	123.00
DCC	7.00
ACC	4.00
mean	127.23
std	18.79
LTV	15.86
STV	5.26
Max correlation	0.15
Lag max correlation (sec)	-1381.75

(b) Linear check of SVC (baseline $= 123$)

Figure 4.10: Interpretability of Support Vector classifier with LIME and linear check over the extended features set.

4.4 First regression trials

A first approximation of regression models interpreted with the SHAP technique has been done following a similar methodology to the one described in Section 4.2.2. The main difference in these experiments is that the model output is the pH value, so subjects are not separated in two classes, instead for each of them the pH value is predicted in a continuous range.

In general, in the first attempts made with RF and SVM regressors, the results obtained have been poor, it seems that the models are not learning well since the problem to be solved is more complex and the database is limited. The outcomes obtained are not yet good enough to conclude anything clearly, but as an example of this type of approach, the analysis of the best models obtained are shown in Appendix C. It would be necessary to extend the work on this approach in more detail in order to obtain better performance.

It is necessary to emphasize that the interpretability provided by SHAP technique on the models is very complete. A detail to take into account is that SHAP is an algorithm that is still under development for ML applications and, as it was seen in Section 3.4.2, it calculates the marginal contribution of each variable taking into account all possible combinations, so it is a slow algorithm. However, despite this, it is already optimized to work with trees quickly and efficiently.

5 Conclusions and future work

This last chapter presents the conclusions reached during this master thesis and the possible future lines of work on the detection of intrapartum fetal hypoxia and its interpretability.

5.1 Conclusions

To date, many contributions have been made in the clinical field with ML, but less attention has been paid to the interpretation of the models, which is essential in this field. In the development of this project, we had intentionally sought to address a real clinical problem, the detection of intrapartum fetal hypoxia, while making model performance visual and interpretable.

First, the fetal distress problem was reviewed with the intention of finding out what were the patterns or signs that currently allow the clinicians to detect it. As a result, it was found that, apart from a set of traditional features from FHR signal, the correlation between FHR and UC signals was one of the most relevant features for the detection of fetal hypoxia. For this reason, it was decided to study it in depth and to extract it in order to introduce correlation-based features in the training of the models.

Then, we studied the importance of the IML and different models interpretable by definition and other more complex ones. By interpreting the operation of the trained models, it was possible to analyze if the features used by the models to decide if, for example, a fetus was healthy were really important or not. It was seen that tree-based models, both DT and RF models, consider important features with clinical sense and very similar to each other. Similarly, another complex model, SVM, also consider relevant some features used in the clinical field, although slightly different from those considered by tree-based models. It is worth mentioning that the interpretability methods used, LIME and SHAP, have proven to be really useful tools to understand the operation of complex models and to test how model predictions change with small perturbations in the predictor variables.

As it is seen throughout this work, the interpretability and the encouraging results obtained in the classification tasks demonstrate the interest and feasibility of this approach to detect intrapartum fetal hypoxia in this way.

5.2 Future Work

In spite of the promising results, there is still great potential for further progress in this research on detection of intrapartum fetal hypoxia.

The approach used in this project to solve the missing values problem present in the signals was to apply a previous interpolation routine. This step allowed later on to extract the correlation between FHR and UC signals. However, it is possible to find in the literature, other ways to handle this problem. For example, it could be performed through a time lag propagation approach. This method consists of applying different time lags steps to the signals, and eliminate the missing values of both signals before calculating the correlation over them at each time lag. This process will be repeated for all the different time lags defined.

On the other hand, with respect to the pH threshold, the same or similar studies could be carried out by changing this threshold to check how the results obtained change and what effect it has on the model's accuracy and on the interpretability. In addition, it would be necessary to try to perform the experiments on a larger database with greater patient variability, since the database used is limited. If it is not possible to work with another database, data augmentation techniques could be considered to increase the number of instances of the database.

Finally, as it has been seen in the development of this work, the priority has not been to build models with high performance and excellent metrics. Therefore, once it has been seen that it is possible to interpret the ML models, with which tools it is possible to do it and how this interpretability can be analyzed, it is necessary to go deeper into improving the performance of the models. It would be desirable to improve the parameter search, increase the number of models and improve their construction to make them more robust and accurate. Once the performance of the classification models is improved, it would be very interesting to re-examine the regression models tested in this work. Similarly, it would be interesting to study more complex models, such as neural networks, and add new features to try to improve fetal distress detection.

Bibliography

[1] S. Yeasmin, “Benefits of artificial intelligence in medicine,” in *2019 2nd International Conference on Computer Applications Information Security*, 2019.

[2] C. Molnar, *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. 2019. Book online source.

[3] A. Sbrollini, A. Agostinelli, L. Burattini, M. Morettini, F. Di Nardo, S. Fioretti, and L. Burattini, “CTG Analyzer: A graphical user interface for cardiotocography,” in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2606–2609, 2017.

[4] D. Hutter, J. Kingdom, and E. Jaeggi, “Causes and mechanisms of intrauterine hypoxia and its impact on the fetal cardiovascular system: A review,” *International Journal of Pediatrics*, vol. 2010, pp. 1–9, 2010.

[5] Z. Cömert, A. F. Kocamaz, and V. Subha, “Prognostic model based on image-based time-frequency features and genetic algorithm for fetal hypoxia assessment,” *Computers in Biology and Medicine*, vol. 99, pp. 85–97, 2018.

[6] A. Pinas and E. Chandraharan, “Continuous cardiotocography during labour: Analysis, classification and management,” *Best Practice & Research Clinical Obstetrics & Gynaecology*, vol. 30, pp. 33–47, 2016.

[7] F. Ekengård, M. Cardell, and A. Herbst, “Impaired validity of the new FIGO and swedish CTG classification templates to identify fetal acidosis in the first stage of labor,” *The Journal of Maternal-Fetal & Neonatal Medicine*, pp. 1–8, 2021.

[8] SFOG, “CTG kort slutversion Sweden: SFOG,” 2009. Online source.

[9] D. Ayres-de Campos, C. Y. Spong, E. Chandraharan, and FIGO Intrapartum Fetal Monitoring Expert Consensus Panel, “FIGO consensus guidelines on intrapartum fetal monitoring: Cardiotocography,” *International Journal of Gynecology & Obstetrics*, vol. 131, no. 1, pp. 13–24, 2015.

[10] DSOG, “Fosterovervågning under fødslen - indikationer Denmark: DSOG,” 2017. Online source.

[11] S. Pereira and E. Chandraharan, “Recognition of chronic hypoxia and pre-existing foetal injury on the cardiotocograph (CTG): Urgent need to think beyond the guidelines,” *Porto Biomedical Journal*, vol. 2, no. 4, pp. 124–129, 2017.

[12] W. Alsaggaf, Z. Cömert, M. Nour, K. Polat, H. Brdesee, and M. Toğaçar, “Predicting fetal hypoxia using common spatial pattern and machine learning from cardiotocography signals,” *Applied Acoustics*, vol. 167, p. 107429, 2020.

[13] A. Pasarica, C. Rotariu, R. G. Bozomitu, and O. D. Eva, “Dynamic of couplings between fetal heart rate and uterine contractions,” in *2015 International Symposium on Signals, Circuits and Systems*, pp. 1–4, 2015.

[14] J. Spilka, V. Chudáček, M. Koucký, L. Lhotská, M. Huptych, P. Janků, G. Georgoulas, and C. Stylios, “Using nonlinear features for fetal heart rate classification,” *Biomedical Signal Processing and Control*, vol. 7, no. 4, pp. 350–357, 2012.

[15] V. Chudáček, J. Spilka, M. Burša, P. Janků, L. Hruban, M. Huptych, and L. Lhotská, “Open access intrapartum CTG database,” *BMC Pregnancy and Childbirth*, vol. 14, no. 1, 2014.

[16] F. Doshi-Velez and B. Kim, “Towards A Rigorous Science of Interpretable Machine Learning,” 2017.

[17] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 ed., 2009.

[18] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, 2006.

[19] S. B. Kotsiantis, “Decision trees: A recent overview,” *Artif. Intell. Rev.*, vol. 39, no. 4, p. 261–283, 2013.

[20] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. Springer, 2013.

[21] TIBCO, “What is a Random Forest? | TIBCO Software.” Online source.

[22] J. L. Rojo-Álvarez, G. Camp-Valls, A. Caamaño-Fernández, and J. Guerrero-Martínez, *ECG Signal Processing, Classification and Interpretation. A Review of Kernel Methods in ECG Signal Classification*. Springer, 2012.

[23] M. Ribeiro, S. Singh, and C. Guestrin, “Why Should I Trust You?: Explaining the Predictions of Any Classifier,” pp. 97–101, 02 2016.

[24] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in *Advances in neural information processing systems*, pp. 4765–4774, 2017.

[25] C.-R. Hsiao and T. Raghavan, “Shapley value for multichoice cooperative games, I,” *Games and economic behavior*, vol. 5, no. 2, pp. 240–256, 1993.

[26] SHAP, “Welcome to the SHapley Additive exPlanations (SHAP) documentation — SHAP latest documentation.” Online source.

[27] R. G. McClarren, “Chapter 10 - Interpolation,” in *Computational Nuclear Engineering and Radiological Science Using Python*, pp. 173–192, Academic Press, 2018.

[28] I. Santamaria, P. Pokharel, and J. Principe, “Generalized correlation function: definition, properties, and application to blind equalization,” *IEEE Transactions on Signal Processing*, vol. 54, no. 6, pp. 2187–2197, 2006.

[29] L. Chen and A. Buja, “Local Multidimensional Scaling for Nonlinear Dimension Reduction, Graph Drawing, and Proximity Analysis,” *Journal of the American Statistical Association*, vol. 104, no. 485, pp. 209–219, 2009.

Acronyms

ACC	Acceleration.
ANN	Artificial Neural Network.
bpm	beats per minute.
CSP	Common Spatial Patterns.
CTG	Cardiotocography.
DCC	Deceleration.
DT	Decision Tree.
FHR	Fetal Heart Rate.
FIGO	International Federation of Gynecology and Obstetrics.
IML	Interpretable Machine Learning.
k-NN	k-Nearest Neighbors.
LIME	Local Interpretable Model-agnostic Explanations.
LR	Logistic Regression.
LS-SVM	Least Square Support Vector Machine.
LTV	Long Term Variability.
ML	Machine Learning.
RBF	Radial Basis Function.
RF	Random Forest.
RFR	Random Forest Regressor.
ROC-AUC	Area Under the ROC Curve.
SHAP	SHapley Additive exPlanations.
std	Standard Deviation.
STV	Short Term Variability.
SV	Support Vectors.
SVC	Support Vector Classifier.
SVM	Support Vector Machine.
SVR	Support Vector Regressor.
SWE	Swedish.
UC	Uterine Contractions.

APPENDICES

A

Code

Code A.1: Model definition.

```python
def model(N_models, X, y, estimator, param_grid): 

    roc_auc = [] 
    acc = [] 

    # Create dictionary to save the best hyperparameters 
    best_param_dict = param_grid.copy() 
    for key in best_param_dict: 
        best_param_dict[key] = [] 

    for iteration in range(N_models): 

        # Internal cross-validation and GridSearch 
        val_int = StratifiedKFold(n_splits=5, random_state=iteration, shuffle=True) 
        cv_estimator = GridSearchCV(estimator, 
                                    param_grid=param_grid, 
                                    cv=val_int, 
                                    scoring='balanced_accuracy', 
                                    return_train_score = True) 

        # Train-test split 
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, stratify=y, random_state=iteration) 

        # Fit and save or load the model 
        print('Model ',str(iteration)) 
        path = base_path + str(iteration) + '.pkl'
        cv_estimator = read_or_new_pickle(path, cv_estimator, X_train, y_train) 

        best_model = cv_estimator.best_estimator_ 
        best_param = cv_estimator.best_params_ 
        for key in best_param_dict: 
            best_param_dict[key].append(best_param[key]) 

        # Predictions 
        predict_proba_pred = best_model.predict_proba(X_test)[:, cv_estimator.classes_==1] 
        predict = best_model.predict(X_test) 
        roc_auc.append(roc_auc_score(y_test, predict_proba_pred)) 
        acc.append(accuracy_score(y_test, predict)) 

    # Choose the intermediate score model 
    rounds = [round(num, 2) for num in roc_auc] 
    idx = rounds.index(statistics.mode(rounds)) 

    # Load again the idx model and its partitions in order to return them 
    # ... 

    return X_train, X_test, y_train, y_test, roc_auc, acc, best_param_dict, best_model, cv_estimator 
```

Code A.2: Proximity matrix function to perform *proximity plot* graph.

```
def proximityMatrix(model, X, normalize=True):

    terminals = model.apply(X)
    nTrees = terminals.shape[1]
    a = terminals[:,0]
    proxMat = 1*np.equal.outer(a, a)

    for i in range(1, nTrees):
        a = terminals[:,i]
        proxMat += 1*np.equal.outer(a, a)

    if normalize:
        proxMat = proxMat / nTrees

return corr
```

B
Functions

The following are the functions used from the `scikit-learn` library to build the models:

- `Pipeline` [1]
- `StandardScaler` [2]
- `GridSearchCV` [3]
- `LogisticRegression` [4].
- `DecisionTreeClassifier` [5]
- `RandomForestClassifier` [6].
- `SVC` [7]
- `Random Forest Regressor (RFR)` [8]
- `Support Vector Regressor (SVR)` [9]

This source [10] has been used for `LIME` development and the functions shown below have been used to obtain the SHAP graphs from Appendix C:

- `shap.plots.force` [11] to obtain individual interpretation of the instances.
- `TreeExplainer` [12] for SHAP interpretability.
- `KernelExplainer` [13]

[1] https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html
[2] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
[3] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
[4] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[5] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
[6] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[7] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[8] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
[9] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html
[10] https://github.com/marcotcr/lime
[11] https://shap.readthedocs.io/en/latest/generated/shap.plots.force.html
[12] https://shap-lrjball.readthedocs.io/en/docs_update/generated/shap.TreeExplainer.html
[13] https://shap-lrjball.readthedocs.io/en/latest/generated/shap.KernelExplainer.html

C

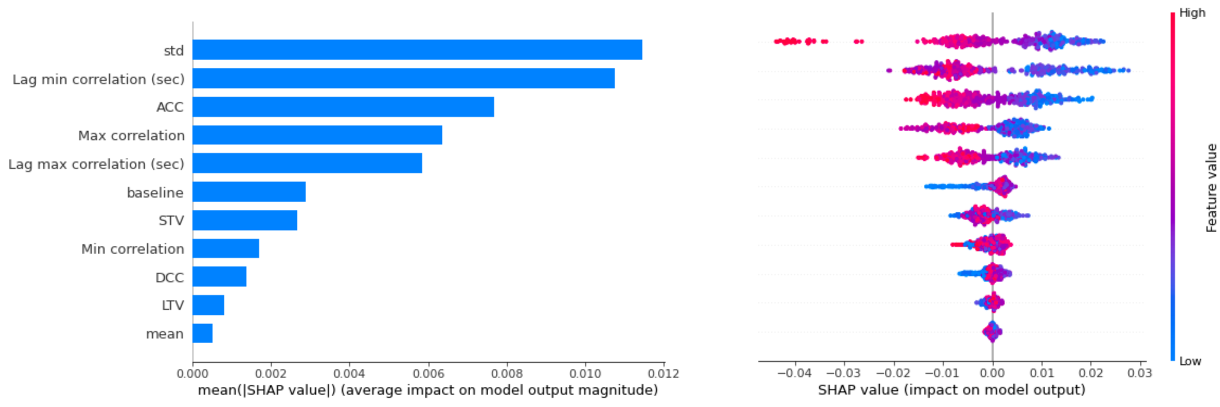
First regression trials

As mentioned above, in this appendix is shown an example of the regression approach to detect intrapartum fetal hypoxia through the umbilical pH value and the analysis of the best models obtained. The SHAP functions used to obtain the graphs shown below can be found in Appendix B.

C.1 Random Forest Regressor

The model is almost the same as in classification tasks, so this embedded model is built also with 99 estimators and the maximum depth of the tree and the minimum number of samples to split at a node have been optimized. The performance of this approach was 0.147 of R2 for the best model, but the most important aspect of this section is the interpretability offered by SHAP.

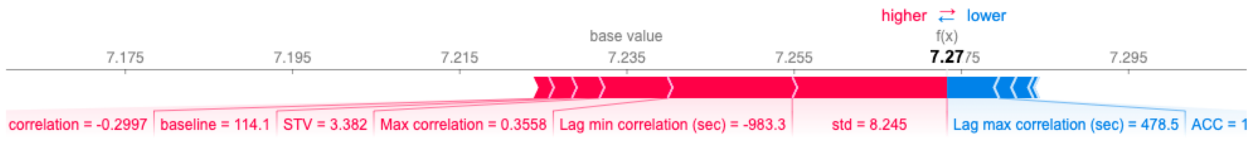
SHAP allows to extract not only the importance of features such as the `scikit-learn` library over Random Forest models, but also provides more detailed information, as it can be seen in Figure C.1. Figure C.1(a) shows the common feature importance, while Figure C.1(b), on the right, shows the *summary plot*, which presents how the value of each individual variable affects the output predicted by the model. For example, the higher the std values, the lower the SHAP value and, therefore,



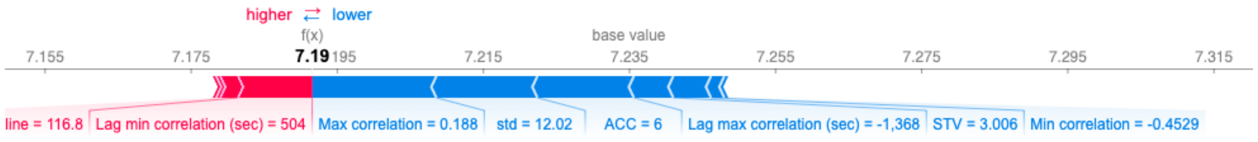
(a) Feature importance.

(b) Summary plot.

Figure C.1: Feature importance and summary plot of Random Forest regressor with SHAP.



(a) Interpretability of healthy subject with SHAP.



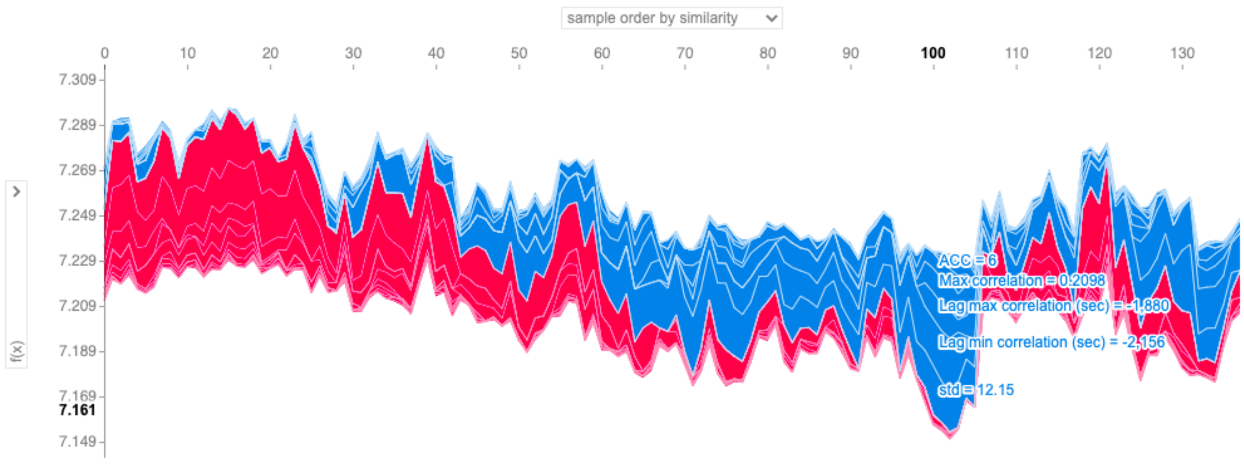
(b) Interpretability of pathological subject with SHAP.

Figure C.2: Interpretability of individual cases of Random Forest regressor with SHAP.

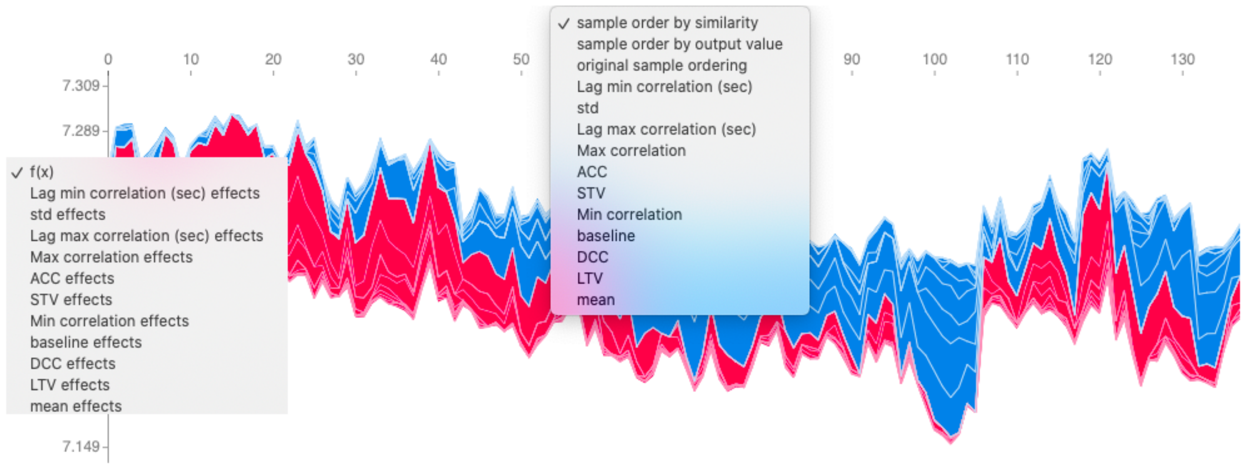
the lower the predicted pH value. As it can be seen, this visualization gives great interpretability to the performance of the model.

In addition to that, SHAP has other functionalities, such as correlation graphs between features or being able to interpret each of the individual predictions. This later application can be seen in Figure C.2, which allows us to obtain an interpretation similar to the one seen with LIME, since it shows how the value of each variable push the output of that particular case from the base value, which is the average output over the training set. In Figures C.2(a) and C.2(b) it is possible to see the interpretation of healthy and pathological cases respectively, where features marked in blue have made the subject more pathological while those shaded in red have made the fetus healthier. For example, as it can be seen in this graphs, when the FHR variability (std) is higher the fetus has a lower pH. This is a clearly sign of fetal distress that makes clinical sense (see Section 2.1) and concordance with FIGO rules.

If the individual explanations of the whole set are grouped, rotated 90 degrees and joined together, the graph in Figure C.3(a) is obtained. The X-axis shows the number of the fetus and the Y-axis shows the predicted pH values for each of them. In addition, by placing the pointer over one subject, we can see how each feature affects the prediction, i.e., for the fetus of the example shows in Figure C.3(a), the ACC patterns, the value of maximum correlation, the time lags and the std have made its predicted pH value lower (7.161). If we were to select a subject further to the left of the X-axis, its pH prediction would be higher and possibly these feature values would be different. SHAP allows to get this overview directly and so that, the operation of the model over the whole dataset can be seen together. This information could be used for other applications such as clustering tasks by grouping similar instances together [2], among others. For these kind of purposes, SHAP allows to select different visualizations according to the desired layout (see Figure C.3(b)). For example, the effect of a single feature on the whole set can be seen or the cases can be ordered according to their similarity over the output, among others. In our application, this allows us to directly relate the model working to the FIGO rules, we can see if, for example, more DCC patterns makes the pH predictions lower or not.



(a) Interpretability of all cases with SHAP.



(b) Options to visualize the interpretation of all cases with SHAP.

Figure C.3: Interpretability of all cases of Random Forest regressor with SHAP.

C.2 Support Vector Regressor

In this last model, the only methodology difference from SVC is that the *epsilon* (ϵ) parameter is searched, $\epsilon \in \{10^i : i \in [-4, -2]\}$, in order to improve the model performance. This parameter is related to the errors that the loss function ignores. The kernel width, γ, and the regularization parameter, C, have been optimized in the same way as in the classification tasks.

SHAP is not optimized for kernel models, so the computation time is longer. In spite of this, SHAP offers, as already seen in RFR, very good interpretability as it gives very detailed information. In fact, applying SHAP on kernel models, provides a great interpretability that cannot be obtained with other libraries such as `scikit-learn`. For example, this later library does not allow to obtain directly the importance of the features on kernel models.

In Figure C.4 is shown the importance of each feature and how their values affects the output target. As it can be seen in the Figure C.4, unlike what was seen in the previous section, this model does not give importance to the correlation features. It may be that this model does not consider them important or that it is not learning adequately as it is indicated by its R2 metric of 0.155.

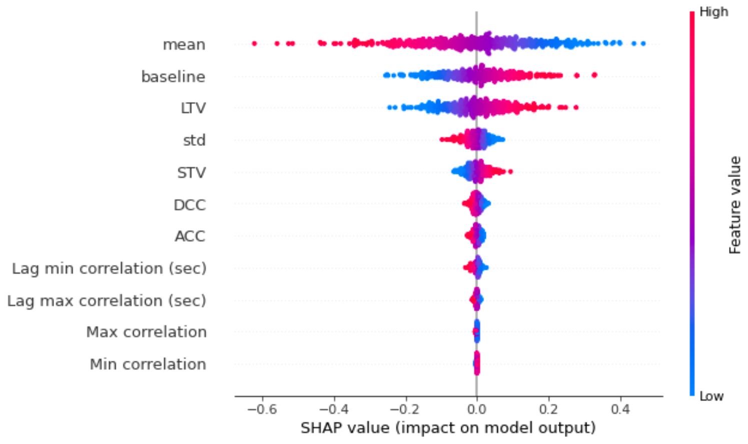


Figure C.4: Summary plot of Support Vetor regressor from SHAP.