

# Efficient Novelty Detection Methods for Early Warning of Potential Fatal Diseases

By

Sèdjro Salomon Hotegni (salomon.hotegni@aims.ac.rw)  
African Institute for Mathematical Sciences (AIMS), Rwanda

Supervised by: Professor Ernest Fokoué  
Rochester Institute of Technology, United States

June 2022

*AN ESSAY PRESENTED TO AIMS RWANDA IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF  
MASTER OF SCIENCE IN MATHEMATICAL SCIENCES*



**AIMS**

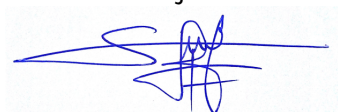
African Institute for  
Mathematical Sciences  
**RWANDA**

# DECLARATION

This work was carried out at AIMS Rwanda in partial fulfilment of the requirements for a Master of Science Degree.

I hereby declare that, except where due acknowledgement is made, this work has never been presented wholly or in part for the award of a degree at AIMS Rwanda or any other University.

Student: Sèdjro Salomon Hotegni



# ACKNOWLEDGEMENTS

I have had the benefit of many helpful persons while completing this research. Those I would want to thank are listed here in a very condensed form. I will be forever thankful to everyone else.

Prof. Ernest Fokoué for being the most understanding supervisor in the world. There was no time at which you threw up your hands and gave up on me. I am grateful for the academic direction, emotional support, and personal counsel you provided me, and I hope that one day I will be as wonderful a professor to others as you have been to me.

Dr. Prudence Djagba for all of his help and counsel. Your generosity and ingenuity have always impressed me.

Dr. Jamal Adetola, for always being a great mentor to me since my undergraduate studies and for encouraging me to pursue graduate studies.

My classmates and cohort members, for their editing assistance, and moral support.

My family and friends for their unfailing love and support throughout my life.

My mother, Justine Mehounou, is my strength, and this experience was no exception to her regular goodness.

# Abstract

Fatal diseases, as Critical Health Episodes (CHEs), represent real dangers for patients hospitalized in Intensive Care Units. These episodes can lead to irreversible organ damage and death. Nevertheless, diagnosing them in time would greatly reduce their inconvenience. This study therefore focused on building a highly effective early warning system for CHEs such as Acute Hypotensive Episodes and Tachycardia Episodes. Such a system must be reliable, so it must be able to predict CHEs earlier with great precision, while avoiding false alarms as much as possible. To facilitate the precocity of the prediction, a gap of one hour was therefore considered between the observation periods (Observation Windows) and the periods during which the event occurs (Target Windows). The MIMIC II dataset was considered for the compilation of the data used, to evaluate the performance of the proposed system. This system first includes extracting additional features using three different modes. Then, the feature selection process allowing the selection of the most relevant features was performed using the Mutual Information Gain feature importance. Finally, the high-performance predictive model LightGBM was used to perform episode classification. This approach called MIG-LightGBM was evaluated using five different metrics: Event Recall (ER), Reduced Precision (RP), average Anticipation Time (aveAT), average False Alarms (aveFA), and Event F1-score (EF1-score). A method is therefore considered highly efficient for the early prediction of CHEs if it exhibits not only a large aveAT but also a large EF1-score and a low aveFA. Compared to systems using Extreme Gradient Boosting, Support Vector Classification or Naive Bayes as a predictive model, the proposed system was found to be highly dominant. It was also compared to the Layered Learning approach from a literature, where it confirmed its superiority. The limitation of MIG-LightGBM is that it can sometimes fail to diagnose CHEs due to its Event Recall, which is not extremely high.

**Keywords:** Early warning system, Critical Health Episodes, Intensive Care Units, Time series, Early anomaly detection, Mutual Information Gain, Light Gradient Boosting Machine.

# List of Figures

3.1	Example of two consecutive sub-sequences . . . . .	8
3.2	Proposed early AHE and TE prediction system . . . . .	9
3.3	Level-wise tree growth VS leaf-wise tree growth (Merghadi et al., 2020) . . . . .	14
3.4	Maximum-margin hyperplane (Wikipedia, b) . . . . .	21
4.3	AHE: Selected features per fold with MIG . . . . .	28
4.6	TE: Selected features per fold with MIG . . . . .	31

# List of Tables

4.1	Train data for AHE . . . . .	25
4.2	Test data for AHE . . . . .	25
4.3	AHE: Average Results on the Full Data (mean $\pm$ std) . . . . .	26
4.4	AHE: Average Results After LightGBM Feature Selection (mean $\pm$ std) . . . . .	27
4.5	AHE: Average Results After MIG Feature Selection (mean $\pm$ std) . . . . .	28
4.6	Average Running Time (in seconds) for AHE Prediction (mean $\pm$ std) . . . . .	28
4.7	Train data for TE . . . . .	29
4.8	Test data for TE . . . . .	29
4.9	TE: Average Results on the Full Data (mean $\pm$ std) . . . . .	29
4.10	TE: Average Results After LightGBM Feature Selection (mean $\pm$ std) . . . . .	30
4.11	TE: Average Results After MIG Feature Selection (mean $\pm$ std) . . . . .	32
4.12	Average Running Time (in seconds) for TE Prediction (mean $\pm$ std) . . . . .	32
5.1	MIG-LightGBM vs LL for AHE Early Prediction . . . . .	33
5.2	MIG-LightGBM vs LL for TE Early Prediction . . . . .	33

# Contents

<b>Declaration</b>	<b>i</b>
<b>Acknowledgements</b>	<b>ii</b>
<b>Abstract</b>	<b>iii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Aim and objective . . . . .	2
1.3 Structure of the study . . . . .	2
<b>2 Literature Review</b>	<b>3</b>
2.1 Acute Hypotensive Episodes prediction . . . . .	3
2.2 Tachycardia Episodes prediction . . . . .	4
2.3 Gap filled . . . . .	6
<b>3 Methodology</b>	<b>7</b>
3.1 Problem statement . . . . .	7
3.2 Data compilation . . . . .	7
3.3 Feature extraction . . . . .	9
3.4 Feature Selection . . . . .	11
3.5 Predictive model . . . . .	14
3.6 Hyperparameter Tuning . . . . .	22
3.7 Evaluation Metrics: . . . . .	23
<b>4 Results</b>	<b>25</b>
4.1 AHE Early Prediction . . . . .	25
4.2 TE Early Prediction . . . . .	29
<b>5 Discussion</b>	<b>33</b>

<b>6 Conclusion</b>	<b>35</b>
<b>References</b>	<b>39</b>



# 1. Introduction

## 1.1 Motivation

An early warning approach in healthcare is an algorithm that provides information about upcoming risks to susceptible people before a Critical Health Episode occurs. This enables action to be taken to reduce possible harm and, in some situations, prevent it from occurring. The purpose of an early warning system for a Critical Health Event is then to provide health specialists and the public with as much notice as possible of the episode's probability of occurrence, thereby extending the range of viable responses. It's of great importance to the reduction of mortality in healthcare. In Rwanda, for instance, the Intensive Care Units (ICUs) mortality rate is around 43.8%, which is exceedingly high (Nkundimana, 2017). The ICUs are special units where patients with life-threatening health issues are monitored and treated, and support units are provided when necessary to maintain vital functions. Therefore, they represent a direct target for Critical Health Episodes. Adverse events (CHEs) that frequently occur in the ICUs include the deterioration of the patient's situation due to a failure to rescue or act in a timely manner. Acute Hypotensive Episodes (AHE) and Tachycardia Episodes (TE) are two of the most dangerous Critical Health Episodes in the Intensive Care Units.

- *Acute Hypotensive Episodes (AHE):* Low blood pressure, often known as hypotension, is a deadly condition in and of itself, as well as a vital marker for the advancement of certain disorders and diseases. An Acute Hypotensive Episode is defined as any interval of 30 minutes or longer during which at minimum 90% of the Mean Arterial Pressure (MAP) values were at or below 60 mmHg (PhysioNet). In patients in critical care, both single and persistent occurrences of hypotension are related to greater mortality, and studies have demonstrated that hypotension is one of the primary predictors of death (Ghassemi, 2011). Since it is a common precursor to many other medical problems, detecting it is an important step in Intensive Care Unit monitoring. The faster a hypotensive event is anticipated, the higher the chances of survival.
- *Tachycardia Episodes (TE):* Tachycardia is a clinical term that describes a faster heart rate that goes beyond the usual upper limit. Tachycardia Episodes can be defined as any interval of 30 minutes or more during which at least 90% of heart rate measurements are over 100 bpm (Cerqueira et al., 2020). They can occur in a variety of clinical situations with the dangers of further instability, in critically ill patients. Certain tachycardia occurrences in the Intensive Care Units can result in cardio-respiratory instabilities such as shock and multi-organ hypoperfusion, resulting in severe morbidity and death. Developing an effective tachycardia prediction algorithm would allow early detection and prevention of the high-risk group, leading to better overall outcomes.(Mu et al.).

Therefore, Critical Health Episodes like AHE and TE are major risk factors for mortality in ICUs, and diagnosing them in time is vital to enhance healthcare. This work is thus motivated by

the need to create innovative medical approaches in critical care, which is made feasible by the use of certain algorithms, in particular effective sequential pattern mining of time series data, to recognize clinically significant events.

## 1.2 Aim and objective

This study aims to provide a highly efficient approach for the timely detection of Critical Health Events such as Acute Hypotensive Episodes and Tachycardia Episodes in ICU patients using the LightGBM algorithm, which is a rapid, distributed, high-performance gradient boosting architecture. The specific objective is then to provide clinicians with at least one-hour warning of an imminent AHE or TE by providing a warning system that is extremely fast, uses a wide range of information from currently accessible patient monitoring data, and surpasses existing models of AHE and TE early detection.

## 1.3 Structure of the study

The results of a review of academic sources that provides an overview of methods for early warning of critical health events in ICUs are presented in the next section (Section 2), along with the gaps in these sources that this study helps to fill.

The data collection technique for the data used to examine the performance of the proposed prediction method follows in Section 3. The proposed early prediction system is then presented, followed by the description of all the machine learning tools that compose it. Other predictive models used in order to compare their performance with the LightGBM have also been described. The Section 4 presents the results obtained in this study.

Finally, a discussion part explaining, analyzing, and interpreting the results was conducted in Section 5, before concluding this study.

## 2. Literature Review

Timely prediction of Critical Health Events can be defined as a branch of activity monitoring, related to early anomaly detection in time series data (Fawcett and Provost, 1999). Since it records a varied and very huge population of patients in intensive care and provides clear and comprehensive clinical data, which include physiological waveforms and minute-by-minute trends for a sample of records, the database Multiparameter Intelligent Monitoring for Intensive Care (MIMIC) is the most used for the critical health events prediction (Moody and Mark, 1996).

### 2.1 Acute Hypotensive Episodes prediction

Using observations gathered 30 minutes prior, Lee and Mark (2010a) were able to predict the occurrence of hypotensive episodes 1 hour in advance by training Artificial Neural Networks (ANNs) for binary classification and regression on the MIMIC II dataset. Two data compilation modes were used: the single data compilation mode where each record offered only one sample, either hypotensive or control, and the multiple data compilation mode where each record was browsed by 30 minutes sliding window with no overlap, and as many samples as available were assembled. Although ANNs for classification outperformed regression with a mean Area Under Receiver Operating Characteristic (AUROC) curve of 0.918, with multiple data compilation, it had a poor Positive Predictive Values (PPVs) of  $0.138 \pm 0.012$ .

Due to the overall poor PPVs caused by the low prevalence of HE, this trained algorithm's efficacy as a direct predictor of HE is restricted.

The same observation was made when the same authors conducted another study on the prediction of hypotensive episodes, this time focused on the comparison of different gap sizes and the weighted prediction (Lee and Mark, 2010b). Principal Component Analysis (PCA) was applied to decrease the dimensionality of the feature space. It is worth noting that, as their analysis showed, projecting further into the future gets more difficult as gap size increases, and therefore prediction performance reduces.

Jiang et al. (2015) suggested the Hilbert-Huang Transform and Multiple Genetic Programming Classifier as a technique for predicting Acute Hypotensive Events, using both the *2009 PhysioNet and Computers Cardiology Challenge* dataset and the MIMIC II database. The Hilbert-Huang transformation method was used to compute the Mean Arterial Pressure (MAP) time series and extract additional features. Multiple Genetic Programming (Multi-GP) was used to construct the classification models for the diagnosis of AHE. Genetic Programming (GP) is an automated programming approach that can handle issues across a broader number of fields and can be more robust than Neural Networks. This technique outperforms the Support Vector Machine model, achieving an accuracy of 82.41% on MIMIC II and 91.89% on the second dataset.

However, as Multi-GP is a random and evolving method, it is not stable in problem resolution.

Yoon et al. (2020) on their side, used a Random Forest (RF) classifier to identify the risk of hypotension every minute using time series data from 30 minutes overlapping time windows from the MIMIC-3 dataset. The RF model demonstrated an Area Under the Receiver Operating Char-

acteristic (AUROC) curve of 0.93 and 0.88 at 15 and 60 minutes before hypotension, respectively, and an Area Under the Precision-Recall Curve (AUPRC) of 0.77 at 60 minutes before hypotension.

Nevertheless, this study should have used additional features, such as temporal correlations between physiologically imposed features. Doing this would have increased the range of variables with the inclusion of more complex and physiology-inspired variables, and this could lead to better performance of the algorithm.

## 2.2 Tachycardia Episodes prediction

Segyeong et al. (2016), like Lee and Mark (2010a) in AHE prediction, referred to Artificial Neural Networks for Ventricular Tachycardia (VT) early detection using real-time physiological monitoring data gathered from 15 patient monitors at Asan Medical Center's Cardiovascular Critical Care Unit (CCU). The proposed VT prediction model showed a sensitivity of 0.88, a specificity of 0.82, and an AUC of 0.93.

Despite the fact that the data were collected over a two years period using 15 patient monitors, only a few real Ventricular Tachycardia (VT) events were recorded (52 recordings prior to one hour from a VT occurrence). As a result, the statistical power of the analysis is limited.

Yoon et al. (2019), on their side, used the MIMIC II database to predict Tachycardia Events in three separate case and control groups using regularized Logistic Regression (LR) and Random Forest (RF) classifiers. All Tachycardia incidents were predicted in the first group. Since future tachycardia signals might have shared etiologies with the initial event, the second group solely assessed the prediction of the first tachycardia episode. The third group was created to model real-life conditions by using the initial observational interval of non-tachycardia that preceded the onset of a tachycardia episode within the same individual. The RF model outperformed the LR model, with an AUROC of 0.93 and 0.88 at 15 and 60 minutes, respectively, before the occurrence of a Tachycardia Event.

However, due to technical challenges such as timestamp mismatching and data sparsity, the matching high-density waveform data could not be adequately characterized. So, the data could not be wholly leveraged to exhibit the highest achievable classification efficiency.

To develop the TOP-Net Prediction Model, Liu et al. (2021) combined two types of data from the MIMIC III database. First, details from biological sensors, such as heart rate, respiration rate, and  $SpO_2$  were used. Second, patient details from electronic health records, which represent their individual health status once admitted to the hospital, such as age, gender, admission type, first care unit, and historical memory of cardiovascular disease were also added.

Data from *SensEcho*, a self-developed physiological signal monitoring system, was used to assess the performance of this model. But it was only deployed to the clinic for a year after the study project began. This has caused the data obtained to be therefore minimal. Therefore, the product model is not a generic service model. Furthermore, treatments such as beta-blocker medication can influence the onset of tachycardia and lead it to be missed by the input features.

In 2020, Cerqueira et al. (2020) developed a hierarchical approach for the early prediction of both

Acute Hypotensive Episodes and Tachycardia Episodes. This requires splitting the initial problem into two hierarchical levels by introducing the concept of pre-conditional events, which represent arbitrary but executable relaxed forms of the event of interest. The objective is to first model the pre-conditional events in reference to normal activity, and then model the main events in reference to the pre-conditional events.

- *Pre-conditional Events Sub-task:*

Let  $P$  denote a pre-conditional event. The first prediction task's purpose is to establish a function  $g^P$  that relates the input predictor variables  $\mathcal{X}$  to the output  $\mathbf{y}^P$ , which is defined as follows:

$$y_i^P = \begin{cases} 1, & \text{if } P \text{ happens} \\ 0, & \text{otherwise} \end{cases}$$

where a subsequence is seen as  $\delta^P = (t_i, \mathcal{X}_i, y_i^P)$ , with  $t_i$  designating the start time of the subsequence and  $\mathcal{X}_i \in \mathcal{X}$ ,  $y_i^P \in \mathbf{y}^P$

- *Main Events Sub-task:*

Let  $M$  be described as the incidence: "given  $P$ , there is an imminent main event in the target window of the actual sub-sequence." For the sub-sequences that at least result to a pre-conditional event, the purpose of the second prediction task is to establish a function  $g^M$  that relates the input predictor variables  $\mathcal{X}$  to the output  $\mathbf{y}^M$ , which is defined as follows:

$$\text{Given } y^P = 1, y_i^M = \begin{cases} 1, & \text{if } M \text{ happens in the target window "i"} \\ 0, & \text{otherwise} \end{cases}$$

where a subsequence is seen as  $\delta^M = (t_i, \mathcal{X}_i, y_i^M)$ , with  $t_i$  designating the start of the subsequence and  $\mathcal{X}_i \in \mathcal{X}$ ,  $y_i^M \in \mathbf{y}^M$

- *Making predictions about future anomalies:*

A model  $h$  was built to forecast the occurrence of the main event in a specific sub-sequence based on the multiplication of the results predicted by both  $g^P$  and  $g^M$ .

$$h : \mathbb{X} \longrightarrow \mathbb{Y}$$

$$\mathcal{X}_i \longmapsto g^P(\mathcal{X}_i) \times g^M(\mathcal{X}_i)$$

In an ideal scenario, there are three possibilities:

- Both event  $P$  and event  $M$  occur, indicating that the main event is about to occur:  
( $g^P = 1$  and  $g^M = 1$ )  $\implies h = 1$
- Event  $P$  occurs, but Event  $M$  does not:  
( $g^P = 1$  and  $g^M = 0$ )  $\implies h = 0$
- Event  $P$  does not occur, and as a result, event  $M$  does not occur:  
( $g^P = 0$  and  $g^M = 0$ )  $\implies h = 0$

In the Layered Learning algorithm, Extreme Gradient Boosting (XGBoost) was used as a predictive model.

This method of *Layered Learning* has outperformed the Standard Classification, the Isolation Forest, the Regression Approach, and the Ad-hoc Methods. The measure of its performance showed an Event Recall (ER) of  $0.830 \pm 0.054$  and an average Anticipation Time (avg.AT) of  $46.9 \pm 3.3$  for the AHE prediction. For the TE prediction, it has an Event Recall of  $0.938 \pm 0.027$  and an average Anticipation Time of  $53.0 \pm 2.2$ .

Because Layered Learning separates a predictive task into two or simpler predictive tasks, the learning process inside one layer influences the learning process of the following layer. Furthermore, before using the LL approach, a pre-conditional event must be explicitly defined. This procedure is highly domain-dependent. But, end-to-end machine learning techniques are gaining ground.

## 2.3 Gap filled

Tsur et al. (2018) stated that for AHE prediction tasks, Extreme Gradient Boosting (XGBoost) gives a good predictive performance. It has been verified by Cerqueira et al. (2020) in their proposed Layered Learning method for AHE and TE prediction. However, in general, existing warning systems for AHE and TE early prediction typically have limited performance and high computational costs in real-time alerting when dealing with large amounts of data. So, this work presents a highly efficient warning system based on the LightGBM model, an extension of XGBoost, for the early detection of Critical Health Episodes.

LightGBM algorithm, is a suitable solution to handle Big Data and a huge proportion of features, which was recommended in 2017 by Microsoft (Ke et al., 2017).

# 3. Methodology

## 3.1 Problem statement

Critical Health Events, such as AHE and TE, are known to be related to severe internal bleeding and mortality. Detecting them in time is therefore vital since it may allow treatments to be improved to minimize or limit their consequences. As proposed by Weiss and Hirsh (1998), this kind of event prediction problem can be formalized as follows:

Let  $\mathcal{P} = \{\mathcal{P}_1; \dots; \mathcal{P}_{|\mathcal{P}|}\}$  denotes a set of patients followed in the intensive care unit of a hospital. Each  $\mathcal{P}_i$  is a time series data containing observations on a given patient over time. More simply; it is a set of sub-sequences

$$\mathcal{E}_i = \{\delta_1; \dots; \delta_i; \dots; \delta_n\}$$

where  $\delta_i = (t_i; \mathcal{X}_i; y_i)$ . Here,  $t_i$  is the timestamp marking the start of the sub-sequence  $\delta_i$ ,  $\mathcal{X}_i \in \mathbb{X} = \{\mathcal{X}_1, \dots, \mathcal{X}_n\}$ , are the input variables (predictors) and  $y_i \in \mathbb{Y} = \{0; 1\}$  is the target variable, which is binary and represents whether there is an imminent critical event in the near future in the respective time series.

A general idea behind early warning systems for CHEs is to simulate a discriminating model  $g$  using a patient's previous physiological data as input, and to anticipate whether a critical event will occur in the near future (Fawcett and Provost, 1999):

$$\begin{aligned} g : \mathbb{X} &\longrightarrow \mathbb{Y} \\ \mathcal{X}_i &\longmapsto y_i \end{aligned}$$

Since the efficacy of early prediction models for these incidents is often restricted, the average Anticipation Time is quite short and the proportion of False Alarms is large. The purpose of this research is to develop an intelligent, fast, and strong system for the earlier detection of Critical Health Events such as AHE and TE. This could therefore contribute to a great improvement in the outcomes of Intensive Care Units.

## 3.2 Data compilation

The data used in this research is a subset of the Multi-parameter Intelligent Monitoring for Critical Care (MIMIC) II database (Clifford et al., 2010). It contains minute-by-minute time series of Heart Rate (HR), Systolic Blood Pressure (SBP), Diastolic Blood Pressure (DBP), and Mean Arterial blood Pressure (MAP) arranged into records, each of which corresponds to an adult patient's ICU stay.

Following Lee and Mark (2010a)'s data compilation procedure, the values of HR (in bpm) and MAP (in mmHg) in each 30 minutes target window must be between 10 and 200; otherwise, the

corresponding sub-sequence was eliminated from this research. In addition, for all the four signals to be regarded valid and therefore taken into consideration for this study, their values must be at least 95% between 10 and 200 in the 60 minutes Observation Windows. If not, the corresponding sub-sequence is also eliminated from the analysis. As many examples of sub-sequences, with valid observation and Target Windows, as possible were compiled by traversing each time series with a 30 minutes sliding window without overlap, irrespective of whether a patient experienced a Critical Health Event. This means that if sub-sequence  $\delta_i$  begins at time  $t$ , sub-sequence  $\delta_{i+1}$  follows at time  $t + 30$ . (Figure 3.1)

Finally, 86 different patient records were used for the analysis.

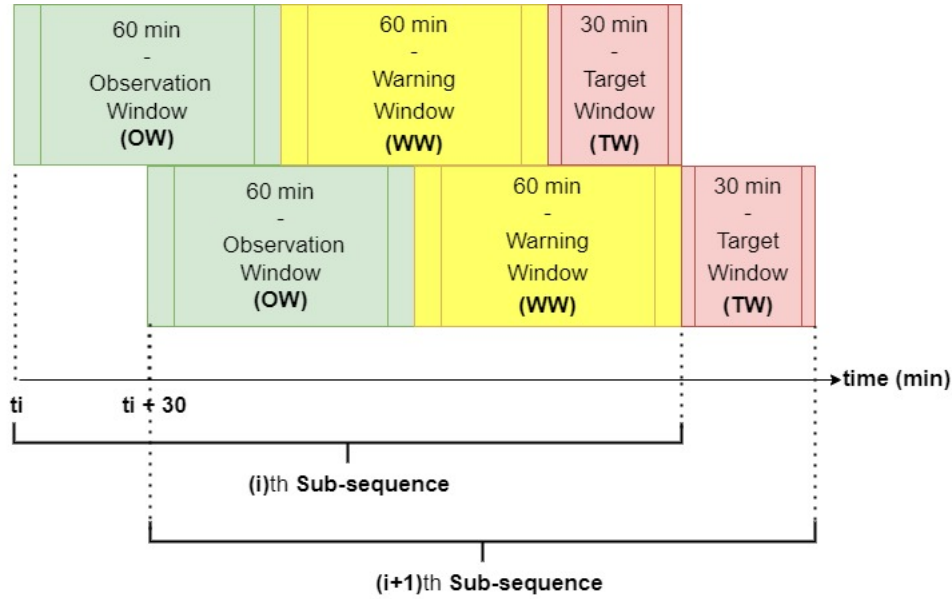


Figure 3.1: Example of two consecutive sub-sequences

**3.2.1 Data labelling.** Using the target window of each episode, two feature targets  $y_1$  and  $y_2$  were added to the compiled time series data of each patient.

The variable  $y_1$  is binary and indicates whether a sub-sequence is a Hypotensive Episode or not. In the case of Hypotensive, the value 1 is assigned to all the observations of the corresponding Observation Window, for  $y_1$ . Otherwise, 0 is assigned.

Similarly,  $y_2$  is binary and indicates whether a sub-sequence is a Tachycardia Episode or not. In the case of Tachycardia, the value 1 is assigned to all the observations of the corresponding Observation Window, for  $y_2$ . Otherwise, 0 is assigned.

So, for both AHE and TE, the early prediction task was seen as a classification task.

**3.2.2 Training and Testing data.** A 5-fold Cross-Validation (CV) was performed on the dataset, considering patients instead of observations (each fold contains patients whose data was used for analysis, some for training and the rest for testing). To permit the classifier to properly learn the unbalanced data, each data in the training folds was under-sampled 10 times without replacement.

- *Random Under-sampling:*



This approach consists of first identifying the minority class (which is here class 1, because the Normal Episodes are more than the Critical Health Episodes in the compiled data.) and privileging it. To do this, the observations of the majority class (class 0) are randomly selected so that we end up with the same number of observations in both classes. Therefore, this approach results in equal numbers of Critical Health Episodes and Normal Episodes within each training data.

The Test data, on the other hand, were left imbalanced.

The Figure 3.2 shows the framework of the proposed system for the early prediction of AHE and TE.

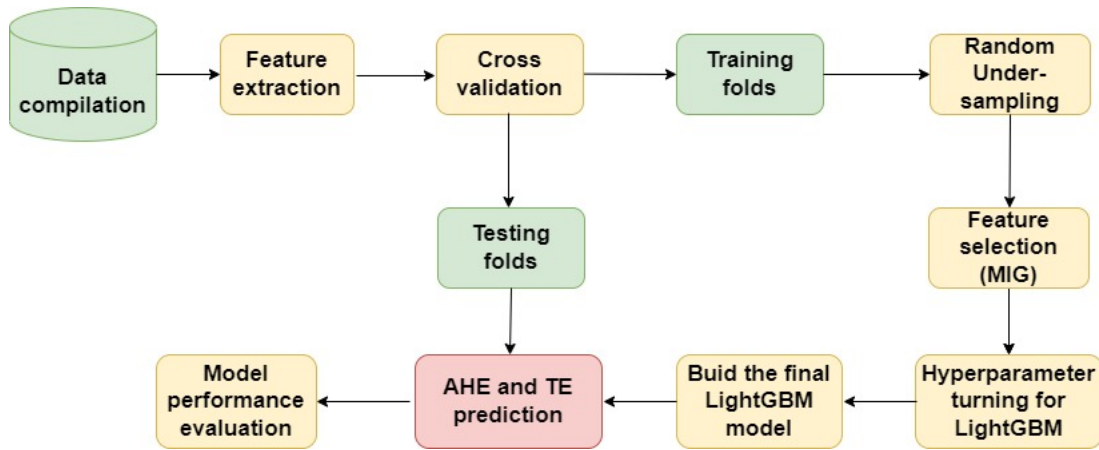


Figure 3.2: Proposed early AHE and TE prediction system

### 3.3 Feature extraction

Feature extraction, also known as feature engineering, is the process of extracting features from raw data using domain expertise. The objective is to utilize these extra attributes to increase the quality of a machine learning model outputs when compared to simply giving the raw data to the machine learning model (Wikipedia, c). In this work, the feature extraction process was carried out according to the techniques used by Tsur et al. (2018) and Lee and Mark (2010a).

- *Knowledge-based features:*

Using the four existing time series features, two more were derived:

Pulse Pressure (PP):  $PP = SBP - DBP$ .

Cardiac Output (CO):  $CO = HR \times PP$

- *Statistical features:*

When investigating a data set for an insightful view, statistical features are usually among the first statistical tools used. In this study, the skewness, kurtosis, slope, median, minimum, maximum, variance, mean, standard deviation, and interquartile range were computed for

each of the 6 prior time series features (HR, SBP, DBP, MAP, PP and CO) based on the Observation Windows. This implies the extraction of  $10 \times 6 = 60$  new features. Note that the skewness is used to measure the level of asymmetry in the data, while kurtosis measures the degree of peakness of a frequency distribution.

- *Cross-correlation features:*

The cross-correlation technique compares the 6 time series features two-by-two and objectively determines how well they match each other.

Let  $\mathcal{R} = \{r_1, \dots, r_n\}$  and  $\mathcal{S} = \{s_1, \dots, s_n\}$  be two different time series features, where  $r_i$  and  $s_i$ ,  $i = 1, \dots, n$ , are the observations within a considered Observation Window. The cross-correlation at lag 0 between them was calculated by using the following formula (Derrick and Thomas, 2004):

$$CC_{RS}(0) = \frac{1}{n} \sum_{i=1}^n r_i s_i.$$

In total,  $C_6^2 = 15$  additional features had therefore been added.

- *Wavelet features*

The Meyer wavelet was used to perform a 5-level discrete wavelet decomposition of each of the 6 signals to extract relative energies in distinct wavelength bands. A time-domain signal  $x[n]$ 's Discrete Wavelet Transform (DWT) is computed by running it through a succession of high-pass filters  $h[n]$  and low-pass filters  $l[n]$ . The outputs of the high-pass and low-pass filters are called the wavelet coefficients and are given by:

$$d_{j+1}[n] = \sum_{k=-\infty}^{\infty} h[k - 2n]a_j[n] \quad (3.3.1)$$

where  $j = 1, \dots, 5$  are the different levels, and

$$a_{j+1}[n] = \sum_{k=-\infty}^{\infty} l[k - 2n]a_j[n].$$

DWT extracts then the signal's important components from the noisy one, according to Ali et al. (2017)

The final result of this decomposition is represented by:

$$\mathcal{W}_5 = [a_5, d_1, d_2, d_3, d_4, d_5].$$

The energy in the approximation signal  $a_5$  is

$$E_{a_5} = ||a_5||^2$$

using the Euclidean norm, and the energy in the  $j^{th}$ -level detail signal is

$$E_{d_j} = ||d_j||^2, \text{ for } j = 1, \dots, 5.$$

Finally, using

$$E_T = E_{a_5} + \sum_{k=1}^5 E_{d_k}$$

the relative energy production within each decomposition level was computed as follows:

$$Er_{a_5} = \frac{E_{a_5}}{E_T}; \quad Er_{d_j} = \frac{E_{d_j}}{E_T} \quad j = 1, \dots, 5.$$

So,  $5 \times 6 = 30$  new features were added.

The total number of features considered for the analysis was therefore  $6 + 60 + 15 + 30 = 111$ .

## 3.4 Feature Selection

Feature selection is an important phase in data cleaning since it determines the key features. It not only removes the redundant data but also aids in the discovery of the most relevant ones, hence improving the model's performance. Apart from the feature importance ranking provided by the proposed predictive model, the *Mutual Information Gain for Classification* approach was also used for the feature selection process.

- *LightGBM feature importance:*

Two methods of calculating feature importance are available in LightGBM:

- *Split Importance:*

Split importance is a measurement of feature relevance in tree-based algorithms. It just counts the number of times the nodes divide on a feature. It is assumed that the more significant the feature, more the times it gets divided. Split importance is used in LightGBM as the default feature importance measure.

- *Gain:*

The Gain of a feature indicates its contribution to the model and is determined by adding the contributions of each feature for each tree in the model. A feature is then more significant for making a prediction if, when comparing its gain with that of another feature, it is high.

- *Mutual Information Gain for Classification:*

Mutual Information Gain (MIG), as one of the most efficient feature selection methods, is a measure of the "mutual dependency" of two random variables ([Wikipedia, a](#)). MIG creates a quantifiable link between a feature and the target. Using MIG as a feature selector has two advantages: It is model-neutral, which means it can be applied to a wide range of machine learning models; and it is also fast ([Andrew Zhu](#)).

MIG is a non-negative value that equals zero if and only if two variables are independent, and a larger value indicates stronger dependency. So, one would like to maximize the mutual information between the subset of selected features and the target variable.

Let  $\mathbf{x}_j$  be a feature and  $\mathbf{y}$  the target variable. The Mutual Information Gain for the two discrete random variables  $\mathbf{x}_j$  and  $\mathbf{y}$  is given by

$$I(\mathbf{x}_j; \mathbf{y}) = I(\mathbf{y}; \mathbf{x}_j) = \sum_{y_i \in \mathbf{y}} \sum_{x_{ji} \in \mathbf{x}_j} p_{xy}(x_{ji}, y_i) \cdot \log \left( \frac{p_{xy}(x_{ji}, y_i)}{p_x(x_{ji})p_y(y_i)} \right)$$

where  $p_x$  and  $p_y$  are the marginal probability density functions and  $p_{xy}$  is the joint probability density function.

Note that given a set of features  $\mathcal{X}_S = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  and a single feature  $\mathbf{y}$ , the Joint Mutual Information between them is given by

$$I(\mathcal{X}_S; \mathbf{y}) = I(\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}; \mathbf{y}) = \sum_j I(\mathbf{x}_j; \mathbf{y} | \mathbf{x}_{j-1}, \mathbf{x}_{j-2}, \dots, \mathbf{x}_1).$$

The process for selecting a subset of important features with MIG is described below.

Let  $|\mathcal{S}| = k$ . be the number of features to be selected.

The feature selection process should identify a subset of features  $\mathcal{X}_{\hat{\mathcal{S}}} = \{\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}\}$ , which maximizes the Joint Mutual Information  $I(\mathcal{X}_S; \mathbf{y})$  between the class label  $\mathbf{y}$  and the possible features subsets  $\mathcal{X}_S$  of size  $k$  (Salem et al., 2021)

$$\{i_1, \dots, i_k\} = \hat{\mathcal{S}} = \underset{\mathcal{S}}{\operatorname{argmax}} \{I(\mathcal{X}_S; \mathbf{y})\}$$

Since the number of potential feature combinations is usually high, this problem is an NP-hard optimization problem (Nondeterministic Polynomial time problem). One technique to address it is to choose features progressively, one at a time, using a *greedy forward step-wise selection* algorithm.

First, according to their mutual information with respect to the target  $\mathbf{y}$ , the features are ranked. The top feature is then selected. Let  $\mathcal{X}_{S^{t-1}} = \{\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_{t-1}}\}$  denote the set of features that were chosen at step  $t - 1$ . The *greedy* technique chooses the next feature  $\mathbf{x}_{i_t}$  in such a way that the greatest improvement in Joint Mutual Information is achieved by using  $\mathcal{X}_{S^t}$ . So,

$$i_t = \underset{i \notin \mathcal{S}^{t-1}}{\operatorname{argmax}} I(\mathcal{X}_{S^{t-1} \cup i}; \mathbf{y})$$

$$i_t = \underset{i \notin \mathcal{S}^{t-1}}{\operatorname{argmax}} I(\mathcal{X}_{S^{t-1}}, \mathbf{x}_i; \mathbf{y})$$

This requires computing across a  $(t - 1)$ -dimensional space, which eventually turns computationally difficult. To make this computation manageable, the data is analyzed under the hypothesis that *the selected features  $\mathcal{X}_S$  are independent and class-conditionally independent given the considered unselected feature  $\mathbf{x}_k$* .

Since  $I(\mathcal{X}_{S^{t-1}}; \mathbf{y})$  is already known, maximize  $I(\mathcal{X}_{S^{t-1}}, \mathbf{x}_i; \mathbf{y})$  is the same as maximize  $I(\mathcal{X}_{S^{t-1}}, \mathbf{x}_i; \mathbf{y}) - I(\mathcal{X}_{S^{t-1}}; \mathbf{y})$  (Peng and Fan, 2017)

Thus,

$$\begin{aligned}
 i_t &= \arg \max_{i \notin \mathcal{S}^{t-1}} \left( I(\mathcal{X}_{\mathcal{S}^{t-1}}, \mathbf{x}_i; \mathbf{y}) - I(\mathcal{X}_{\mathcal{S}^{t-1}}; \mathbf{y}) \right) \\
 &= \arg \max_{i \notin \mathcal{S}^{t-1}} \left( I(\mathcal{X}_{\mathcal{S}^{t-1}}; \mathbf{y}) + I(\mathbf{x}_i; \mathbf{y} | \mathcal{X}_{\mathcal{S}^{t-1}}) - I(\mathcal{X}_{\mathcal{S}^{t-1}}, \mathbf{y}) \right) \\
 &= \arg \max_{i \notin \mathcal{S}^{t-1}} \left( I(\mathbf{x}_i; \mathbf{y} | \mathcal{X}_{\mathcal{S}^{t-1}}) \right) \\
 i_t &= \arg \max_{i \notin \mathcal{S}^{t-1}} \left( I(\mathbf{x}_i; \mathbf{y}) - I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}}) + I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}} | \mathbf{y}) \right)
 \end{aligned}$$

(using the identity  $I(\mathbf{x}_i; \mathbf{y} | \mathcal{X}_{\mathcal{S}^{t-1}}) - I(\mathbf{x}_i; \mathbf{y}) = I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}} | \mathbf{y}) - I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}})$  (Brown et al., 2012))

Therefore,

$$i_t = \arg \max_{i \notin \mathcal{S}^{t-1}} \left( \underbrace{I(\mathbf{x}_i; \mathbf{y})}_{\text{relevancy}} - \left[ \underbrace{I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}}) - I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}} | \mathbf{y})}_{\text{redundancy}} \right] \right) \quad (3.4.1)$$

When this factor is optimized, the *relevance* of a new feature  $\mathbf{x}_i$  with respect to the target  $\mathbf{y}$  is evaluated against the *redundancy* of that information in comparison to the information provided by the variables  $\mathcal{X}_{\mathcal{S}^{t-1}}$  that have been already picked.

The *lower-dimensional approximation* technique helps to make the problem even more solvable (Brown et al., 2012):

$$I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}}) \approx \alpha \sum_{j \in \mathcal{S}^{t-1}} I(\mathbf{x}_j; \mathbf{x}_i)$$

$$I(\mathbf{x}_i; \mathcal{X}_{\mathcal{S}^{t-1}} | \mathbf{y}) \approx \beta \sum_{j \in \mathcal{S}^{t-1}} I(\mathbf{x}_j; \mathbf{x}_i | \mathbf{y})$$

The optimization problem is then simplified for a given  $\alpha$  and  $\beta$  to:

$$i_t = \arg \max_{i \notin \mathcal{S}^{t-1}} \left( \underbrace{I(\mathbf{x}_i; \mathbf{y})}_{\text{relevancy}} - \underbrace{\left[ \alpha \sum_{j \in \mathcal{S}^{t-1}} I(\mathbf{x}_j; \mathbf{x}_i) - \beta \sum_{j \in \mathcal{S}^{t-1}} I(\mathbf{x}_j; \mathbf{x}_i | \mathbf{y}) \right]}_{\text{redundancy}} \right) \quad (3.4.2)$$

Here are some examples of certain values of  $\alpha$  and  $\beta$  (Salem et al., 2021):

- *Mutual Information Maximization (MIM)*:  $\alpha = 0$  and  $\beta = 0$ .
- *Joint Mutual Information (JMI)*:  $\alpha = \frac{1}{t-1}$  and  $\beta = \frac{1}{t-1}$ .
- *Maximum Relevancy, Minimum Redundancy (MRMR)*:  $\alpha = \frac{1}{t-1}$  and  $\beta = 0$ .

Generally, a value of  $\alpha$  near-zero is related to the hypothesis that all features are class-conditionally independent of each other:

$$p(\mathbf{x}_j; \mathbf{x}_i | \mathbf{y}) = p(\mathbf{x}_j | \mathbf{y})p(\mathbf{x}_i | \mathbf{y}) \implies \sum I(\mathbf{x}_j; \mathbf{x}_i | \mathbf{y}) = 0$$

and a value of  $\beta$  near-zero is related to the hypothesis that all features are independent of one another:

$$p(\mathbf{x}_j; \mathbf{x}_i) = p(\mathbf{x}_j)p(\mathbf{x}_i) \implies \sum I(\mathbf{x}_j; \mathbf{x}_i) = 0$$

The feature importance ranking lists provided by LightGBM and MIG were used to choose the most important variables, given a threshold.

### 3.5 Predictive model

The Light Gradient Boosting Machine (LightGBM) is the predictive model used in the proposed early detection of AHE and TE system.

LightGBM is a sophisticated Gradient-Boosted Decision Tree (GBDT) method that splits the input layer parameters into distinct portions and therefore develops the mapping connection between the inputs and outputs (Gan et al., 2021). To accelerate training, it uses leaf-wise tree growth rather than the more often used level-wise tree growth (Figure 3.3).

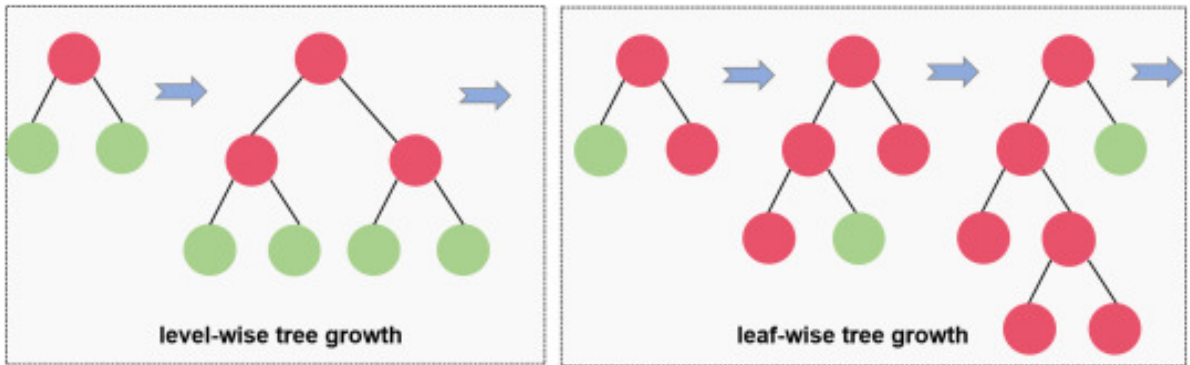


Figure 3.3: Level-wise tree growth VS leaf-wise tree growth (Merghadi et al., 2020)

Let  $\mathcal{X} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$  be the training data and  $T$  the number of additive functions to predict the output. After  $t$  iterations, the final prediction is the sum of the previous  $(1 - t)^{th}$  and  $t^{th}$  predictions. The additive training procedure is as follows:

$$\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(\mathbf{x}_i) = \hat{y}_i^{(0)} + f_1(\mathbf{x}_i) \\
\hat{y}_i^{(2)} &= f_1(\mathbf{x}_i) + f_2(\mathbf{x}_i) = \hat{y}_i^{(1)} + f_2(\mathbf{x}_i) \\
&\dots \\
\hat{y}_i^{(t)} &= \sum_{k=1}^t f_k(\mathbf{x}_i) = \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)
\end{aligned}$$

where  $\hat{y}_i^{(t)}$  represents the prediction of the  $i^{th}$  case at the  $t^{th}$  iteration and  $i_t$  represents the learned function for the  $t^{th}$  decision tree. The LightGBM was designed to minimize the following regularized objective function (*Gradient Tree Boosting* process) (Fafalios et al., 2020):

$$\mathcal{O} = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^T \Omega(f_k) \quad (3.5.1)$$

with  $l(\cdot, \cdot)$  the loss function that evaluates the difference between the prediction  $\hat{y}_i$  and the target  $y_i$ , and

$$\Omega(f_k) = \gamma J + \frac{1}{2} \lambda \|\mathbf{w}\|_2^2$$

is the regularization term that penalizes the model's complexity. Here,  $J$  is the number of leaf nodes in the tree  $k$ .

Let  $F = \{f(\mathbf{x}) = \mathbf{w}_{q(\mathbf{x})}\} (q : \mathbb{R}^n \rightarrow J, \mathbf{w} \in \mathbb{R}^J)$  be the space of regression trees.

$$\hat{y}_i = \sum_{t=1}^J f_t(\mathbf{x}_i), \quad f_t \in F$$

where  $q$  denotes the structure of each tree that maps an example to the corresponding leaf index, each  $f_t$  corresponds to an independent tree structure  $q$ , and  $\mathbf{w}$  is the leaf weights.

Let  $\hat{y}_i^{(t)}$  be the prediction of the  $i^{th}$  instance at the  $t^{th}$  iteration,  $i_t$  needs to be added to minimize the following objective

$$\mathcal{O}^{(t)} = \sum_{i=1}^n l(y_i, \hat{y}_i^{(t)}) + \Omega(i_t) \quad (3.5.2)$$

The logistic loss function is the loss function that has been applied in this algorithm. So,

$$l(y_i, \hat{y}_i^{(t)}) = y_i \ln(1 + e^{-\hat{y}_i^{(t)}}) + (1 - y_i) \ln(1 + e^{\hat{y}_i^{(t)}}) \quad (3.5.3)$$

Since  $\hat{y}_i^{(t)} = f_{t-1}(\mathbf{x}_i) + f_t(\mathbf{x}_i)$ , the objective function can be written as:

$$\mathcal{O}^{(t)} = \sum_{i=1}^n l(y_i, f_{t-1}(\mathbf{x}_i) + f_t(\mathbf{x}_i)) + \Omega(i_t) \quad (3.5.4)$$

Taylor's polynomial interpolation is applied at this point to quickly approximate the objective function:

$$\begin{aligned} l(y_i, \hat{y}_i^{(t)}) &\approx f_t(\mathbf{x}_i) \frac{\partial}{\partial f_{t-1}(\mathbf{x}_i)} \left[ l(y_i, f_{t-1}(\mathbf{x}_i)) \right] + \frac{1}{2} [f_t(\mathbf{x}_i)]^2 \frac{\partial^2}{\partial f_{t-1}(\mathbf{x}_i)^2} \left[ l(y_i, f_{t-1}(\mathbf{x}_i)) \right] \\ l(y_i, \hat{y}_i^{(t)}) &\approx g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i [f_t(\mathbf{x}_i)]^2 \end{aligned}$$

where,

$$\begin{aligned} g_i &= \frac{\partial}{\partial f_{t-1}(\mathbf{x}_i)} \left[ l(y_i, f_{t-1}(\mathbf{x}_i)) \right] \\ h_i &= \frac{\partial^2}{\partial f_{t-1}(\mathbf{x}_i)^2} \left[ l(y_i, f_{t-1}(\mathbf{x}_i)) \right] \end{aligned}$$

are the first and second derivations of the loss function with respect to  $f_{t-1}(\mathbf{x}_i)$ . Then,

$$\mathcal{O}^{(t)} = \sum_{i=1}^n \left[ g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i [f_t(\mathbf{x}_i)]^2 \right] + \Omega(i_t) \quad (3.5.5)$$

Let  $I_j$  denotes the sample set of leaf  $j$ . Using the fact that

$$\Omega(i_t) = \gamma J + \frac{1}{2} \lambda \|\mathbf{w}\|_2^2 = \gamma J + \frac{1}{2} \lambda \sum_{j=1}^J w_j^2$$

it follows:

$$\begin{aligned} \mathcal{O}^{(t)} &= \sum_{j=1}^J \sum_{i \in I_j} \left[ g_i w_j + \frac{1}{2} h_i w_j^2 \right] + \sum_{j=1}^J \left( \frac{1}{2} \lambda w_j^2 \right) + \gamma J \\ \mathcal{O}^{(t)} &= \sum_{j=1}^J \left[ \sum_{i \in I_j} g_i w_j + \frac{1}{2} \left( \sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma J \end{aligned}$$

where  $w_j = f_t(\mathbf{x}_i)$  at each leaf node on the  $t^{th}$  tree is the unknown variable to be found. Except for  $w_j$ , all other parameters are provided or can be determined through the  $(t - 1)$  learning process. Minimizing the loss function  $\mathcal{O}^{(t)}$  requires solving the following equation

$$\frac{\partial \mathcal{O}^{(t)}}{\partial w_j} = 0 \quad (3.5.6)$$



for a fixed structure  $q(x)$ , and find the optimal weight  $w_j^*$  of leaf  $j = 1, \dots, J$ . It follows,

$$\begin{aligned} \frac{\partial \mathcal{O}^{(t)}}{\partial w_j} = 0 &\iff \left( \sum_{i \in I_j} g_i \right) + \left( \sum_{i \in I_j} h_i + \lambda \right) w_j = 0 \\ &\iff w_j^* = - \frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \end{aligned}$$

Then, the optimal value of the objective function is

$$\mathcal{O}^{(t)*} = -\frac{1}{2} \sum_{j=1}^J \frac{\left( \sum_{i \in I_j} g_i \right)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma J$$

Suppose that  $I_L$  and  $I_R$  are the instance sets of the split's left and right nodes. If we assume that  $I = I_L \cup I_R$ , then the loss reduction after the split is determined by:

$$Gain = \frac{1}{2} \left( \frac{\left( \sum_{i \in I_L} g_i \right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left( \sum_{i \in I_R} g_i \right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left( \sum_{i \in I} g_i \right)^2}{\sum_{i \in I} h_i + \lambda} \right) - \gamma \quad (3.5.7)$$

When the training loss reduction is less than the regularization parameter  $\gamma$ , the gain of the best split might be negative, and the split is ended. But if a maximum depth is specified, the tree grows to that depth, and then all leaf splits with a negative gain are removed recursively (Chen, 2014).

Furthermore, LightGBM is optimized to identify the optimal feature splitting points while also reducing the quantity of samples and features (Ke et al., 2017). Gradient-based One Side Sampling (GOSS) and leaf-wise growth are its two key benefits.

#### 1. Gradient-based One Side Sampling:

Standard GBDT systems must go over every feature of every data point while calculating the information gain for all potential splits. GOSS attempts to address this computational complexity problem. Its key insight is that data instances with stronger gradients play larger roles in information gain computation. So, it preserves data instances with big gradients and randomly picks data with small gradients when determining the optimal split. To do so, GOSS ranks the training instances based on the absolute values of their gradients. Then, it selects the top  $a\%$  of the total instances with the largest gradient (a subset  $A$  is thus obtained) and  $b\%$  of instances from the remaining  $(1-a)\%$  (another subset  $B$  is obtained).

The gradients of the  $b\%$  of instances are multiplied by  $\frac{(1-a)}{b}$ , which amplifies the lower gradients. This guarantees that the original data distribution does not change much, and avoids the following boosting step from totally disregarding these data points with lower gradients. Finally, the instances are splitted based on the projected variance gain  $\hat{V}_j(d)$  over the subset of selected instances:

$$\hat{V}_j(d) = \frac{1}{n} \left( \frac{(\sum_{\mathbf{x}_i \in A_l} k_i + \frac{1-a}{b} \sum_{\mathbf{x}_i \in B_l} k_i)^2}{n_l^j(d)} + \frac{(\sum_{\mathbf{x}_i \in A_r} k_i + \frac{1-a}{b} \sum_{\mathbf{x}_i \in B_r} k_i)^2}{n_r^j(d)} \right)$$

where  $A_l = \{\mathbf{x}_i \in A | x_{ij} \leq d\}$ ,  $B_l = \{\mathbf{x}_i \in B | x_{ij} \leq d\}$ ,

$A_r = \{\mathbf{x}_i \in A | x_{ij} > d\}$ ,  $B_r = \{\mathbf{x}_i \in B | x_{ij} > d\}$ ,

$n_l^j(d) = \{\sum I(\mathbf{x}_i \in A \cup B | x_{ij} \leq d)\}$ ,

$n_r^j(d) = \{\sum I(\mathbf{x}_i \in A \cup B | x_{ij} > d)\}$ ,

and  $\{k_1, \dots, k_n\}$  are the negative gradients of the loss function with respect to the output of the model.

On the subset  $A \cup B$ , for each feature  $j$ , GOSS selects  $d_j^* = \underset{d}{\operatorname{argmax}} \hat{V}_j(d)$  and computes the largest gain  $\hat{V}_j(d_j^*)$ . The data is then split based on the feature  $j^* = \underset{j}{\operatorname{argmax}} \hat{V}_j(d_j^*)$  at the point  $d_{j^*}^*$ .

This technique has proven to be more successful and faster than traditional methods.

## 2. Leaf-wise growth:

In LightGBM, the leaf-wise tree growth chooses the leaf that minimizes loss the most and splits only that leaf, ignoring the rest of the leaves at the same level. As a result, the tree becomes asymmetrical, and additional splitting may occur only on one side of the tree. (Figure 3.3)

The steps below explain how a leaf-wise tree is constructed (Only binary decision trees are considered: every node has a most two branches)(Shi, 2007).

- Choose a feature for the root node and create some branches for it depending on specified criteria.
- Divide the training dataset into subsets, one per branch, out from the root node.
- The preceding step is repeated for the leaf which leads to the lowest loss. For each of the following expansions, only the subset of training data that reaches a considered node should be utilized.
- This building procedure is repeated unless all the nodes are pure or a certain number of growth are attained.

The leaf-wise tree growth technique achieves lower loss than the level-wise growth strategy, but it also overfits, especially in small datasets. So, LightGBM adds a maximum depth boundary on a leaf-by-leaf basis to ensure high efficiency while avoiding overfitting.

There are another two reasons why LightGBM is very fast: Histogram-based splitting and Exclusive Feature Bundling (EFB).

## 1. Histogram-based splitting:

The decision tree task is to discover the best feature dividing points. A very few split

points (bins) may result in information loss, but it can also prevent overfitting in some cases. A high proportion of splits preserves information while increasing training time and processing resources. Iterating over the data points numerous times until the ideal split points that provide the maximum information gain or decrease variance are determined is a standard approach. Rather, LightGBM separates the data into a given number of bins of uniform length, similar to a histogram. The algorithm then iterates across these bins to determine the best split spots. The complexity of building the histogram for all features is  $O(data \times features)$ , and the complexity of identifying the ideal split points after this process is proportional to  $O(bins \times features)$ . In general,  $bins \ll data$ . As a result, this strategy is substantially more extremely fast than the previous one.

## 2. Exclusive Feature Bundling:

The Histogram-based splitting technique requires  $O(bins \times features)$  complexity. The model will speed up tree learning if the *features* can be down-sampled. LightGBM does this by grouping features together. Several attributes of a big data are mutually exclusive (they never take zero values simultaneously). LightGBM detects such features and combines them into a single feature to decrease complexity to  $O(bins \times bundle)$ , with  $bundle \ll features$ .

In order to conduct a comparative study on LightGBM and other predictive models, the Extreme Gradient Boosting (XBoost), Naive Bayes (NB), and Support Vector Classification (SVC) have also been used.

- *Extreme Gradient Boosting for classification*

Like LightGBM, Extreme Gradient Boosting (XGBoost) is an optimized, distributed Gradient-Boosted Decision Tree (GBDT) algorithm that supports parallel tree boosting (Chen and Guestrin, 2016).

Thus, its tree construction process also considers the minimization of the regularized objective function in 3.5.1, in the same way, leading to the Gain obtained in 3.5.7. The key difference between these approaches is how they grow trees. LightGBM uses leaf-wise tree growth, whereas XGBoost uses level-wise tree growth (Figure 3.3). The level-wise technique builds the tree level-by-level. So, it results in symmetric trees. The steps below explain how a leaf-wise tree is constructed (Only binary decision trees are considered: every node has a most two branches)(Shi, 2007)

- Choose a feature for the root node and create some branches for it depending on specified criteria.
- Divide the training dataset into subsets, one per branch, out from the root node.
- The preceding step is repeated level-by-level, from left to right, for example. For each of the following expansions, only the subset of training data that reaches a considered node should be utilized.
- This building procedure is repeated unless a certain number of growth are attained or all the nodes are pure (When all observations at a node have the same class label: splitting is completed and the node becomes a terminal node.)

For small data, level-wise growth is typically preferable, whereas leaf-wise growth may overfit. However, with the big data, leaf-wise growth outperforms level-wise growth since it is quicker.

- *Support Vector Classification:*

Support Vector Classification (SVC), a Support Vector Machine method, is a supervised and linear Machine Learning technique that is often used to solve classification issues (Awad and Khanna, 2015).

Let  $\mathbf{x}_i \in \mathbb{R}^p, i = 1, \dots, n$  be the training features, each belong to class 1 or  $-1$ . A data point is seen as a  $p$ -dimensional vector in SVC and a  $(p - 1)$  hyperplane is used to separate such locations. One feasible alternative for the best hyperplane is the one with the greatest difference between the two classes. Therefore, the hyperplane is chosen in such a way that the distance between it and the closest data point on every side is maximized. This hyperplane is defined as the set of points  $\mathbf{x}$  that fulfill:

$$\mathbf{w}^T \mathbf{x} - b = 0, \quad (3.5.8)$$

where  $\mathbf{w}$  is its normal vector, and  $b$  a constant. To find it, SVC solves the following optimization problem:

$$\begin{aligned} & \text{minimize} \left( \frac{1}{2} \|\mathbf{w}\|^2 + \mathcal{C} \sum_{i=1}^n \zeta_i \right) \\ & \text{subject to} \quad \begin{cases} y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1 - \zeta_i, \\ \zeta_i \geq 0, \forall i = 1, \dots, n. \end{cases} \end{aligned}$$

where  $y_i \in \{-1, 1\}$  are the class labels, and  $\zeta_i = \max \left( 0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b) \right)$  is the smallest non-negative value that satisfies  $y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1 - \zeta_i$ .

The goal is to maximize the margin by minimizing  $\|\mathbf{w}\|^2 = \mathbf{w}^T \mathbf{w}$  while imposing a penalty if a sample is incorrectly classified (Figure 3.4). The value  $y_i(\mathbf{w}^T \mathbf{x}_i - b)$  must be greater than one for a correct prediction. As problems are often not fully separable using a hyperplane, some samples are tolerated to be  $\zeta_i$  away from their correct margin boundary.  $\mathcal{C}$  is an inverse regularization parameter that manages the strength of the penalty (Pedregosa et al., 2011).

The SVC algorithm utilizes a group of mathematical functions known as kernel. A kernel function converts input space into high-dimensional feature space via non-linear projections. It is the inner product of two vectors  $\mathbf{x}_i$  and  $\mathbf{x}_j$  in the feature space  $\phi(\mathbf{x}_i)$  and  $\phi(\mathbf{x}_j)$ :

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \quad (3.5.9)$$

The kernel used in this study is the Gaussian Radial Basis Function (RBF). It is usually utilized in SVC when no previous knowledge of the data is available. The RBF of two samples  $\mathbf{x}_i$  and  $\mathbf{x}_j$  is given by:

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2) \quad (3.5.10)$$

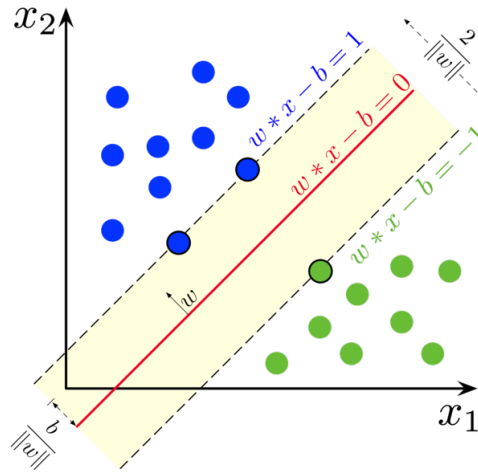


Figure 3.4: Maximum-margin hyperplane (Wikipedia, b)

where  $\gamma = \frac{1}{2\sigma^2}$  with  $\gamma > 0$ ,  $\sigma$  the hyperparameter and  $\|\mathbf{x}_i - \mathbf{x}_j\|^2$  is the euclidean distance between the samples  $\mathbf{x}_i$  and  $\mathbf{x}_j$ .

The RBF kernel is widely used because of its excellent characteristics: it can successfully handle both linear and non-linear input-output mapping.

- *Gaussian Naive Bayes Classifier :*

The Bayes' Theorem — a method for calculating conditional probability of previous knowledge and the naive assumption that each feature is independent of the others — is the foundation of Naive Bayes. The most huge benefit of Naive Bayes is that, whereas most machine learning algorithms require a vast volume of training data, it performs excellently even with little amounts of data (Vijaykumar and Vikramkumar, 2014).

Let  $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  be a set of features, and  $\mathbf{y}$  a class variable. According to the Bayes Theorem:

$$P(\mathbf{y}|\mathbf{x}) = \frac{P(\mathbf{x}|\mathbf{y}) \times P(\mathbf{y})}{P(\mathbf{x})} \quad (3.5.11)$$

The idea is to compute the posterior probability  $P(\mathbf{y}|\mathbf{x})$  from the likelihood  $P(\mathbf{x}|\mathbf{y})$  and prior probabilities  $P(\mathbf{y})$ ,  $P(\mathbf{x})$ . The probability  $P(\mathbf{x}|\mathbf{y})$  can be rewritten as:

$$\begin{aligned} P(\mathbf{x}|\mathbf{y}) &= P(\mathbf{x}_1, \dots, \mathbf{x}_n|\mathbf{y}) \\ P(\mathbf{x}|\mathbf{y}) &= P(\mathbf{x}_1|\mathbf{x}_2, \dots, \mathbf{x}_n, \mathbf{y}) \times P(\mathbf{x}_2|\mathbf{x}_3, \dots, \mathbf{x}_n, \mathbf{y}) \times \dots \times P(\mathbf{x}_n|\mathbf{y}), \end{aligned}$$

using the chain rule.

With the fact that the conditional probabilities are independent of each other under the Naive's conditional independence hypothesis, it follows:

$$P(\mathbf{x}|\mathbf{y}) = P(\mathbf{x}_1|\mathbf{y}) \times P(\mathbf{x}_2|\mathbf{y}) \times \dots \times P(\mathbf{x}_n|\mathbf{y}).$$

Then,

$$\begin{aligned}
 P(\mathbf{y}|\mathbf{x}) &= P(\mathbf{x}|\mathbf{y}) \times \frac{P(\mathbf{y})}{P(\mathbf{x})} \\
 &= P(\mathbf{x}_1|\mathbf{y}) \times P(\mathbf{x}_2|\mathbf{y}) \times \dots \times P(\mathbf{x}_n|\mathbf{y}) \frac{P(\mathbf{y})}{P(\mathbf{x})} \\
 &= \left( \prod_{i=1}^n P(\mathbf{x}_i|\mathbf{y}) \right) \frac{P(\mathbf{y})}{P(\mathbf{x}_1) \times P(\mathbf{x}_2) \times \dots \times P(\mathbf{x}_n)} \\
 P(\mathbf{y}|\mathbf{x}) &= P(\mathbf{y}|\mathbf{x}_1, \dots, \mathbf{x}_n) = \left( \prod_{i=1}^n P(\mathbf{x}_i|\mathbf{y}) \right) \frac{P(\mathbf{y})}{\prod_{i=1}^n P(\mathbf{x}_i)}.
 \end{aligned}$$

Therefore, since  $\prod_{i=1}^n P(\mathbf{x}_i)$  is constant with respect to  $y$  values, it comes:

$$P(\mathbf{y}|\mathbf{x}_1, \dots, \mathbf{x}_n) \propto P(\mathbf{y}) \prod_{i=1}^n P(\mathbf{x}_i|\mathbf{y}). \quad (3.5.12)$$

This approach is used with a decision rule in the Naive Bayes classifier: choose the most likely option. The Maximum Posteriori decision rule is given by:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} P(\mathbf{y}) \prod_{i=1}^n P(\mathbf{x}_i|\mathbf{y}), \quad (3.5.13)$$

where,  $\hat{\mathbf{y}}$  is the predicted value.

The Gaussian Naive Bayes, used in this study, is a Naive Bayes classifier that assumes that features follow a normal distribution. In this case,

$$P(\mathbf{x}_i|\mathbf{y} = c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp\left(-\frac{(\mathbf{x}_i - \mu_c)^2}{2\sigma_c^2}\right), \quad \forall i = 1, \dots, n \quad (3.5.14)$$

where  $\mu_c$  and  $\sigma_c$  are the mean and variance of  $\mathbf{x}_i$  calculated for a given class  $c$  of  $\mathbf{y}$ .

## 3.6 Hyperparameter Tuning

The Hyperparameter Tuning is a method by which the best parameters of a model are chosen. The measurement can be made according to the Accuracy of the model or the Area Under the Receiver Operator Characteristic curve (AUC-ROC curve), for each list of candidate parameters. Because a model's parameter space may contain real-valued or unbounded value spaces for certain parameters, manually setting boundaries and discretization is usually required before using certain Hyperparameter Turning techniques.

A simpler technique is to evaluate all possible combinations of parameters in the sets of values of the considered parameters (*Grid search*). But, in this study, *Random search* was used. Instead of evaluating all potential parameter combinations, *Random Search* chooses them at random in every iteration. It can surpass *Grid Search*, notably when only a few hyperparameters impact the model's optimum performance.

## 3.7 Evaluation Metrics:

As Fawcett and Provost (1999) pointed out, the purpose of early identification of anomalous phenomena is not just to categorize each positive or negative subsequence. Rather, the primary task is to detect an impending abnormal event in a timely way. Therefore, following Cerqueira et al. (2020) to evaluate the performance of the predictive model, four metrics were assessed first: Event Recall (ER), Reduced Precision (RP); the average number of False Alarms (aveFA), and the average Anticipation Time (aveAT).

**3.7.1 Event Recall (ER).** The Recall of a model assesses its ability to recognize positive cases. The more positive cases identified, the larger the Recall. In this study, a case corresponds to a whole subsequence instead of an individual observation. Therefore, the classical Recall metric can be misleading. An example to better understand is the case where the model identifies several individual positive observation in the same subsequence. The classical Recall metric would consider these predictions as different predictions while they refer to the same event. To avoid this, the Event Recall measure considers only the first predicted positive observation of a subsequence as sufficient to classify this subsequence as a positive case. During the computation of the Recall, this approach then ignores any other positive alarm following the first one, in the same subsequence.

The following equation calculates the  $ER$  for a model  $m$ :

$$ER_m = \frac{\hat{T}_m}{T}, \quad (3.7.1)$$

where  $\hat{T}_m$  denotes the total number of the correctly predicted events, and  $T$  is the total number of the main events in the test data.

**3.7.2 Reduced Precision (RP).** The Precision of a model is described as the proportion of properly recognized positive cases (True Positive) to the overall number of positively classified cases (either True Positive or False Positive). Because of the scenario stated in Event Recall (ER), the classical Precision metric can be misleading. So, RP also considers the number  $\hat{T}_m$  of the correctly predicted events as the number of properly recognized positive cases. Similarly, consider the scenario where several false alarms appear in the same subsequence. RP avoids making a mistake by considering the first false alarm as sufficient to count the corresponding subsequence as a False Positive case. So, the RP metric changes the number of False Positives with the number of Discounted False Positives: the number of subsequences related to the False Positive alarms.

The following equation gives the  $RP$  for a model  $m$ :

$$RP_m = \frac{\hat{T}_m}{\hat{T}_m + DFP_m} \quad (3.7.2)$$

where  $DFP_m$  is the number of Discounted False Positives.

**3.7.3 Average number of False Alarms (aveFA).** A high-performance model must avoid false alarms as much as possible. So the smaller the average False Alarms, the efficient the model. It

is calculated with the following formula:

$$aveFA = \frac{numFA}{numFPC}, \quad (3.7.3)$$

with  $numFA$ , the total number of False Alarms (False Positive Observations) and  $numFPC$  the number of False Positive Cases.

**3.7.4 Average Anticipation Time (aveAT):** The average Anticipation Time (aveAT) is the average interval of time left in advance by the model to correctly predict a Positive Case. Each anticipation time is calculated with respect to the start of the Warning Window. So, the bigger the  $aveAT$  of a model, the stronger it is.

It is calculated as follows:

$$aveAT = \frac{sumAT}{TPPC} \quad (3.7.4)$$

where  $sumAT$  is the summation of the Anticipation Times, and  $TPPC$  is the total number of the True Predicted Positive Cases.

It is now clear that an efficient model, in addition to having a long Average Anticipation Time must also have in a balanced way, its Event Recall and Reduced Precision high. It should easily predict the main events in time and also avoid false alarms as much as possible. To easily identify such a model, another metric has been added: *Event F1-score (EF1-score)*.

**3.7.5 Event F1-score (EF1-score).** The F1-score is the harmonic mean of Precision and Recall. Similarly, the Event F1-score is defined here as the harmonic mean of the Reduced Precision and the Event Recall. It uses the following formula to merge RP and ER into a single value:

$$EF1\text{-score} = 2 \times \frac{ER \times RP}{ER + RP}. \quad (3.7.5)$$

The goal is to maximize both Event Recall and Reduced Precision such that given pairs of their values for various models, one could compare and determine which is the best.

Note that in this study, a model is considered to be a highly efficient algorithm for the early prediction of Acute Hypotensive Episodes and Tachycardia Episodes, if it presents a high Event F1-score (EF1-score) with a large average Anticipation Time (aveAT), and a low average False Alarms.



## 4. Results

The findings of this study on the timely prediction of the Acute Hypotensive Episodes (AHE) and the Tachycardia Episodes (TE) are presented in this section. First, the whole dataset was examined before performing the feature selection. The whole data contains 111 features and after the feature selection processes, almost half were eliminated. This technique makes it possible to evaluate whether performing a feature selection is necessary or not in order to improve the model's performance. As previously stated in Subsection 3.4, two distinct feature selection approaches were used: Mutual Information Gain for classification (considering the Mutual Information Maximization (MIM) technique) and LightGBM Feature Importance Ranking. So, there are three sorts of outcomes for both early prediction of AHE and TE, expressed in terms of the mean and the standard deviation of each Evaluation Metric. Note that the average Anticipation Time (aveAT) was computed here in minutes for each model.

### 4.1 AHE Early Prediction

Considering the patient data in each of the 5 folds obtained after a 5-fold Cross-Validation, the statistics presented in the Tables 4.1 and 4.2 were obtained for the Train set and the Test set.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
AHE					
Observation Windows	2718	2337	2497	2757	2535
Main Events	286	188	206	255	241

Table 4.1: Train data for AHE

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
AHE					
Observation Windows	903	1277	916	603	1153
Main Events	13	122	94	43	63

Table 4.2: Test data for AHE

So each fold of the Training set contains at least 2,337 Observation Windows (2,337 hours) with at least 188 of them related to the Acute Hypotensive Episodes. In addition, each fold of the Test set contains at least 603 Observation Windows (603 hours) with at least 13 of them related to the Acute Hypotensive Episodes.

**4.1.1 Full Dataset:.** After evaluating the performance of the predictive models XGBoost, LightGBM, SVC and Naive Bayes, the average results obtained across the 5 folds are presented in the Table 4.3.

Models	ER	RP	aveFA	aveAT	EF1-score
XGBoost	0.687±0.12	0.364±0.177	2.585±0.836	46.781±7.69	0.462±0.184
LightGBM	0.695±0.122	<b>0.37±0.18</b>	<b>2.376±0.677</b>	46.813±6.426	<b>0.469±0.186</b>
SVC	0.648±0.245	0.067±0.035	25.18±11.531	45.939±7.896	0.12±0.063
NB	<b>0.946±0.05</b>	0.116±0.047	21.256±5.235	<b>56.595±1.611</b>	0.204±0.077

Table 4.3: AHE: Average Results on the Full Data (mean ± std)

The Naive Bayes has the highest Event Recall ( $0.946 \pm 0.05$ ) and the highest average Anticipation Time ( $56.595 \pm 1.611$ ). However, with a high average False Alarms and low Reduced Precision, it also has the lowest Event F1-score ( $0.204 \pm 0.077$ ) behind the Support Vector Classification ( $0.12 \pm 0.063$ ). Meanwhile, LightGBM has the highest Reduced Precision ( $0.38 \pm 0.185$ ) and the lowest average False Alarms ( $2.419 \pm 0.683$ ). Although it ranks second behind Naive Bayes for the Event Recall and the average Anticipation Time, it holds the highest Event F1-score ( $0.478 \pm 0.189$ ). The XGBoost model, on the other hand, is behind LightGBM for each metric.

**4.1.2 Feature Selection with LightGBM:.** Using the *Split importance* method for the LightGBM feature importance ranking, features with non-zero importance were selected. Some information about the selected features by Fold are provided below:

- Fold 1: 20 features selected, with the most important being *meanMAP* (the statistical feature denoting the mean of the Mean Arterial Blood Pressure).  
*minMAP* and *maxMAP* are in the top 5.
- Fold 2: 23 features selected, with the most important being *minDBP* (the statistical feature denoting the minimum of the Diastolic Blood Pressure).  
*minMAP* and *maxMAP* are in the top 5.
- Fold 3: 18 features selected, with the most important being *minMAP* (the statistical feature denoting the minimum of the Mean Arterial Blood Pressure).  
*meanMAP* is also in the top 5.
- Fold 4: 22 features selected, with the most important being *meanMAP* (the statistical feature denoting the mean of the Mean Arterial Blood Pressure).  
*maxMAP* and *medianMAP* are also in the top 5.
- Fold 5: 16 features selected, with the most important being *minMAP* (the statistical feature denoting the minimum of the Mean Arterial Blood Pressure).  
*meanMAP* and *medianMAP* are also in the top 5.

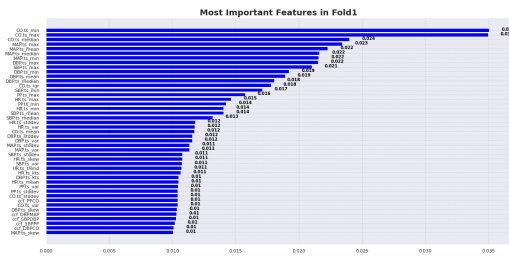
The average results obtained across the 5 folds are then presented in the Table 4.4:

Models	ER	RP	aveFA	aveAT	EF1-score
XGBoost	0.686 $\pm$ 0.124	0.359 $\pm$ 0.178	2.684 $\pm$ 0.685	47.374 $\pm$ 6.63	0.458 $\pm$ 0.189
LightGBM	0.708 $\pm$ 0.095	<b>0.38<math>\pm</math>0.185</b>	<b>2.419<math>\pm</math>0.683</b>	48.668 $\pm$ 2.932	<b>0.478<math>\pm</math>0.189</b>
SVC	0.834 $\pm$ 0.04	0.079 $\pm$ 0.033	34.063 $\pm$ 6.254	55.663 $\pm$ 3.745	0.143 $\pm$ 0.057
NB	<b>0.959<math>\pm</math>0.031</b>	0.14 $\pm$ 0.053	16.672 $\pm$ 4.527	<b>55.693<math>\pm</math>3.371</b>	0.24 $\pm$ 0.081

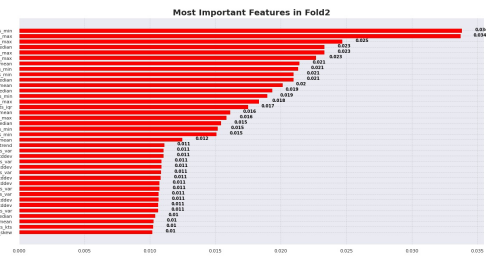
Table 4.4: AHE: Average Results After LightGBM Feature Selection (mean  $\pm$  std)

The LightGBM presents the highest Reduced Precision ( $0.38 \pm 0.185$ ) and the lowest False Alarm ( $2.419 \pm 0.683$ ) in front of the XGBoost. Although it ranks third for Event Recall and average Anticipation Time, behind Naive Bayes and Support Vector Classification, it holds the highest Event F1-score ( $0.478 \pm 0.189$ ).

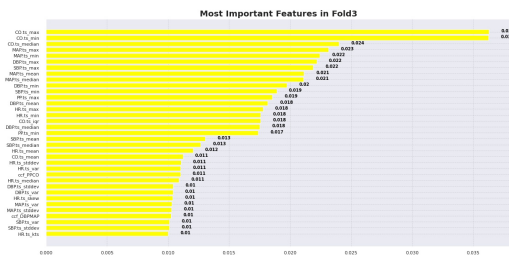
**4.1.3 Feature Selection with MIG:.** Each feature was considered with the target variable to calculate the MIG corresponding to them. After normalizing the computed feature importance, features with at least 1% (0.01) importance were selected. The figures below show the distributions of the selected features per fold.



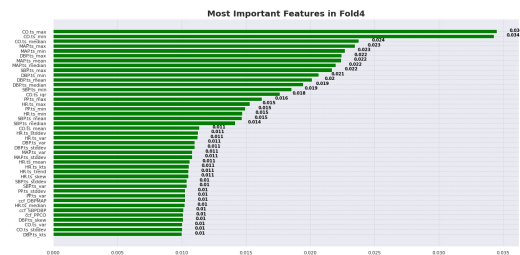
(a) Fold 1: 45 features selected, with the most important being *minCO* (the statistical feature denoting the minimum of the Cardiac Output). *maxMAP*, and *meanMAP* are also in the top 5.



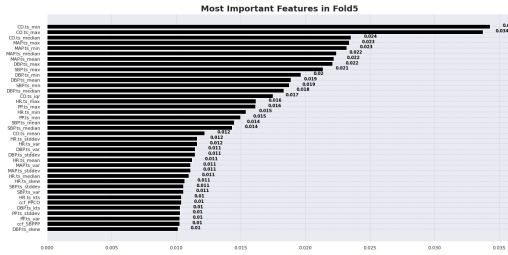
(b) Fold 2: 38 features selected, with the most important being *minCO* (the statistical feature denoting the minimum of the Cardiac Output). *maxMAP* is also in the top 5.



(a) Fold 3: 35 features selected, with the most important being *maxCO* (the statistical feature denoting the maximum of the Cardiac Output). *maxMAP*, and *minMAP* are also in the top 5.



(b) Fold 4: 43 features selected, with the most important being *maxCO* (the statistical feature denoting the maximum of the Cardiac Output). *maxMAP*, and *minMAP* are also in the top 5.



(a) Fold 5: 39 features selected, with the most important being *minCO* (the statistical feature denoting the maximum of the Cardiac Output). *maxMAP*, and *minMAP* are also in the top 5.

Figure 4.3: AHE: Selected features per fold with MIG

The Table 4.5 presents the average results obtained across the 5 folds:

Models	ER	RP	aveFA	aveAT	EF1-score
XGBoost	0.726±0.142	0.347±0.167	3.176±0.7	48.941±4.628	0.458±0.19
LightGBM	0.703±0.129	<b>0.405±0.177</b>	<b>2.128±0.336</b>	47.744±6.39	<b>0.502±0.191</b>
SVC	0.68±0.191	0.065±0.031	30.367±11.069	49.455±6.088	0.118±0.055
NB	<b>0.967±0.03</b>	0.114±0.048	22.909±5.136	<b>56.798±1.646</b>	0.2±0.078

Table 4.5: AHE: Average Results After MIG Feature Selection (mean ± std)

The LightGBM, once again, has the greatest Reduced Precision ( $0.405 \pm 0.177$ ), the lowest average False Alarms ( $2.128 \pm 0.336$ ) and completely dominates the Event F1-score with  $0.502 \pm 0.191$ . It is, however, in the last position for the average Anticipation Time ( $47.744 \pm 6.39$ ) and in the third position for the Event Recall ( $0.703 \pm 0.129$ ) behind the Naive Bayes and the XGBoost.

**4.1.4 Run-time Analysis.** The Table 4.6 shows the average running time (in seconds) of the 4 models for the prediction of Acute Hypotensive Events.

Models	Full Data	MIG Selection	LightGBM Selection
XGBoost	10.382±4.453	8.441±1.724	20.038±5.507
LightGBM	4.24±1.114	1.564±0.347	0.816±0.123
SVC	11.169±4.967	5.852±3.1	3.204±1.083
NB	0.263±0.037	0.026±0.006	0.057±0.013

Table 4.6: Average Running Time (in seconds) for AHE Prediction (mean ± std)

It is observed that the LightGBM model has a very low running time, behind the Naive Bayes model.

## 4.2 TE Early Prediction

The Tables 4.7 and 4.8 show the number of Observation Windows contained in each fold, as well as the number of associated main events, for the Train set and the Test set.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
AHE					
Observation Windows	2718	2337	2497	2757	2535
Main Events	554	519	610	658	435

Table 4.7: Train data for TE

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
AHE					
Observation Windows	903	1277	916	603	1153
Main Events	155	380	95	57	285

Table 4.8: Test data for TE

Thus, each fold of the Training set contains at least 2337 Observation Windows (2337 hours) with at least 435 of them related to the Tachycardia Episodes. In addition, each fold of the Test set contains at least 603 Observation Windows (603 hours) with at least 57 of them related to the Tachycardia Episodes.

**4.2.1 Full Dataset:.** The average results obtained across the 5 folds for the predictive models XGBoost, LightGBM, SVC and Naive Bayes, are presented in the Table 4.9.

Models	ER	RP	aveFA	aveAT	EF1-score
XGBoost	<b>0.853±0.058</b>	0.583±0.099	4.843±1.373	<b>54.272±3.466</b>	0.691±0.087
LightGBM	<b>0.853±0.045</b>	<b>0.601±0.105</b>	<b>4.323±1.09</b>	53.945±3.743	<b>0.703±0.086</b>
SVC	0.772±0.11	0.229±0.082	20.285±3.224	52.646±2.21	0.342±0.094
NB	0.633±0.161	0.398±0.109	6.589±3.604	50.676±4.171	0.47±0.09

Table 4.9: TE: Average Results on the Full Data (mean ± std)

The LightGBM model shares first place with XGBoost for the Event Recall (0.853 in mean). But it outperforms all others for Reduced Precision (0.601±0.105), average False Alarms (4.323±1.09) and Event F1-score (0.703 ± 0.086). XGBoost exceeds it in terms of average Anticipation Time (54.272 ± 3.466 against 53.945 ± 3.743).

**4.2.2 Feature Selection with LightGBM:.** The features with non-zero importance were selected, using the *Split importance* method for the LightGBM feature importance ranking. Some information about the selected features by Fold are provided below:

- Fold 1: 26 features selected, with the most important being *medianDBP* (the statistical feature denoting the median of the Diastolic Blood Pressure).  
*minHR* and *meanHR* are also in the top 5.
- Fold 2: 6 features selected, with the most important being *meanHR* (the statistical feature denoting the mean of the Heart Rate).  
*minHR*, *maxHR* and *medianHR* are also in the top 5.
- Fold 3: 7 features selected, with the most important being *meanHR* (the statistical feature denoting the mean of the Heart Rate).  
*minHR*, *maxHR*, *medianHR* and *slopeHR* are also in the top 5.
- Fold 4: 35 features selected, with the most important being *minHR* (the statistical feature denoting the minimum of the Heart Rate).  
In the top 10, *slopeHR* and *maxHR* are also present.
- Fold 5: 6 features selected, with the most important being *meanHR* (the statistical feature denoting the mean of the Heart Rate).  
*minHR*, *maxHR*, *medianHR* and *slopeHR* are also in the top 5.

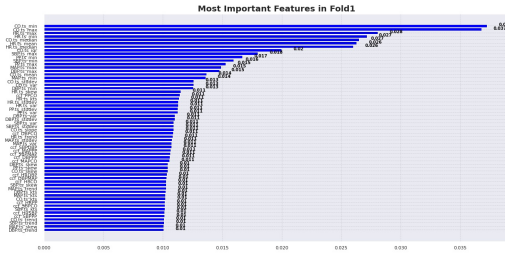
The average results obtained across the 5 folds are shown in the Table 4.10:

Models	ER	RP	aveFA	aveAT	EF1-score
XGBoost	0.845±0.053	<b>0.604±0.107</b>	4.417±1.217	54.303±2.806	<b>0.702±0.088</b>
LightGBM	0.851±0.043	0.595±0.118	<b>4.232±1.091</b>	53.886±3.359	0.696±0.095
SVC	0.466±0.12	0.365±0.198	10.014±6.954	50.923±3.928	0.358±0.087
NB	<b>0.895±0.103</b>	0.441±0.141	9.689±3.12	56.32±2.262	0.584±0.14

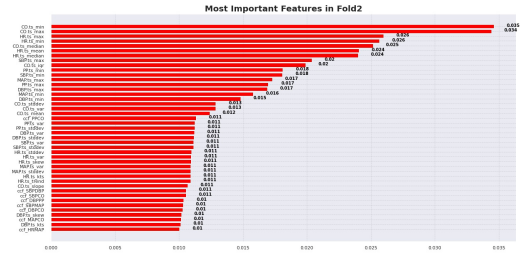
Table 4.10: TE: Average Results After LightGBM Feature Selection (mean ± std)

The LightGBM holds the lowest average False Alarms ( $4.232 \pm 1.091$ ) but only came second for Event Recall ( $0.851 \pm 0.043$ ) behind Naive Bayes, Reduced Precision ( $0.595 \pm 0.118$ ) and Event F1-score ( $0.696 \pm 0.095$ ) behind XGBoost. It is also at the third position for the average Anticipation Time, ( $53.886 \pm 3.359$ ) behind the XGBoost and the Naive Bayes models.

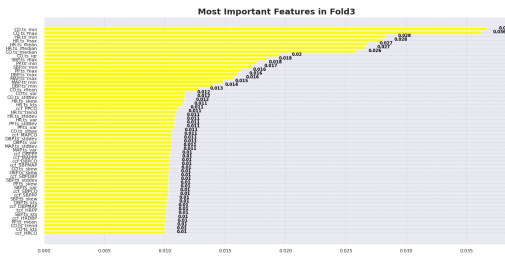
**4.2.3 Feature Selection with MIG:.** Using the normalized MIG feature importance, features with at least 1% (0.01) importance were selected. The distributions of the selected features per fold are shown in the figures below:



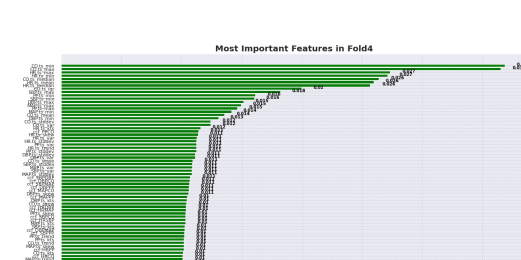
(a) Fold 1: 60 features selected, with the most important being *minCO* (the statistical feature denoting the minimum of the Cardiac Output). *minHR* and *maxHR* are also in the top 5.



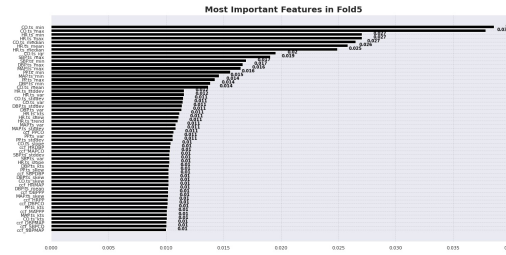
(b) Fold 2: 43 features selected, with the most important being *minCO* (the statistical feature denoting the minimum of the Cardiac Output). *minHR* and *maxHR* are also in the top 5.



(a) Fold 3: 54 features selected, with the most important being *minCO* (the statistical feature denoting the minimum of the Cardiac Output). *minHR*, *maxHR* and *meanHR* are also in the top 5.



(b) Fold 4: 63 features selected, with the most important being *minCO* (the statistical feature denoting the minimum of the Cardiac Output). *minHR* and *maxHR* are also in the top 5.



(a) Fold 5: 54 features selected, with the most important being *minCO* (the statistical feature denoting the minimum of the Cardiac Output). *minHR* and *maxHR* are also in the top 5.

Figure 4.6: TE: Selected features per fold with MIG

The average results obtained across the 5 folds are shown in the Table 4.11:

Models	ER	RP	aveFA	aveAT	EF1-score
XGBoost	<b>0.853±0.058</b>	0.583±0.099	4.843±1.373	<b>54.272±3.466</b>	0.691±0.087
LightGBM	<b>0.853±0.045</b>	<b>0.601±0.105</b>	<b>4.323±1.09</b>	53.945±3.743	<b>0.703±0.086</b>
SVC	0.772±0.11	0.229±0.082	20.285±3.224	52.646±2.21	0.342±0.094
NB	0.633±0.161	0.398±0.109	6.589±3.604	50.676±4.171	0.47±0.09

Table 4.11: TE: Average Results After MIG Feature Selection (mean ± std)

The LightGBM model completely dominated the other models with almost all metrics, with notably an Event F1-score of  $0.703 \pm 0.086$ ! It ranks second for the average Anticipation Time behind the XGBoost model ( $53.945 \pm 3.743$ ).

**4.2.4 Run-time Analysis.** The Table 4.12 shows the average running time (in seconds) of the 4 models for the prediction of Tachycardia Events.

Models	Full Data	MIG Selection	LightGBM Selection
XGBoost	12.641±1.835	12.532±2.579	22.451±14.734
LightGBM	3.173±0.19	2.28±0.406	4.657±7.151
SVC	49.233±17.467	40.396±19.571	15.941±8.596
NB	0.402±0.07	0.084±0.017	0.078±0.036

Table 4.12: Average Running Time (in seconds) for TE Prediction (mean ± std)

Once again, it is observed that the LightGBM model has a very low running time behind the Naive Bayes model.



## 5. Discussion

The objective of this study was to provide a highly efficient approach for the early prediction of the Acute Hypotensive Episodes and the Tachycardia Episodes. Not only should such an approach have a large average Anticipation Time, but it should also primarily have a very high Event F1-score. This expresses the model's ability to predict Critical Health Episodes earlier with great efficiency, while avoiding false alarms as much as possible. The proposed AHE and TE early prediction system (MIG-LightGBM) includes the Mutual Information Gain feature selection process and the LightGBM predictive model.

According to the results on the early prediction of AHE, this approach can capture up to 70% of the Acute Hypotensive Events at more than 1 hour 47 minutes before their appearance (a gap of 1 hour had been considered) while maintaining the highest EF1-score of 50% compared to other methods. When it comes to predicting TE, it can capture up to 85% of the Tachycardia Events at more than 1 hour 53 minutes before their appearance while maintaining the highest and remarkable EF1-score of 70%. Although Naive Bayes has a high average Anticipation Time, it has the major drawback of launching several False Alarms as much as it predicts several Critical Health Episodes. The same is true for Support Vector Classification. As for Extreme Gradient Boosting, it does better than the last two, but was overtaken by LightGBM.

The comparison of the performance of the Layered Learning approach (LL) suggested by [Cerqueira et al. \(2020\)](#) and the MIG-LightGBM method is presented in Table 5.1 for AHE early prediction.

	ER	RP	aveFA	aveAT	EF1-score
MIG-LightGBM	0.703±0.129	<b>0.405±0.177</b>	<b>2.128±0.336</b>	<b>47.744±6.39</b>	<b>0.502±0.191</b>
LL	<b>0.830±0.054</b>	0.205±0.044	3.3±9.1	46.9±3.3	0.328±0.056

Table 5.1: MIG-LightGBM vs LL for AHE Early Prediction

The MIG-LightGBM approach completely dominates the LL method with the main efficiency measurement metrics (EF1-score, aveAT, and aveFA). It notably exceeds LL by about 20% on both Event F1-score and Reduced Precision.

Moreover, LL has an average run-time of  $102.3 \pm 12.5$  seconds, while MIG-LightGBM has only  $1.564 \pm 0.347$  seconds.

For TE early prediction, the Table 5.2 shows the comparison of the performance of the Layered Learning approach and the MIG-LightGBM method.

	ER	RP	aveFA	aveAT	EF1-score
MIG-LightGBM	0.858±0.054	<b>0.602±0.108</b>	<b>4.285±1.146</b>	<b>53.131±4.182</b>	<b>0.705±0.088</b>
LL	<b>0.938±0.027</b>	0.136±0.018	35.9±8.8	53.0±2.2	0.237±0.027

Table 5.2: MIG-LightGBM vs LL for TE Early Prediction

As in the case of AHE, the MIG-LightGBM approach completely dominates the LL method with the main efficiency measurement metrics for TE early prediction. It notably exceeds LL by about 50% on both EF1-score and RP.

About the average execution time, LL ( $70.9 \pm 4.9$ ) takes more than 68 seconds longer than MIG-LightGBM ( $2.28 \pm 0.406$ ).

The proposed method, MIG-LightGBM, is therefore a highly efficient approach for the early prediction of AHE and TE. Its greatest strength is its ability to identify numerous Critical Health Episodes earlier while avoiding false alarms as much as possible, and more than existing methods. It gives ample time for vital actions to be taken after an impending alarm, and its warnings are highly reliable. Systems that avoid false alarms are highly recommended in healthcare. They avoid alarms fatigue and therefore constitute an excellent improvement in ICU results and the living conditions of nursing staff. Alarm fatigue occurs when busy personnel are subjected to a high volume of regular safety warnings and become insensitive to them. In 2020, [Lewandowska et al. \(2020\)](#) investigate on the impact of alarm fatigue on the work of nurses in an intensive care Environment. Frequent monitoring alerts may reduce caregivers' capabilities to react when an alarm signals an emergency that demands immediate attention. This desensitization might result in longer response times or the ignoring of vital alerts, causes of deaths in ICUs.

However, MIG-LightGBM has a main limitation. By trying to avoid false alarms as much as possible, it can lose sight of a few Critical Health Episodes. This study focused much more on improving the model's ability to avoid false alarms while predicting CHEs early and efficiently. To perfect the proposed warning system, it would also be good to significantly increase the model's ability not to miss the CHEs while maintaining the current strength.

## 6. Conclusion

This study focused on building a highly effective early warning system for the Critical Health Episodes such as Acute Hypotensive and Tachycardia. This system is able to predict them earlier with great efficacy, while avoiding false alarms as much as possible. With a high Reduced Precision, a large average Anticipation Time and a very strong Event F1-score on the MIMIC II dataset, the proposed method, MIG-LightGBM, dominates the existing approaches. Its performance is therefore reliable. In practical situations, it can be applied on patients to make a reliable early prediction on Critical Health Episodes, having knowledge of at least their latest one-hour observations. This system also has the benefit of being able to avoid alarms fatigue. But because of its Event Recall which is not extremely high, it can sometimes omit some Critical Health Episodes. A direct future work could therefore focus on strengthening the system's ability to identify CHEs without losing its current strength.

# References

- 2009 PhysioNet. Predicting acute hypotensive episodes: The physionet/computing in cardiology challenge 2009. George Moody, <https://physionet.org/content/challenge-2009/1.0.0/>, Accessed April 2022.
- Mohammed Nabih Ali, EL-Sayed A El-Dahshan, and Ashraf H Yahia. Denoising of heart sound signals using discrete wavelet transform. *Circuits, Systems, and Signal Processing*, 36(11): 4482–4497, 2017.
- Andrew Zhu. Select features for machine learning model with mutual information. Andrew Zhu, <https://towardsdatascience.com/select-features-for-machine-learning-model-with-mutual-information-534fe387d5c8>, Accessed May 2022.
- Mariette Awad and Rahul Khanna. *Support Vector Machines for Classification*, pages 39–66. 01 2015. ISBN 978-1-4302-5989-3. doi: 10.1007/978-1-4302-5990-9\_3.
- Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. *The journal of machine learning research*, 13(1):27–66, 2012.
- Vitor Cerqueira, Luis Torgo, and Carlos Soares. Early anomaly detection in time series: a hierarchical approach for predicting critical health episodes. *arXiv preprint arXiv:2010.11595*, 2020.
- Tianqi Chen. Introduction to boosted trees. *University of Washington Computer Science*, 22 (115):14–40, 2014.
- Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.
- Gari D Clifford, Daniel J Scott, and Mauricio Villarroel. User guide and documentation for the mimic ii database (version 2, release 1). 2010.
- Thomas M Cover, Joy A Thomas, et al. Entropy, relative entropy and mutual information. *Elements of information theory*, 2(1):12–13, 1991.
- Timothy Derrick and Joshua Thomas. Time series analysis: the cross-correlation function. 2004.
- Stefanos Fafalios, Pavlos Charonyktakis, and Ioannis Tsamardinos. Gradient boosting trees. 2020.
- Tom Fawcett and Foster Provost. Activity monitoring: Noticing interesting changes in behavior. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 53–62, 1999.
- Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

- Min Gan, Shunqi Pan, Yongping Chen, Chen Cheng, Haidong Pan, and Xian Zhu. Application of the machine learning lightgbm model to the prediction of the water levels of the lower columbia river. *Journal of Marine Science and Engineering*, 9(5):496, 2021.
- Marzyeh Ghassemi. *Methods and models for acute hypotensive episode prediction*. PhD thesis, Oxford University, UK, 2011.
- Dazhi Jiang, Liyu Li, Bo Hu, and Zhun Fan. An approach for prediction of acute hypotensive episodes via the hilbert-huang transform and multiple genetic programming classifier. *International Journal of Distributed Sensor Networks*, 11(8):354807, 2015.
- Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.
- Joon Lee and Roger G Mark. An investigation of patterns in hemodynamic data indicative of impending hypotension in intensive care. *Biomedical engineering online*, 9(1):1–17, 2010a.
- Joon Lee and Roger Greenwood Mark. A hypotensive episode predictor for intensive care based on heart rate and blood pressure time series. In *2010 Computing in Cardiology*, pages 81–84. IEEE, 2010b.
- Katarzyna Lewandowska, Magdalena Weisbrot, Aleksandra Cieloszyk, Wioletta Medrzycka-Dabrowska, Sabina Krupa, and Dorota Ozga. Impact of alarm fatigue on the work of nurses in an intensive care environment—a systematic review. *International Journal of Environmental Research and Public Health*, 17(22):8409, 2020.
- Xiaoli Liu, Tongbo Liu, Zhengbo Zhang, Po-Chih Kuo, Haoran Xu, Zhicheng Yang, Ke Lan, Peiyao Li, Zhenchao Ouyang, Yeuk Lam Ng, et al. Top-net prediction model using bidirectional long short-term memory and medical-grade wearable multisensor system for tachycardia onset: algorithm development study. *JMIR Medical Informatics*, 9(4):e18803, 2021.
- Abdelaziz Merghadi, Ali P Yunus, Jie Dou, Jim Whiteley, Binh ThaiPham, Dieu Tien Bui, Ram Avtar, and Boumezbeur Abderrahmane. Machine learning methods for landslide susceptibility studies: A comparative overview of algorithm performance. *Earth-Science Reviews*, 207: 103225, 2020.
- George B Moody and Roger G Mark. A database to support development and evaluation of intelligent intensive care monitoring. In *Computers in Cardiology 1996*, pages 657–660. IEEE, 1996.
- Lidan Mu, Lujie Chen, Joo Yoon, Gilles Clermont, and Artur Dubrawski. Predicting the onset of tachycardia for patients in intensive care units.
- Gerard Nkundimana. *Intensive care unit (ICU) outcomes in Rwanda and the US: clinical course, morbidity and mortality*. PhD thesis, University of Rwanda, 2017.

- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- Hanyang Peng and Yong Fan. Feature selection by optimizing a lower bound of conditional mutual information. *Information Sciences*, 418:652–667, 2017.
- PhysioNet. A challenge from physionet and computers in cardiology 2009. Webots, <https://archive.physionet.org/challenge/2009/>, Accessed April 2022.
- Mohammed Saeed, Mauricio Villarroel, Andrew T Reisner, Gari Clifford, Li-Wei Lehman, George Moody, Thomas Heldt, Tin H Kyaw, Benjamin Moody, and Roger G Mark. Multiparameter intelligent monitoring in intensive care ii (mimic-ii): a public-access intensive care unit database. *Critical care medicine*, 39(5):952, 2011.
- Omar AM Salem, Feng Liu, Ahmed Sobhy Sherif, Wen Zhang, and Xi Chen. Feature selection based on fuzzy joint mutual information maximization. *Mathematical Biosciences and Engineering*, 18(1):305–327, 2021.
- Segyeong, Hyojeong Lee, Soo-Yong Shin, Myeongsook Seo, Gi-Byoung Nam, and Joo. Prediction of ventricular tachycardia one hour before occurrence using artificial neural networks. *Scientific reports*, 6(1):1–7, 2016.
- Haijian Shi. *Best-first decision tree learning*. PhD thesis, The University of Waikato, 2007.
- Elad Tsur, Mark Last, Victor F Garcia, Raphael Udassin, Moti Klein, and Evgeni Brotfain. Hypotensive episode prediction in icu via observation window splitting. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 472–487. Springer, 2018.
- B Vijaykumar and Trilochan Vikramkumar. Bayes and naive-bayes classifier. 2014 *arXiv:1404.0933*, 2014.
- Gary M Weiss and Haym Hirsh. Learning to predict rare events in event sequences. In *KDD*, volume 98, pages 359–363, 1998.
- Wikipedia. Mutual information. Webots3, [https://en.wikipedia.org/wiki/Mutual\\_information](https://en.wikipedia.org/wiki/Mutual_information), Accessed May 2022a.
- Wikipedia. Support-vector machine. Webots4, [https://en.wikipedia.org/wiki/Support-vector\\_machine#Primal](https://en.wikipedia.org/wiki/Support-vector_machine#Primal), Accessed May 2022b.
- Wikipedia. Feature engineering. Webots5, [https://en.wikipedia.org/wiki/Feature\\_engineering](https://en.wikipedia.org/wiki/Feature_engineering), Accessed May 2022c.
- Joo Heung Yoon, Lidan Mu, Lujie Chen, Artur Dubrawski, Marilyn Hravnak, Michael R Pinsky, and Gilles Clermont. Predicting tachycardia as a surrogate for instability in the intensive care unit. *Journal of Clinical Monitoring and Computing*, 33(6):973–985, 2019.

---

Joo Heung Yoon, Vincent Jeanselme, Artur Dubrawski, Marilyn Hravnak, Michael R Pinsky, and Gilles Clermont. Prediction of hypotension events with physiologic vital sign signatures in the intensive care unit. *Critical Care*, 24(1):1–9, 2020.