Frequency and Time Domain Feature Engineering and Predictive Modeling Based on ECG, SpO2, and Respiration Signals

Luiz Fernando Medeiros

School of Science

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 24.2.2020

Supervisor

Prof. Esa Ollila

Advisor

Dr. Kaveh Samiee

Aalto University
School of Science

Copyright © 2020 Luiz Fernando Medeiros

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Luiz Fernando Medeiros

Title Frequency and Time Domain Feature Engineering and Predictive Modeling Based on ECG, SpO2, and Respiration Signals

Degree programme Computer Science

Major Computer Science **Code of major** SCI3042

Supervisor Prof. Esa Ollila

Advisor Dr. Kaveh Samiee

Date 24.2.2020 **Number of pages** 77+1 **Language** English

Abstract

Today, there exists a challenge in simplifying biosignals into features that are well suited for machine learning and clinician understanding. This work reports the feature engineering exercise involved with such challenge, along with the predictive modeling. We primarily tackle ECG, Respiration (Thoracic Impedance), and SpO2 (Plethsmographic) signals extracted from a proprietary dataset used by GE Healthcare. Throughout the study, we analyze biosignals while searching for general characteristics which may help describe (and even highlight) human function for a machine learning model, while maintaining clinical value. Wave Morphology Analysis in the Time Domain, Wavelet Decomposition and Fast Fourier Transforms were the main methods explored for feature engineering. Finally, results from a Convolutional Neural Network and a Random Forest model are reported, whereby the best performing model is able to predict Sepsis with 77% accuracy at least three (3) hours in advance.

Keywords Signal Processing, Feature Engineering, Machine Learning, Healthcare

Preface

I want to thank first and foremost, my wonderful wife for the unconditional support and love throughout this whole journey. Thank you! Then, I have to also thank my friends, supervisor and advisers, whose support was essential throughout this whole process. Thank you!

Otaniemi, 05.5.2019

Luiz F. F. Medeiros

Contents

Symbols and abbreviations

Symbols

$\mathbb{Z}$	Set of Integers
$a \leftarrow b$	Setting variable value b to variable a
$\Re$	Set of real Numbers
f	frequency related variable

Operators

$\frac{\mathrm{d}}{\mathrm{d}t}$	derivative with respect to variable t
$\frac{\partial}{\partial t}$	partial derivative with respect to variable t
$\sum_i$	sum over index i
$\mathbf{A} \cdot \mathbf{B}$	dot product of vectors $\mathbf{A}$ and $\mathbf{B}$
$x[n]$	Discrete data or signal
$x(t)$	Continuous or analog data

Abbreviations

EHR	Electronic Health Record
EMR	Electronic Medical Record
ECG	Electrocardiogram
SpO2	Plethysmographic Wave
Resp	Respiration Signal
ICU	Intensive Care Unit
QRS	Peaks Q, R and S from an ECG beat
AWS	Amazon Web Services
EC2	Elastic Compute Cloud
S3	AWS storage system
CPU	Central Processing Unit
GPU	Graphics Processing Unit
RAM	Random Access Memory
HW	Hardware
UML	Unified Modeling Language
OOP	Object Oriented Programming

1 Introduction

Today, there exists a wealth of information that is captured in the healthcare domain. In fact, if one focuses solely on Intensive Care Units (ICUs), as it was done on this study, a patient is generally being monitored by at least three different devices: ECG, SpO2, and Impedance Respiration monitoring devices. In addition, a numerous other types of data (lab results, imaging work, etc...) are recorded in the hospital's Electronic Health Record (EHR) system. This information is then available for medical professionals to assess and produce patient recovery plans. Yet, in spite of all of this monitoring and information availability, patients still deteriorate and perish due to the difficulty that clinicians have to mine through all of the data available.

An important special case of this happens with patients that suffer from Sepsis. Septic patients contract Sepsis as a consequence of some infection that happens (in this case we are only considering patients in ICUs). As a consequence of the generic nature, doctors generally produce treatment plans that are not fully aware of the condition. Consequently, patient's deteriorate rapidly [1], causing irreparable damages [2] [3].

Because of the health damages that may be caused by this disease and the availability of data in a scenario such as the ICU, GE Healthcare (GEHC) Finland saw an opportunity to study whether or not it may be possible to see patterns and perhaps predict this deterioration early enough, in order to provide a relevant care plan for patients and eventually save lives. *The ultimate goal of this study is then to investigate whether it may be possible to leverage the data which is most commonly available to clinicians and help not only foresee the disease before its too late, but also understand health deterioration.* For this specific study, Sepsis was chosen as the reference disease due to its relevance and ability to go unnoticed in hospital care.

In the following sections of this chapter, Sepsis will be discussed in more depth, along with the approach that was conceived to pursue this study.

1.1 Sepsis and Health Deterioration

Sepsis is a disease that affects millions of people in the hospitals around the world. In fact, Sepsis leads the rank in causes of death during a hospital stay [1]. In addition, the longer a patient goes without being properly diagnosed, the more likely the patient is to sustain lifelong impairment (given that the patient survives) [1].

The costs of the disease are also staggering. In 2013 Sepsis earned the spot for most costly disease in the U.S., adding 24 billion dollars to health care related costs in American hospitals. The damages and staggering costs create space for this study, in which methods to detect the disease early enough are explored and developed. In machine learning terms, the aim is to predict the event (of having Sepsis) before it has traditionally being diagnosed and logged in the EHR systems.

From the medical perspective, and how Sepsis may be identified in the data that is available, the disease's presence is not so stark, specially in the early stages of its development. The general conditions are: fever, rapid breathing and heart rate, confusion, and disorientation [2]. As one may suspect, the symptoms themselves are

quite generic. However, for an expecting eye (meaning a trained clinician), they may be apparent in the data.

Rapid breathing and heart rate, along with confusion and disorientation are conditions that can be detected by monitoring devices capturing Impedance Respiration, Electric Cardiogram, Electrical Encephalogram, or even Electrical Myograms. In addition, Plethysmographic waves (SpO2) can also demonstrate some of these conditions.

In addition to the general conditions mentioned, Sepsis emerges from infection. This infection may be due to complex conditions, such as pneumonia, or something as simple as a paper cut [1]. But even conventional infections may produce indicators in the *parameters*[1] mentioned in the previous paragraph, as they may cause internal bleeding and debilitation on the surrounding organs.

It is based on the above logic, that there exists cues in the data coming from the various medical devices and information systems monitoring the patient, that the idea of using high frequency data (waveforms) and Electronic Medical Records (EHR) emerges.

1.2 Proposed Approach

The hospital's EHR system allows one to identify when a patient was diagnosed, what was the diagnosis, lab results retrieved during that period, as well as consult any waveform that has been stored for that specific patient. The approach proposed here builds on the ability to access waveform data, or electric biosignals, along with information about a patient's diagnoses (as targets), and the predictive potential of machine learning algorithms in the medical science domain [4]. *The goal is then to predict sepsis at least 1-3 hours before it happens, therefore allowing clinicians time to act on the information .*

[1]Parameters is the term used in the Healthcare Device industry for biosignals such as ECG, Respiration and SpO2

2 Background

In this section, the general background regarding the methods and data that will be used throughout this work will be presented, along with a description on what a predictive problem in machine learning is about. In addition, there will also be a justification on why and how these ideas tie in with the current medical practice.

2.1 Predictive Learning

As mentioned in the introductory chapter, the current environment that will be studied is composed of a specific hospital scenario, namely ICU, a disease named Sepsis, and a whole lot of data. The goal is to use this data within the context provided to help find out whether a patient will contract Sepsis before it has been historically found. In addition, if possible, provide some clinical explainability so clinicians can learn from the decisions provided.

The above can be formulated in a predictive problem for a machine learning algorithm that uses Supervised Learning to pair input data $\mathbf{x}$ to expected outputs y [5]. Ideally, the algorithm is able to learn from a training set $\mathbf{X}$, whilst maintaining a sufficiently generic predictive capability such to perform well on an unseen set $\mathcal{T}$ (generally referred to as test sets).

More specifically to the context and goal pursued herein, the input data $\mathbf{x}$ is some combination of explainable features extracted from the biosignals that were recorded at least 1-3 hours before some information about the target diagnose y was provided to the EHR system. But before any suitable input set $\mathbf{x}$ is created containing features that may be useful for clinicians, a general $\mathbf{x}_{raw}$ can be retrieved directly from the hospitals' systems. Figure 1 should help illustrate what the raw input resembles.

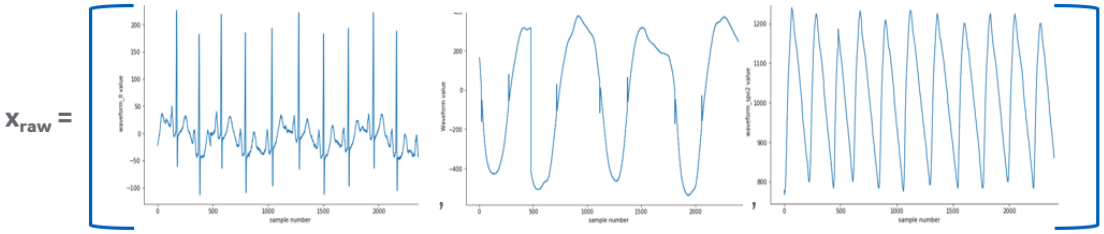


Figure 1: Figure illustrating that the raw input $\mathbf{x}_{raw}$ is composed of information from the different waveforms.

However, in order to reach a more ideal $\mathbf{x}$, that has a lower dimension, it was necessary to observe what a clinician actually looks for when attempting to diagnose Sepsis, since what is being used as a target is a clinical note indicating that the patient was considered Septic at time t.

2.2 General Data and Biosignals: EHR and Waveforms

By consulting clinicians a set of general notes were created regarding what composes a septic condition, and which sorts of signals were most readily available in the EHR system. The notes were as follows:

1. Clinicians often dissect the waveform they are observing into features (or use software packages provided for these purposes)

2. Specific waveforms seem to be more readily available and consulted by clinicians, namely: ECG, SpO2, Respiration, and Continuous Blood Pressure information [3] [2]

3. The volume of information that is created and collected in ICUs make it difficult for clinicians to analyze the data by hand

With the above points in mind, it was then decided that: in order to reach higher probability of success, the machine learning algorithm should use only information that is readily available. In addition, due to the dimension the combined set of waveforms, the input, should also be dissected into features which clinicians understand.

2.2.1 EHR

EHR data comes in the form of tables containing specific information regarding patients. This information then travels with the patient throughout his/hers hospital stay. Beyond a hospital stay, the data is stored for future use and reference, whether it may be for a future stay, or research, such as what is being done in this study [6].

For this particular work, there was a necessity to traverse an EHR table containing the diagnoses information for the chosen cohort. On this diagnoses table, information such as diagnosis name, date, and patient specific information was retrieved. This was then cross matched with the waveform information for that specific patient, so to produce a timeline in the patient's stay that contained identified diagnoses dates and waveform events. Below is an example of a subset of columns which such a table may contain:

Table 1: Table demonstrating a subset of columns that may be found in an EHR diagnoses table

encounter_id	patient_id	diagnosis_start_date_unix	diagnosis_name
12345	67891	1363132800	Sepsis(995.91)

Using data from Table 1, it is possible to use the unix epoch time stamp, encounter and patient id information and pick the exact date for which that diagnosis, in this example Sepsis, was imputed by a clinician.

2.2.2 Waveforms

The second data format used were waveforms, or data sampled through medical monitoring devices. These devices are generally attached to the patient, and they sample information at some regular interval throughout the patient's stay in the ICU. The data is then sent through the Hospital's network system to a local server that stores this information. This information is then available to the staff of the hospital.

For this experiment, the waveforms had to go through a series of conversion processes. Initially the waveforms are sampled by their respective devices. The sampling frequencies were as follows:

- $f_{ECG} = 240\text{Hz}$
- $f_{RR} = 60\text{Hz}$
- $f_{SpO2} = 60\text{Hz}$

The two latter waveforms were then up-sampled to 240Hz, so all three waveforms contained the same number of samples for a given moment in time. Since this work was mainly an offline research venture, the waveforms were translated into SQL tables which contained the following columns:

Table 2: Table demonstrating a subset of columns that may be found in the waveform tables in the [7]

patient_id	epoch_ms	Variable	sample_number	value
12345	1363132800	II	3	100

The format shown above pertains strictly to [7]. Datasets such as [8] contain a different scheme of storing waveforms data. See the MIT Mimic III dataset [9] as reference.

Now, the initial size of the raw dataset was very large. A specific waveform containing about 48 hrs worth of recording could be as large as 10Gbs. This created a significant burden in loading the waveforms for processing. As a consequence, mid-tables were created that contained less information. Namely, a reduction in dimension and gain in access times was achieved by reformatting the tables to the above set of columns, where "patient_id" and "Variable" were placed in the name of the file. This reduced the size of the file by 40 %.

In addition to the above notes, it is also important to mention that each waveform used was not ingested as a whole. The waveforms were stripped from the original set in order to comply with a set of delimiting rules. These are:

Definition 2.2.1.

- *Every waveform group, e.x., a set containing an ECG, Respiration and SpO2 signal, must contain strips that correspond to each other in time. Therefore,*

start time t_s and end time t_f of each waveform used to create a $\mathbf{x}_{raw}$ should be synced.

- *The waveforms should correspond to some standard sampling window*

$$\mathbf{t}^{ECG}, \mathbf{t}^{RR}, \mathbf{t}^{SpO2} = [t_s, t_f, t_{g,s}, t_{g,f}, t_{d,s}, t_{d,f}]$$

t_s: *Start time of sampling,*

$t_f, t_f > t_s$: *End time of sampling,*

$t_{g,s}, t_{g,s} > t_f$: *Start time of gap,*

$t_{g,f}, t_{g,f} > t_{g,s}$: *End time of gap,*

$t_{d,s}, t_{d,s} > t_{g,f}$: *Start time of diagnosis (in EHR),*

$t_{d,f}, t_{d,f} > t_{d,s}$: *End time of diagnosis (in EHR)*

- *$t_f - t_s = \tau$ is a fixed sampling duration. The same τ should be used throughout the experiment.*

Therefore, whenever a particular waveform is mentioned, or a group of waveforms that have been chosen to be used for feature engineering (or whose features have been selected as input), a background work that satisfies the rules presented in definition 2.2.1. In order to better support the above statements, see Figure 2.

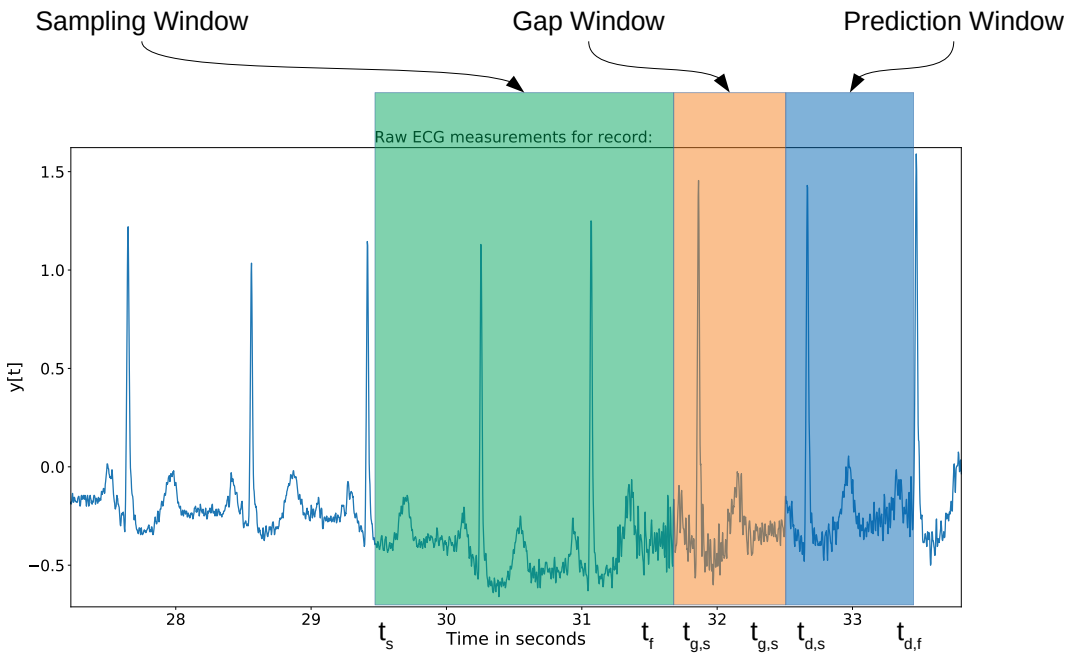


Figure 2: The windows are defined per definition 2.2.1. The waveform used as an example is a raw ECG waveform from MIT BIH dataset [8].

2.2.3 ECG

ECG or Electrocardiogram, is an electrical measurement of the heart's activity. It is the most widely used, and consequently available, type of electrical biosignal in medical environments. The ECG signals used in this research were retrieved from an Internal Care Units (ICU).

The human heart (illustrated in Figure 3) produces a pumping action. That action is activated by electrical pulses that emerge in a point called the Sinoatrial Node [10]. As the electrical pulse propagates through the heart, heart muscles get activated and contract, creating the pumping action.

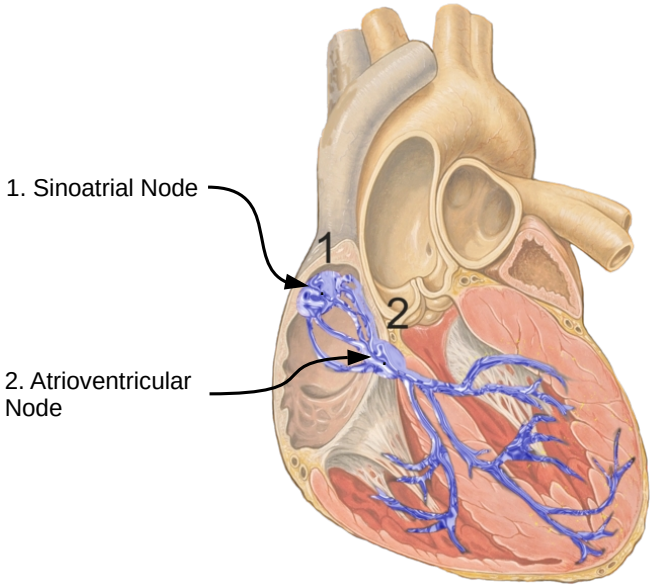


Figure 3: The human heart, along with the Sinoatrial and Atrioventricular Nodes. Heart illustration from [11]

By placing electrical sensors at specific points of the body, it possible to create different view of this action. This is called Holter Monitoring [10]. See Figure 4 for an illustration showing the various points of interest in the body that help retrieve the electrical information from the heart.

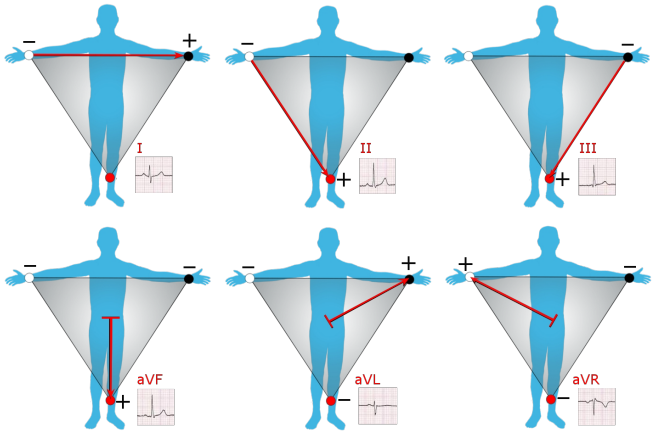


Figure 4: By placing electrodes in different areas of the body, it is possible to create different views [10]. Image credited to [12]

By utilizing different leads and reference points, one is able to get a picture on the electrical activity that is controlling the circulation of blood. Much more than that, it is possible to see how the heart tissues are responding, delays in response between different heart areas, and much more. Moreover, ECGs can be extracted in different number of leads (or views): 3, 6, 12, 32, or 128 in order to produce different perspectives of the heart. It all depends on the type of information and knowledge that the physician is trying to extract from the patient. Figure 5 helps visualize the monitoring leads positioned in the chest for a 12 lead ECG.

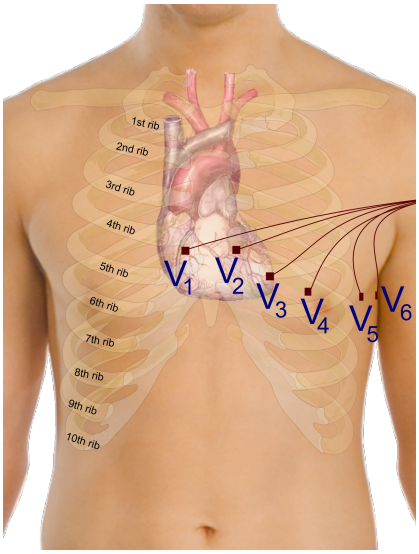


Figure 5: In order to produce the 12 lead ECG [10], numerous leads are placed on the chest of the patient. Image credited to [12]

For the purposes of this research, Lead II was chosen to be the representative view of the heart (or the ECG signal). Lead II is equivalent to the following:

$$II = V_{LL} - V_{RA} \tag{1}$$

Figure 4 illustrates the electrode positioning of Lead II. Taking Lead II was an arbitrary choice. It was partly filled by the performance of the QRS and beat identification algorithm (see subsubsection 3.3.1 for further details). Testing against the MIT BIH dataset produced good results, whose data was extracted from a Modified Lead II. In addition, there was a goal to keep the work simple, by first experimenting with one lead, and eventually (in future works) evaluate the performance with multiple leads.

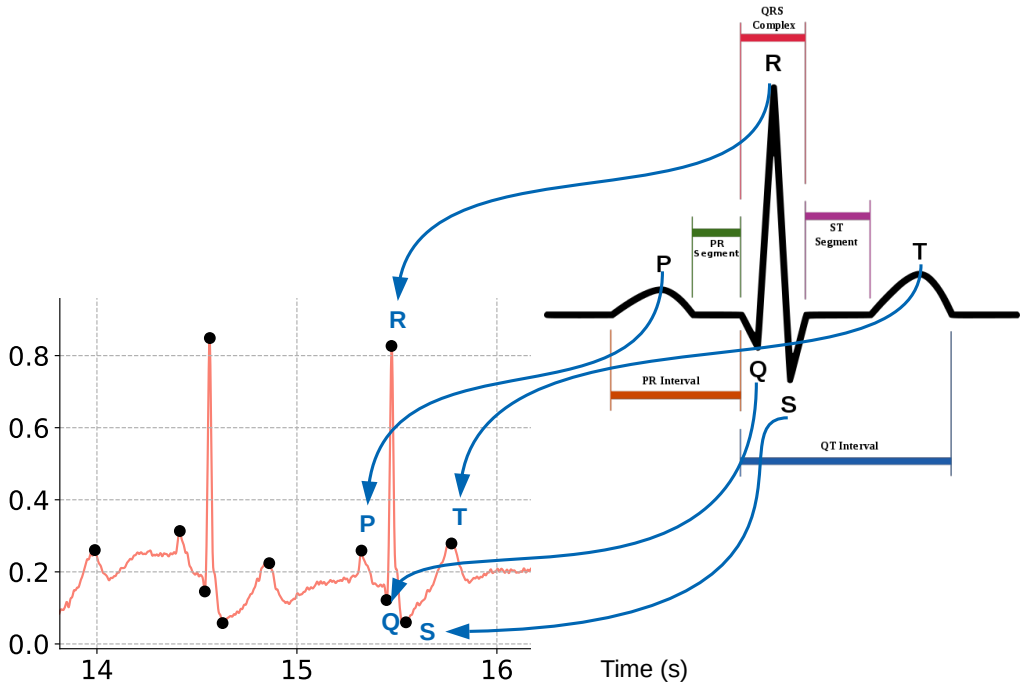


Figure 6: An ECG signal with annotations pointing to the PQRST peaks. In addition to the general peaks, segments and their time differentials are also considered, so to analyze different portions of the heart are responding to the Sinoatrial impulse. These are: PR interval and segment, QRS Complex, QT Interval, and ST Segment.

2.2.4 SpO2

SpO2 signals refer to the data collected by the Pulse Oximetry devices used in hospitals. In this specific case, the data was collected by an SpO2 device installed in an ICU (as previously mentioned, the scenario in which all data for this work was collected is the ICU).

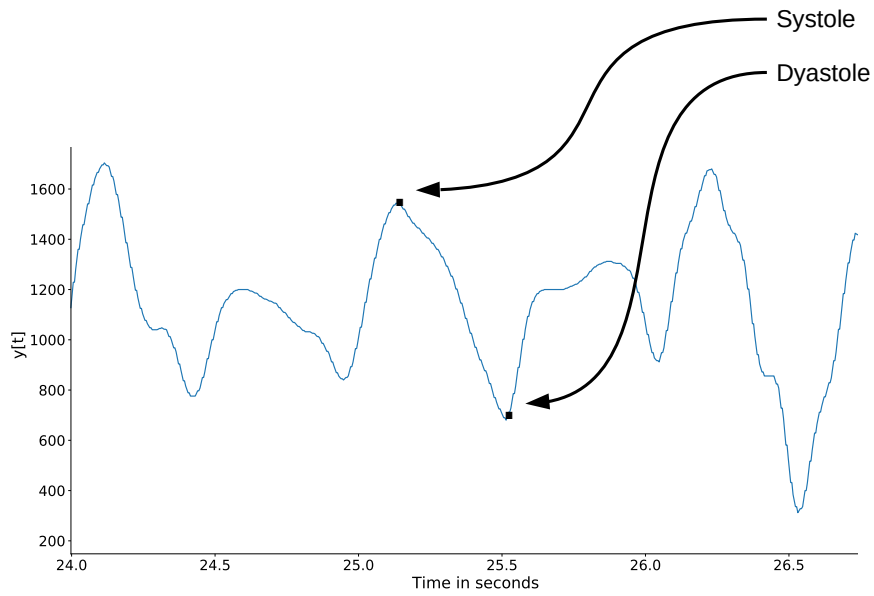


Figure 7: Raw SpO2 signal attained from the GEHC dataset. It contains the Systole and Dyastole peak annotated. These denote the max and min of light absorption by the patient.

SpO2 signals generally refers to an indication of how light is being absorbed by the blood. A sample of blood which is saturated with oxygen will demonstrate different absorbance compared to a sample which is not saturated with oxygen [13]. In more specific terms, this is Oximetry by Spectrophotometry, where the oxygen molecules connected to the haemoglobin (see Figure 8) produce a different depth of color in the blood, thereby allowing one to differentiate a sample that is saturated with oxygen vs a sample that is deficient with blood.

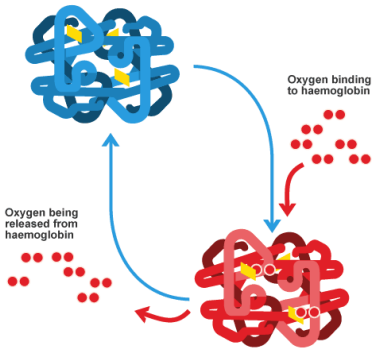


Figure 8: Oxygen binding to haemoglobin [14]

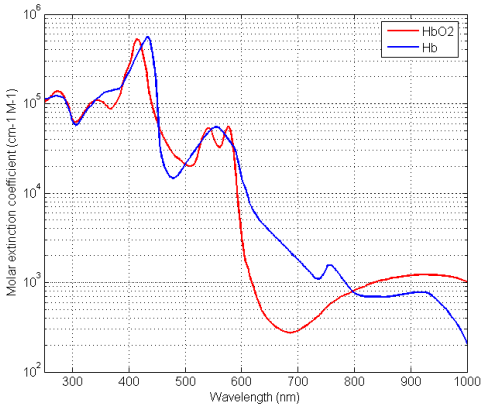


Figure 9: Curve demonstrating saturated (HbO2) and non saturated blood (Hb) [15]

Figure 9 illustrates that at a certain wavelength, there is a clear distinction

between values.

SpO2 signals (as seen in Figure 7) have become progressively more common due to the ease of adaptation with current mobile devices. One can produce a low quality SpO2 signal with a commercially available camera phone. However, the SpO2 signals used for this research belonged to a GEHC device installed in an ICU.

2.2.5 Respiration

The respiratory signal is in fact an impedance measurement across the chest. It's purpose is to derive information about lung function, general ventilation and air volume changes in the lungs [16]. These signals can generally be derived from an ECG monitoring device by selecting specific leads and applying a reference signal.

Lead	Composition
I	LA – RA
II	LL – RA
II	LL – LA

Figure 10: Image showing the physical electrode placement for transthoracic impedance measurement using an ECG monitoring device [17]

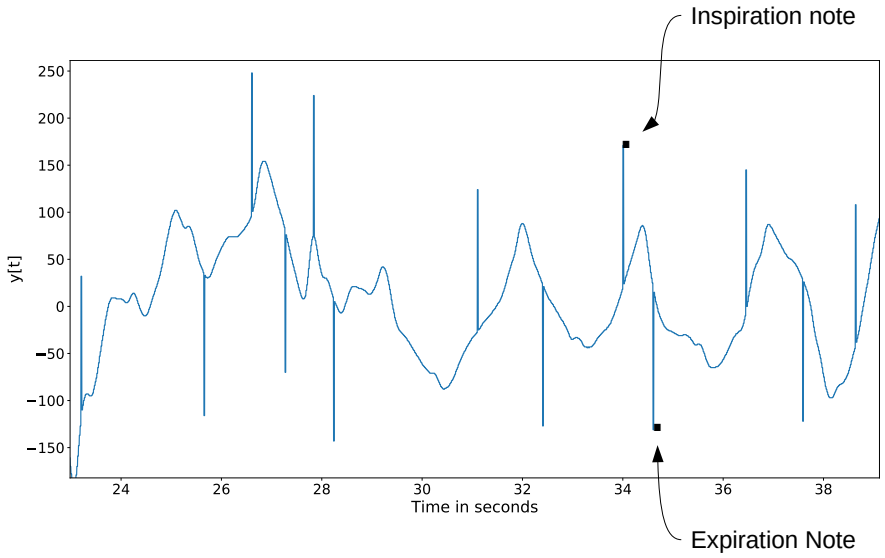


Figure 11: The raw Respiration signal used included inspiration and expiration annotation (artifacts created in the signal by the monitor) which facilitated the calculation of respiration rate.

Respiration signals (as seen in Figure 11) allows physicians to gain insight into volume of air, elasticity of the lungs, rate of response by the respiratory system, and general knowledge on the patient's breathing mechanisms. This is crucial information when attempting to determine whether a patient is becoming pneumonic or undergoing any significant internal infection, as the lungs generally get flooded by liquids.

The Respiration signals used in this dataset were extracted by GEHC monitors that were installed in the same ICU as the previous two waveforms mentioned. It contains, as noted in Figure 10, notes about inspiration and expiration events of the patient. These notes are used to calculate respiration rate and a number of other features mentioned in the next chapter.

2.3 Datasets

2.3.1 MIT BIH Dataset

The MIT BIH Dataset [8] was chosen to validate the feature engineering algorithms developed for ECG. It contains ECG information that is labeled on a beat per beat basis. It is also considered the defacto dataset to validate ECG algorithms.

The dataset contains recorded ECG waveforms from 49 different patients containing different type of conditions. The data was digitized at 360 Hz, therefore containing a different sampling frequency as the dataset used for Sepsis prediction (GEHC). A convenience found with this dataset is that most of the data is preprocessed, there is software available to handle waveforms, and the whole dataset is not more than a few hundred megabytes (depending on how one expands the waveforms).

The criteria used to select patients was as follows: randomly select patients that have rare, but relevant cardiac clinical phenomena. Each recording is 30 minutes long, extracted from patients in the Beth Israel Hospital Arrythmia Laboratory.

2.3.2 GEHC Proprietary Dataset

Information regarding the data structure of the the GEHC dataset has already been mentioned in 2.2.1 and 2.2.2. Beyond this information, it should be noted that this dataset contains information from patients who were admitted to an Intensive Care Unit (ICU).

The entire dataset was initially comprised of about 2300 patients. Due to conditions mentioned in the Exploratory Data Analysis (EDA) section 4 the number of patients used for this research decreased dramatically.

The selection criteria for the dataset was in general: patients admitted to the ICU during a certain time frame. Because the dataset is private, the specifics cannot be disclosed.

The dataset is comprised of different types of waveform data, including Continuous Blood Pressure, SpO2, ECG, Respiration, EMG, EEG, among other signals. Not all signals are available for every patient. The most available group of signals is ECG, SpO2 and Respiration. As a consequence to this, and the fact that most consumer devices can also capture these signals, ECG, SpO2, and Respiration waveforms were chosen for this study.

In contrast to the MIT BIH dataset [8], this dataset contained variable length recordings, with as much as weeks worth of recorded waveform information, and no pre-processing (containing several discontinuous strips), and was 7 terabytes (Tb) large. This incurred a significant extra work, which involved figuring out the structure of the dataset (EHR and Waveforms) and processing it in the most timely fashion in order to have the work concluded within the timeline of a Master's Thesis work.

2.4 Software Design and Engineering

Since this research entailed an applied solution, with a significant amount of software engineering from the very beginning, a number of concepts and tools were chosen at the start. These concepts and tools helped develop the research and exercise concepts studied throughout the academic program. This section first describes the tools used, then the software engineering concepts applied in the project. Finally, notes regarding how all of these tools and ideas helped shape the results achieved.

2.4.1 Infrastructure and Development Tools

To begin with, the list of tools used for this project includes programming languages, Interactive Development Environments (IDEs) and infrastructure setups (see Figure 12 for an illustration of the services). The language of choice for the development of all the tools was Python. This was mainly due to the following reasons:

- Syntax readability (close to writing pseudo code)
- Computational support (Numpy, Scikit-Learn, Pytorch packages)
- General Data organization support (Pandas, Pickle and OS packages)
- Data visualization and prototyping support (Matplotlib, Jupyter Notebooks, Dash and Plotly)

After choosing the programming language, the next tool to consider was the IDE to be used. From the very beginning, the idea was to create tools that would be as close to production quality as possible. For these purposes, IDEs such as Pycharm provide great support by giving syntax highlighting, comprehensive search and refactoring options, version control support, and project organization. In addition, the Professional version of it also provides support for remote development. Consequently, Pycharm Professional was chosen. Figure 12 helps illustrate the logos and tools used to produce the development infrastructure.

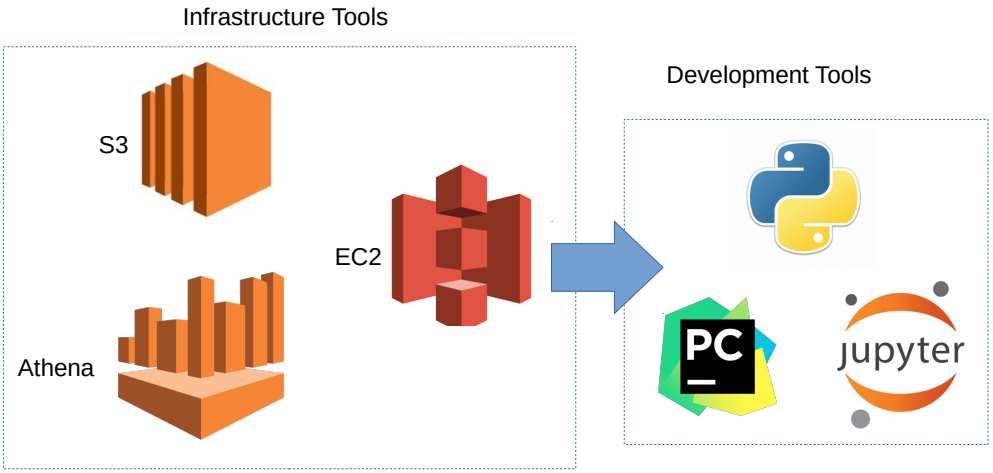


Figure 12: Infrastructure and development tools chosen

AWS	Amazon Web Services
S3	AWS Simple Storage Services
EC2	Elastic Compute Cloud
Athena	DBMS like service from AWS
Boto3	Python library that allows communications with AWS tools

The GEHC dataset contained several terabytes of data. Initially, it was contained 7.1 Tb. However, this quickly expanded as waveforms had to be sliced and diced, features computed, and supporting data structures created (for example, tables to keep track of continuous strips of waveforms (section 4). In addition, for the computation of features, which produced the mapping between $\mathbf{x}_{raw} \mapsto \mathbf{x}$, a significant amount of resources were necessary. In order to achieve this, and be able to scale according to the tasks at hands, a development environment was created within the Amazon Web Services (AWS) environment (see Figure 13 to get a glimpse in the environment). Specific services used included EC2 instances, to create environments with appropriate GPU and CPU computation capabilities, as well as Athena and S3, to easily access data and information.

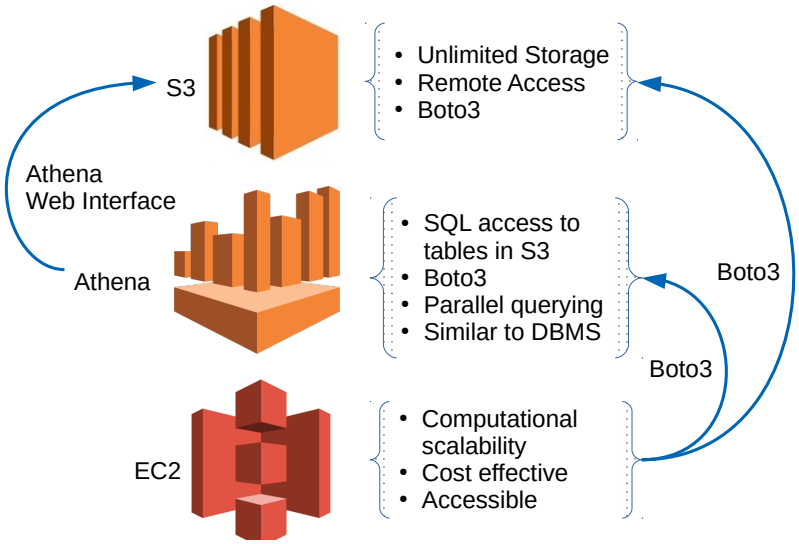


Figure 13: Noting the benefits of the AWS services used, along with the direction of access.

Now,as mentioned in the previous paragraph, the GEHC dataset was very large (7.1Tb). As described in subsubsection 5.2.1, the hardware required to accomplish tasks such as feature extraction, data analysis, data warehousing, and model training varied widely. For this reason AWS EC2 machines were used, such to scale hardware according to the task at hand, while maintaining the same software stack. Figure 14 illustrates the idea.

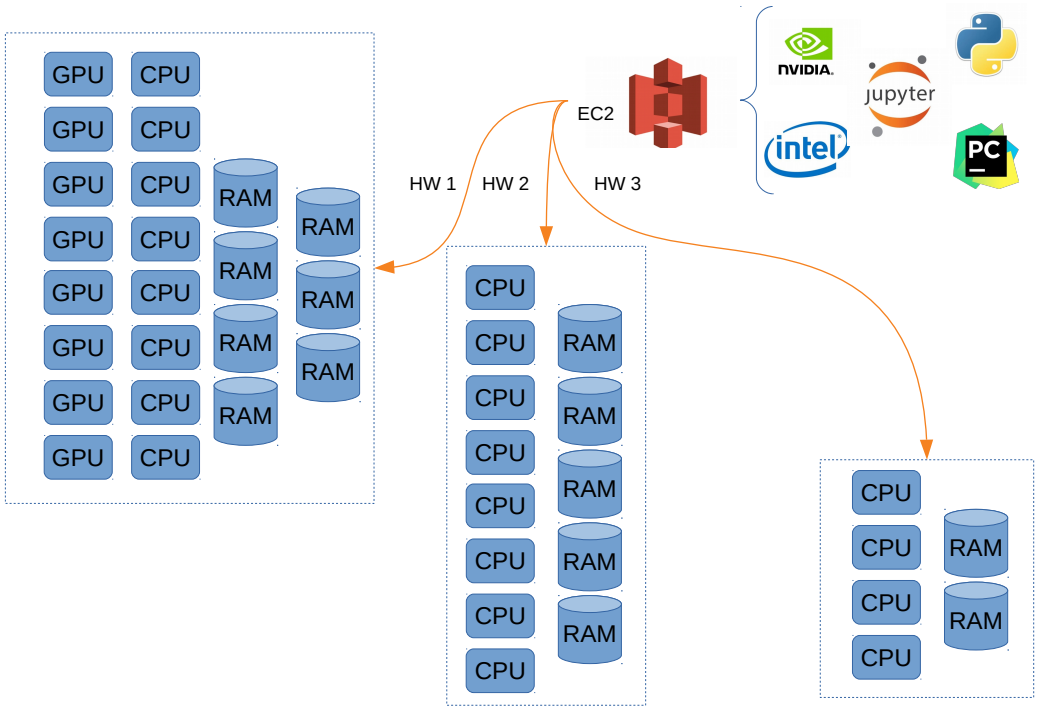


Figure 14: Illustrating how one is able to have multiple hardware configuration, while maintaining one software stack

2.4.2 Fundamentals Used

Moving on to the software engineering concepts, two main ideas were used during the design phase of this project: Object-Oriented Design patterns [18], Automata Theory [19], and general concepts in algorithms [20]. The latter concepts were used rather indirectly, as they contain some of the fundamental ideas that either complement or help understand the logic behind object oriented design (OOD).

The Unified Modeling Language (UML) is a sub-concept of OOD patterns. It is also a very helpful grammar to use when tackling a problem. During this research, Class Diagrams and State Diagrams were developed.

Class Diagrams are a class of graphical notation that allows one to specify class names, attributes and methods, along with inheritance notations. Figure 15 depicts a graph containing a high level definition:

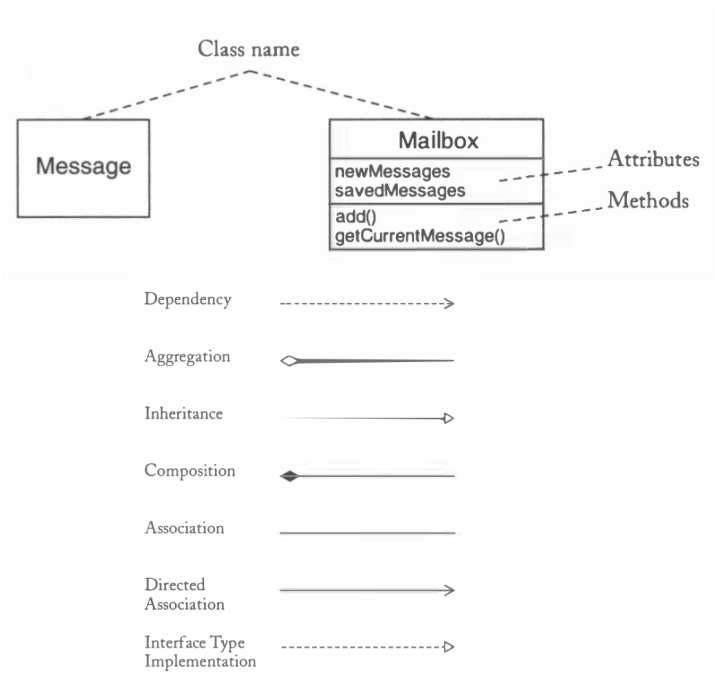


Figure 15: Class Diagram rules and inheritance definitions. Graphs retrieved from Horstmann's book [18]

Figure 16 depicts a high level UML Class Diagram that was designed for the Feature Extraction System used in the MIT BIH and GEHC dataset to produce the

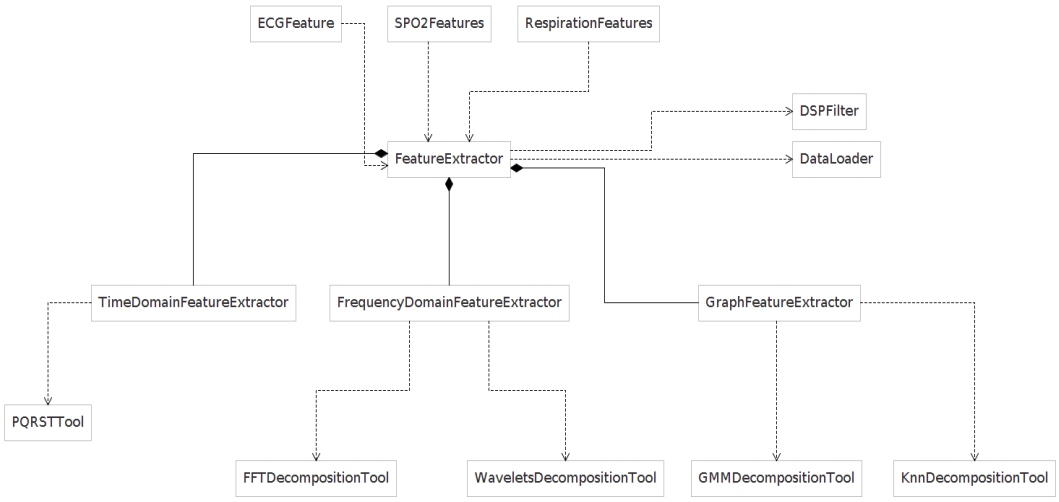


Figure 16: High level Class Diagram, containing only class names

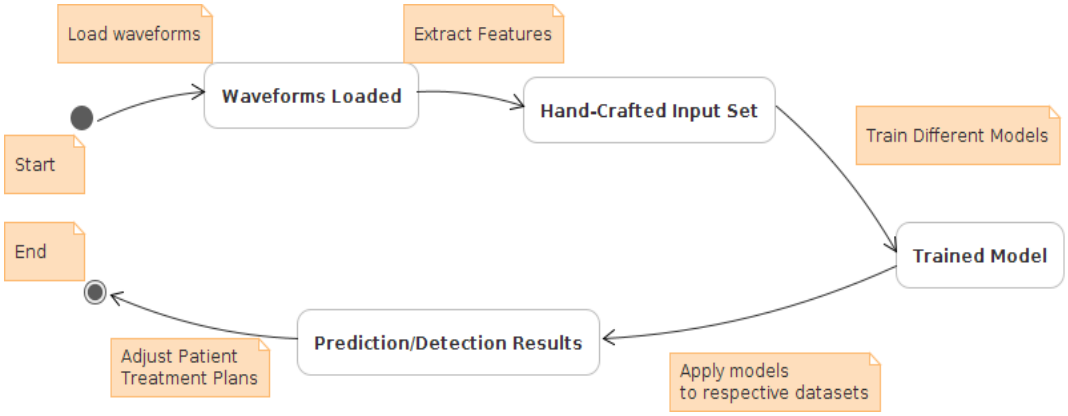


Figure 17: It is possible to see the different states pertaining to an input set of waveforms. The colored notes denote the different operations that are performed to reach that specific state.

The above tools and ideas allowed for a structured development environment that was adaptable, yet performance oriented. By observing OOD patterns and designing computational systems that made use of efficient algorithms, while being careful about its input and output grammar, it was possible to load, process, and store large chunks of data (tens of gigabytes per waveform) in an efficient and structured fashion (see Appendix A). In the end, since the data was well structured and traceable, it was possible to adjust the data structure and optimize the machine learning training.

In a final concluding statement, the end goal with all of this structure and discipline in the software engineering part is to attain a higher degree of confidence on the results achieved.

3 Feature Engineering

Feature engineering composed a significant part of this work. In essence, the idea was to dissect biosignals into features that could be better digested by machine and clinician. In order to achieve this, it was necessary to make decisions (when choosing features) that were conscious and considerate of optimization concepts (for the underlying optimization algorithms used by most machine learning algorithms), and clinically explainable (meaning that a clinician could understand directly or indirectly).

In order to help a clinician understand a feature directly, it should make a complete one-to-one matching with already used features or methodologies. For example, features such as heart rate and respiration rate are information that produce this one-to-one matching. In order to accomplish an indirect understanding, it should be possible to explain a feature in clinical terms. Features such as the information frequency distribution of an ECG can be indirectly mapped to a physician language. For example, a large peak at 9 Hz is understood to be related to the P wave [21].

Now, in terms of optimization, every feature that is developed shall have the base requirement to map a raw signal, say an ECG signal $\mathbf{x}_{raw}$ into a domain that is easier for the algorithm to classify correctly. This could entail filtering and normalizing the signal (as it was done on the implementation for the Feature Extraction System created for this research) [22] [23] [24]. The goal is to help the optimizing algorithm, so that it will not get stuck in sharp local minimas or irrelevant solutions.

With the above principles in mind, the next step is to consider the nature of biosignals. Per the definition proposed by Oppenheim and Shaffer in [24], a signal refers to information or data that is extracted with a time dependent component. Biosignals on the other hand, are a category of signals whose origins are biological processes. The signals used in this research are all under the class of biosignals.

Now, there are two classes of signals: *discrete* and *analog* signals. Analog signals, are those which are continuous in nature, such as audio signals. Discrete time signals is generally related to information that is being sampled (or recorded) at some regular interval. These can be defined as follows:

$$x[n] = x_a(nT_s), -\infty < n < \infty \tag{2}$$

In Equation 2, $x_a(nT_s)$ represents an analog signal that is being sampled at a sampling period of T_s, and n is the specific sample number [24]. Mathematically, these can also be referred to as sequences, as their domain is defined by a set of integers (in this case n). Biosignals for example, most often carry a periodic characteristic which denote some consistent function of an organ. This means that there is some component in $x[n]$ which recurs every T_p. Therefore, in an ideal scenario: $x[2 + T_p] = x[2]$. In a more realistic world, it will be something of the form: $x[2 + T_p] = x[2] + \epsilon[2 + T_p]$. The term $\epsilon[k]$ refers to noise that may be incurred over time in the signal. However, the structure and value should be approximately the same. We can refer back to the ECG signal extracted from the GEHC dataset.

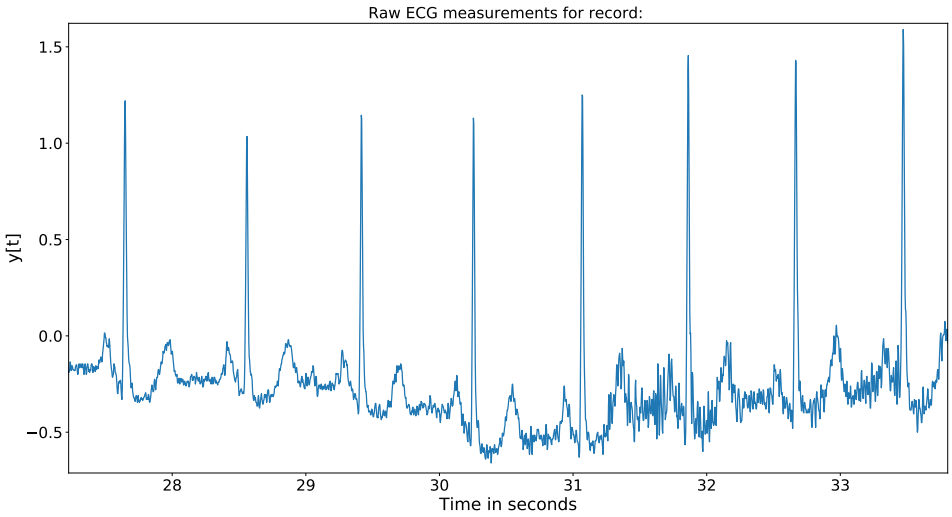


Figure 18: Raw ECG signal. It is possible to see the recurring beats, which demonstrate the natural periodicity in the ECG biosignal.

These periodic components allow a view into how a specific organ (or a group of organs, by inference) is functioning. For example, the ECG provides a beat-to-beat view of the heart. This shows how heartbeats are changing over time. If the heart has malfunctioned or produced abnormal behavior over the last few minutes, or hours, it should be possible to see this in the ECG recording.

In the example in Figure 18, it is possible to see the noise, preventing strict periodicity. However by loosening the definition one can claim periodicity. By referring back to the SpO2 signal in Figure 7 and Respiration signal in Figure 11, it is possible to see these periodic waves that occur.

Considering this periodicity in biosignals, the following is hypothesized:

Hypothesis 3.0.1. *There is value in the periodic components of a biosignal time series.*

Beyond the view in the time domain, it is also possible to analyze a signal in the frequency domain. The frequency domain is just another representation of the original signal. It is similar to viewing the same information from a different angle. An angle that highlights how the energy is distributed across frequency components.

In order to see and understand how another insightful angle may be achieved, and why it may interesting, it is first necessary to consider the results achieved by Joseph Fourier [25]. These results demonstrate that any signal can be approximated by a sum of complex exponentials (or complex sinusoids).

$$x(t) = \sum_{k=-\infty}^{\infty} \{c_k \cdot e^{2\pi \cdot t \cdot f_0 \cdot k}\} \tag{3}$$

$f_0 =$ fundamental sampling frequency

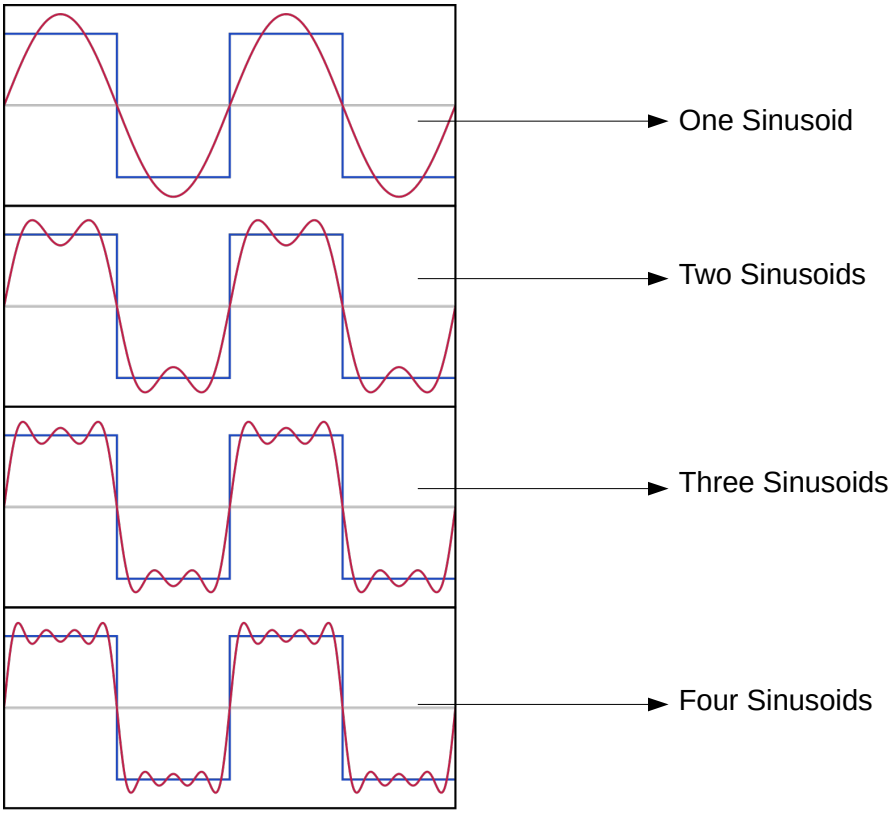


Figure 19: By summing different number of sinusoids, it is possible to a simple sine wave to approximate a square wave.

The idea that it is possible to represent different functions through the sum of complex sinusoids leads to the Fourier Transforms (FT). The Fourier Transform is the tool that allows this different view point into a biosignal. For the purposes of this work, the Discrete Fourier Transform is more important, and thus its definition will be presented as follows:

$$X[k] = \sum_{n=0}^{N-1} \left\{ x[n] \cdot e^{-\frac{j2\pi}{N} \cdot kn} \right\} = \sum_{n=0}^{N-1} \left\{ x[n] \cdot (\cos(\frac{j2\pi}{N} \cdot kn) - j \sin(\frac{j2\pi}{N} \cdot kn)) \right\} \quad (4)$$

Similarly, the time representation may be achieved by the inverse discrete Fourier Transform:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} \left\{ X[k] \cdot e^{-\frac{j2\pi}{N} \cdot kn} \right\} = \frac{1}{N} \sum_{k=0}^{N-1} \left\{ X[k] \cdot (\cos(\frac{j2\pi}{N} \cdot kn) - j \sin(\frac{j2\pi}{N} \cdot kn)) \right\} \quad (5)$$

$x[n]$:	Input (time domain) sequence
$X[k]$:	Output (frequency domain) Sequence
N	Number of elements in the input sequence
k:	Output sequence index
n:	Input sequence index
j:	Imaginary number ($\sqrt{-1}$)

The Discrete Fourier Transform (DFT) defined in Equation 4 allows the mapping of an input sequence $x[n]$ to an output sequence $X[k]$. The sequence $X[k]$ is an alternate representation of $x[n]$ in what is called the frequency domain.

The frequency domain provides insight into the periodic components that occur in the input sequence, as well as how predominant these components are in comparison to other recurring components. More generally, it is possible to see how fast information is changing in the input sequence.

Traditionally, in areas of signal processing, wireless communications and image processing, the frequency domain has been crucial to produce valuable insight on input sequences [24].

The valuable insight provided by the frequency domain leads us to the following hypothesis:

Hypothesis 3.0.2. *There is added value in the information contained in the frequency domain for a certain time interval of a biosignal.*

Hypothesis 3.0.2 and Hypothesis 3.0.1 are the crucial ideas that support the features and feature engineering produced in this research work. By considering the value in the time domain of the sequences, it is possible to derive features that hold information from the time domain, while maintaining direct relation to features and ideas that physicians understand.

The remainder of this chapter will be divided in the following topics: Signal Processing, Feature Definition, Distinct Features Developed, and Conclusion.

3.1 Signal Processing

When considering the data used for this project, it is necessary to be attentive to the structure. As each respective input data is a discrete time signal, two specific concepts must be accounted for when processing this information: Sampling and Noise.

3.1.1 Signal Sampling

For the MIT BIH dataset, where only ECG was considered, the sampling frequency was 360 Hz. This was after hardware processing of the information. For simplicity, only the post-hardware signal processing will be considered here.

As previously mentioned, the actual monitoring device sampling frequencies for the GEHC dataset varied between 60Hz to 240Hz. However, when the waveforms were stored, those which were below 240Hz were up-sampled to 240Hz.

$$T_s = \frac{1}{f_s} = \frac{1}{240} = 0.004167 \text{ (s)} \tag{6}$$

3.1.2 Noise Filtering

Noise Filtering was tailored for each individual waveform. More specifically, a Butterworth Filter [24] with different specifications was used for each signal. Figure 20 provides a simple block diagram that demonstrates the computational flow of signal through filtering.

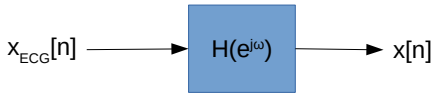


Figure 20: Signal $\mathbf{x}_{ECG}[n]$ going through a filter $H(e^{j\omega})$

The raw waveform signal $\mathbf{x}_{ECG}$ is a discrete-time signal that contains noise. This signal is passed through a filtering function $H(e^{j\omega})$ that attempts to keep the frequency bands of interest intact, while attenuating what is considered noise.

Attenuating a certain *frequency band* that is considered noise is the general goal of signal filtering. A frequency band is a range of frequencies, generally defined in terms of radians. Another term that usually goes with frequency bands is *passband*. Passband refers to range of frequencies that one wishes to keep in the signal. Equation 7, 8, and 9 outline the passbands used for the Butterworth filters used to clean the ECG, Respiration and SpO2 waveforms.

$$\pi \cdot \frac{0.5}{T_s} \leq \omega_{ECG} \leq \pi \cdot \frac{45}{T_s} \tag{7}$$

$$\pi \cdot \frac{0.1}{T_s} \leq \omega_{RR} \leq \pi \cdot \frac{4}{T_s} \tag{8}$$

$$\pi \cdot \frac{0.1}{T_s} \leq \omega_{SpO2} \leq \pi \cdot \frac{20}{T_s} \tag{9}$$

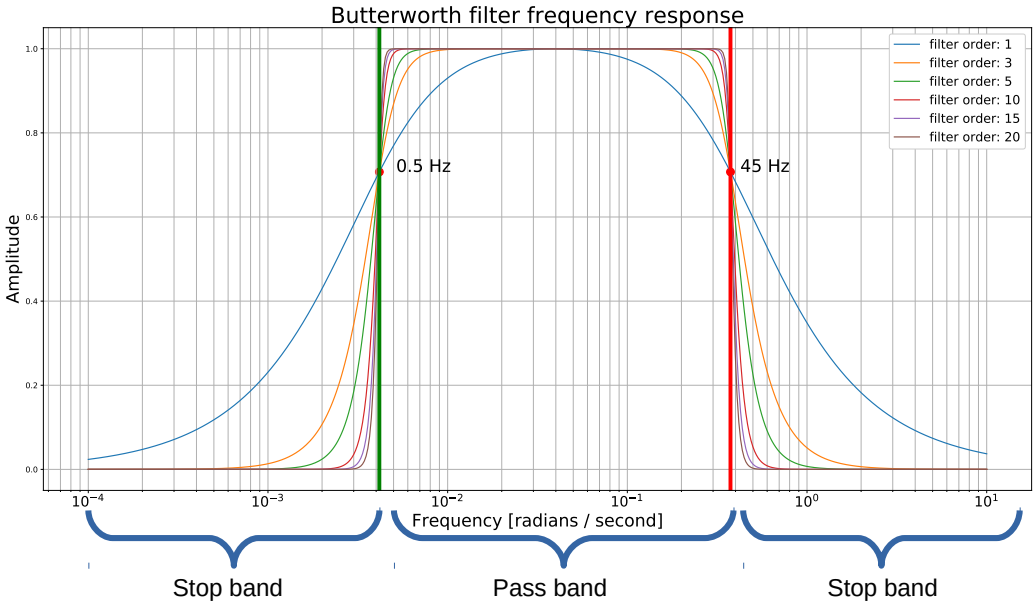


Figure 21: In the figure above, a number of different filter orders for a Butterworth Bandpass Filter is demonstrated. The parameters are similar to the ones tested for ECG waveform

Figure 21 further illustrates this idea of a passband when considering a filtering operation. Filters of different order are displayed. These filter contain a range of values from [0,1]. This is generally referred to as the gain of the filter. Ideally one would like to have a gain of unity (or $|H(e^{j\omega})| = 1$) in the passband, and a gain of 0 in the stopband. The *Stopband* is the region of frequencies that is undesirable in the signal. However, in a more realistic scenario, there is some error that distorts the signals in the passband.

The Butterworth Filter, was chosen due to the fact that it attempts to keep a constant gain of unity in the passband. This ensures the signal achieves minimal modification (amplification or attenuation) within the frequency band of interest.

One important artifact that is filtered in an ECG waveform is the baseline wandering. Figure 22 helps the reader note the variation of the waveform around 0 in the y axis. This variation is known as the baseline wander. These are low frequencies that are created by artifacts such as other organs, movement, and general noise between electrode and skin.

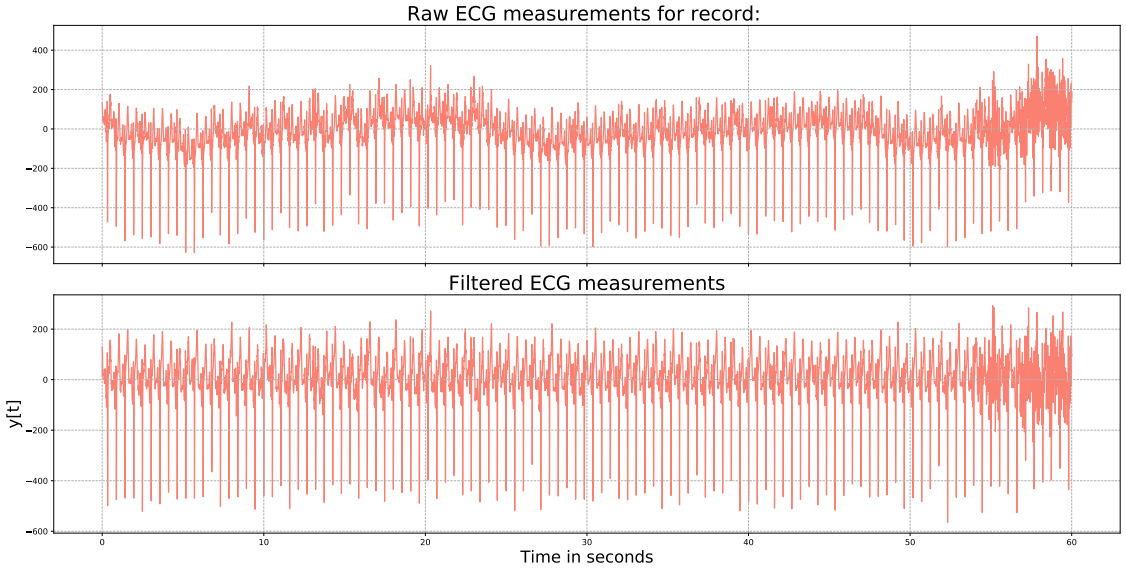


Figure 22: Baseline wandering for an ECG signal is demonstrated in the top graph. On the lower graph, it is possible to see the filtered signal.

The topic of Filtering is a large area of Signal Processing. The attempt here is only to provide sufficient information such to understand the steps that were taken during the feature engineering performed in this research work. For further reference regarding filtering and sampling of signals, please refer to [24] and [10] for further material.

3.2 Defining Features

In the previous section, the pre-processing concepts and supporting ideas that motivated and propelled the feature engineering portion of this work was explored. In this section, the feature developed, along with the algorithms designed to extract those features from raw waveform signals are explained.

To begin, the concept of feature shall be explicitly defined:

Definition 3.2.1. *A feature is a vector* $\mathbf{z}_{i,k}$ *that may be achieved by applying a feature mapping function* $\phi_{i,k}(\cdot)$.

i: Represents domain name, e.g. t for time, f for frequency
k: Represents function name, e.g. Δ *for peak differential feature mapping*

$$\phi_t = \{\phi_P, \phi_Q, \phi_R, \phi_S, \phi_T, \phi_{Resp}, \phi_{Dyast}, \phi_\Delta, \phi_\sigma\} \tag{10}$$

$$\phi_f = \{\phi_{Max}, \phi_{Spec}, \phi_{Wave}, \phi_{WaveStats}\} \tag{11}$$

The feature mapping functions designed for this project sit within two main classes: Time Domain functions, denoted by $\phi_{t,j}$, and Frequency Domain function, denoted by $\phi_{f,k}$, where j and k denote some specific feature name. The specific

names and their respective domain denominations are expressed in Equation 10 and Equation 11. It should also be noted that as the text progresses, the notation style $\phi_{t,\Delta}$, where both subscripts are mentioned, will be switched to ϕ_{Δ} for convenience.

Now, an example to better illustrate Definition 3.2.1 is the following: consider a filtered ECG waveform as shown in Figure 22. Feature mapping function $\phi_{t,\Delta}(\cdot)$ (further explained in subsection 3.3) computes the time changes over a specific reference peak. For an ECG signal, this means calculating a raw version of a Heart Rate [2]

$$\mathbf{z}_{t,\Delta} = \phi_{t,\Delta}(\mathbf{x}_{ECG}) \tag{12}$$

In the example presented in Equation 12, Δ is a feature name that represents changes with respect to specific reference points in the waveforms. By applying $\phi_{t,\Delta}$, a vector $\mathbf{z}_{t,\Delta}$ containing information about heart rate is created.

In order to extract these reference points, lower level features must be created, whose role is to extract the reference points (or points of interest). In the case of ECG waveforms, these reference points generally come from the detection of the QRS complex (further discussed in subsubsection 3.3.1.

A final high level note regarding the features: features related to the time domain structure of the data are generally waveform specific. For example, developing an algorithm for identifying the P,Q,R,S, and T peaks is specific to the ECG (as these are clinically acceptable features of interest). For SpO2, identifying the Systole point and Dyastole is very specific to a Plethysmographic wave. This holds an important distinction in the implementation of the system to extract the features, as the algorithms developed to exercise the frequency domain feature mapping are data structure agnostic. This means that these algorithms can be applied to any of the waveforms. Whereas algorithms developed to implement the time domain feature mapping are dependent on the structure of the data (or waveform, such as ECG, SpO2, or Respiration).

3.3 Time Domain Features

In this section, time domain features will be explored, along with the algorithms that were developed to allow the extraction of these features.

3.3.1 PQRST Detection

In order to analyze the periodic functioning of the heart, clinicians examine the beat-to-beat behavior of the heart. This beat-to-beat behavior provides information such as Heart Rate, Heart Rate Variability, and consequently valuable insight on how the heart has been functioning over a specific window of time. From a computational perspective, it is possible to design an algorithms that performs such actions reliably. To achieve this, an algorithm must find some reference point within the beat to identify that there has been a beat. This brings up the term *QRS Complex*. The QRS Complex, is basically three extremas generally found in a heart beat.

[2] Regular Heart Rate is calculated over the difference between R peaks of regular heart beats. Here, raw version means that Heart Rate is calculated over any beat.

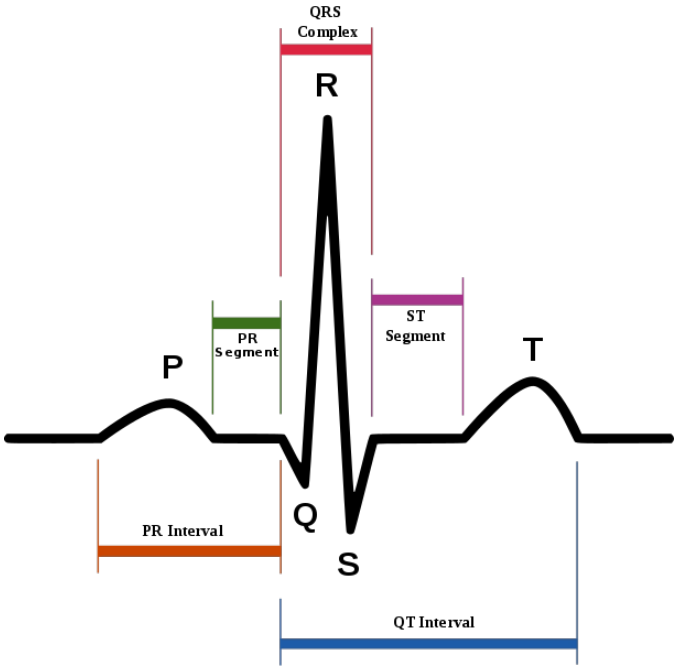


Figure 23: QRS Complex, where the more accentuated R peak helps algorithms such as the Pan Tompkins [26] identify the QRS Complex

As noted in Figure 23, the Q and S peaks provide valleys which help accentuate the most acute peak in the heart beat, the R peak.

In order to identify the QRS complex, the Pan Tompkins algorithm [26] was chosen as a starting point for a QRS Detection algorithm, as it helps accentuate and identify this change. In short, Pan-Tompkins is a classic QRS detection algorithm proposed in the 1985. It uses differentiation, squaring and integration steps in order to reliably identify the R peaks of the wave. The product of applying these operations in the data is a highlight of acute changes, such as the one produced by the QRS peaks. Figure 24 helps illustrate how the Pan-Tompkins algorithm affects the original signal. Moreover, it was validated with the MIT BIH [8] dataset.

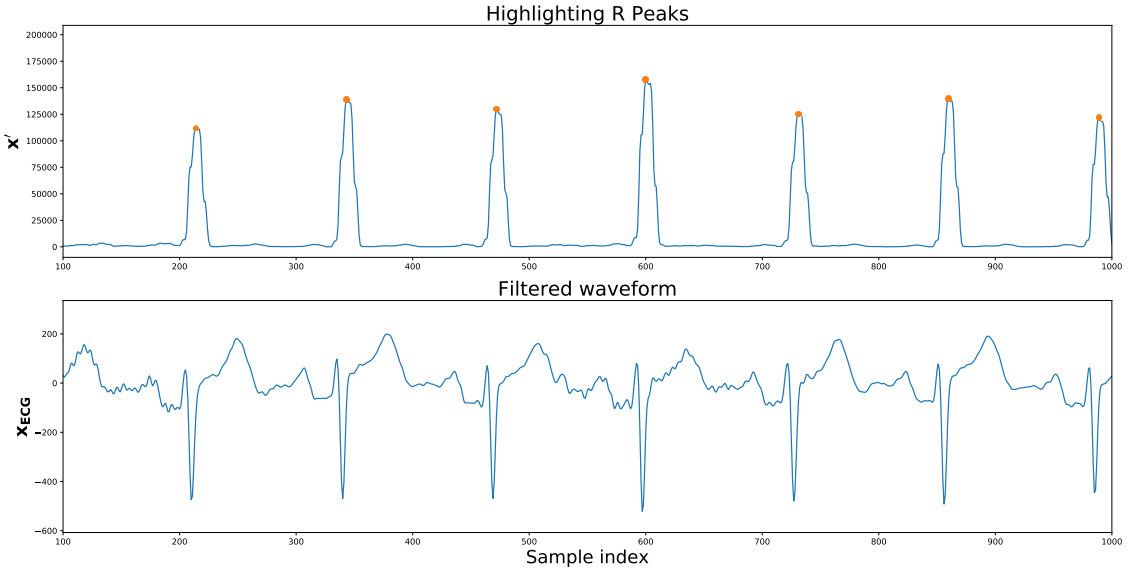


Figure 24: $\mathbf{x\prime}$ on the top plot shows the result of applying the Pan-Tompkins algorithm [26]. The orange dots illustrate the peaks that were found. The lower plot shows the input $\mathbf{x}_{ECG}$ that was used.

Algorithm 1 gives further insight into how the Pan-Tompkins algorithm was used to detect the QRS Complex.

Algorithm 1 R Peak Detection ϕ_R

Require: ECG Signal $\mathbf{x}_{ECG} \in \Re$
Ensure: Peak array $\mathbf{R}$ is initialized
  Apply Pan-Tompkins [26] on $\mathbf{x}_{ECG}$ to achieve $\mathbf{x\prime}$
  Apply [27] `find_peaks` function to $\mathbf{x\prime}$ and store peaks p in peak set $\mathbf{R}$
  Define window w to validate peak $p \in \mathbf{R}$
  Define a new peak set $\mathbf{R\prime}$
  for all Peak $p \in \mathbf{R}$ **do**
    Find peak position p_0 within the array of values $\mathbf{x}_{ECG}[p - w/2 : p)$ that is the largest
    Find peak position p_1 within the array $\mathbf{x}_{ECG}[p : p + w/2)$ that is the largest
    if $\mathbf{x}_{ECG}[p_0] > \mathbf{x}_{ECG}[p_1]$ **then**
      Append p_0 to $\mathbf{R\prime}$
    else
      append p_1 to $\mathbf{R\prime}$
    end if
  end for
  $\mathbf{R} \leftarrow \mathbf{R\prime}$
  return $\mathbf{R}$

Algorithm 1 was validated in the MIT BIH [8] dataset, which is considered a de facto dataset to validate algorithms for beat detection. The performance tests

showed 95% accuracy.

Now, given that the R peaks are detected, it is possible to find the latter peaks by using heuristics derived from the physical capabilities of the heart. Namely, from the R peaks one can investigate a small window that predates the R peak, and find minima Q . Similarly, placing a small window after the R peak, one can find minima S (Algorithm 2). A similar idea, however now parting from S can be used to detect T (Algorithm 4). Finally, investigating prior to Q allows the detection of P (Algorithm 3).

Algorithm 2 describes the mapping function ϕ_S for extracting the S peaks. The S peak is in fact a valley that is part of the QRS Complex, as noted in Figure 23. It sits between P and R peaks, and is generally the starting point for a segment called the *ST Segment*. The ST Segment represents a window of time during which the ventricles are in active depolarization. Abnormalities in the duration and form of this segment can indicate anomalies in the ventricles.

Algorithm 2 S Peak Detection ϕ_S

Require: ECG Signal $\mathbf{x}_{ECG}$, R peak array $\mathbf{R}$, ECG Sampling Frequency f_s
Ensure: Peak array $\mathbf{S}$ is initialized
  for all Peak $r \in \mathbf{R}$ **do**
    Window size $M_w \leftarrow 0.11 \cdot f_s$
    Starting position $w_0 \leftarrow r + 1$
    End position $w_f \leftarrow r + M_w$
    Get a normalized version of the data to check: $\mathbf{d} \leftarrow \texttt{norm}(\mathbf{x}_{ECG}[w_0, w_f])$
    Apply [27] `find_peaks` on $\mathbf{d}$ to get s
    Append s to $\mathbf{S}$
  end for
  return $\mathbf{S}$

The S peak should sit directly after the R peak. Consequently, Algorithm 2 uses a constant (0.11) that can be translated to a window of sampled data, whose length is 110 ms. This window of information containing the array $\mathbf{x}_{ECG}[w_0, w_f]$ is used to find the peak. In the case of the S peak, it should be a minima within this window.

An additional note regarding 2, is the `norm` that is used. It is defined as follows:

$$\texttt{norm}(\mathbf{x}) = \frac{\mathbf{x} - \overline{\mathbf{x}}}{\sigma} \tag{13}$$

In Equation 13, σ is the standard deviation of the input vector. Moreover, $\overline{\mathbf{x}}$ represents the mean of the input vector $\mathbf{x}$.

Similarly to the S peak, the Q peak is a valley that sits directly before the R peak. It dictates the beginning of the QRS Complex. The algorithm to find the Q peaks is very similar to Algorithm 2, except for the window that is chosen. The constant is the same, and the peak of direct dependence, R peak, is also the same.

The P peak indicates the start of what is called *depolarization* process [10]. Depolarization is when changes in potential in the membranes of the cells in the heart occurs. This depolarization begins in the right and left atria. The cells depolarize

producing an electrical impulse that traverses the heart's *myocardium*. Myocardium is the name of the muscle cells that composes the heart's walls. Algorithm 3 describes how the feature extraction system shall record this information.

Algorithm 3 P Peak Detection ϕ_P

Require: ECG Signal $\mathbf{x}_{ECG}$, Q peak array $\mathbf{Q}$, ECG Sampling Frequency f_s
Ensure: Peak array $\mathbf{P}$ is initialized
  for all Peak $q \in \mathbf{Q}$ **do**
    Window size $M_w \leftarrow 0.2083 \cdot f_s$
    Starting position $w_0 \leftarrow q - M_w$
    End position $w_f \leftarrow q - 1$
    Get a normalized version of the data to check: $\mathbf{d} \leftarrow \texttt{norm}(\mathbf{x}_{ECG}[w_0, w_f])$
    Apply [27] `find_peaks` on $\mathbf{d}$ to get p
    Append p to $\mathbf{P}$
  end for
  return $\mathbf{P}$

Algorithm 3 contains a constant, 0.2083 which helps define the size of window that will be used to look for the P peak. It basically means that we are considering 208.3 ms of information for that window that will be checked. This constant was reached by experimentation and the fact that a P wave is expected to last for about 120 ms (if it is there at all). By extending the window, experiments showed a more reliable identification of the P peak.

Now, Algorithm 4 describes more precisely how to identify the T peak. The T wave detection algorithm contains some interesting points that are worth mentioning. The first is the fact that the T wave illustrates the ventricular *repolarization* (last peak seen in Figure 23) Depolarization means that the heart is undergoing a relaxation phase, after the contraction period. This process should last for about 300 ms [10]. As one may suspect, this is highly dependent on the rate at which the heart is beating. If it is slower, something slightly above 300ms may be expected. If faster, this window is expected to be smaller. This is where the constant comprising M_w comes from. We consider about 347 ms worth of information, in order to make sure that the algorithm is able to withstand some error, and in the worst case, it will capture more information than necessary. However, not sufficient such to interfere with the peak formation of the T wave.

Algorithm 4 T Peak Detection ϕ_T

Require: ECG Signal $\mathbf{x}_{ECG}$, S peak array $\mathbf{S}$, ECG Sampling Frequency f_s
Ensure: Peak array $\mathbf{T}$ is initialized
  for all Peak $s \in \mathbf{S}$ **do**
    Window size $M_w \leftarrow 0.3472 \cdot f_s$
    Repolarization offset $r_{off} \leftarrow M_w \cdot 0.10$
    Starting position $w_0 \leftarrow s + r_{off}$
    End position $w_f \leftarrow s + M_w$
    Get a normalized version of the data to check: $\mathbf{d} \leftarrow \texttt{norm}(\mathbf{x}_{ECG}[w_0, w_f])$
    Get Average Gradient $\mathbf{d}'_{avg}$ of $\mathbf{d}$
    if $\mathbf{d}'_{avg} > 0$ **then**
      Apply [27] `find_peaks` on $\mathbf{d}$ to get t
    else
      Apply [27] `find_peaks` on $-\mathbf{d}$ to get t
    end if
    Append t to $\mathbf{T}$
  end for
  return $\mathbf{T}$

Ideally Algorithm 4 should include some information about the time difference between the current R peak and the previous. This is so the algorithm is able to include the information about whether or not the heart is beating faster, and adjust the T wave detection. This was initially implemented, however it required more testing validation. And so, at the end Algorithm 4 was used as the performance was sufficiently good.

Another couple of points worth mentioning is the 10% starting gap between the S peak detected and the window that will be used for detecting the T peak. This is due to the beginning of the repolarization event, therefore causing some unpredictability in this portion of the signal. As a consequence, only the latter part is used. Finally, the gradient is used because it is able to give some indication about whether the T wave is normal or inverted [3].

P Peak Detection should be something similar to T, except we select a window before Q. Figure 25 illustrates the result of the algorithms for peak selection in practice. It is possible to see the peak selection on a beat-to-beat basis.

[3]The difference between normal and inverted T waves can be an indicator of Myocardial Ischaemia [10]

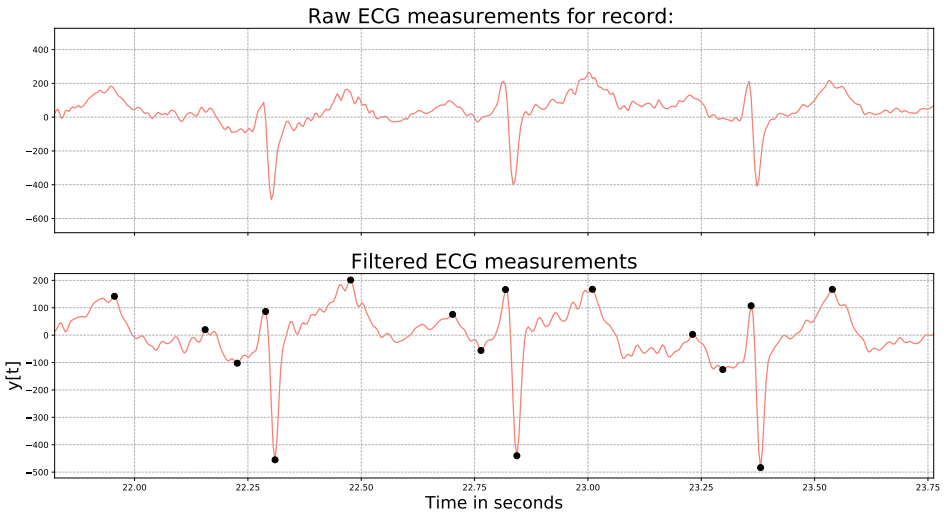


Figure 25: PQRST peak selection feature.

The resulting peak selection shown in Figure 25 is the basis for the rest of the features developed for ECG Time Domain.

3.3.2 Respiration Peak Detection

The concepts developed in subsubsection 3.3.1 served as basis for the peak detection of the Respiration signal. In essence, the respiration signal contained in the GEHC dataset was extracted by a GEHC monitoring device. This monitoring device imputed an artifact into the signal to highlight *inspiration* and *expiration* events. Inspiration is when the device detected that the patient was inhaling. Expiration highlighted exhaling. Figure 11 better highlights the points discussed.

The strategy was then as follows: use these imputed artifacts [4] that are created by the monitor as reference peaks. This is very similar to what was done with the R Peak detection algorithm.

Algorithm 5 Respiration Detection ϕ_{Resp}

Require: Respiration Signal $\mathbf{x}_{Resp} \in \Re$
Ensure: Peak array $\mathbf{E}$ is initialized
  Apply Pan-Tompkins [26] on $\mathbf{x}_{Resp}$ to achieve $\mathbf{x\prime}$
  Apply [27] `find_peaks` function to $\mathbf{x\prime}$ and store peaks p in peak set $\mathbf{E}$
  Initialize peak set $\mathbf{E}'$ to be half of the size of $\mathbf{E}$
  Append even indexes of $\mathbf{E}$ to $\mathbf{E}'$
  Reset $\mathbf{E} \leftarrow \mathbf{E}'$
  return $\mathbf{E}$

[4] In reality, these are not artifacts. These are peaks created by a sophisticated peak detection algorithm that is proprietary to GEHC.

Algorithm 5 shows more precisely how the mapping function ϕ_{Resp} achieves the reference peaks for the respiration signal. In essence, the steps are very similar to R peak detection for ECG, however here it is necessary to drop half of the peaks to isolate only inspiration, or expiration.

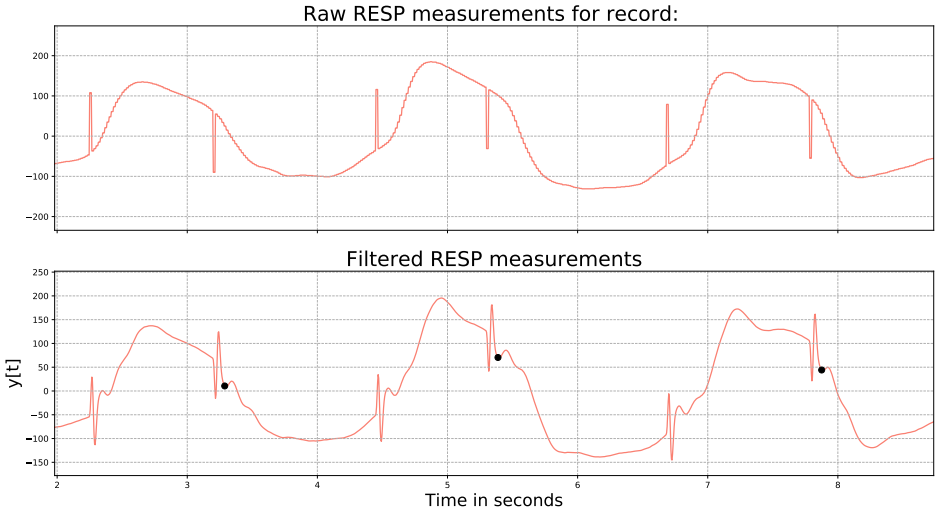


Figure 26: Respiration Peak Selection. In the top graph, it is possible to see the raw respiration signal from GEHC dataset, which contains the artifacts of interests. The lower image demonstrates the peak selection of expiration points, achieved by Algorithm 5.

Figure 26 shows not only the resulting peak selecting for Algorithm 5, but also the raw respiration signals. It is possible to note the digitization artifacts that are contained in the raw respiration signal. Although the filtering process produces some deformation to the final time morphology of the respiration signal, the main point of interest is a clearer frequency spectrum (digitization artifact corrupt the frequency spectrum), and the location of the peaks.

3.3.3 Plethysmographic Peak Detection

SpO2 peak detection was performed similarly to respiration peak detection. In essence, the core algorithm was similar, along with the results. The difference is the peak of reference, which in this case of the Systole. The user may refer to Figure 26 as a reference for Systole Detection.

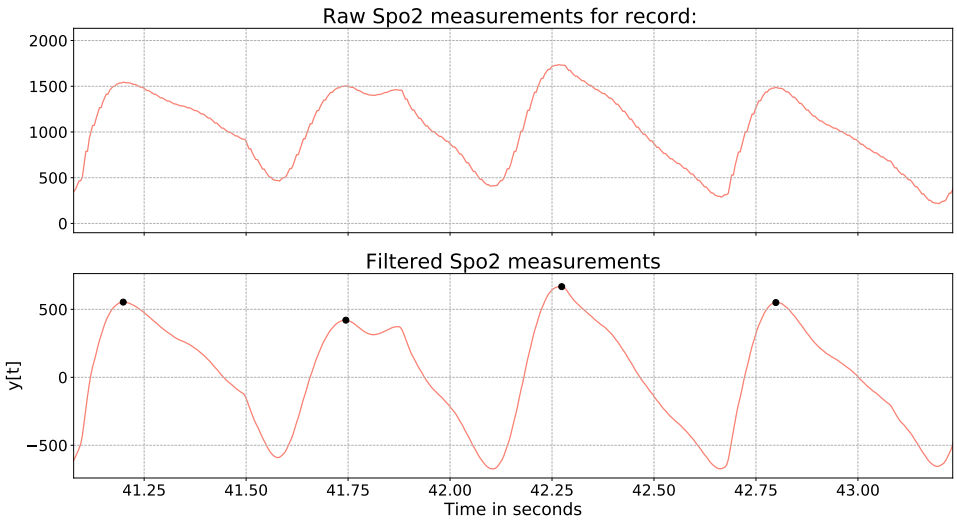


Figure 27: Systole Peak Selection. In the top graph, it is possible to see the raw SpO2 signal from GEHC dataset. The lower image demonstrates the peak selection of expiration points.

Similar to the filtering deformations created in the respiration signal, the filtered SpO2 signal also has a morphological difference compared to its original, raw signal. However, the digitization artifacts created by the up-sampling is filtered.

3.3.4 Higher Level Features

The peak detection features are considered as lower level features, since they are the basis for higher level features such as ϕ_Δ and ϕ_σ feature mappings. Algorithm 6 describes one of these higher level feature mappings which help produce features that are equivalent to Respiration Rate, Heart Rate, and Pulse Rate.

Algorithm 6 Delta Feature Mapping ϕ_Δ

Require: Peak array $\mathbf{P}$,
Ensure: Peak array $\mathbf{D}$ is initialized
  for all Index i in range of $|\mathbf{P}| - 1$ **do**
    Calculate time distance $d = \frac{\mathbf{P}[i+1]-\mathbf{P}[i]}{f_s}$ in seconds
    Append per minute time interval $\frac{60}{d}$ to $\mathbf{D}[i]$
  end for
  return $\mathbf{D}$

Now, Algorithm 7 builds on Algorithm 6. It's goal is to return the standard deviation of an input array containing the time differences in a per minute basis computed by Algorithm 6.

Algorithm 7 Sigma Feature Mapping ϕ_σ

Require: Delta array $\mathbf{D}$,
Ensure: D is initialized
Apply [27] `std` to set $D \leftarrow \texttt{std}(\mathbf{D})$
return D

In Algorithm 7, `std` stands for standard deviation. The function is further defined in Equation 16.

$$\texttt{mean}(\mathbf{x}) = \overline{\mathbf{x}} = \frac{1}{N}\sum_{k=1}^{N}\{x_k\}, \text{where } N \text{ is the number of items in } \mathbf{x} \tag{14}$$

$$\texttt{dev}(\mathbf{x}) = \mathbf{x}_d = |\mathbf{x} - \overline{\mathbf{x}}| \tag{15}$$

$$\texttt{std}(\mathbf{x}) = \mathbf{x}_\sigma = \sqrt{\texttt{mean}(\mathbf{x}_d^2)} \tag{16}$$

Part of the value in the Sigma Feature Mapping ϕ_σ lies in the ability to reduce the dimension of feature vector $\mathbf{z}_\Delta$ to a scalar containing the standard deviation z_σ. The importance of dimensionality reduction will be further discussed in section 5.

3.4 Frequency Domain Features

This section explains on the frequency domain features that were developed. As in subsection 3.3, where the time domain features along with the algorithms that composed the feature mappings were discussed, this section aims to provide the similar value and explanation.

In Equation 4 the DFT was explored. Soon after, Hypothesis 3.0.2 was posed. The Fourier Transform Decomposition (FTD) module that was developed to extract features for the eventual machine learning experiment built on these ideas and assumptions. Moreover, it was also driven by the fact that these algorithms are domain agnostic features. Therefore, the same algorithm can be applied to any input waveform.

This section will then be split into two subdomain decomposition modules: Fourier Transform Decomposition and Wavelet Decomposition.

3.4.1 Fourier Transform Decomposition

The FTD mainly produced two features: Max Spectrum components ϕ_{Max}, and Spectrogram Images ϕ_{Spec}. A *power spectrum* describes the distribution of energy in the different frequencies encountered in the signal. A *spectrogram* is a visual representation that is created to help analyze how the frequency bands change over time. So for example: how did the spectrum distribution look over the first 5 seconds of ECG recording? Were there high levels of energy in range between 8-12hz? Or perhaps between 2-4Hz?

These questions are relevant because it has been shown that it's possible to identify the presence of specific waves, such as T or P waves in the case of ECG

waveforms, in the power spectrum [10]. For example, the T wave is contained mostly within 0 to 4Hz. The P wave can be found in the range between 0 to 8Hz. The QRS complex is usually concentrated in the range between 8 to 20 Hz. This information shows that the peaks representing these waves will generally be within a certain frequency range. If that changes, then something may be happening which may be worth examining. The idea with feature mapping ϕ_{Max} is to provide this information to the machine learning model.

Algorithm 8 Max Frequencies Feature Mapping ϕ_{Max}

Require: Signal $\mathbf{x}_{raw}$, Peak percentage α
Ensure: Empty peak array $\mathbf{D}$ and $\mathbf{V}$ are initialized
  Apply FFT from [27] FFT package to attain array $\mathbf{X}[k]$
  $N \leftarrow$ `length`$(\mathbf{X}[k])$
  Reset $\mathbf{X}[k] \leftarrow |\mathbf{X}[0 : N/2 - 1]|$
  Find max peak value p in $\mathbf{X}[k]$
  Set height of interest $h \leftarrow p \cdot \alpha$
  Apply [27] `find_peaks` to $\mathbf{X}[k]$ and store peak locations in $\mathbf{P}$
  for all peak index i in $\mathbf{P}$ **do**
    if $\mathbf{P}[i] > h$ **then**
      Append $\mathbf{P}[i]$ in $\mathbf{D}$
      Append peak value $\mathbf{X}[\mathbf{P}[i]]$ in $\mathbf{V}$
    else
      Keep Going
    end if
  end for
  return $\mathbf{D}$, $\mathbf{V}$

Particular to Algorithm 8 is the peak percentage parameter, which one must decide before moving forward with the algorithm. Once paired with the maximum peak achieved in the spectrum, will translate to the minimum height of interest for a peak that will be used for prediction. This was done so to decrease the amount of data that would be sampled, since most frequency spectra is generally sparse.

Figure 28 illustrates the results of the implementation of Algorithm 8.

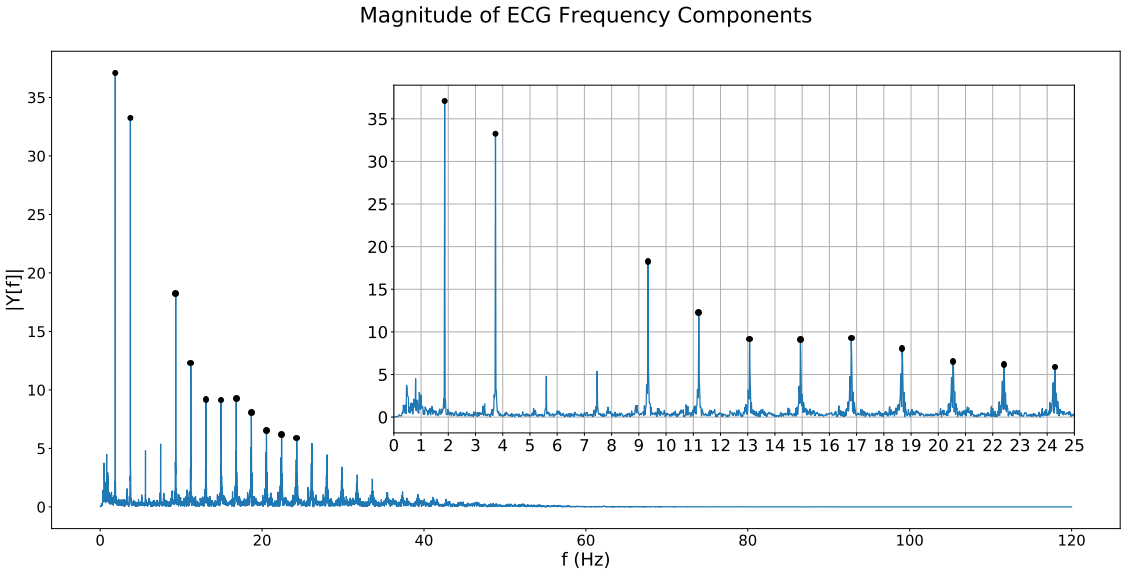


Figure 28: Periodogram showing the magnitude of the frequency values, along with the highlighted peaks that will be included in features vector ϕ_{Max}

Biosignals usually have a very sparse, and rather localized frequency spectra, as shown in Figure 28 and Figure 29. As a consequence, it is possible to allow the machine learning algorithm to focus on the parts that are interesting by selecting the most prominent peaks.

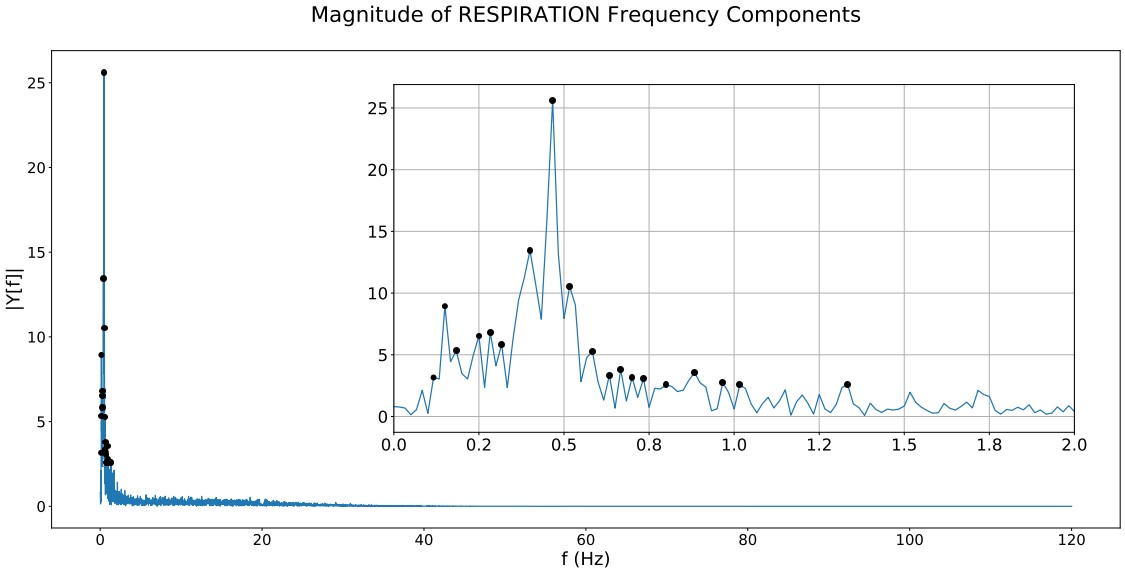


Figure 29: Respiration Periodogram. It's possible to see in the smaller, focused plot, that the majority of energy lies between 0.1 to 0.8 Hz.

A final feature that builds on the same concepts is the spectrogram images. These images are a product of feature mapping function ϕ_{Spec}. To simplify the idea, it is like applying the FFT for selected windows of time.

Algorithm 9 Spectrogram Images Feature Mapping ϕ_{Spec}

Require: Signal $\mathbf{x}_{raw}$, time window T
Ensure: Empty peak array $\mathbf{D}$
  for all for t in range $[0, T)$ **do**
    Apply FFT from [27] FFT package to attain array $\mathbf{X}[k]$ for first second of interest
    Reset $\mathbf{X}[k]$ to only one half of the spectrum
    Append $\mathbf{X}[k]$ to $\mathbf{D}$
  end for
  return $\mathbf{D}$

Algorithm 9 is rather simple: for every window size that is given, the FFT is calculated and stored. At the end, the resulting image is composed of columns that contain frequency information for that window of time. Figure 30 illustrates the result of this sequential FFT calculation. The lines which are easier to see are frequency bands that had significant levels of energy at that moment in time. Moreover, it's also possible to see that the lines fade as frequencies become larger. Additionally, in specific windows of times there is more or less activation at higher frequencies. These interesting characteristics are vital for capturing and understanding information about events that lead to abnormal health states.

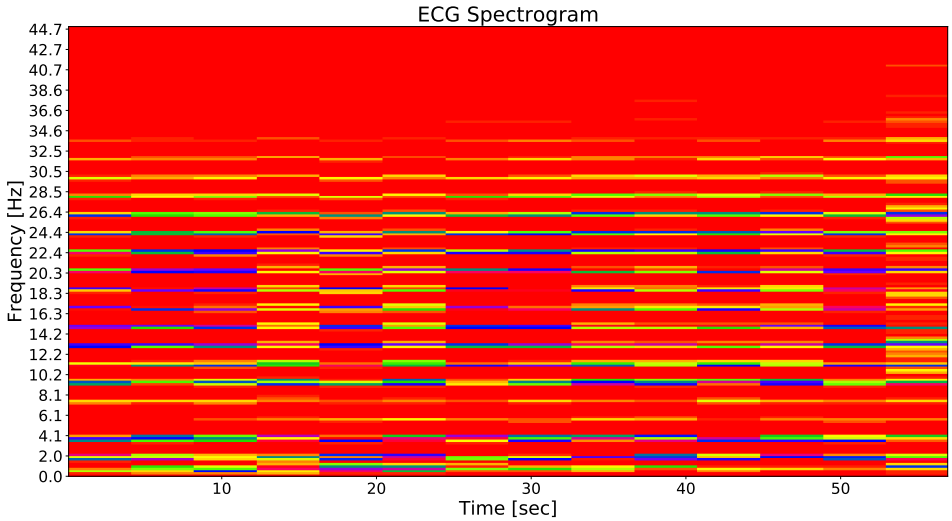


Figure 30: Spectrogram of an ECG signal.

3.4.2 Wavelets Decomposition

Wavelet Decomposition (WD) was another method that was used to extract features. From WD two feature mappings were created: component coefficients ϕ_{Wave}, and a collection of statistics from every wavelet level and component extracted, $\phi_{WaveStats}$. Wavelet decomposition (or expansion) of a signal is an analytical method that

allows one to reduce the dimension of the signal, while providing a time-frequency localization [28]. This means that with a small number of coefficients we can represent most of the energy in the signal. In addition, most discrete wavelet transform algorithms have similar run time as the FFT, O(N log(N)), making it an efficient data structure agnostic option for feature extraction.

There are different types of wavelets, and they may be tailored for the problem of interest. The most common types are Haar and Daubechies Wavelets [28]. The wavelets chosen for a decomposition task are referred to as "mother wavelets", or "prototype wavelets" [29], and can be generalized as per Equation 17.

$$\Psi_{a,b}(t) = \frac{1}{\sqrt{a}} \cdot \Psi(\frac{t-b}{a}), \text{ where } a, b \in \mathbb{Z} \tag{17}$$

$$\mathbf{X}_W(a,b) = \int_{-\infty}^{\infty} x(t) \cdot \Psi_{a,b}(t) \cdot dt \tag{18}$$

a	Scaling factor
b	Translation amount
Ψ	Specific wavelet function used
t	continuous time variable
$x(t)$	Continuous time signal
$\mathbf{X}_W$	Resulting wavelet transform

Equation 18 provides the basic definition of the Wavelet Transform. This definition shows how the scaling and translation properties of a mother wavelet $\Psi_{a,b}(t)$ can be used to transform an input function $x(t)$, or in the discrete case $x[n]$ to an alternate representation that provides a different view of the initial information. Further details on the topic are beyond the scope of this thesis, however the reader may refer to [25] and [28] for a more comprehensive treatment on wavelets and wavelet transform.

In addition to Equation 18, an alternate form used is the Multi-Resolution Wavelet Analysis definition. This form highlights the components that emerge from the transformation and gives insight on how a signal can be completely reconstructed. Equation 19.

$$x(t) = \sum_{a=-\infty}^{\infty} c_a \phi(t-a) + \sum_{a=-\infty}^{\infty} \sum_{b=0}^{\infty} d_{a,b} \cdot \Psi_{a,b}(t) \tag{19}$$

The concept of multiresolution wavelet decomposition provides an opportunity to explore *Low* and *High* frequency components. These components are achieved by passing the signal through low pass and high pass filters for some determined amount of l times (or levels). At each pass the low pass filter $\mathbf{L}$ cuts the sequence in length by half by only keeping its lower spectra components. The high pass filter $\mathbf{H}$ produces the same effect. This set of operations decimates the signal in time and frequency, allowing one to pick and choose which coefficients and the desired resolution of said coefficients. Figure 31 helps illustrate how this is achieved.

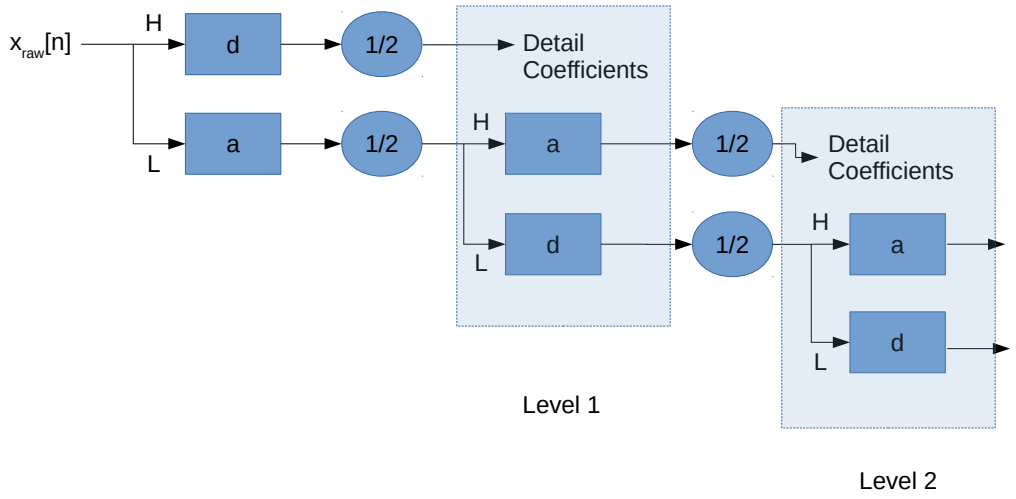


Figure 31: Graph showing the wavelet level decomposition. **a** contains the low level components, while **d** contains the high level, detail coefficients

Different kinds of wavelets and levels of decomposition were explored in an attempt to find the optimal feature vector. The decomposition scheme chosen to produce the feature used in the machine learning step was the Low Frequency, Second Level Component, of a ten level WD produced with a Daubechies 5 (DB5) wavelet.

Algorithm 10 Wavelet Feature Mapping ϕ_{Wave}

Require: Signal $\mathbf{x}$, output level l
Ensure: Empty peak arrays $\mathbf{A}$ and $\mathbf{D}$ are initialized
  set $\mathbf{a} \leftarrow \mathbf{x}$
  Set `dwt` $(\cdot)$ to `dwt` from [30] PyWavelets package
  for all l in range $[0, 10]$ **do**
    $\mathbf{a}, \mathbf{d} \leftarrow$ `dwt`$(\mathbf{a}, \text{db5})$
    Append $\mathbf{a}$ to $\mathbf{A}$
    Append $\mathbf{d}$ to $\mathbf{D}$
  end for
  return $\mathbf{A}[l]$

Algorithm 10 shows how a low level frequency component in level l is extracted to be used by a machine learning algorithm. Figure 32 illustrates the resulting waveform that is created by the coefficients extracted from an ECG waveform.

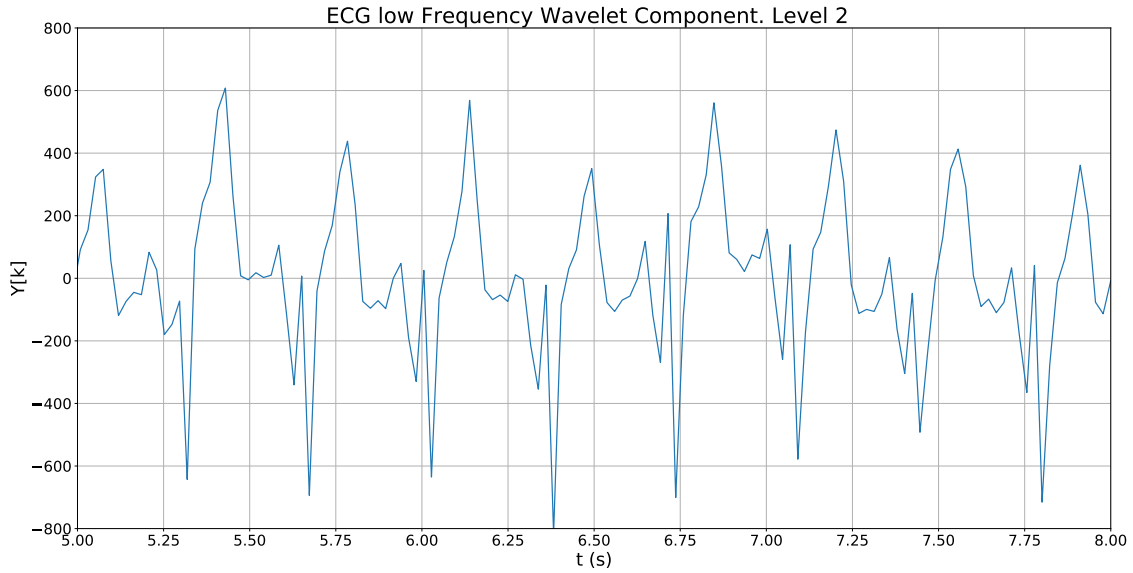


Figure 32: ECG Wavelet Components.

In addition to the chosen wavelet component statistics about the components were also computed and concatenated into a feature of its own. The driving idea behind this was to further reduce the dimension of the wavelet information available to a set of statistics and evaluate its predictive power. Algorithm 11 describes the specific instructions that comprise $\phi_{WaveStats}$. As previously mentioned, the goal to is further reduce the dimension of the resulting feature vector, while also including information about every component computed.

Algorithm 11 Wavelet Statistic Feature Mapping $\phi_{WaveStats}$

Require: Data Structure containing frequency components **X**
Ensure: Empty array **D** is initialized
  for all Component **c** in Data Structure **X** **do**
    Get statistics max, min, mean and medium $c_{max}, c_{min}, c_{mean}, c_{medium}$
    Append $c_{max}, c_{min}, c_{mean}, c_{medium}$ to **D**
  end for
  return **D**

3.5 Conclusions

The research, implementation and validation of the features implemented in this section was the most challenging and extensive portion of this work. It required exhaustive study of the datasets and tools available to more efficiently (in terms of development time and computational efficiency) develop the feature mapping functions.

As it will be further detailed in section 4 and section 5, not all features developed were used in the final predictive modeling and evaluation phase. This is because it would have taken much more time to implement support and evaluate the predictive

benefit of all features. In future works, the features that were not used in the current research work will be evaluated, so to validate whether there is value in the information they provide.

Finally, it should be noted that the feature engineering exercise produced in this section proved to be a great software engineering and signal processing venture that allowed for the creation of a waveform Feature Extraction System that was sufficiently modular such to be able to compute features efficiently for SpO2, Respiration and ECG waveforms.

4 Exploratory Data Analysis

This chapter shall discuss the efforts pertaining to Exploratory Data Analysis(EDA) that were developed. The main goals are to bring insight into the data that is being used and difficulties that exists with the use of this data.

4.1 Cohort Selection

The first topic that will be explored is the cohort selection. This refers to the act of choosing the population for an experiment. The cohort selection was produced by exploring a set of patient data that included patients whose diagnoses are related to sepsis. By identifying patients who were diagnosed with Sepsis it was possible to create a positive cohort, as defined in Definition 4.1.1 and a negative cohort, defined in Definition 4.1.2.

Definition 4.1.1. *The positive cohort shall primarily be composed of distinct encounters*[5] *that have the following inclusion criteria:*

1. *Patients containing the word "sepsis" or "septic" in their diagnoses name*
2. *Patients who do not have a septic diagnosis when they were admitted to that particular encounter*
3. *At least 48 hrs of waveform data*[6]

The exclusion criteria goes as follows:

1. *Patients that do not have waveform information that is verified.*

Definition 4.1.2. *The negative cohort is composed of patients that have different encounters, are not in Definition 4.1.1, and satisfy the following criteria:*

- *At least 48 hrs of waveform data*

The exclusion criteria goes as follows:

- *Patients that do not have waveform information that is verified.*
- *Patients that contain the following words in their diagnoses:*
  - *sepsis*
  - *septic*
  - *cardiac*
  - *atrial*
  - *tachycardia*
  - *artery*
  - *ventricular*
  - *aortic*
  - *respiration*

[5] Encounters are defined by distinct periods that a patient may have spent in the hospitals. Encounters may be days, months or years apart. Consequently, the assumption is that every different encounter may contain a distinct set of information, even if it pertains to the same patient, and therefore contains information that may be considered distinct.

[6] This is to make sure that the cohort contains enough waveform data for prediction. This is further explained in the following sections.

- *cardio*
- *heart*
- *pulmonary*
- *lung*
- *breath*
- *diastolic*

The positive cohort in Definition 4.1.1 had all diagnostic entries composed of some septic related condition. The cohort achieved with Definition 4.1.2 contained patients that did not have sepsis in their EHR history. In order to provide light into the different diagnoses present in the GEHC dataset, Table 3 provides a list containing the fifteen (15) most predominant diagnoses encountered in the EHR:

Diagnoses Name	Number of Values
Sepsis(995.91)	870
Unspecified septicemia(038.9)	862
Other specified cardiac dysrhythmias(427.89)	772
Severe sepsis(995.92)	724
Septic shock(785.52)	466
Cardiac arrest	262
Urosepsis	252
Sepsis	180
Cardiac pacemaker in situ	155
Automatic implantable cardiac defibrillator in situ	151
Sepsis, unspecified organism	144
Cardiac dysrhythmia, unspecified	134
Severe sepsis with septic shock	133
Septic shock	114
Cardiac complications	90
Total Number of diagnoses encountered :	230981

Table 3: Table demonstrating the top 15 diagnoses encountered in the Diagnoses table

A closer look into the numbers of similar diagnoses encountered provides insight into the different ways (and therefore inconsistent) format in which clinicians enter information on the EHR systems. Although a significant portion of the rows in Table 3 contains some form of sepsis, there does not exist a consistent entry measure, making the cohort selection more difficult. This difficulty lies on the fact that the same condition may be expressed in different formats in the Diagnoses EHR tables. Consequently, in order to capture cases which are sufficiently general, where only patients that have septic related conditions are chosen, one must control inclusion parameters by checking the diagnoses name fields for specific words. Hence the list of conditions included in Definition 4.1.1 and Definition 4.1.2.

4.2 Waveform Continuity Analysis

An important case to consider with biosignals is discontinuity. Discontinuity may be caused by medical personnel disconnecting the monitoring devices for short periods, patient movement, connectivity issues, etc. These artifacts must be taken into account because they cause noise in the input data.

Now, as one may expected, a patient's movement, or the case where a physician may disconnect one monitoring device, for example the SpO2, but leave ECG and Respiration connected. This causes discontinuity in one waveform, while the others are intact. Therefore, as a starting point to settle any mutual waveform analysis or extraction of continuity information, it is necessary to align start and end points of each the waveforms that will be used (in this case Respiration, SpO2 and ECG) to a mutual start and end point. For this purpose the following Algorithm 12 was designed and implemented:

Algorithm 12 Selecting mutual start and end points for RR, SpO2 and ECG pertaining to same encounter

Require: Respiration, ECG, and SpO2 signals $\mathbf{X} \in \Re^{d \times 3}$
Require: Start time t_s and end time t_e within an encounter
Ensure: Same length of valid signals $\mathbf{Y} \in \Re^{d' \times 3}$
  for all waveform $\mathbf{x} \in \mathbf{X}$ **do**
    Find index set $\mathbf{i}$ which does not contain $\emptyset$ values or values ≤ -32700 (Very small int denomination)
    Append smallest time stamp v_i to $\mathbf{v}$, containing start values
    Append largest time stamp v_j to $\mathbf{r}$, containing end values
  end for
  Set mutual start time t'_s to $max(\mathbf{v})$
  Set mutual end time t'_f to $min(\mathbf{r})$
  return t'_s, t'_f

In order to further explore the situation of discontinuity within the waveforms, it was necessary to retrieve all the windows of continuity in the waveform. This would provide time stamps that point to every start and end point of a continuous strip of data. Algorithm 13 describes how this is done.

Algorithm 13 Find continuous time points for a specific waveform

Require: Some waveform signal $\mathbf{x} \in \Re^{d\times 1}$, and a vector $\mathbf{t} \in \Re^{d\times 1}$ containing time stamps for each entry in $\mathbf{x}$
Require: t_m = Max time difference accepted (may also be seen as max period between entries)
Ensure: A table $\mathbf{X} \in \Re^{d\times 2}$ is created with a time stamp
Set table $\mathbf{Y}$ to the list of all values in $\mathbf{X}$ which are > -32700 or **not** $\emptyset$
Initialize reference point array $\mathbf{r} \in \Re^{z\times 2}$ to contain start and end points encountered

for all index $i \in \mathbf{Y}$ **do**
  Set data from the following sample $j = i + 1$ to $\mathbf{v}$
  Set data from current index i to $\mathbf{w}$
  $dt = \mathbf{v}_t - \mathbf{w}_t$, where $\mathbf{w}_t$ indicates the time entry of $\mathbf{w}$
  if $dt > t_m$ **then**
    Append start time stamp $\mathbf{v}_t$ to $\mathbf{r}_{t_0}$, containing start values
    Append end time stamp $\mathbf{w}_t$ to $\mathbf{r}_{t_f}$, containing end values
  end if
end for
return $\mathbf{r}$

Algorithm 13 was applied to a total of 500 waveforms. For this set of waveforms every start and end point of a continuous window was retrieved. This helped identify how much discontinuity existed in the waveforms that were being used.

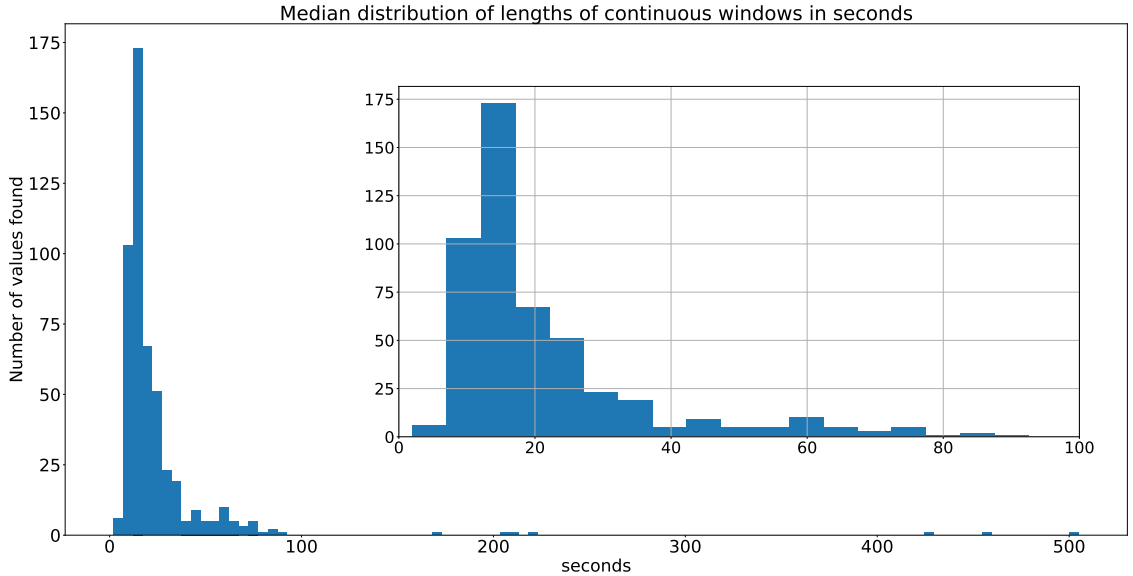


Figure 33: Distribution of the length of continuous windows found in the waveforms examined.

Figure 33 illustrates the distribution of lengths for the continuous windows encountered. The result implies that there is a significant amount of discontinuities that

occur consistently in all 500 recordings. In fact, it was found that most discontinuities are one (1) second or less, and occur approximately every twenty (20) seconds.

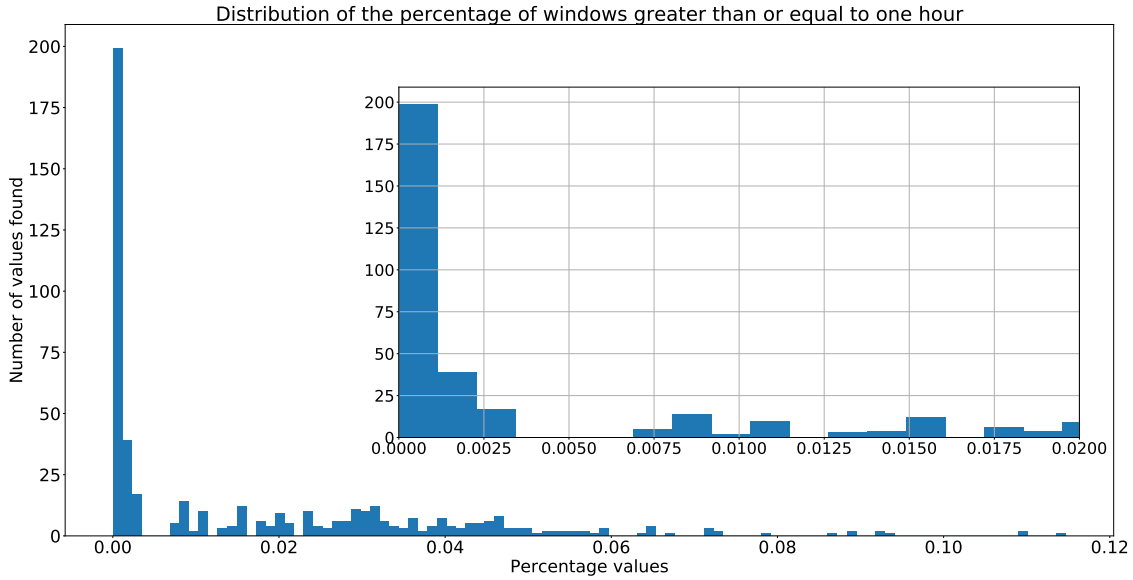


Figure 34: Distribution of percentages of windows which are at least one hour long, for every waveform chosen in the cohort set.

Figure 34 further highlights the sparsity of *significant*[7] continuous windows. Over the set of waveforms analyzed, only three (3) waveforms had more than 5 significant continuous windows. In fact, if one considers Figure 33, it is possible to note that the majority of continuous windows are no longer than 30 seconds.

4.2.1 Expected Feature Impact

Since a significant amount of discontinuity was found in the data, questions were raised as to how the features would be impacted by these artifacts. This section will show the results of this consideration.

The waveforms have values that are equal to or less than -32000 for points that are considered discontinuous. Therefore, from a computational perspective, the feature extraction system process the discontinuous strips as if they are regular waveform values. An example of these discontinuities for an ECG waveform is presented in Figure 35, where the effect of discontinuity in the raw waveform and in the filtered waveform is shown.

[7]For this experiment, we have defined significant continuous windows as a vector $\mathbf{x}$ which contains at least one (1) hour of information. Since the sampling rate $f = 240$, $|\mathbf{x}|$: length of vector $\mathbf{x} \rightarrow f \cdot 3600 = 864000$ samples

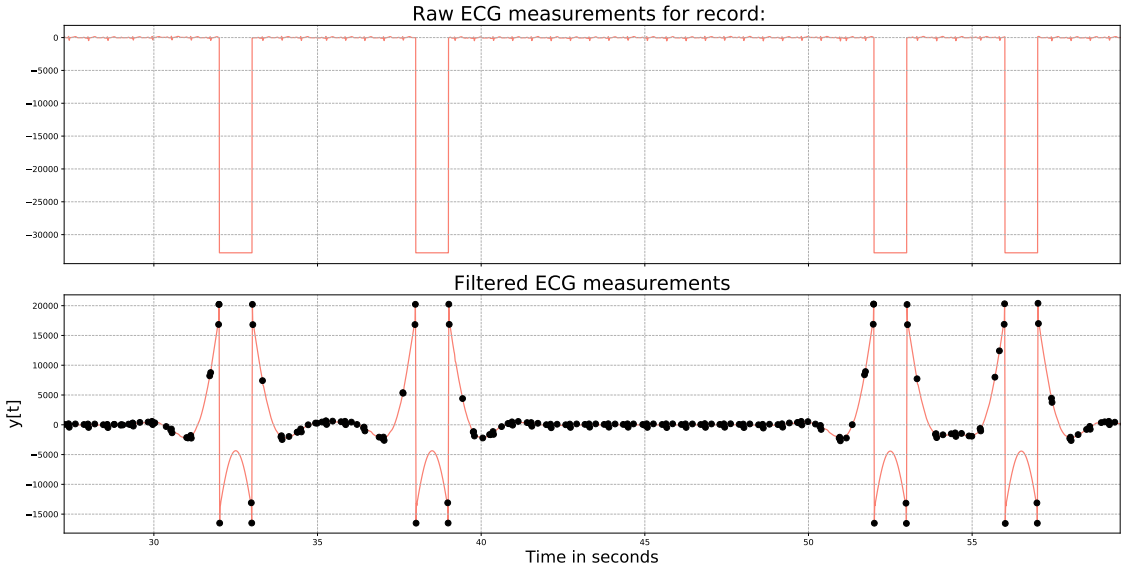


Figure 35: Providing an example of discontinuous points in an ECG waveform

Figure 35 shows the effect of discontinuity in the raw waveform and the filtered waveform. Moreover, the effects on the peak selecting algorithm are also displayed.

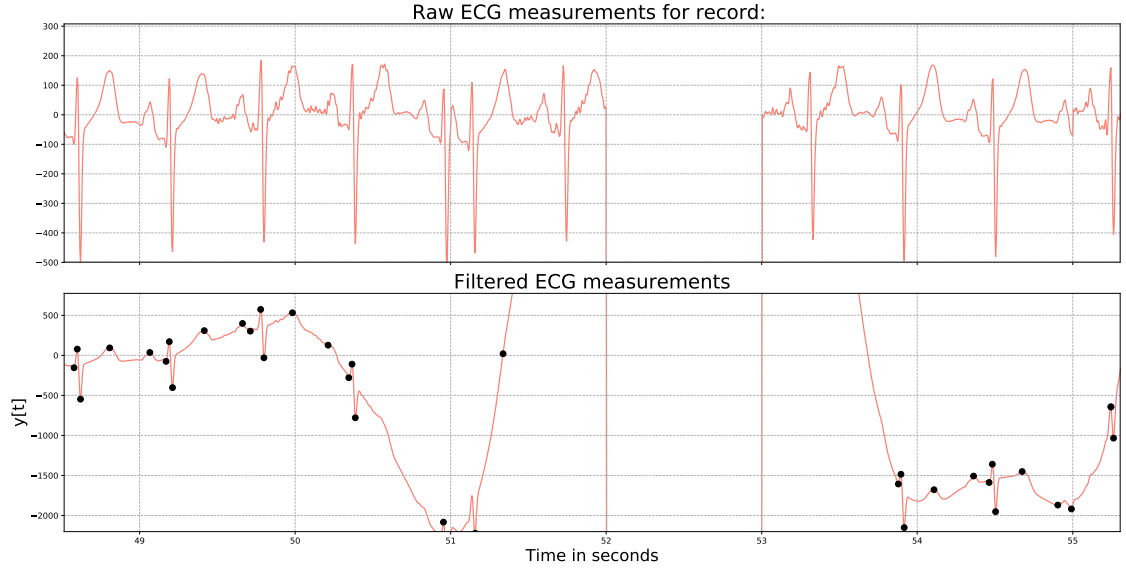


Figure 36: By zooming into the discontinuity, its possible to note the impact of discontinuity in the peak selection algorithm

Figure 36 shows that the peak selection algorithm manages to select real beats and artifacts. However, due to the noise incurred by the artifacts, there are distortions in the waveform morphology, as well as added spurious peaks which are not produced by the human heart.

Since these values are not changing for some length of time, features that represent changes over time, such as heart rate, respiration rate, time variability statistics will demonstrate zero change. However, in contrast frequency related features such as the

spectrogram images will have most of its energy in the windows of time at which the discontinuity occurs, since the filtered signal shows high peaks during those moments of time.

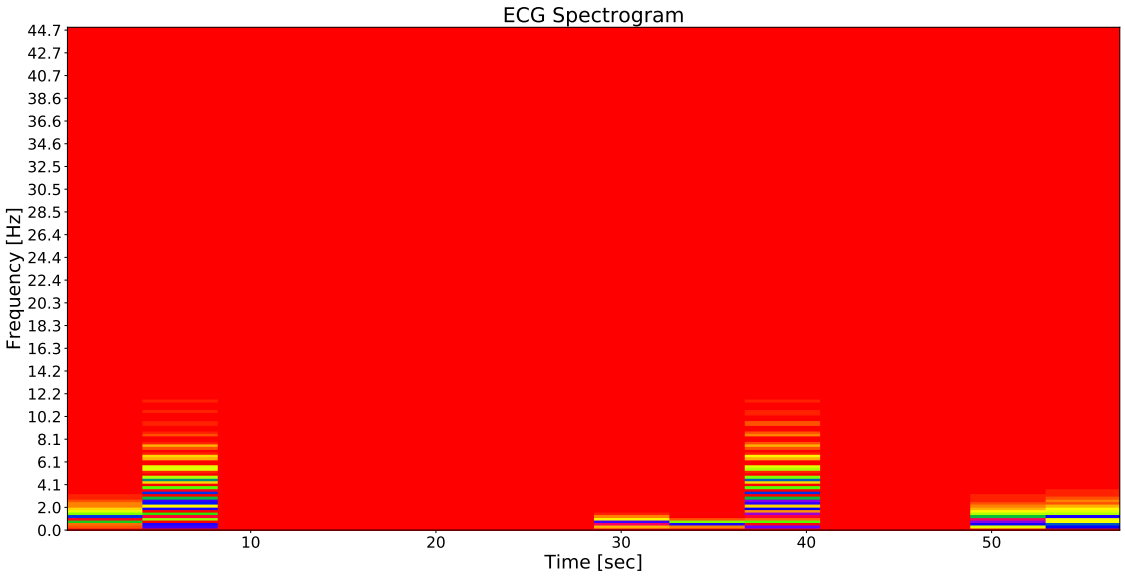


Figure 37: Spectrogram image of one minute window containing discontinuities.

Figure 37 shows the spectrogram containing discontinuities. Energy is seen in windows of time where discontinuities occurred. This high concentration of energy dwarfs the energy produced by the true periodic components of the biosignal, consequently distorting the feature.

Due to the latter, an arbitrary choice was made to change the very low numbers representing discontinuity into 0 values. This choice was based on the fact that for the filtered signals, the very high energy is generated by imputing large negative values. If the values are replaced by 0, the large distortions incurred in the filtered waveform should be resolved.

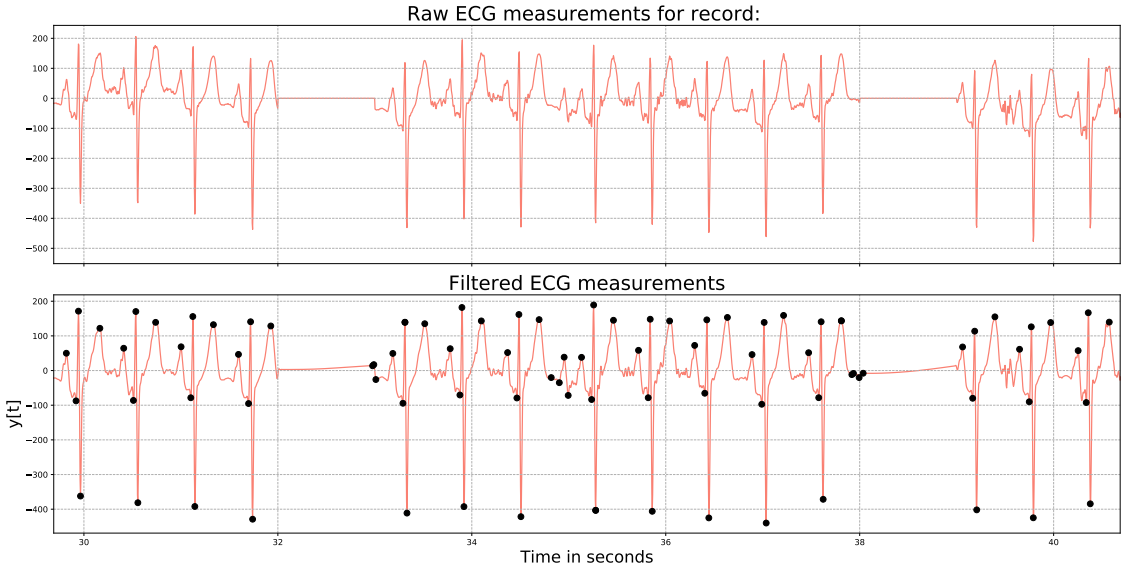


Figure 38: Raw and filtered ECG waveforms containing the zero (0) values for discontinuous strips.

Figure 38 shows the resulting effect of the zero based discontinuous value change. There is less distortion to the filtered signal, and peak selection is more reliable. Finally, Figure 39 also supports the benefit of the change from very low integer values to 0, as the spectrogram images appear to more closely illustrate the energy distribution of the ECG signal.

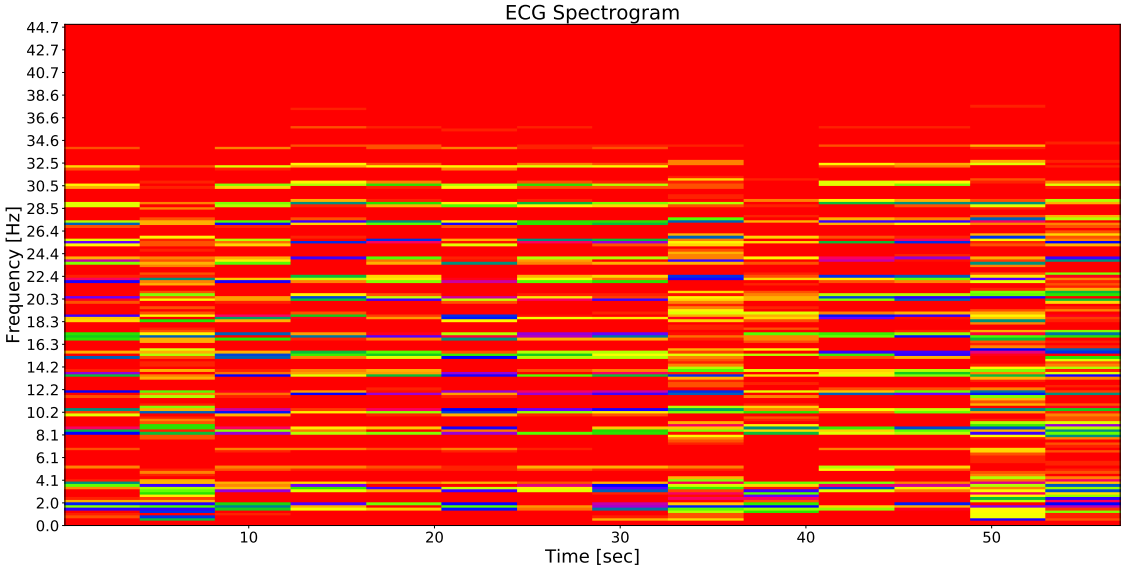


Figure 39: Spectrogram image of one minute window containing discontinuities with zero based discontinuous values

4.3 Statistical Evaluation

In addition to the data evaluation performed in the previous section, a statistical evaluation of the data was also performed for a selected set of features. The main goal for this study was evaluate whether or not there existed some statistical difference between septic and non-septic patients.

4.3.1 Defining Statistical Difference

A number of methods which help define the statistical difference between two sets of number are available. The method chosen here was the Pearson's Chi-Squared Test [31].

$$\chi^2 = \sum_{i=1}^{N} \left\{ \frac{(O_i - E_i)^2}{E_i} \right\} \tag{20}$$

O_i	Number of observations for the *ith* value
E_i	Expected number of observation for the *ith* value
N	Number of values (or bins) used in the split
χ^2	Pearson's test statistic

The reason for choosing Equation 20 (and Pearson's work [31]) was due to the fact that the aim was to find out whether the difference between an array containing the *frequencies* of observed values **O** was statistically different than the expected frequencies contained in an array **E**. Frequencies refer to the number of observations found for a specific value (or value range). For example, if 20 features were found to have the value of 5, then 20 will be stored in **O**. If in turn we expected there to be 23 features to have the value of 5, then 23 will be will be in **E**. More specifically, array **O** contained the frequencies encountered for septic patients and array **E** contained the frequencies encountered for non-septic patients. If the case was that there existed no difference between observed and expected arrays, then a null hypothesis would be accepted. However, if some significant difference was encountered, than the null hypothesis stating that there is no difference between septic and non-septic patients cannot hold. The following formal hypotheses can be posed:

Definition 4.3.1. *H_0: Null hypothesis implies there is no significant statistical difference between* **O** *and* **E**

Definition 4.3.2. *H_1: Alternate hypothesis implies that there is a significant statistical difference between* **O** *and* **E**

Significant in Definition 4.3.1 and Definition 4.3.2 is defined by p-values of the χ^2-statistic < 0.05. If the observed p-value of the test statistic is smaller than the chosen level(i.e. false alarm rate) of the test, chosen as $p_{FA} = 0.05$, then the null hypothesis is rejected.

4.3.2 Results

in order to restrict the evaluation into an exercise that fits the timeline of the thesis, the experiments were performed only for features that capture time variability, as described in Algorithm 7, and were extracted in time chunks of 600 seconds (10 minutes). This means that for every window of ten minutes ϕ_σ was applied to achieve an array $\mathbf{z}_\sigma$. These arrays are stored for each specific waveform, and patient. Finally, the resulting time variability values $\mathbf{z}_\sigma$ are stored with labels that contain information about whether or not that patient was diagnosed with some Septic related illness or would be diagnosed with a septic related illness in the next 24 hours.

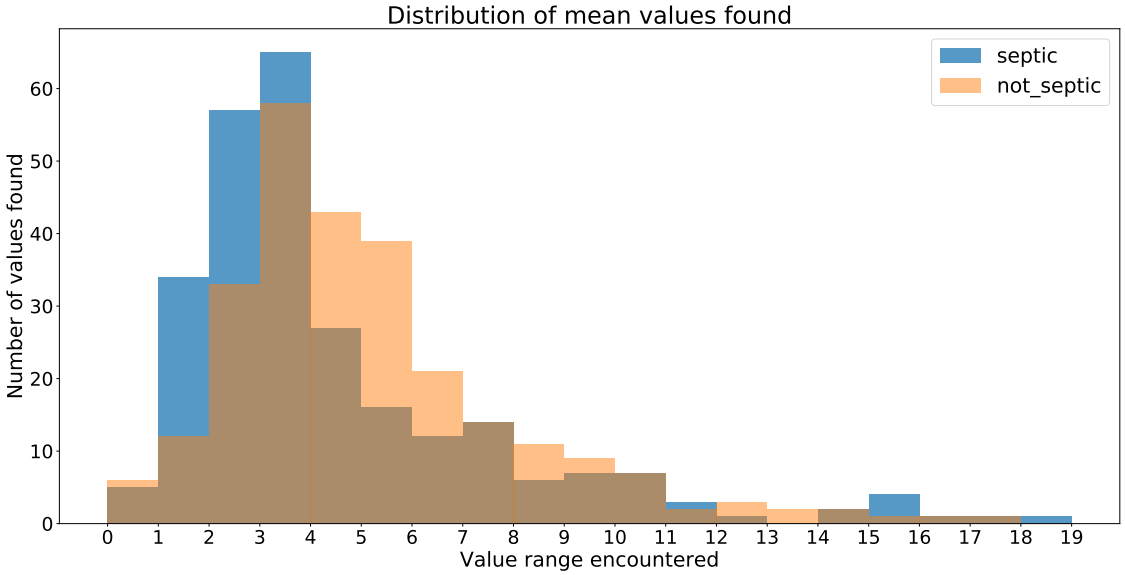


Figure 40: Distribution of time variability values $\mathbf{z}_\sigma$ encountered in the dataset

The distribution plot displayed in Figure 40 has values distributed in the range of [0,20), where every integer values is a bin. The value range was chosen based on experiments that showed the majority of $\mathbf{z}_\sigma$ the [0,20] value range. Both sets, septic and non-septic are divided with the same structure, so the statistical difference between the two can be evaluated.

Finally, based on the division of values presented above, the sets septic and non-septic present a $\chi^2 = 40.85$ and a p-value equal to 0.00159, based on 18 degrees of freedom. This result leads to the rejection of the null hypothesis Definition 4.3.1, thereby allowing the assertion that there exist a significant statistical difference between time variability values of septic and non-septic patients.

5 Predictive Modeling

The development of predictive models for the biosignals used in this work and the form in which results were produced is explored with details in this chapter. Finally, results and concluding notes are presented.

5.1 Chosen Features

In section 3, a number of features (as defined in Definition 3.2.1) and mapping functions (as defined in Equation 10 and Equation 11) were proposed. Due to the large number of features, and the necessity to produce a predictive analysis that was within the time frame of a Master's thesis, four (4) features were chosen to be analyzed separately. The main goals behind this feature choice were the following:

1. Find information on the predictive potential of the different domains explored
2. Investigate the effects of further reducing the dimension of the features

Based on the above notes, the following features were chosen: $\mathbf{z}_{\sigma}$, $\mathbf{z}_{Max}$, $\mathbf{z}_{Wave}$, $\mathbf{z}_{WaveStats}$. When considering the chosen features, $\mathbf{z}_{\sigma}$ for example, its possible to note that it provides insight on the value of the time domain of the signals. It is a feature that is built on a number of other lower level features (ϕ_{Δ} is one of them), while still reducing the time domain information to a lower dimension (the standard deviation of heart rate over a specific window). On the other hand, $\mathbf{z}_{Max}$, provides direct information about the most predominant frequencies in the time window of interest. Feature $\mathbf{z}_{Wave}$ provides a time-frequency value to the analysis. Finally, $\mathbf{z}_{WaveStats}$, brings about insight into how a statistical collection of time-frequency components may add value to the predictive modeling.

5.2 Training, Validation and Test Setups

This section will explore how the training and validation steps were done. In addition, the format in which the data was split will also be explored.

The data was split in a patient wise format, where the superset of patients (or cohort) $\mathcal{S}$ containing positive and negative patients were divided randomly. This resulted in three different sets: Training set $\mathcal{X}$ containing 50%, Validation set $\mathcal{V}$ containing 25% and Test set $\mathcal{T}$ containing 25% of observations as illustrated in Figure 41. This was done in order make sure that the patterns that were found for one patient would also be true for different patients [32].

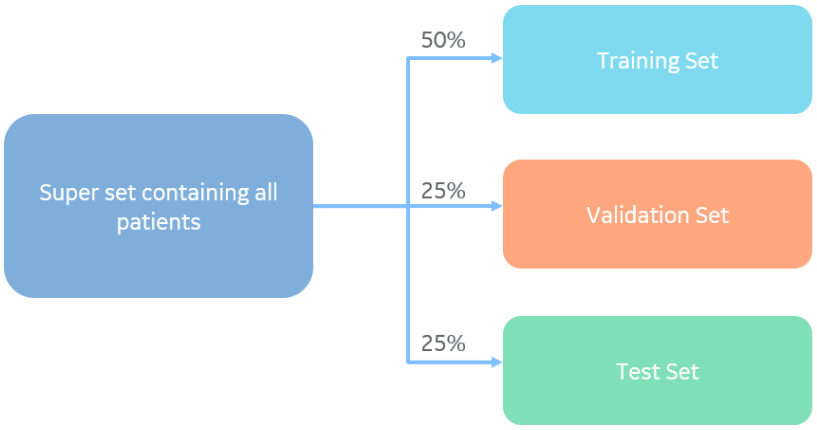


Figure 41: Illustration demonstrating how the different sets were broken down.

During training, the validation set was used to verify whether or not the model was learning from the data. In the final verification step, the trained model was tested against a tertiary set called the Test set $\mathcal{T}$. This provided the final performance metric for the model.

5.2.1 Computation and Data Operations

Storing, loading and computing was done predominantly in the infrastructure service provided by AWS. In essence, different EC2 instances with different capabilities were created in order to handle the computation loads. The datasets were kept in an S3 bucket, where it was accessed and loaded on demand. Python Jupyter Notebooks were used for visualization and experimentation.

In order to load and organize the data, different types of EC2 instances with different CPU and RAM capabilities were used. For example, an instance containing 96 cores and 380 Gb of RAM was extensively used for feature engineering and data organization, where CPU and RAM requirements were very high. Figure 42 helps illustrates the steps that were necessary to produce the training.

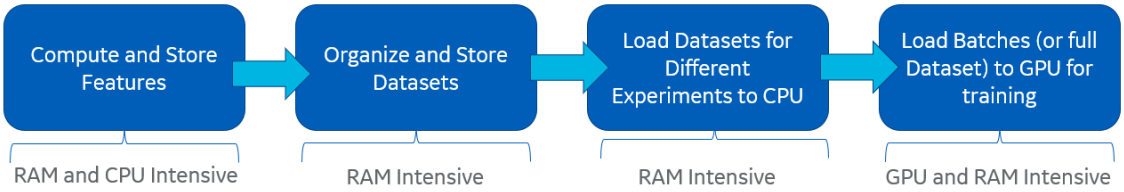


Figure 42: Illustration demonstrating the computation flow and different areas of impact on RAM, CPU and GPU.

Moreover, by organizing the data in Pandas DataFrame tables and storing the information in AWS S3 Storage, it was possible to produce massively parallel [8]

[8]The parallelization was achieved by using Python's Multiprocessing library.

computations that computed features and organized the data in a structured format (Pandas DataFrame). Finally, splitting the data frames as illustrated in Figure 43 allowed the computing units to be able to load smaller chunks of information, thereby accelerating load times and preventing RAM deficiency problems.

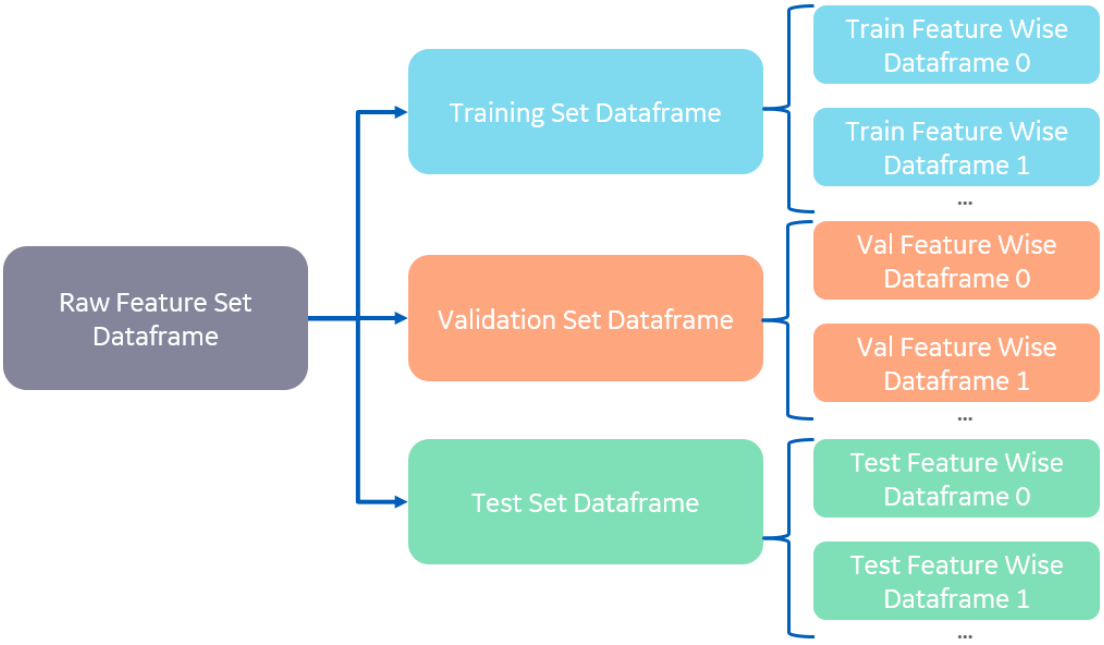


Figure 43: Illustration demonstrating how the data is subsequently split, and its format

5.3 Data Transformation

The first transformation to the input data was setting the final sampled time window to 6 hrs. As a result, each sample in $\mathcal{T}, \mathcal{X}$ or $\mathcal{V}$ corresponds to a feature, say $\mathbf{z}_{Max}$, sampled for some 6 hrs period within the sampling window as illustrated in Figure 2. The sample is any 6 hrs period within the sampling window. The gap window is set to 3 hrs, while the prediction window is constant at 24 hrs (since diagnoses information are set at day resolution). All data transformations from this point builds on 6 hrs worth of feature information.

5.3.1 Reducing Dimensions

A significant challenge encountered during the predictive modeling phase was the large high dimensionality of the inputs. Table 4 shows the dimensions of $\mathbf{z}_{Max}, \mathbf{z}_{Wave}$, $\mathbf{z}_{WaveStats}$ and $\mathbf{z}_{\sigma}$ for sampling lengths of 6 hrs.

An example which illustrates this complication is the number of training samples in $\mathcal{X}$: 250, while the input dimension for feature vector $\mathbf{z}_{Wave}$ is 658252. As it is mentioned in [33] and [22], having a training set where the number of observations are much smaller than the number of features, leads to complications in the optimization

phase of the machine learning. This is due to the fact that there are many more parameters to learn than there are observations.

Even with the feature engineering introduced, where signals were reduced to much smaller dimensions, there was still a need to further reduce the dimension of input features $\mathbf{z}_{Max}$ and $\mathbf{z}_{Wave}$. The fact that the largest number of observations in $\mathcal{X}$ was 250 samples further reinforced this necessity. Table 5 provides insight into the specific number of encounters (time period that a patient spends in the hospital) within each set used in training, validation, and testing.

Sampling Length	Feature Name	Dimension	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{T}\rvert$
21600s (6 hrs)	Time Variability Values ($\mathbf{z}_{\sigma}$)	36	250	134	115
	Wavelet Statistics Values($\mathbf{z}_{WaveStats}$)	1584			
	Wavelet Values ($\mathbf{z}_{Wave}$)	658252			
	Frequency Hz Values ($\mathbf{z}_{Max}$)	216000			

Table 4: Table demonstrating the final feature dimensions for features extracted in time chunks of 10 minutes

Sampling Length	Set	Number of encounters
21600s (6 hr)	$\mathcal{X}$	60
	$\mathcal{V}$	30
	$\mathcal{T}$	30

Table 5: Table demonstrating the number of encounters per set

As a consequence of very high dimensionality, specific methods were explored in order to map the features into smaller dimensions. Namely, Principal Component Analysis (PCA).

In order to increase the chance to reach a unique global solution the number of features should be much smaller than the number of observations available [34] [35] [22]. As a consequence, the dimension sought for during dimensionality reduction was a tenth (10%) of the number of observations available for the training set $\mathcal{X}$.

Aside from feature engineering, PCA was the main method of reducing the dimension for the feature vectors. The general idea is to find an orthogonal projection of the data into a small subspace [33]. The projection is produced by leveraging the covariance matrix of the data and its eigen-value decomposition. Projecting observations onto eigen-vectors generates a number of components from which one is then able to choose the whole set, or a subset of the components. The specific implementation used for the computation was Scikit-Learn's PCA implementation [36].

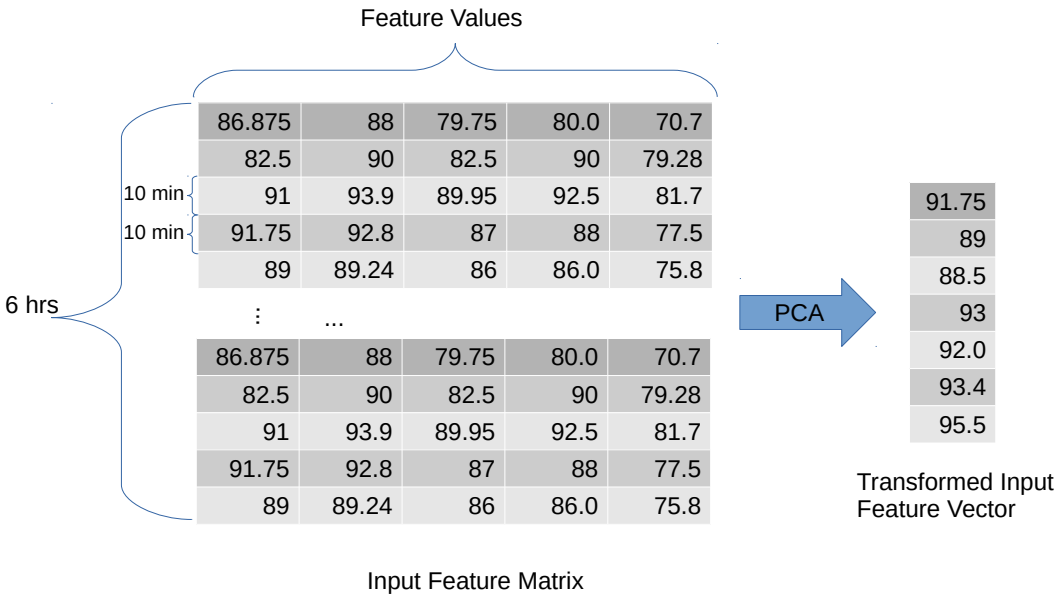


Figure 44: Illustration of how PCA was applied to the input feature vectors. The numbers in the cells are for illustration purposes only.

Figure 44 shows how PCA is applied in the context of this research work. An input matrix containing features, say for example $\mathbf{z}_{Max}$ for every 10 minutes of a 6 hour window is used as input. PCA is applied to find the most relevant set of values that can represent this matrix in a smaller dimension. The resulting transformed matrix (now a vector), is used as the input to either the machine learning model, or a scaling transform algorithm. Finally, although scaling algorithms were tested, the best set of test results achieved (presented in subsection 5.5) did not include scaling.

5.3.2 Data Augmentation and Batching

Since the number of samples available was limited and dimensions were large, an alternative to expanding the current sample set was desirable. Reference [22, Chapter 7] provides an explanation on the benefits of data augmentation and noise. Building from that work the augmentation work and noise infliction was explored. First samples were replicated and then a small amount of noise was added to the features, so they may be perceived as different input. Figure 45 illustrates the different steps.

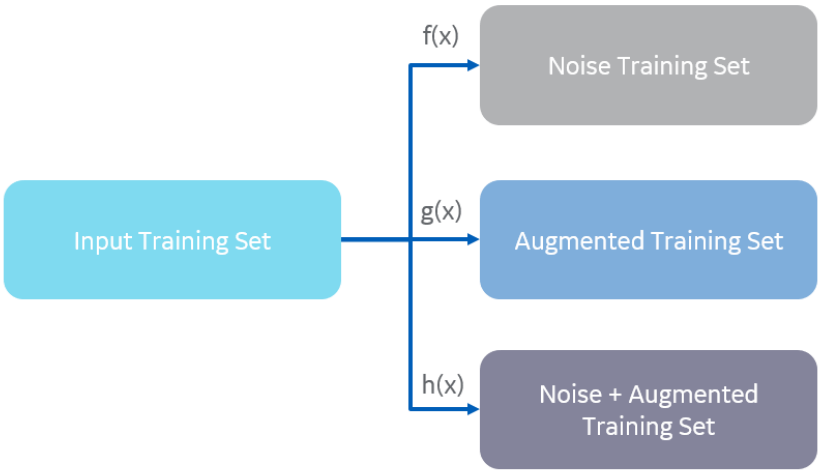


Figure 45: Procedures used for augmenting and inflicting noise to input data.

In Figure 45 each function, $f(\mathbf{x}), g(\mathbf{x})$, and $h(\mathbf{x})$, illustrates a mapping between the input set into another set which is used for training. In the case of $f(\mathbf{x})$, the noise function used to induce noise into the input was a Rayleigh distribution. This was chosen due to the work performed in the EDA section and distribution fitting seen in Figure 40.

Now, function $g(\mathbf{x})$ performs augmentation by randomizing indexes and concatenating the result to the input array $\mathbf{x}$. Finally, function $h(\mathbf{x})$ is a combination of $f(\mathbf{x})$ and $g(\mathbf{x})$, which performs first the augmentation, and then it adds noise to added samples. The result is a larger number of observations with a few noisy features.

5.4 Machine Learning Models

In order to validate the predictive potential of the chosen features within their sampled time windows, two models were explored: Random Forest (RF) and Convolutional Neural Network (CNN) model. The following sections will explain the reasoning behind each model and the final model structure achieved.

5.4.1 Random Forest

Random Forest is a method that derives from a class of classification algorithms called *bagging* [34]. Bagging is essentially regularizing a set of predictions by their mean, and utilizing the result. This produces a solution that is able to perform well with unseen data and has lower variance. Random Forest builds on these ideas by building a large set of de-correlated decision trees and averaging them. Algorithm 15.1 in [34] provides further details on how Random Forest works.

Now, a diagnosis is a conclusion which clinicians make based on a set of patterns that have been identified during some window of time which they have observed the patient. Clinicians usually have a consistent set of patterns which they search. For example, high heart rate, or high heart rate variability will lead to a certain set of

conclusions. The justification behind the choice of using RF models derives from the algorithms randomized construction of decision trees and averaging over predictions. By randomly creating a set of decision trees that capture prevalent value ranges, and further generalizing by averaging the predictions, it is possible to capture consistent patterns that emerge in the sampling window predating the diagnoses.

Parameter Name	$\mathbf{z}_{\sigma}$	$\mathbf{z}_{Max}$	$\mathbf{z}_{Wave}$	$\mathbf{z}_{WaveStats}$
Criterion	gini	gini	gini	gini
Min Samples per leaf	1	7	3	1
Min Sample Split	2	14	2	2
Min Weight Fraction	0	0.09871	0	0
Number of estimators	36	35	34	36
Max Depth	25	418	None	25
Input Vector size	108	108	36	4752

Table 6: RF model parameters.

The final hyperparameters for the RF models used are shown in Table 6. The exact description of every parameter named can be found in the documentation of [36]. Moreover, "None" values in Max Depth indicates to the algorithm that there is no restriction on the Depth of the trees created. In addition to the normal hyperparameters of the RF model, Table 6 also mentions the input vector size of each feature. Although only $\mathbf{z}_{Max}$ and $\mathbf{z}_{\sigma}$ have the same size, these were the dimensions which provided the best results during training and testing. $\mathbf{z}_{WaveStats}$ is a concatenation of the result of $\phi_{WaveStats}$ applied to all three waveforms, ECG, Respiration and SpO2. However, all other features are a concatenation of the resulting transformation that was applied to the respective feature mapping function used.

5.4.2 Convolutional Neural Nets

In order to complement the predictive experiments made with the RF model, a classical Convolutional Neural Network (CNN) deep learning model, LeNet5 [37], was used to evaluate the potential of the features of interest. The architecture was implemented the same way as it was done in [37], except that the implementation done for this work included one dimensional signals, with three channels (ECG, Respiration and SpO2 features). Figure 46 helps illustrates the architecture. For further details, please see an extensive description regarding the model in [37].

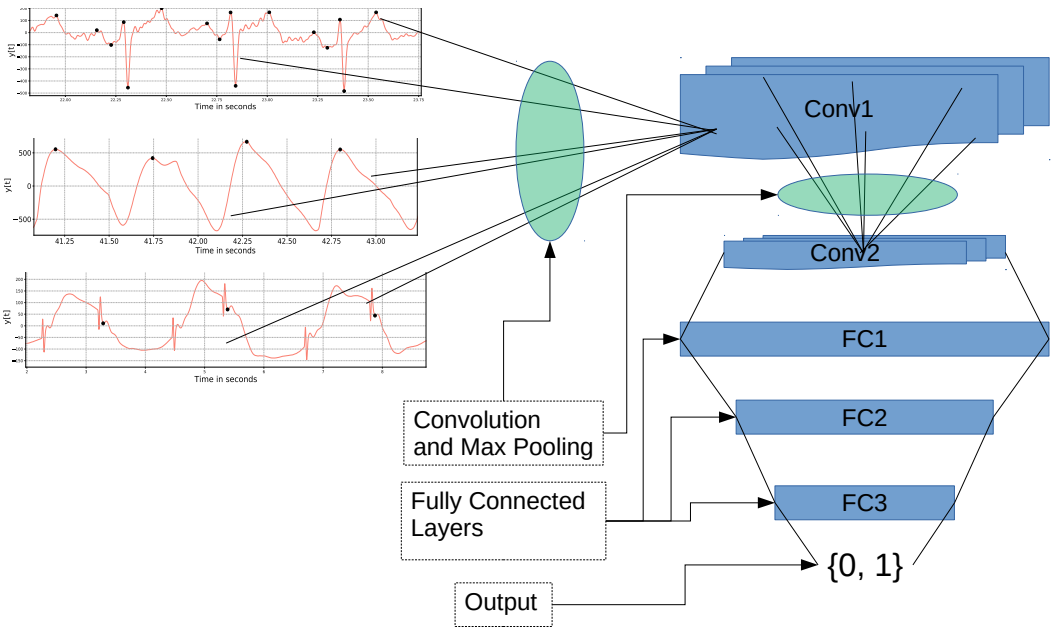


Figure 46: Implemented LeNet5 architecture.

Since the main goal was to evaluate the predictive potential of the features engineered, a very complex and state-of-the-art deep learning model was not desirable. Instead, the aim was to find a model which encapsulated proven performance and ease of implementation. LeNet5 provides this ease of implementation (and customization) along with a proven performance with images [37].

Parameter Name	Values
Optimizer	Adam
Optimizer Learning rate	0.001926
Weight Decay	No
Number of Mid Layer Channels	18
Batch Size	100
Dropout Probabilities	[0.18, 0.3, 0.43, 0.29 , 0.21]
Number of epochs	62
Number of estimators	36
Max Depth	25
Input Vector size	36 features per channel (3 channels)

Table 7: LeNet5 model parameters.

Table 7 provides the specific parameters used when training the model. Dropout probabilities is provided as an array, since the probability for every layer is given, from the first (in the left), to the last fully connected layer. Moreover, a significant difference between this model and the previous is the fact that the feature vectors

were provided in three channels (as opposed to the concatenation done in the RF model). Each channel represented a specific waveform. Finally, note that weight decaying was used. This was because dropout was enough to prevent overfitting and allow for generalization.

5.5 Results

In this section, the results concerning Sepsis prediction will be presented. The main goal was to evaluate the performance of the individual features given the CNN or RF model. For these purposes a five (5) fold cross validation was performed on the training data $\mathcal{X}$, while the best model was then tested against an unseen test set $\mathcal{T}$.

5.5.1 Random Forest

The RF model was implemented with the Scikit-Learn Python library [36]. It contains an implementation of the RF model, along with a set of tools to produce cross validation. In order to evaluate the potential of the individual features with the RF model, a few different experiments were made. The first experiment was a cross validation experiment with the training data $\mathcal{X}$. Namely, a five (5) fold cross validation was performed, since there was a limited number of data samples to train and validate against.

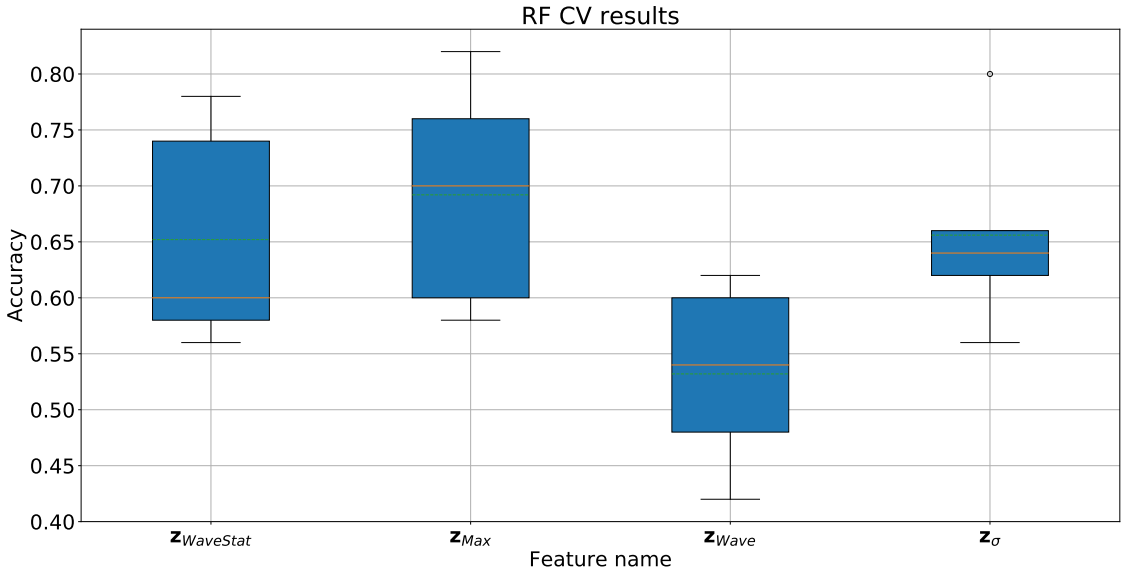
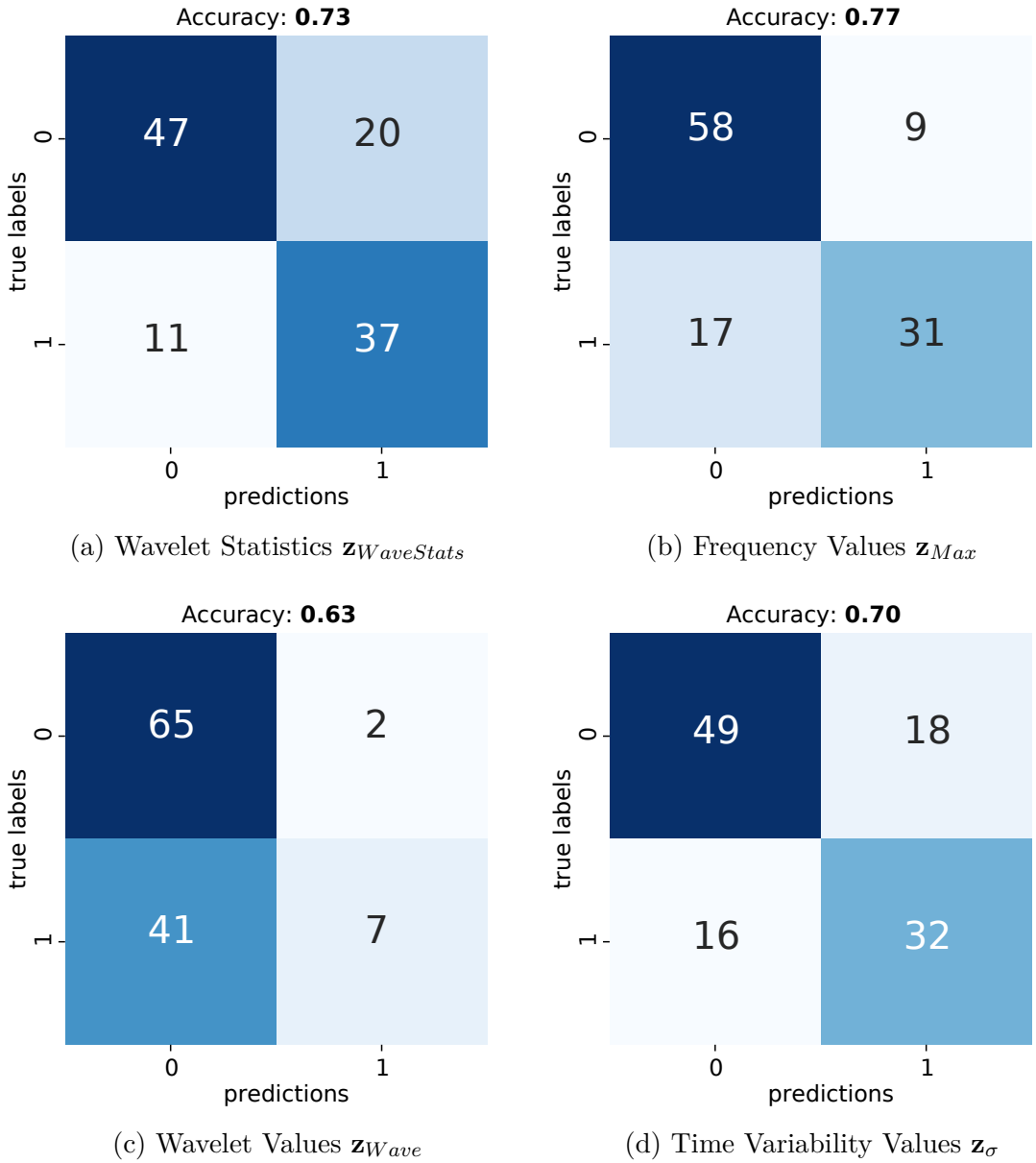


Figure 47: Five (5) fold cross validation results for RF model.

Figure 47 demonstrates the five fold cross validation results for the RF model. Cross validation results show that the best performing feature is the frequency related information $\mathbf{z}_{Max}$. Figure 48b further validates this result by showing the Confusion Matrix pertaining to the Test set $\mathcal{T}$ results.



(a) Wavelet Statistics $\mathbf{z}_{WaveStats}$

(b) Frequency Values $\mathbf{z}_{Max}$

(c) Wavelet Values $\mathbf{z}_{Wave}$

(d) Time Variability Values $\mathbf{z}_{\sigma}$

Figure 48: Confusion Matrix showing the different predictive results achieved for the Random Forest Model.

The worst performing feature, $\mathbf{z}_{Wave}$ achieved an accuracy of 63%, as shown in Figure 48c. The conclusion for this lower performance is that the wavelet component contain too much information, and perhaps the PCA distorts the information in the wavelet component. As Table 8 shows, $\mathbf{z}_{Wave}$ was the worst performing.

Feature Name	Class	Precision	Recall	f1-score	support
Wavelet Statistics	0	**0.81**	0.70	0.75	67
	1	0.65	**0.77**	0.70	48
Wavelet	0	0.61	**0.97**	0.75	67
	1	0.78	0.15	0.25	48
Frequency Components	0	0.77	0.87	**0.82**	67
	1	**0.78**	0.65	0.70	48
Time Variability	0	0.75	0.73	0.74	67
	1	0.64	0.67	0.65	48

Table 8: Table demonstrating the classification results for Random Forest

Finally, Figure 48d and Figure 48a shows the test results for $\mathbf{z}_\sigma$ and $\mathbf{z}_{WaveStats}$ respectively. Both features have an average performance. Overall, these results demonstrate not only that the features contain predictive value, but the features paired with a fine tuned RF model can provide a respectable predictive performance.

5.5.2 LeNet5

LeNet5 is a CNN deep learning model first proposed in [37]. Over the years, it has been tested in several applications, including time-series and image classification. In order to implement the deep learning model LeNet5, the Python Pytorch Machine Learning framework [38] was used. It's a framework that provides an extensive library Machine Learning tools that are GPU friendly.

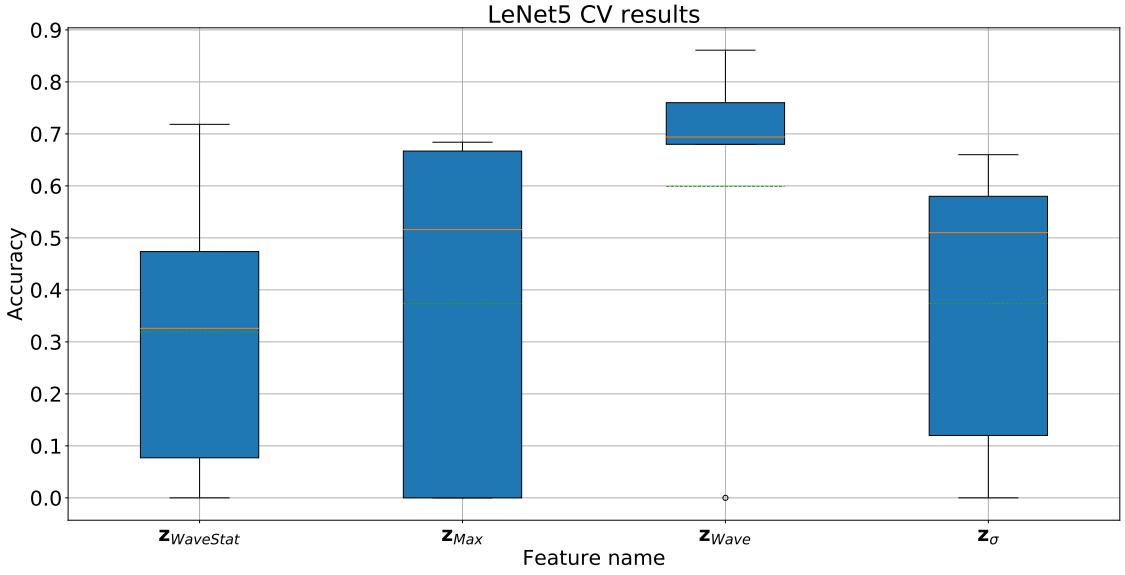
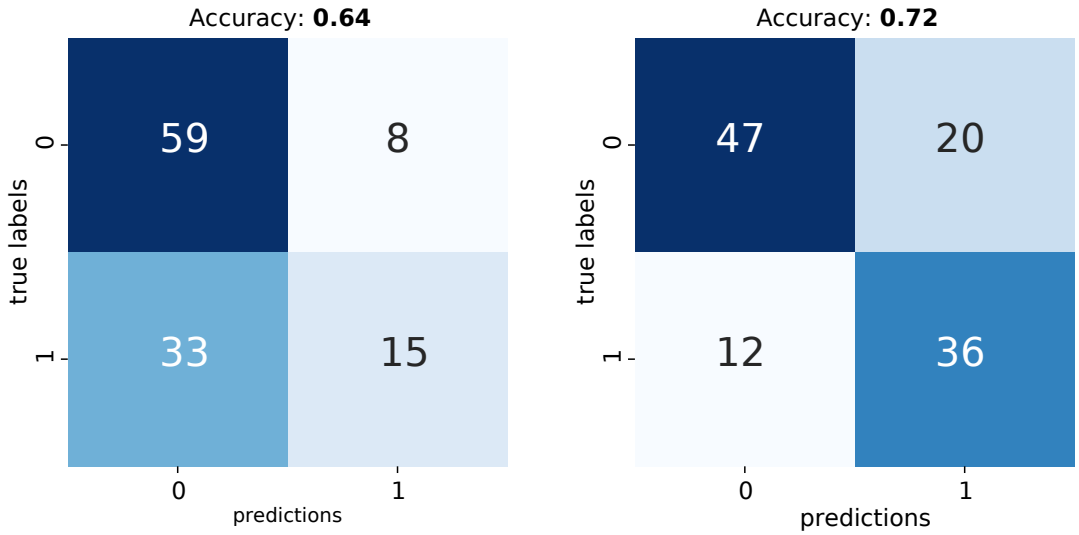


Figure 49: Five (5) fold cross validation results for LeNet5 model.

In order to evaluate the potential of the individual features with this CNN model, a five (5) fold cross validation test was performed. Figure 49 shows the results of

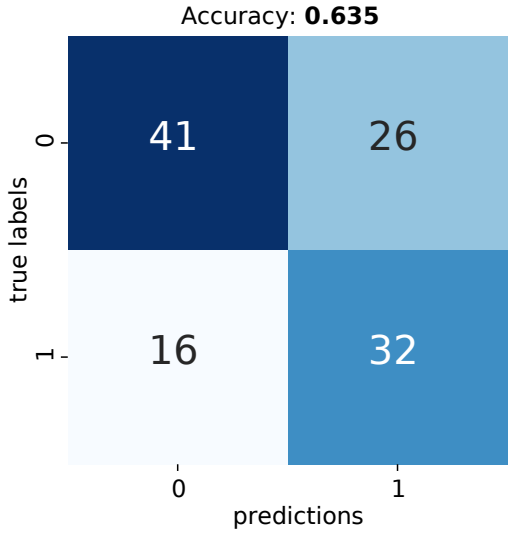
the cross validation procedure on the training data $\mathcal{X}$. It's possible to note that the variance is much larger than the one found with the RF model. This may be justified by the general necessity for deep learning models to use more data.

Now, in addition to the cross validation experiment, the best performing models were also tested against the test set $\mathcal{T}$. The best performing features on the test set for this model was $\mathbf{z}_{\sigma}$ (Figure 50d), followed by $\mathbf{z}_{Max}$ (Figure 50b).

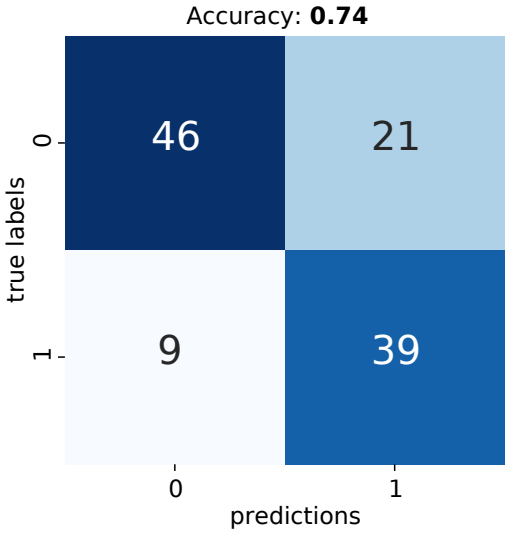


(a) Wavelet Statistics $\mathbf{z}_{WaveStats}$

(b) Frequency Values $\mathbf{z}_{Max}$



(c) Wavelet Values $\mathbf{z}_{Wave}$



(d) Time Variability Values $\mathbf{z}_{\sigma}$

Figure 50: Confusion Matrix showing the different predictive results achieved for the LeNet5 Model.

On the other end, $\mathbf{z}_{WaveStats}$ (Figure 50a) and $\mathbf{z}_{Wave}$ (Figure 50c) show a much lower performance, just above 60%. The low performance on the test set $\mathcal{T}$ may

indicate that the amount of training data for these particular features were not sufficient, given the CNN deep learning model.

Feature Name	Class	Precision	Recall	f1-score	support
Wavelet Statistics	0	0.65	0.69	0.67	67
	1	0.52	0.48	0.50	48
Wavelet	0	0.68	0.67	0.68	67
	1	0.55	0.56	0.56	48
Frequency Components	0	0.78	0.73	0.75	67
	1	0.65	0.71	0.63	48
Time Variability	0	0.84	0.69	0.75	67
	1	0.65	0.81	0.72	48

Table 9: Table demonstrating the classification results for LeNet5

Finally, Table 9 shows more specific information about the performance of the selected features along with the LeNet5 model. Overall, the conclusions drawn from these results indicate that in order to use a deep learning model to learn from the features produced, a much larger sample size for training is needed. Moreover, although features $\mathbf{z}_{Wave}$ and $\mathbf{z}_{WaveStats}$ did not perform well with this CNN model, $\mathbf{z}_{Max}$ and $\mathbf{z}_{\sigma}$ indicates predictive potential even with the low amount of training data.

6 Conclusions and Future Work

When the study begun, the main goal was to help medical patients in ICU by foreseeing the deterioration in their health status. This was thought to be possible by taking advantage of the clinical data available for any given patient during his/her hospital stay.

As the study progressed, data was further specified to be ECG, Respiration and Spo2 biosignals as these were the most prevalent sources of information. In addition, it was also found that ICUs are a good place to retrieve these signals, since patients are generally still and have better monitoring devices. Finally, a condition that is often found in ICUs, Sepsis, provided an opportunity to focus on a specific target that causes patients health to deteriorate in a rapid rate, eventually leading to organ failure.

Throughout the initial stages of this research, the main activities were to explore and understand the data surrounding this problem. The eventual route chosen, to create features that closely match clinical understanding or may be clinically explainable, showed to be very challenging. An extensive data exploration, signal processing and software engineering exercise was undertaken so as to reach a stage where features such as $\mathbf{z}_{Max}$ and $\mathbf{z}_{\sigma}$ were available to be used in a Machine Learning model.

The features whose predictive evaluation are presented in section 5 illustrate a set of clinically interpretable features. Although $\mathbf{z}_{Max}$ is a view into the frequency domain, it stills holds logic to clinicians as they are able to identify the different frequency components that are expected in normal heart functioning. A feature that is not as easy to interpret is $\mathbf{z}_{WaveStats}$, a statistical decomposition of $\mathbf{z}_{Wave}$. The latter, containing time-frequency localization proposes a broken down reconstruction of the original information.

Moving towards the latter part of the feature engineering and predictive modeling, the goal then became to evaluate the value and predictive potential of the subset of features chosen. As a result, complex models were not appropriate for the task. On the contrary, models which would be able to encapsulate in one way or another the information provided by the features, while maintaining simplicity and proven performance, would be the more suitable choices. Consequently, RF and LeNet5 were chosen to be used to evaluate the potential of the features. Two models that are well established and have straight-forward implementations.

In spite of the small training dataset, the final results proposed that the features engineered have predictive value, with $\mathbf{z}_{Wave}$ reaching predictive accuracy of 86% in one of the cross validation tests. Altogether, the results achieved support the higher goal of providing information that can be interpretable, while maintaining predictive value.

Now, although the results are promising, they are not conclusive or ideal. The highest accuracy reached in a test set was 77%. This implies that future work should explore models that have been proven to provide higher performance than LeNet5, such as very deep CNNs [39] or RNNs [22]. Finally, another beneficial step would be to evaluate the features which were not included in the predictive modeling study, so

to gain insight into their predictive performance.

Altogether, a significant amount of knowledge was gained with this research work. From software engineering for Big Data to signal processing and data science. The results achieved demonstrate that it is possible to not only engineer features that are easier to be interpreted by clinicians, but also predict Sepsis hours in advance.

References

[1] S. Alliance, "Sepsis fact sheet." https://www.sepsis.org/get-involved/fundraising-toolkit/attachment/sepsis-fact-sheet-2018, 2018.

[2] A. Oberholzer, C. Oberholzer, and L. L. Moldawer, "Sepsis syndromes: understanding the role of innate and acquired immunity.," *Shock (Augusta, Ga.)*, vol. 16, no. 2, pp. 83–96, 2001.

[3] S. S. Campaign, R. Dellinger, M. Levy, A. Rhodes, D. Annane, H. Gerlach, S. Opal, J. Sevransky, C. Sprung, I. Douglas, *et al.*, "International guidelines for management of severe sepsis and septic shock: 2012," *Critical Care Med*, vol. 41, no. 2, pp. 580–637, 2013.

[4] R. C. Deo, "Machine learning in medicine," *Circulation*, vol. 132, no. 20, pp. 1920–1930, 2015.

[5] D. Barber, *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[6] P. Garrett and J. Seidman, "Emr vs ehr–what is the difference," *HealthITBuzz, January*, 2011.

[7] G. Healthcare, "Ge healthcare icu dataset," 2018.

[8] G. B. Moody and R. G. Mark, "The impact of the mit-bih arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

[9] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, p. 160035, 2016.

[10] L. Sörnmo and P. Laguna, *Bioelectrical signal processing in cardiac and neurological applications*, vol. 8. Academic Press, 2005.

[11] Wikipedia, "Sinoatrial node." https://en.wikipedia.org/wiki/Sinoatrial_node, 2019.

[12] Wikipedia, "Electrocardiography." https://en.wikipedia.org/wiki/Electrocardiography, 2019.

[13] J. T. Moyle, *Pulse Oximetry*. BMJ Books, 2002.

[14] Bitesize, "Transport systems- animals." https://www.bbc.com/bitesize/guides/zc8pqhv/revision/2, 2019.

[15] Wikipedia, "Pulse oximetry." https://en.wikipedia.org/wiki/Pulse_oximetry, 2019.

[16] P. Berghuis, N. Cohen, M. Decker, and A. Gettinger, *Respiration*, vol. Biophysical Measurement Series. Space Labs, Dec. 25 2007.

[17] C. Redmond, "Trans-thoracic impedance measurements in patient monitoring," *EDN Network*, 2013.

[18] C. Horstmann, *Object-oriented design and patterns*. John Wiley & Sons, 2009.

[19] J. E. Hopcroft, *Introduction to automata theory, languages, and computation*. Pearson Education India, 2008.

[20] S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, *Algorithms*. McGraw-Hill Higher Education, 2008.

[21] C. Saritha, V. Sukanya, and Y. N. Murthy, "Ecg signal analysis using wavelet transforms," *Bulg. J. Phys*, vol. 35, no. 1, pp. 68–77, 2008.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[23] A. Gacek and W. Pedrycz, *ECG signal processing, classification and interpretation: a comprehensive framework of computational intelligence*. Springer Science & Business Media, 2011.

[24] R. W. Schafer and A. V. Oppenheim, *Discrete-time signal processing*. Prentice Hall Englewood Cliffs, NJ, 1989.

[25] J. Gomes and L. Velho, *From fourier analysis to wavelets*, vol. 3. Springer, 2015.

[26] J. Pan and W. J. Tompkins, "A real-time qrs detection algorithm," *IEEE Trans. Biomed. Eng*, vol. 32, no. 3, pp. 230–236, 1985.

[27] E. Jones, T. Oliphant, and P. Peterson, "Scipy: Open source scientific tools for python." http://www.scipy.com, 2014.

[28] C. S. Burrus and R. A. Gopinath, *Introduction to wavelets and wavelet transforms*. Prentice Hall Upper Saddle River, NJ, 1998.

[29] M. Vetterli and C. Herley, "Wavelets and filter banks: Theory and design," *IEEE transactions on signal processing*, vol. 40, no. 9, pp. 2207–2232, 1992.

[30] G. Lee, F. Wasilewski, R. Gommers, K. Wohlfahrt, A. O'Leary, and H. Nahrstaedt, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, 2006.

[31] K. Pearson, "X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 50, no. 302, pp. 157–175, 1900.

[32] J. T. Behrens, "Principles and procedures of exploratory data analysis.," *Psychological Methods*, vol. 2, no. 2, p. 131, 1997.

[33] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[34] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, vol. 1. Springer series in statistics New York, 2001.

[35] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[37] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[38] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," *OpenReview*, 2017.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

A Algorithms Run-Times

Below is the output of the benchmark program created to time the feature extraction system. This is true for a group of waveforms containing ECG, SpO2 and Respiration signals. The higher level program that performs this task creates a massive pool of parallel nodes that compute each waveform separately.

As an example, for a cohort of 150 patients (approximately the cohort size used in this work), there will be 450 nodes created for computation.

Sample window	Computation Time
10 minutes	1.5s
60 minutes	4.89s
300 minutes	15.62s

Table A1: Computation times for features in Equation 10 and Equation 11.