

**BABEȘ-BOLYAI UNIVERSITY CLUJ-NAPOCA**  
**FACULTY OF MATHEMATICS AND COMPUTER**  
**SCIENCE**  
**SPECIALIZATION [Secție]**

**DIPLOMA THESIS**

**[Titlu lucrare]**

**Supervisor**  
**[Grad, titlu si nume coordonator]**

*Author*  
*[Nume student]*

2024



---

## ABSTRACT

---

Abstract: un rezumat în limba engleză cu prezentarea, pe scurt, a conținutului pe capitole, punând accent pe contribuțiile proprii și originalitate

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation and Goals . . . . .	1
1.2	Thesis Structure . . . . .	1
1.3	Original Contributions . . . . .	1
<b>2</b>	<b>Theoretical background</b>	<b>2</b>
2.1	Cardiovascular Diseases (CVD's) . . . . .	2
2.1.1	What are cardiovascular diseases? . . . . .	2
2.1.2	CVD's impact over population . . . . .	3
2.1.3	Current detection scheme for CVD's . . . . .	5
2.2	AI and Machine Learning . . . . .	6
2.2.1	Logistic Regression . . . . .	7
2.2.2	Support Vector Machine . . . . .	8
2.2.3	Tree Based Models . . . . .	11
<b>3</b>	<b>State of the Art results</b>	<b>15</b>
3.1	Datasets . . . . .	15
3.1.1	Heart Disease Dataset . . . . .	15
3.1.2	Heart Failure Prediction Dataset . . . . .	16
3.1.3	Framingham Dataset . . . . .	17
3.2	Related Work . . . . .	17
3.2.1	Heart Disease Dataset Research . . . . .	17
3.2.2	Heart Failure Prediction Dataset Research . . . . .	18
3.2.3	Framingham Dataset Dataset Research . . . . .	19
3.2.4	Combined Datasets Research . . . . .	21
3.3	Conclusions . . . . .	22
<b>4</b>	<b>Study case - Heart Disease Prediction</b>	<b>23</b>
4.1	Approach and Methodology . . . . .	23
4.2	Prediction of developing a CVD . . . . .	23
4.2.1	Data Preprocessing . . . . .	24

4.2.2	Experiments . . . . .	25
4.3	Comparison between results . . . . .	26
4.3.1	Comparison between best models for each dataset . . . . .	28
4.3.2	State of the art results compared to current results . . . . .	30
<b>5</b>	<b>Heart Disease Prediction Portal</b>	<b>32</b>
5.1	Problem Statement and App features . . . . .	32
5.2	Design and Architecture . . . . .	33
5.2.1	Front-end . . . . .	33
5.2.2	Back-end . . . . .	34
5.3	User Guide . . . . .	37
<b>6</b>	<b>Conclusions and future improvements</b>	<b>40</b>
	<b>Bibliography</b>	<b>41</b>

# **Chapter 1**

## **Introduction**

### **1.1 Motivation and Goals**

### **1.2 Thesis Structure**

### **1.3 Original Contributions**

Introducere: obiectivele lucrării și descrierea succintă a capitolelor, prezentarea temei, prezentarea contribuției proprii, respectiv a rezultatelor originale și menționarea (dacă este cazul) a sesiunii de comunicări unde a fost prezentată sau a revistei unde a fost publicată.

# Chapter 2

## Theoretical background

### 2.1 Cardiovascular Diseases (CVD's)

The domain of this research paper is the medical one and how to use the power of machine learning to ease the detection mechanisms of detecting Cardiovascular Diseases. In the first part of this chapter it will be introduced the medical field and more precisely the field of CVD's and what they mean to population. After that, the second part of this chapter will present the machine learning techniques used to upgrade the prevention rate of the CVD's, in the detriment of the classic medical approach presented before.

#### 2.1.1 What are cardiovascular diseases?

Heart failure is a complex and potentially life-threatening condition that significantly burdens healthcare systems worldwide. It is a pathophysiologic condition in which the heart's inability to pump blood at a rate sufficient to meet the needs of the body's metabolizing tissues results from faulty cardiac function[5]. Cardiovascular diseases (CVDs) are a group of disorders affecting the heart and blood vessels. They are the leading cause of death globally, claiming millions of lives each year. The primary forms of CVD include coronary artery disease (leading to heart attacks), cerebrovascular disease (leading to strokes), raised blood pressure (hypertension), peripheral artery disease, rheumatic heart disease, congenital heart disease, and heart failure.

At the cellular level, CVDs are often the result of atherosclerosis, a condition characterized by the buildup of fatty deposits inside the arteries. This buildup can reduce or block blood flow to the heart, leading to serious complications such as heart attack or stroke. The heart muscle requires a constant supply of oxygen-rich blood; a reduction in this supply can damage the heart muscle, affecting its ability to function effectively. Unfortunately, often times there are no symptoms of the

underlying disease of the blood vessels, so a heart attack or a stroke could be the first sign for the illness, which can be concernedly late.

### **2.1.2 CVD's impact over population**

Globally, CVDs are responsible for more deaths annually than any other cause. According to the "World Health Organization", an estimated 17.9 million lives are lost each year due to cardiovascular diseases, accounting for 32% of all global deaths[13]. These deaths are primarily due to two major conditions: heart attacks (myocardial infarctions) and strokes, which are the acute manifestations of underlying vascular diseases, these go up to 85% for the causes of death from CVD.

The impact of cardiovascular diseases is not uniform across different regions or populations. Lower- and middle-income countries bear a disproportionate share of the CVD burden, largely due to the limited healthcare infrastructure available to prevent and treat such complex conditions. Additionally, the demographic shift towards older age groups in many parts of the world is likely to increase the prevalence of CVDs due to the association of these diseases with aging.

The economic burden of cardiovascular diseases on global economies is substantial and multifaceted. The direct costs include hospital care, medications, and medical procedures, while indirect costs arise from loss of productivity due to illness or death. According to estimates, CVDs cost governments billions of dollars each year. Beyond the economic costs, cardiovascular diseases have a profound impact on the quality of life of individuals and their families. Chronic heart conditions can lead to significant limitations in daily activities, reducing mobility and independence. Patients often experience physical pain and discomfort, as well as psychological effects such as depression and anxiety, which are common in those diagnosed with chronic illnesses. Cardiovascular diseases place a significant strain on healthcare systems around the world. They are among the leading causes of hospital admissions and require substantial healthcare resources for management and treatment.

In the figure 2.1 is the leading causes of death worldwide, from the year 2019. It is a well known fact that the cardiovascular heart diseases are the number one cause of death worldwide, this graphic reinforcing this, the number of people who died from CVDs in this year is 18.56 million.

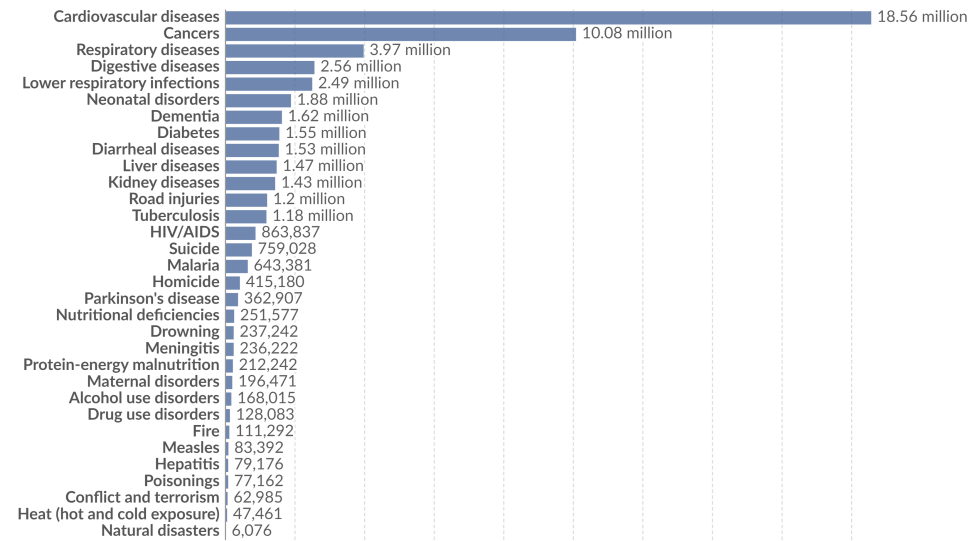
This is no exception for Romania; according to "Our World in Data", we can see that from the year 1990, up to 2019, the CVD remained the number one cause of death in Romania, the number of deaths per year fluctuating, but we can see that it is increasing, from 1990 with 136.528 deaths to 2019 with 150.427 deaths. This is the chart 2.2 that shows the leading causes of death in Romania from 1990 to 2019.



## Causes of death, World, 2019

Our World  
in Data

The estimated annual number of deaths from each cause. Estimates come with wide uncertainties, especially for countries with poor vital registration<sup>1</sup>.



Data source: IHME, Global Burden of Disease (2019)

OurWorldInData.org/causes-of-death | CC BY

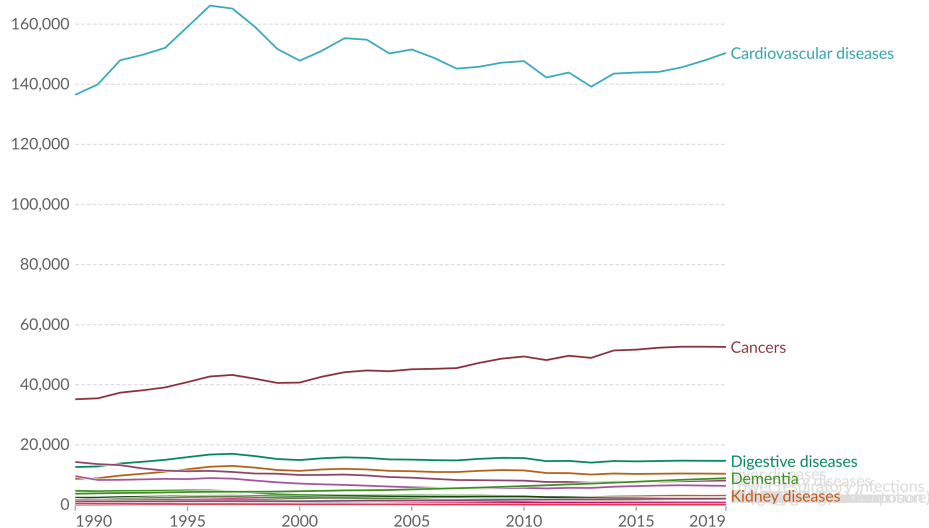
1. **Civil Registration and Vital Statistics system:** A Civil Registration and Vital Statistics system (CRVS) is an administrative system in a country that manages information on births, marriages, deaths and divorces. It generates and stores 'vital records' and legal documents such as birth certificates and death certificates. You can read more about how deaths are registered around the world in our article: [How are causes of death registered around the world?](#)

Figure 2.1: Causes of death worldwide in 2019

## Causes of death, Romania, 1990 to 2019

Our World  
in Data

The estimated annual number of deaths from each cause. Estimates come with wide uncertainties, especially for countries with poor vital registration<sup>1</sup>.



Data source: IHME, Global Burden of Disease (2019)

OurWorldInData.org/causes-of-death | CC BY

1. **Civil Registration and Vital Statistics system:** A Civil Registration and Vital Statistics system (CRVS) is an administrative system in a country that manages information on births, marriages, deaths and divorces. It generates and stores 'vital records' and legal documents such as birth certificates and death certificates. You can read more about how deaths are registered around the world in our article: [How are causes of death registered around the world?](#)

Figure 2.2: Causes of death in Romania from 1990 to 2019

### 2.1.3 Current detection scheme for CVD's

Cardiovascular diseases (CVDs) are the leading cause of mortality globally, necessitating efficient and effective detection techniques. Modern detection methods integrate clinical assessments with advanced diagnostic tools to accurately identify various types of cardiovascular conditions. This subsection explores the primary modalities used in the detection of cardiovascular diseases, including physical examinations, imaging techniques, and biochemical markers.

The initial approach in detecting cardiovascular disease involves a comprehensive physical examination and detailed patient history. Physicians assess risk factors such as age, family history of CVD, lifestyle factors (like smoking and diet), and comorbid conditions such as diabetes and hypertension. Physical signs such as pulse rate, blood pressure, and sounds of the heart and lungs are also evaluated. This initial screening is crucial for identifying individuals at risk and deciding on further diagnostic testing.

Advanced imaging techniques such as Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) scans are increasingly used for a more detailed assessment of cardiovascular health. Cardiac MRI provides high-resolution images of the heart's anatomy and can evaluate both its structure and function without the need for radiation. CT scans, particularly coronary CT angiography, are effective in visualizing the coronary arteries and identifying blockages or atherosclerosis.

A comprehensive study from the New England Journal of Medicine regarding the impact of modifiable risk factors on cardiovascular disease (CVD) and mortality?? analyzed individual-level data from 1,518,028 participants across 112 cohort studies in 34 countries, aiming to understand how five key risk factors—body mass index, systolic blood pressure, non-HDL cholesterol, smoking, and diabetes—contribute to cardiovascular disease incidence and mortality.

Modifiable risk factors significantly contribute to the incidence of cardiovascular disease and mortality. The study found that 57.2% of cardiovascular disease cases among women and 52.6% among men could be attributed to these risk factors, with similar impacts on all-cause mortality (22.2% in women and 19.1% in men). The prevalence of these risk factors and their effects on health outcomes varied significantly across different regions and between genders.

This thorough investigation provides a robust framework for understanding the impact of lifestyle and biological risk factors on cardiovascular health on a global scale, illustrating the crucial role of prevention and management strategies in public health. It also shows the impact these factors have on developing a CVD and how they can be used to train a model to detect a person that is susceptible to developing CVDs, so it can be prevented from from early stages.

Cardiovascular disease (CVD) is influenced by several modifiable risk factors, including smoking, high blood pressure, high cholesterol, diabetes, obesity, physical inactivity, and poor diet. Smoking accelerates atherosclerosis and increases blood pressure, while diets high in saturated and trans fats can elevate harmful cholesterol levels, leading to heart disease. Physical inactivity and obesity further exacerbate the risk by affecting blood pressure, cholesterol levels, and body glucose regulation, which are tightly linked with heart health. Managing these risk factors through lifestyle changes and medical interventions can significantly reduce the incidence of CVD.

There are also non-modifiable risk factors that contribute to the development of CVD, including age, gender, genetic predisposition, and ethnic background. Men are generally at higher risk than women until advanced age, when the risk becomes comparable. Family history of CVD also plays a crucial role, especially if close relatives were diagnosed at a young age. Additionally, emerging risks like sleep apnea, chronic stress, excessive alcohol consumption, and inflammatory diseases like lupus or rheumatoid arthritis are increasingly recognized for their impact on heart health. Understanding both modifiable and non-modifiable factors is essential for comprehensive cardiovascular risk assessment and management.

## **2.2 AI and Machine Learning**

In the theoretical chapter of this thesis, we dive into the fundamental concepts of Artificial Intelligence (AI) and Machine Learning (ML) algorithms, which are at the core of contemporary computational techniques used in predicting heart disease. AI encompasses a wide range of technologies capable of mimicking human intelligence and executing tasks that typically require human-like cognition, including learning, reasoning, and problem-solving. Machine Learning, a critical subset of AI, employs statistical models and algorithms to enable computers to learn from historical data, thereby enhancing their ability to make predictions or decisions. This chapter will explore various machine learning models, focusing on their principles, capabilities, and the specific algorithms that have shown efficacy in the field of medical diagnostics, particularly in predicting cardiovascular abnormalities. Through this exploration, we aim to establish a solid theoretical foundation that supports the practical application of these technologies in predicting heart disease, highlighting their significance and potential in advancing medical research and practice.

### 2.2.1 Logistic Regression

Logistic regression is a supervised learning algorithm employed for classification purposes, aiming to predict the likelihood of an instance being classified into a specific category. Logistic regression is used when the dependent variable (target) is categorical. For binary classification, this target is typically coded as 0 or 1, representing two classes (e.g., "No" or "Yes"), in our case if the person will develop a heart disease, or not.

The logistic regression model calculates probabilities using the logistic( sigmoid) function 2.3, which is an S-shaped curve that maps any real-valued number into the range (0, 1): 2.1

$$Probability = \frac{1}{1 + e^{-(b_0 + b_1 x)}} \quad (2.1)$$

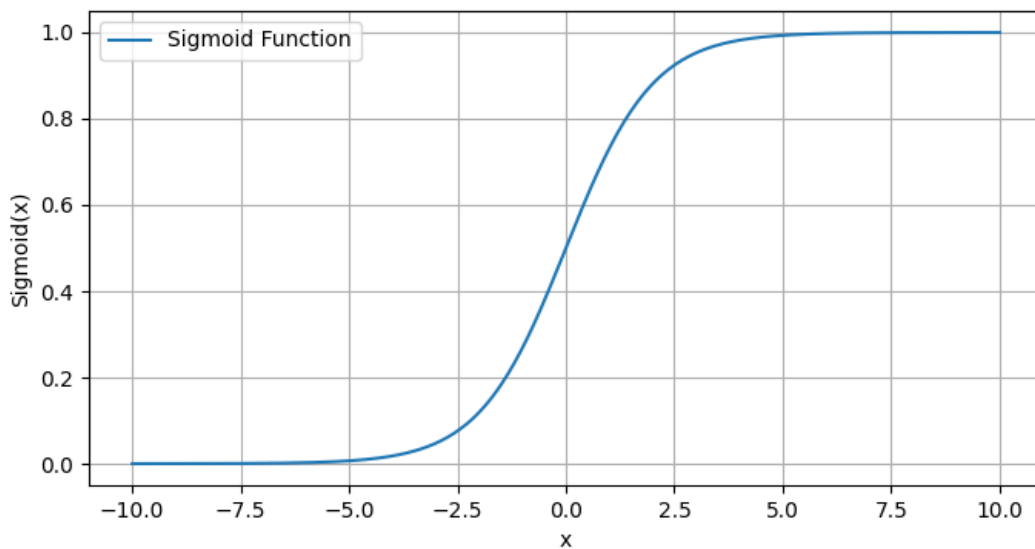


Figure 2.3: Plot for the Sigmoid function

The coefficients of logistic regression are estimated using maximum likelihood estimation (MLE), which seeks to find the parameter values that maximize the likelihood of the observed sample. In logistic regression, each coefficient tells you how the likelihood of the outcome changes with a one-unit increase in its corresponding predictor variable, when expressed in terms of log odds. Log odds describe the logarithm of the odds of the outcome occurring. For instance, if a coefficient is positive, a one-unit increase in the predictor variable increases the log odds of the outcome, making the outcome more likely. Conversely, a negative coefficient means a one-unit increase in the predictor decreases the log odds, making the outcome less likely. This relationship is a direct, quantifiable link between changes in predictor

variables and changes in the likelihood of the outcome.

Logistic regression is highly regarded for several advantages that make it a popular choice for binary classification problems. Firstly, its simplicity and ease of interpretation stand out, as the relationship between the predictor variables and the probability of the outcome is presented in a clear and understandable way. This allows for straightforward explanations of results, which is particularly valuable in business and medical fields where decisions need clear justification. Additionally, logistic regression is computationally efficient, making it suitable for scenarios with limited computational resources. It also provides probabilities for the outcomes, offering more information than just a classification, which can be crucial for risk assessment and decision-making processes where the likelihood of outcomes needs to be considered.

Logistic regression does have some drawbacks, especially when used for predicting outcomes like cardiovascular diseases. One significant limitation is that it assumes a straightforward, linear relationship between factors influencing heart health (like cholesterol levels, blood pressure, etc.) and the likelihood of developing cardiovascular disease. This assumption might not hold if the real-world relationships are more complex or curved, which means logistic regression could miss important patterns in the data. Additionally, logistic regression might not be very reliable if the data set is small or if there are many factors being considered at once, which can lead to a model that fits the training data too closely but doesn't perform well in predicting new cases. The model also requires that each patient's data is unaffected by any other patient's data, which might not always be the case, particularly in scenarios where environmental or genetic factors play a role across populations.

### **2.2.2 Support Vector Machine**

Support Vector Machines (SVM) are a set of supervised learning methods used for classification. Support Vector Machines work by finding a hyperplane in an  $N$ -dimensional space ( $N$  — the number of features) that distinctly classifies the data points. To separate two classes of data points, there are many possible hyperplanes that could be chosen. The goal is to find a plane that has the maximum margin, i.e., the maximum distance between data points of both classes. The SVM algorithm builds a model that assigns new data points to one category or another, making it a non-probabilistic binary linear classifier. It focuses on the data points that are hardest to classify, known as support vectors, and the hyperplane is influenced most by these points.

Linear SVM is the simplest form of SVM used when the data is linearly separable,

meaning the data can be divided by a straight line into two classes. The algorithm finds the line (in 2D) or hyperplane (in higher dimensions) that separates the classes with the maximum margin 2.4.

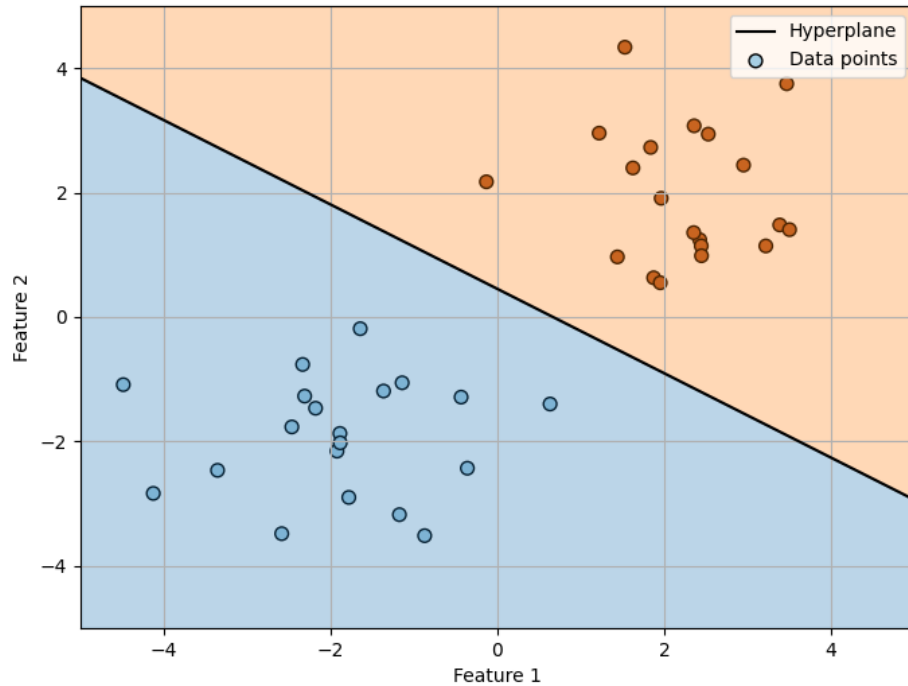


Figure 2.4: Plot for linear SVM

When data is not linearly separable, SVM uses a kernel trick to transform the input space into a higher-dimensional space where a hyperplane can be used to separate the classes. Common kernels include polynomial, radial basis function (RBF), and sigmoid 2.5.

- **RBF Kernel:** The Radial Basis Function (RBF) kernel, also known as the Gaussian kernel, transforms data into a higher-dimensional space using the squared Euclidean distance between data points, making it exceptionally effective for handling non-linear relationships in data.
- **Polynomial Kernel:** The polynomial kernel, which raises the dot product of two vectors to a specified power, is adept at capturing interactions between features up to the degree of the polynomial, providing a flexible and controlled way of increasing model complexity.
- **Sigmoid Kernel:** The sigmoid kernel, inspired by the neural activation function, transforms the data by applying a sigmoid function on the dot product of

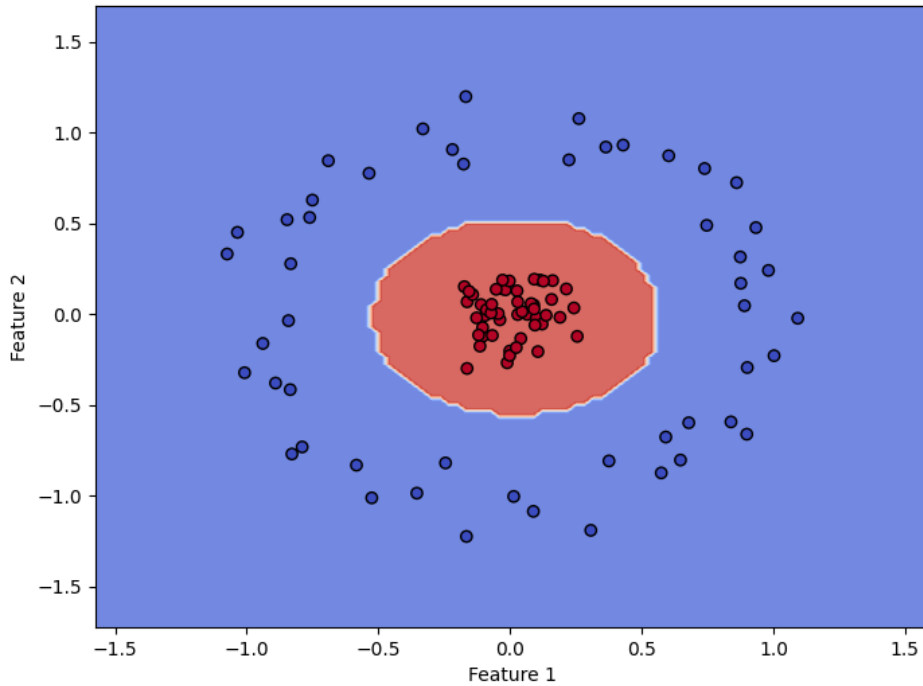


Figure 2.5: Plot for RBF-kernel SVM

feature vectors, and is used in neural networks to introduce non-linear properties into SVMs.

Support Vector Machines (SVMs) offer a number of significant advantages that make them a preferred choice for many machine learning tasks. One of the primary strengths of SVMs is their ability to handle high-dimensional data effectively, making them ideal for applications involving complex datasets such as gene expression, text, and image classification. The kernel trick is a key feature that allows SVMs to operate in a transformed feature space without the need to compute the coordinates of the data in that space explicitly, which can save computational resources. Additionally, SVMs are known for their robustness against overfitting, especially in high-dimensional spaces. This robustness is partly due to their decision function being based only on a subset of the training data, the support vectors, which means they are less influenced by noise in the data.

However, SVMs also have several limitations that can impact their utility in certain scenarios. They are notably sensitive to the choice of the kernel parameter and the regularization parameter  $C$ , which can require careful tuning to achieve optimal performance. This model selection process often involves an extensive search, such as grid search, which can be computationally expensive and time-consuming. Furthermore, SVMs do not naturally provide probability estimates for predictions,

which are often desirable in real-world settings(e.g. in the medical field, where all the risks should be taken into consideration). Calculating these probabilities involves an additional layer of cross-validation and can significantly slow down the model's performance. Lastly, SVMs' performance degrades with the increase in the size of the data; they are not particularly efficient with very large datasets due to their high training time complexity and intensive memory requirements.

### 2.2.3 Tree Based Models

This subsection focuses on two tree based models that are usually used in this type of setting, because they manage to achieve very good results and the last model is a model close to the first two (Decision Tree and Random Forest), but a little unexplored to this domain.

#### A. Decision Tree

Decision Trees are a versatile machine learning method used for both classification and regression tasks. This model represents decisions and their possible consequences, including chance event outcomes, resource costs, and utility, in a tree-like structure of nodes and leaves. Decision Trees are built on simple decision-making concepts, which makes them easy to understand and interpret. A decision tree splits the data into branches at each node, making decisions based on the input features.

A decision tree consists of nodes that form the decision points, branches that represent the outcomes of a decision, and leaves that represent the final outcomes or classes. The root node at the top of the tree makes the first decision, which divides the data into smaller subsets. [15]

Building a decision tree involves selecting features and splitting the dataset recursively until certain criteria are met. This process is known as recursive partitioning. Decision Trees use metrics such as Gini impurity or entropy to determine which feature and which value of that feature to use at each split. These metrics help in choosing the splits that will most effectively separate the data into classes based on the target variable. The two most common splitting criteria are:

- Gini Impurity: A measure of how often a randomly chosen element would be incorrectly classified. It is defined as:

$$GiniIndex = \sum_i p_i(1 - p_i) \quad (2.2)$$



where  $p_i$  is the probability of class  $i$  and the interval of Gini is  $[0, 0.5]$ . For a two-class problem, the Gini impurity for a given node is expressed as:

$$p_1(1 - p_1) + p_2(1 - p_2) \quad (2.3)$$

It is straightforward to understand that in cases where the sample set is completely pure, one of the probabilities becomes 0, leading to the smallest possible Gini score. On the other hand, when  $p_1 = p_2 = 0.5$ , the Gini score reaches its maximum value, indicating the lowest level of purity at that node.[10]

- Entropy: A measure of the degree of disorder or randomness in a system, defined in the context of decision trees as:

$$Entropy = - \sum_i p_i \log_2(p_i) \quad (2.4)$$

where  $p_i$  is the probability of class  $i$ , and the possible values of entropy range from  $[0, 1]$ . For a two-class problem, the entropy is calculated as:

$$Entropy = -p \log_2(p) - (1 - p) \log_2(1 - p) \quad (2.5)$$

where  $p$  represents the proportion of one type of samples.

A perfectly uniform distribution of classes (50/50 split) signifies maximum uncertainty, reflected by an entropy value of 1. Conversely, a complete absence of diversity (all data points belonging to one class) indicates no uncertainty, resulting in an entropy of 0. The core principle behind decision tree construction is to progressively decrease entropy by strategically splitting the data. This process minimizes uncertainty and leads to a more organized classification structure. [10]

To prevent overfitting, trees are often pruned. Pruning reduces the size of decision trees by removing parts of the tree that do not provide power to classify instances. This involves removing branches that have little importance and can help improve the model's accuracy on unseen data.

## B. Random Forest

Random Forests are an ensemble learning technique used primarily for classification and regression tasks that operates by constructing a multitude of decision trees at training time. By outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees, Random Forests improve predictive accuracy and control over-fitting more ef-

fectively than a single decision tree.

A Random Forest is essentially a collection of decision trees, where each tree is slightly different from the others. This method leverages the power of multiple decision trees to produce a more accurate and stable prediction. Random Forests create individual trees through random selection of data points and features at each split point during the training process. This randomization ensures a diverse set of trees and results in a model that is less likely to overfit to the noise in the data.

### C. Extra Tree

The Extra Trees (or Extremely Randomized Trees) classifier extends the concept of Random Forests by introducing more randomness into the model construction. By making cuts in the decision trees at random rather than the best cut and using the whole learning sample to grow the trees, Extra Trees can often achieve higher levels of accuracy than traditional Random Forests, especially in complex datasets.

Extra Trees differs from traditional decision trees and Random Forests in the way splits are chosen. While Random Forests use bootstrapping and search for the optimal splits, Extra Trees use the entire original sample and make splits based on random thresholds for each feature rather than searching for the best possible thresholds. Building an Extra Trees model involves creating multiple trees using randomly selected features and thresholds for each node, which enhances model diversity and robustness.

The Extra Trees classifier is highly regarded for its ability to handle overfitting, making it a robust choice for binary classification tasks. It achieves this by using extreme randomization in selecting cut-points and considering all available data at each split, which enhances the diversity among the trees in the model. This diversity leads to a more generalized model that performs well on unseen data. Additionally, Extra Trees classifiers are computationally efficient because they eliminate the need for bootstrap sampling, allowing for faster training times compared to many other ensemble methods. These characteristics make Extra Trees particularly useful in binary classification scenarios involving complex, noisy, or high-dimensional data.

Despite its strengths, the Extra Trees model has some limitations in binary classification. The randomness introduced in the tree construction process, while beneficial for reducing overfitting, can lead to higher variance if the model is not adequately tuned, particularly with an insufficient number of trees. Furthermore, the model's performance is heavily dependent on proper parameter

tuning, especially the settings for the number of trees and the maximum number of features considered at each split. The lack of bootstrap sampling, though it reduces computation time, may sometimes decrease the diversity that could be beneficial for avoiding biases in the model's predictions.

# Chapter 3

## State of the Art results

The last chapter presented the background of both the medical field that this work focuses on and on the machine learning algorithms used in the application, this new chapter has the main objective of presenting the relating work done so far in this domain, starting with some of the datasets available, their attributes and differences.

### 3.1 Datasets

There are three major datasets found that contain the necessary attributes for predicting the risk of a cardiovascular disease. All of these 3 datasets were constructed with the help of specialised clinicians, the diagnosis being a real one, making the datasets a good training for the machine learning algorithms3.1.

Dataset Name	Nr Attributes	Nr Instances
Heart Disease[9]	14	1025
Heart Failure Prediction [6]	12	918
Framingham[4]	18	5209

Table 3.1: Available Datasets

#### 3.1.1 Heart Disease Dataset

The Heart Disease Dataset [9] is a dataset from 1988, that puts together four databases: Cleveland, Hungary, Switzerland, and Long Beach V. It is an open dataset, collected from Kaggle platform. As the source mentions there were originally 76 attributes, but only 14 of them(including the target attribute) are used in experimental applications. 3.2 The last column, the target column, presents the labeled value, if there is presence of heart disease in the patient (1), or if there is not (0). This is an extensive dataset that consists of 1026 entries, which makes it a very good candidate for

training models on it.

Table 3.2: Description of Features in the Heart Disease Dataset

Variable Name	Role	Type	Demographic	Description	Units	Missing Values
age	Feature	Integer	Age		years	no
sex	Feature	Categorical	Sex			no
cp	Feature	Categorical				no
trestbps	Feature	Integer		resting blood pressure (on admission to the hospital)	mm Hg	no
chol	Feature	Integer		serum cholestoral	mg/dl	no
fbs	Feature	Categorical		fasting blood sugar > 120 mg/dl		no
restecg	Feature	Categorical				no
thalach	Feature	Integer		maximum heart rate achieved		no
exang	Feature	Categorical		exercise induced angina		no
oldpeak	Feature	Integer		ST depression induced by exercise relative to rest		no
slope	Feature	Categorical				no
ca	Feature	Integer		number of major vessels (0-3) colored by flourosopy		yes
thal	Feature	Categorical				yes
num	Target	Integer	diagnosis of heart disease			no

### 3.1.2 Heart Failure Prediction Dataset

The second relevant dataset that should be considered is the Heart Failure Prediction Dataset.[6] Same as the previous data set, this is a dataset created by combining 5 datasets over 11 common features, which makes it the best dataset available so far for this research purpose, the datasets being collected from Cleveland, Hungary, Switzerland, Long Beach VA and Stalog.

### 3.1.3 Framingham Dataset

The Framingham Heart Study is acknowledged as a premier longitudinal cohort study. In the early 1900, up to 1942, the primary cause of mortality was infections. But in 1942, the penicillin was introduced, so the infections became treatable, therefore the leading role for mortality became the CVD's (cardiovascular disease). We would go as far as to say that one in 3 men in United States suffered from CVD before reaching the age of 60 years and that the life expectancy beyond the age of 45 years could not be increased[4].

It was reached the conclusion, that it should be a way to prevent the debut of CVD. In order to do that, the Framingham Heart Study was born. Its purpose was to choose people that were not suffering from CVD and analyze over time attributes like age, sex, and determine if they are factors that influence the development of CVD. The study took place in Framingham, Massachusetts, from where were selected a systematic sample of 2 of every 3 families in the town. The major aim of the study was to secure epidemiological data on CVD. The Framingham study was very successful with an offspring cohort initiated in 1971 (2,489 men and 2,646 women) and a third-generation cohort in 2001 with over 4,000 subjects.[7]

## 3.2 Related Work

Over the course of the years, the idea of predicting the chances of developing a CVD appeared more and more important. So there were a large number of people that decided to build models to predict the appearance of a heart disease, the fact that the databases presented above were classified by a person with a medical background, they stood as a base for these research.

### 3.2.1 Heart Disease Dataset Research

The importance of prediction heart diseases has not gone mistaken in the research community. First of all, in 2019, Fahd Saleh Alotaibi, wrote an article implementing a machine learning algorithm using the UCI dataset to predict the heart failure chances[1]. Furthermore, for the time being, he improved the accuracy score in predicting heart disease comparing to best classification models results at the time.

For this research, it was only used the data from the Cleveland set, having a record of 303 patients and selecting 14 attributes, same attributes that were presented above in the dataset overview. First of the all, there was data preprocessing, that included increasing the volumes using random number generation technique, and imputing missing values using k Nearest Neighbor (KNN). After that, there

were used 5 different models to predict the heart disease: Decision Tree, Naive Bayes, Random Forest, Logistic Regression, SVM. The target of this work was to improve the accuracy of the model therefore different amendment was done in the experiment design. For example, data expansion, 10-fold cross validation, execution of all model at the same time to understand the actual differences in the accuracy measurement. For experiment execution, the Rapid Miner tool was used in this study.

Comparing to the work done so far, these models improved the classification ability, the Naive Bayes classifier performance was the lowest and the performance of Decision Tree classifier computed highest, among all classifier. This study had the final results as: Decision Tree with the best score, improving with 11% as of last best known score, it being now 0.9319%.

The accuracy of the machine learning algorithms' was improved the following year (2020) [16]. This article also uses the UCI dataset, the used models were Random Forest, Logistic Regression, Naive Bayes and Voting Classifier and for feature selection was used the chi2 method. Even though the UCI dataset consist of 13 attributes, for this research purpose only 8 features were selected using the chi2 method. After getting the results for each model, there was used a Voting Classifier to aggregate the prediction of each classifier and predict the class that gets the most votes, it is based on probability.

From using the chi2 method, we can see that the feature with the biggest score is cholesterol, and the one with the lowest score was Fasting blood sugar. The dataset was divided into 75% data used for training and 25% data used for testing(from the Cleveland dataset). Out of the 5 algorithms (excluding for now the Voting Classifier), the Logistic Regression was the one with the best score(92.10), followed by Naive Bayes and Random Forest (0.8947), Decision Tree (0.8026) and the one that had the lowest accuracy was Support Vector Machine (SVM) (0.6973). However, after using the Voting Classifier, the accuracy rise up to 0.9473.

### **3.2.2 Heart Failure Prediction Dataset Research**

The following article was published in 2021 and it also compares 4 different classifiers, but this time using the Heart Failure Prediction Dataset[12]. This article is relevant because it shows a wide range of data statistics and histograms, the dataset was split 80% for training and 20% of data for testing. For the SVM classifier, there were made tests applying 3 kernel functions: linear, RBF, polynomial, but the one that performed the best: linear kernel was used as the SVM function in order to compare its results with the other classifiers. In terms of accuracy, the best performing classifier is a tie between Random Forest and Logistic Regression (0.88), SVM

with (0.85) and Naive Bayes (0.86). Another experiment was to see the importance of the feature selection: the five least important features were eliminated from the training data. It is showed that all features are important, the accuracy decreasing by removing even the least important features. This article has a little to no data preprocessing before using the classifiers.

The next article taken into consideration also uses the third database: Heart Failure Prediction Dataset, and also it is the most relevant, as it is the newest, from end of July 2023 [11]. For data preprocessing it is used a process known as encoding, that converts the categorical feature values into numerical representations. The dataset is splitted into training (736 instances, 2) and testing (184 instances, 2). To prevent overfitting in the stacking method, K-Fold Cross-validation was used, with k equals 4.

The Random Forest Classifier, Gaussian Naive Bayes, and Decision Tree models are employed as estimators, while K-Nearest Neighbors serve as the final estimator. The resulting data structure becomes (736, 4), with four columns representing the predicted results from the Random Forest Classifier model, Gaussian Naive Bayes model, Decision Tree model, and the 'HeartDisease' column of the primary training dataset. These four columns are then used to prepare the metamodel. To obtain the base models, the primary training dataset needs to be trained using the three fundamental models: RFC, GNB and DT. This process allowed to derive the model for predicting heart failure. Finally, the primary test data are passed through the final model to validate and assess the data. The metamodel achieved an accuracy of 0.87 for this dataset. This was compared to other machine learning models including DT, Bagging classifier, LGBM, Ridge classifier, SVR, SGDC, KNN, and GPC, however the metamodel of this report outperformed all the other models tested.

One of the strengths of the metamodel lies in its incorporation of multiple machine learning algorithms, namely Random Forest Classifier, Gaussian Naive Bayes, decision tree models, and k-Nearest Neighbor. By blending these algorithms, the metamodel leverages their individual strengths. This study provided a different approach about combining these classifiers on a different dataset than the ones presented before, the accuracy of the metamodel was a promising one, but we must note that the data preprocessing part was not as prominent as the ones from some of the previous articles.

### **3.2.3 Framingham Dataset Dataset Research**

In 2019, a different article[2] also tried to implement some machine learning models to predict heart diseases, but this time it used the Frammingham data set, described above in the data overview. It's purpose, beside the predicting heart diseases, was



to compare the results from using two different statistical software platforms: Rapid Miner and also R-Studio. This article did comparative research by using supervised classification algorithms: decision tree, random forest, support vector machines, and neural networks, in addition to traditional logistic regression. For comparison, there were prepared 3 models: model A: having original variables from the dataset, without modification. Model B included the same variables, but excluded all observations in which there was at least one missing variable. For the model C the missing values were replaced by the rest of non-missing values for that variable. The data was split into train and test: 80%, respectively 20%.

The algorithm that showed the highest AUC when performing the analysis using R-Studio was neural network applied to Model B (AUC=0.71). As for the Rapid Miner, the highest performing algorithm was SVM (Support Vector Machine), also applied to the Model B (AUC=0.75). Taking by comparison the used algorithms, we should mention that this article highlights that the decision tree could manage the variable with missing values, also prioritized of some variables, just as a clinician would also do; the most valuable feature was glycerin (as for the above mentioned article that used a different database where cholesterol was the most important feature [16]), moreover it was a fast algorithm, both in execution and implementation, the drawback was that the accuracy was low. Next one, the Random Forest algorithm provided better values than the Decision Tree, but it took more processing time. The next step were the Support Vector Machines that managed to improve even more the AUC, but with the cost of having an even longer processing time and code complexity. The Neural Network algorithm was the one that required more programming and processing time, but also the one which offered the best results. Lastly, the traditional logistic regression offered similar results to the previously described algorithms. An important aspect that led to this result is the normalization of the data and the data balancing.

Reaching the scope of this article, the result comparison between the results produced by the softwares used. Even though the processing time required for R-Studio was considerably higher than the Rapid Miner, the first one proved to be more flexible and powerful. On the other hand, Rapid Miner was simpler and more intuitive. The overall AUC values obtained were low close for 0.5 for decision tree and 0.6-0.73 for others.

In the year 2021, the goal of predicting cardiovascular disease kept going and started to expand, trying to compare the same model on different datasets to see the performance differences; moreover, this article focuses on feature selection[14]. The authors of this article found that there is a serious need of an integrated machine learning framework for CVD's where data balancing, optimum feature selection and improved classification should be achieved in a systematic way, presenting in this

article a MaLCaDD("Machine Learning based Cardiovascular Disease Diagnosis") framework . The dataset used for this study was the Framingham dataset.

It is also presented that before this article, the researchers focused on improving accuracy (like the first article presented) via traditional classification methods, and not by focussing on missing values and data imbalance. The first step is data preprocessing that starts with searching for possible outliers(a sample from the dataset witch deviates from the normal behavior of the dataset). After they are found, they are removed using the Boxplot, it works by forming a box structure and anything outside of it, it is considered an outlier. After that, is the step that checks for missing values and replacing them with the average value of that variable. The last step of data preprocessing is data inbalance, handled by using SMOTE ("Synthetic Minority Over-sampling Technique"), increasing the minority class by finding its k nearest neighbour.

After preprocessing of the data, the next step is doing feature selection using 'Feature importance' technique, taking the features with the highest scores, preventing overfitting. Also another new thing that this article brings is the ensemble for classification; it is based on two models: Logistic Regression and K-Nearest Neighbour, after that it analyzes the accuracy using k-fold cross-validation. From the feature selection process, the following 6 attributes were used: SysBP, Age, totChol, CigsPerDay, diaBP, SysBP. With all the setup done before, there were implemented 3 classifiers with the following accuracy: Decision Tree(0.743), K-Nearest Neighbour (0.834), Logistic Regression (0.943). Moreover using the ensemble method the accuracy resulted is a very high one: 0.991.

### **3.2.4 Combined Datasets Research**

In the year 2021, very closely published to the former article, there was also published an article for prediction of CVD's, but it uses more classifiers (6) and it is using the UCI dataset and the Heart Failure Prediction Dataset(both presented in the Data Overview chapter),but not the Framingham one[3]. The first step that this article tackled, was preprocessing of the data which consisted in finding missing or null values and handling categorical variables by creating dummy variables for each categorical variable.

For the UCI heart disease dataset the best test accuracy was obtained for applying the Support Vector Machine (SVM) classifier (0.918), followed by KNN classifier and Logistic Regression (0.9016) , very close to them was the Random Forest classifier (0.9), after that the XGBoost (0.8852) and the classifier with the lowest score was Naive Bayes classifier (0.8689). For the Heart Failure Prediction Dataset the best classifier was Random Forest(0.9412), followed by XGBoost (0.9286), KNN (0.916), SVM

(0.9076), Logistic Regression (0.8445) and the lowest scored: Naive Bayes (0.7983). Moreover, for this research the two datasets were combined (on the 12 common attributes) and the models ran on this new dataset: the best: Random Forest (0.9331), after that XGBoost (0.9164), than two close ones: SVM (0.8361) and KNN (0.8227) and the lowest score for: Naive Bayes (0.7793) and Logistic Regression (0.7726). In all cases, Random Forest, KNN, SVM, and XGBoost algorithms performed really well. The only exception is the Naive Bayes algorithm.

### **3.3 Conclusions**

In conclusion, the review of existing literature on heart prediction using machine learning reveals a diverse array of approaches, ranging from traditional machine learning models to advanced deep learning architectures, also highlighted in this roadmap - state of the art paper[8], that consisted as a base for the choice we took on what model to implement.

The above mentioned articles stand as a foundation for this research, however we can see that, except for the last presented article, each one only referred to one available dataset, not trying to see the comparative results between different datasets. Moreover, we can see that the data and the way it is being preprocessed is very important, like handling data-inbalance, empty values and feature selection. Even though for this research purpose we will not try to implement every model and methodology above we took them into consideration and tried one that was not really addressed: Extra Tree classifier.

# Chapter 4

## Study case - Heart Disease Prediction

The goal of this chapter is to present the original contributions brought to this field of research. This chapter includes both the experiments done and their results in a comparative manner, between the models, the datasets and the state-of-the-art results and also, it includes the web application developed using Java Spring and React for a quality user-experience with the trained model.

### 4.1 Approach and Methodology

In the experiments made for this paper, as mentioned before, we choose to use three datasets to see how each model would perform on different datasets. The approach we choose is to prepare all the datasets in the same manner, to maintain the level. After that, different models were trained on them and tried to optimise the results by fine-tuning the hyper-parameters. The results obtained for the respective models will be presented in the following subsections.

After that, we choose to continue with the Extra Tree Classifier as the model to use in the application, the model being trained on the Heart Failure Dataset[6]. We choose this model, because, as for now, it was not explored for this problem, however it presented comparable results to the state-of-the-art ones, especially for the selected dataset.

### 4.2 Prediction of developing a CVD

This subsection aims to present the experiments made and how each of them was conducted. Moreover, the results will be presented and then it will be conducted a study to compare this results between each other, between datasets and with the state-of-the-art results and how we choose which model to integrate afterwards into the application, all these steps are visually presented in the Figure 4.1.

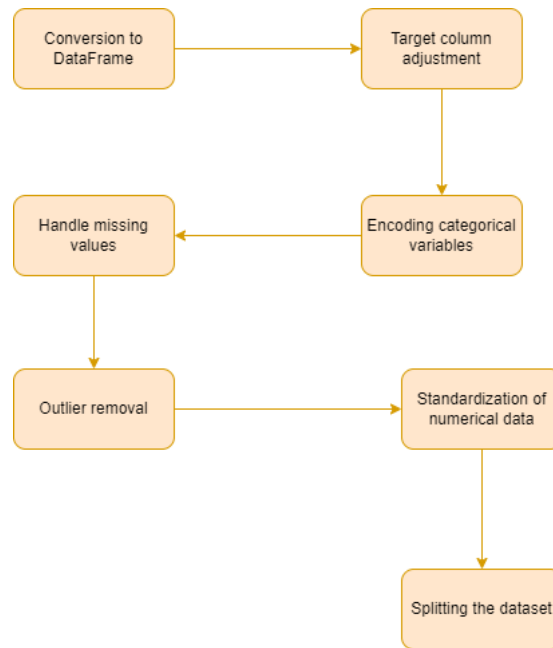


Figure 4.1: Data Preprocessing steps

### 4.2.1 Data Preprocessing

As mentioned above, the data preprocessing was conducted in the same way for all three datasets, because we wanted afterwards to have the same level for the models.

First of all, all three of the datasets were transformed into a DataFrame, the Heart Failure Dataset and the Heart Disease Dataset from a .csv and the Framingham Dataset from a .dta file. Furthermore, we needed to fine-tune the target column in the Framingham Dataset, which shows the actual result. This is because the data indicated where the heart disease was found, and a value of 1–16 indicated the development of a heart condition, while a value of 0 indicated the absence of a heart condition.

The next step was to convert categorical variables to numerical formats using a label encoder. The data frame is split into object-type categorical data and other data types. For each column in the object-type category is assigned unique integers to text labels. The encoded data frame is then concatenated with the rest of the columns to form a new data frame. This transformation ensures that all categorical attributes are suitably encoded for further model training processes.

Following this was the filling out of the dataframes, replacing any missing values with the mean value from that specific column. This method was chosen so we could keep as much data from the dataset as we could. Then we removed the outliers using an Isolation Forest method. We give the specific features to the method and it classifies the data points as outliers or normal. The outliers are those that, according to the model's calculation of the required fraction, considerably vary from

the norm; they are subsequently eliminated from the dataset.

We then standardize numerical data using a `StandardScaler`, which transforms each feature to have zero mean and unit variance, because many models are sensitive to the magnitude of the features. By standardizing the features, all variables contribute equally to the analysis, ensuring that the model's performance isn't biased by the natural variance in the dataset.

The final step in the preprocessing of the data, and a crucial one is splitting the dataset into training and testing sets. The larger segment for training the model (70% of the data) and a smaller segment for testing its accuracy (30% of the data). This division is done randomly but consistently across runs, thanks to a fixed random seed. The function ultimately returns these subsets, ensuring that the model can be trained on one portion of the data and validated on an independent set to evaluate its performance and the capacity to generalize.

### 4.2.2 Experiments

The models trained on this datasets were the ones presented also in the theoretical chapter: Logistic Regression, Support Vector Machines (SVM), Decision Tree, Random Forest and Extra Tree.

The process involves training logistic regression models on separate training sets and then assessing their accuracy using test data. After this we consider the SVM, starting with feature scaling to ensure all variables contribute equally to the analysis, enhancing model accuracy and efficiency. A recursive feature elimination (RFE) process then selects the most influential features, further refining the model input. Hyperparameter tuning is conducted via `GridSearchCV`, optimizing parameters like the regularization constant, kernel type, and gamma setting to find the best configuration for the SVM. This structured framework ensures that the SVM models are not only tuned to deliver optimal performance but are also critically evaluated and understood in the context of their application to various datasets.

A Decision Tree model is configured with a variety of potential hyperparameters, encompassing tree depth, minimum samples per split and leaf, and the splitting criterion. To identify the most effective configuration, the model undergoes a rigorous `GridSearchCV` process with k-fold cross-validation, pinpointing the optimal hyperparameters by assessing accuracy. Once optimized, the decision tree is employed to predict outcomes on test datasets. This thorough methodology guarantees that the decision tree model is properly improved and that its prediction accuracy is evaluated critically, creating a solid foundation for further predictive jobs in the future.

For the Random Forest Classifier, initially, the optimization process involves using a hyperparameter tuning method called `Hyperopt`, which employs a Bayesian

optimization technique to efficiently search the parameter space. Key parameters such as the number of trees, tree depth, and node splitting criteria are varied within specified ranges to find the optimal settings that maximize accuracy. Once the best parameters are identified, the Random Forest model is retrained with these optimized settings to enhance its predictive capabilities.

For the Extra Tree Classifier, the method starts with defining a range of potential settings for the Extra Trees model, such as the number of estimators and the depth of the trees. To find the optimal configuration, Optuna's Bayesian optimization technique is employed, which tests different combinations of parameters to maximize the F1 score—a balanced metric that considers both precision and recall.

Optuna is an open-source optimization library designed to automate the process of tuning hyperparameters for machine learning models. It provides efficient mechanisms to search through a wide range of parameter combinations and find the most optimal settings. The library supports various optimization algorithms, including Bayesian optimization, Tree-structured Parzen Estimator (TPE), and others. Optuna is particularly known for its user-friendly interface and its ability to handle complex, multi-dimensional search spaces effectively. This makes it a valuable tool for researchers and developers aiming to enhance the performance of their models by optimizing the configurations that govern their behavior.

The optimization process is conducted using a StratifiedKFold cross-validation approach to ensure that each subset of the data is proportionately represented in both training and validation phases. To address class imbalance, SMOTE (Synthetic Minority Over-sampling Technique) is integrated into the training process, artificially augmenting the minority class to equalize the influence of each class on the model's learning. We would also like to mention the best hyperparameters we found for the model we will then use in the application: the Extra Tree classifier for the Heart Failure Dataset with the hyperparameters: `n_estimators: 11`, `max_depth: 15`, `min_samples_split: 13`, `min_samples_leaf: 6`, `max_features: 'log2'`. In the Figure 4.2 we shown a visual representation of the improvements brought by the Optuna for the Extra Tree classifier.

Once the optimal parameters are determined, the Extra Trees model is retrained with these settings, and its performance is rigorously evaluated on unseen test data.

### 4.3 Comparison between results

The aim of this subchapter is to present the results obtained in this experiments. The results will be first presented for each dataset, to see the difference between them and after they will be compared to the state-of-the-art results.

It is important also, to present firstly the way they are to be compared. First of all,

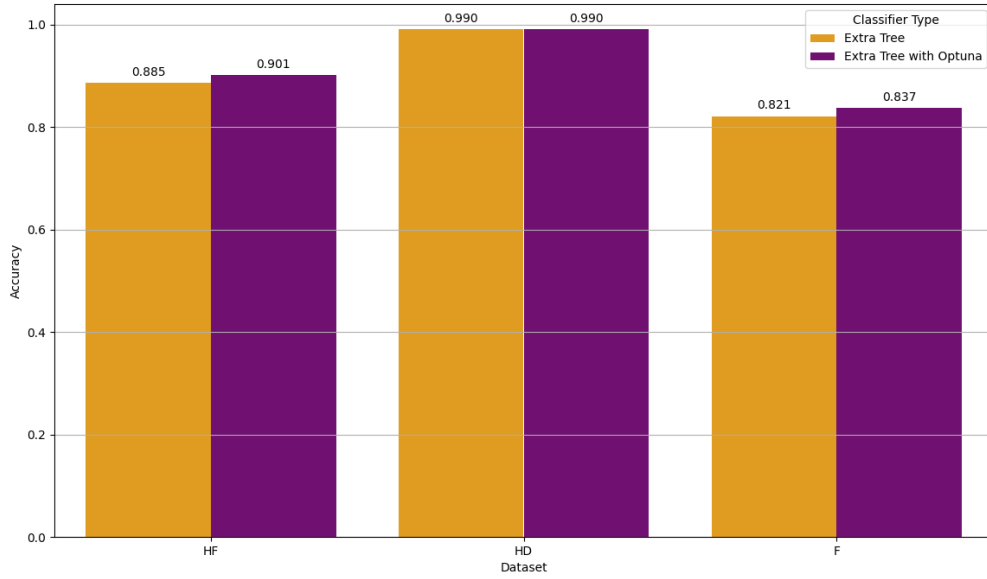


Figure 4.2: Accuracy by Dataset for Extra Tree improved with Optuna

for each model trained on each dataset, there was computed the confusion matrix, a table used in classification to visualize the accuracy of a model. It shows the correct and incorrect predictions of the model categorized by the actual classes. To assess the quality of the results, there were calculated 4 metrics:

1. **Accuracy:** Accuracy measures the proportion of true results (both true positives and true negatives) among the total number of cases examined. It is calculated as:

$$Accuracy = \frac{TruePositives(TP) + TrueNegatives(TN)}{TotalPopulation} \quad (4.1)$$

2. **F1 Score:** The F1 Score is the harmonic mean of precision and recall. It is particularly useful when the classes are imbalanced. The F1 score is calculated as:

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4.2)$$

3. **Recall** (also known as sensitivity): Recall is the ratio of correctly predicted positive observations to all actual positives. It is calculated as:

$$Recall = \frac{TruePositives(TP)}{TruePositives(TP) + FalseNegatives(FN)} \quad (4.3)$$

4. **Precision:** Precision is the ratio of correctly predicted positive observations to



the total predicted positives. It is calculated as:

$$Precision = \frac{TruePositives(TP)}{TruePositives(TP) + FalsePositives(FP)} \quad (4.4)$$

### 4.3.1 Comparison between best models for each dataset

After the models were presented in the above chapter, both their implementation and their optimisations, now will be presented the actual results.

#### Framingham Dataset

The Framingham Dataset was the one that the model provided the poorest results on. We believe that this is also because of the data-inbalance presented in this dataset. For this dataset, the models performed with similar efficiency, the results being close. The worst for this dataset performed the Decision Tree Classifier and the best output was given by the Extra Tree Classifier after optimisation with Optuna and by the Random Forest Classifier. A visual representation of the actual results of the accuracy metric for each model for this dataset is presented in the Figure 4.3.

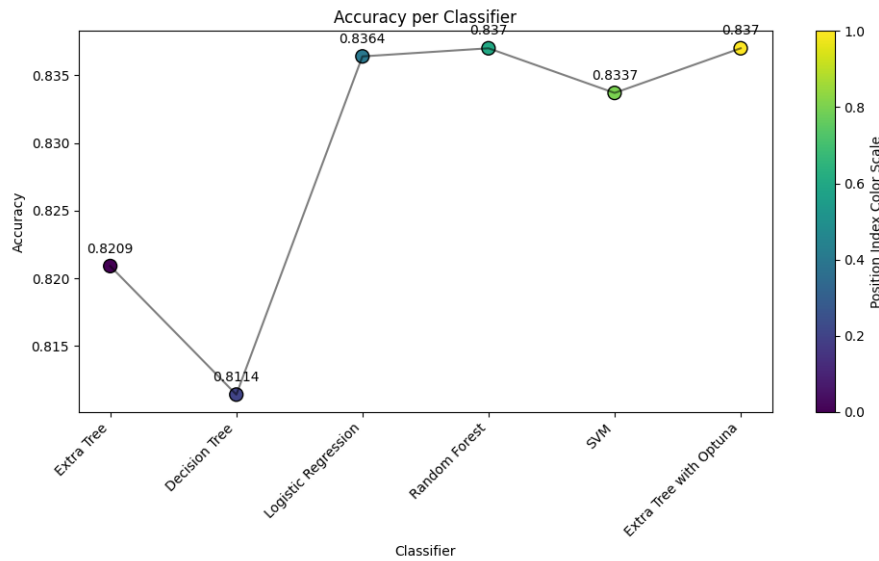


Figure 4.3: Accuracy plot for Framingham Dataset

#### Heart Disease Dataset

The Heart Disease dataset, we believe it was an easier dataset to understand for the models, because we observed that the tree based models managed to obtain an accuracy greater than 0.97, this accuracy being obtained on both the training dataset and the validation dataset. However the Logistic Regression obtained the

lowest accuracy (0.80), followed by the Support Vector Machines (0.93). The visual representation of the accuracy results can be viewed in the Figure 4.4.

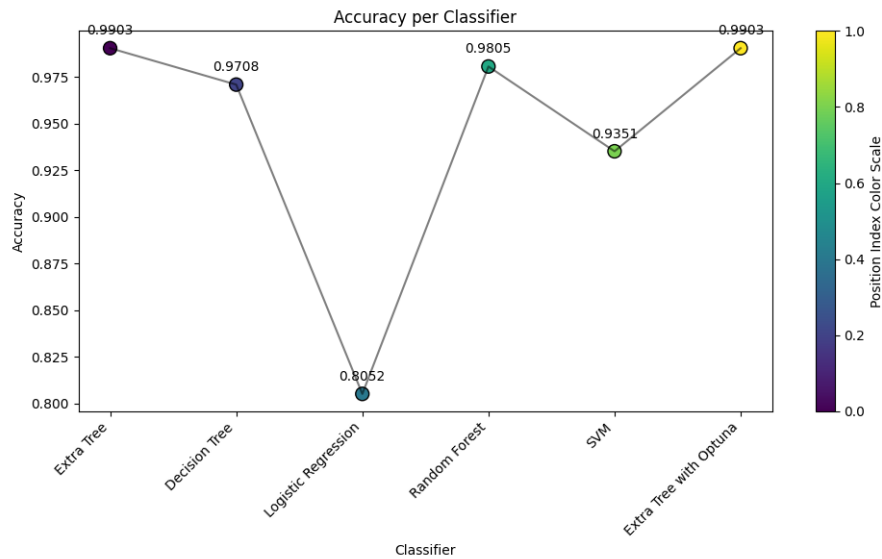


Figure 4.4: Accuracy plot for Heart Disease Dataset

### Heart Failure Dataset

The Heart Failure dataset was the dataset that we choose to continue with in our application. For this dataset, the poorest results were obtained by Decision Tree Classifier (0.83) and the best were obtained by the Extra Tree Classifier optimised with Optuna(0.90), the rest of the models obtaining results around (0.87-0.88), as it is visually represented in the Figure 4.5.

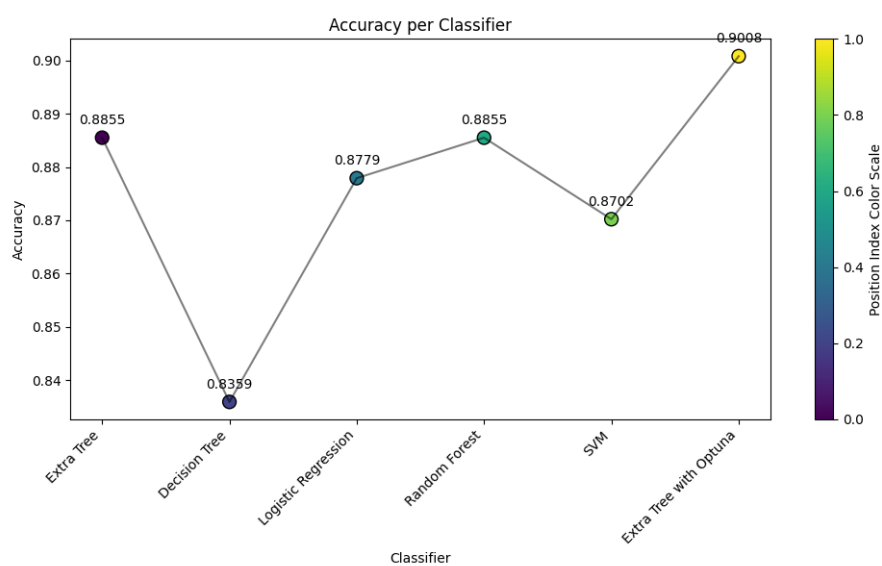


Figure 4.5: Accuracy plot for Heart Failure Dataset

### 4.3.2 State of the art results compared to current results

In the 'State of the Art results' chapter, for the Heart Disease dataset were presented 4 articles, 2 performing the best: the first one [1] selected only the data from the Cleveland dataset (303 rows) and managed to obtain with the Decision Tree an accuracy of 0.93, where we used the whole dataset and obtained an accuracy with the Decision Tree of 0.97. The second article [16] selected only 8 features and managed to obtain 0.94 using a Voting Classifier, but our models surpassed this value using all the features. We managed to improve the results, the best achieved being when the Extra Tree classifier with Optuna was used (0.99), as it can be seen in the table with all our results 4.7.

For the Framingham dataset, we managed to improve the valeus obtained by the article [2], from 0.7 to 0.837 being the highest we achieved using Extra Tree with Optuna, but we did not manage to improve the results obtained in the article [14] witch was the best achieved by Logistic Regression 0.94.

For the Heart Failure dataset we managed to make the most improvements. The original contribution being a model: the Extra Tree Classifier that was not previously used for this purpose. The state-of-the-art results extracted from the following two articles were: 0.88 fro Logistic Regression and Random Forest from the article [12] and 0.87 for the metamodel built in the article [11]. We first managed to get the result 0.88 fro the Extra Tree Classifier, but after optimising its hyperparameters using Optuna we managed to surpass both this value and the state-of-the-art values achieving 0.90, the confusion matrix for this can be observed in the Figure 4.6. For

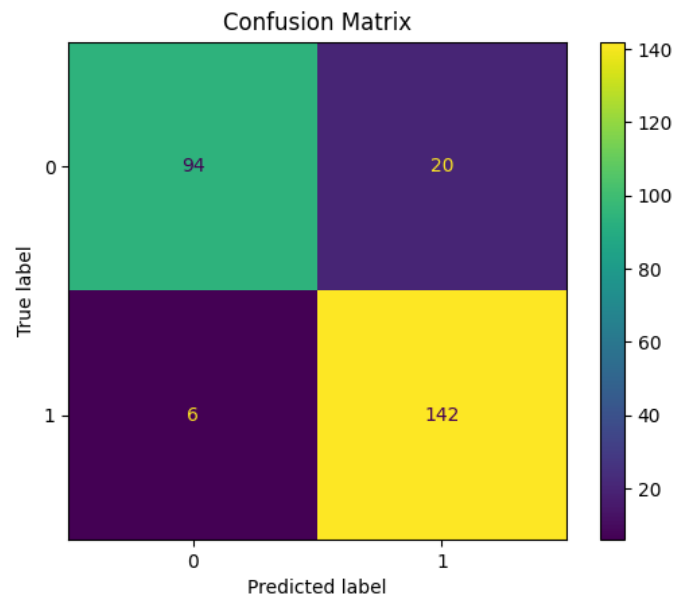


Figure 4.6: Confusion matrix for ExtraTree Classifier with Optuna on the Heart Failure Dataset

the mentioned above reasons we decided that this model on this dataset will be the one used in the application.

All the results obtained from our experiments are presented in the Figure 4.7, with the 4 metrics we calculated and categorised also on the dataset they were trained.

	Classifier	DataSet	Accuracy	F1 Score	Recall	Precision
0	Extra Tree	HF	0.8855	0.8854	0.8855	0.8854
1	Extra Tree	HD	0.9903	0.9903	0.9903	0.9904
2	Extra Tree	F	0.8209	0.8011	0.8209	0.8169
3	Decision Tree	HF	0.8359	0.8347	0.8359	0.8365
4	Decision Tree	HD	0.9708	0.9707	0.9708	0.9723
5	Decision Tree	F	0.8114	0.7935	0.8114	0.8022
6	Logistic Regression	HF	0.8779	0.8772	0.8779	0.8783
7	Logistic Regression	HD	0.8052	0.8048	0.8052	0.8112
8	Logistic Regression	F	0.8364	0.8083	0.8364	0.8633
9	Random Forest	HF	0.8855	0.8847	0.8855	0.8867
10	Random Forest	HD	0.9805	0.9805	0.9805	0.9812
11	Random Forest	F	0.8370	0.8085	0.8370	0.8665
12	SVM	HD	0.9351	0.9350	0.9351	0.9351
13	SVM	HF	0.8702	0.8699	0.8702	0.8701
14	SVM	F	0.8337	0.8077	0.8337	0.8507
15	Extra Tree with Optuna	F	0.8370	0.8085	0.8370	0.8665
16	Extra Tree with Optuna	HF	0.9008	0.8998	0.9008	0.9042
17	Extra Tree with Optuna	HD	0.9903	0.9903	0.9903	0.9905

Figure 4.7: Final Results

# Chapter 5

## Heart Disease Prediction Portal

The "Heart Disease Prediction Portal" is a web application that has the AI model written in a Python Server, that communicates with a Spring back-end and finally with the graphical user interface (GUI) written in React Typescript. We choose to write the AI model in Python, because it provides with a desirable amount of libraries directed for Machine Learning algorithms, such as Tensorflow, Keras, Torch, NumPy, Pandas, Scikit-learn, we are also using this in the AI part. The Java Spring application we choose because of our inclination to this programming language, and the freedom and easiness the Spring brings.

### 5.1 Problem Statement and App features

The healthcare sector has benefited considerably from technological advances, but there is still a constant need for more accurate and reliable applications that can aid doctors and patients alike. As we continue to improve healthcare technologies, the emphasis on early detection and management of diseases becomes increasingly important, particularly for conditions like cardiovascular diseases (CVDs).

Cardiovascular diseases (CVDs) are the leading cause of death globally, which emphasizes the need for better early detection methods. Current diagnostic techniques often detect CVDs when it's already too late for optimal treatment, making them less effective. These methods can also be costly and slow, which burdens healthcare systems, particularly in less wealthy areas. Recognizing these challenges, our goal is to develop an application that uses machine learning to predict CVDs more effectively. Our aim is to create a tool that is not only highly accurate but also extremely user-friendly, enabling quick identification of potential health issues. The user does not have to know how an ML algorithm works or how to fine-tune it because the model incorporated in the application already has the hyperparameters configured. This would allow individuals to realize when they might have a prob-

lem and encourage them to seek medical attention immediately, improving their chances of a better health outcome.

## 5.2 Design and Architecture

“Heart Disease Prediction Portal” is a web application; its development can be logically split into a front-end part and a back-end part. In the following sections, we will discuss both of them and how they interact with each other.

### 5.2.1 Front-end

Front-end refers to the part of the software that users interact with directly, both in the way they see the application and the way they can use its functionalities. Our objective was to create a fantastic user experience that was clear, adaptable, and effective. We opted to develop the front end in React because of its strong community support, modular component-based architecture, and effective rendering, all of which contribute to a reliable and maintainable user experience. Moreover, we chose TypeScript because it enhances JavaScript’s flexibility with strong typing and advanced object-oriented features, allowing us to catch errors early in the development process and streamline the maintenance and scalability of our application. Also, for the styling part of the application, we chose Material Ui, because it elevates the look of the application.

The functionalities that we implemented are split into two pages: the authentication page, where the user can login into his account; and the home page where the prediction would take place. The home page presents a form in which the data can be introduced, a list of previous predictions, along some information about the form’s fields and CVDs in general. This is why the front-end organisation was also split into two packages: Authentication and Predict. Both include an API file that manages the connection with the back-end, this is where calls are made to the back-end and information is exchanged.

In our application, we utilize the Provider design pattern to manage user authentication and prediction data efficiently. The next common component is the Provider, present for both. The *AuthProvider* specifically manages user authentication, handling login sessions, user status, and authentication errors. It centralizes the authentication process, making secure access and user credential management more straightforward throughout the application. On the other hand, the *PredictionProvider* is focused on fetching and managing cardiovascular health prediction data. It interfaces with backend services to retrieve necessary data, managing loading and error states efficiently. This provider uses authentication details from the

AuthProvider to ensure secure API calls, catering to the application's specific needs for health predictions.

After this the directories both contain the page components. For the Authentication directory there is only one, but for the *Predict* one there is a more complex structure. There is a *Main* file in which the React components are connected as it can be seen in the Figure 5.1.

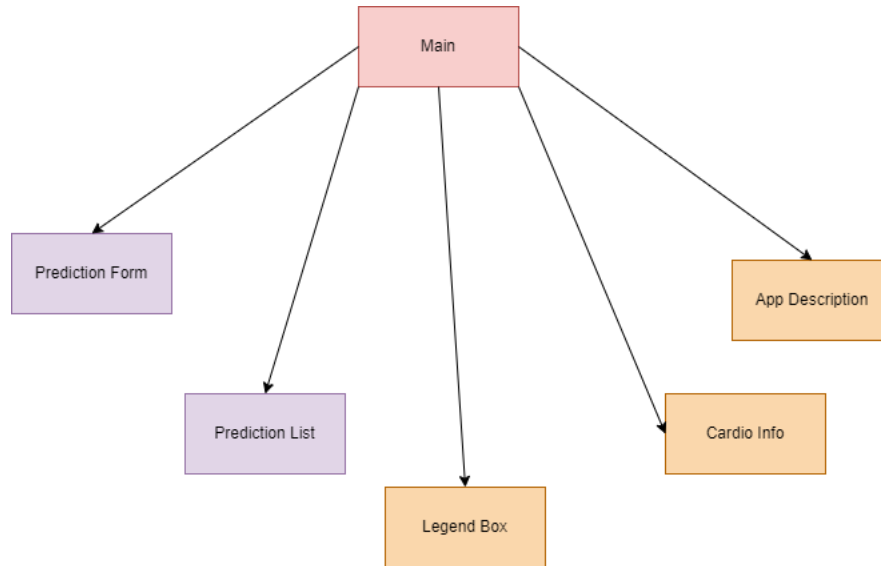


Figure 5.1: Visual Components on the application

### 5.2.2 Back-end

Back-end refers to the server-side part of the application, which handles tasks such as managing database interactions, executing server logic, and most importantly, running a machine learning model.

For the integration of the ML algorithm into the back-end code we used Flask. Flask is a lightweight WSGI web application framework in Python that is widely used for developing web applications. It is designed to be simple and easy to use, enabling rapid development with minimal lines of code. We used it to set up a web server with a specific endpoint that handles POST requests for making predictions based on user-provided data. The code also involves loading a pre-trained machine learning model using joblib. Joblib is a Python library that provides utilities for saving and loading Python objects that make use of NumPy data structures, efficiently, in our case it was particularly useful for persisting the model.

We took a methodical approach to building the application's back-end architecture by following the Model-View-Controller (MVC) architectural paradigm. This strategy made it easier to arrange our software in a methodical manner by dividing up issues related to data processing, presentation, and business logic. Our main

framework of choice turned out to be Java Spring Boot, which provides unmatched assistance in creating RESTful APIs. Our development process was expedited by utilizing its features, which eliminated the need for time-consuming manual settings.

For our database system we choose PostgreSQL, it was essential to our data management approach since it gave our back-end operations stability and scalability. We containerized it into a Docker image. In addition, we took a moral stand on security, using JWT token authentication to protect our endpoints from unwanted access. The confidentiality and integrity of the resources in our application are enhanced by this mechanism. We benefited greatly from the inclusion of the *Lombok* library in terms of code readability and conciseness. We were able to reduce the amount of time we spent creating generic code by using its annotations, which improved the maintainability and readability of our software.

We included Swagger into our back-end ecosystem to achieve our goals of thorough documentation and easy-to-use API administration. Swagger is an effective tool for creating, describing, and testing APIs. It provides developers and other stakeholders with a centralized platform where they can easily explore and interact with our endpoints. Through the automatic generation of interactive API documentation from our code, Swagger promotes clarity and openness in our development process by keeping our API specs current and synchronized with implementation. Additionally, its interactive user interface (UI) makes it simple to explore and test endpoints, enabling developers to effectively validate their hypotheses and troubleshoot problems. We maintain a high quality for API design and communication, improving user experience and expediting teamwork among developers, all thanks to the smooth integration of Swagger into our workflow. 5.2



Figure 5.2: Swagger UI



Knowing the constructs that stay at the base of this application, we can explore its architecture. The architecture is structured in three main layers: controller, service and repository:

- **Controller:** Controllers serve as the orchestrators of incoming requests, responsible for handling and processing client interactions with the application's resources. These controllers encapsulate the business logic required to interpret requests, interact with the appropriate services or data sources, and generate responses conforming to the REST architectural style. Through the use of annotations such as `@RestController`, along with request mapping annotations like `@RequestMapping` or `@GetMapping`, we have defined the URL patterns and HTTP methods that each controller method can handle. In our application existing two controllers: one that handles the authentication requests and one that handles the prediction requests.
- **Service:** Services represent the core business logic and functionality of the application. Services act as intermediaries between controllers, which receive incoming requests, and data repositories which store or provide access to the application's data. In our application, we split once again based on functionalities into more services: *AuthenticationService*, *PredictionService* and *JwtService* that handles JWT token generation, extraction, validation, and manipulation.
- **Repository:** A repository serves as the intermediary between the application's business logic and the underlying data storage mechanism, typically a database. It encapsulates data access operations, such as querying, inserting, updating, and deleting data, providing a clean and abstract interface for interacting with the data source. We, once again, split into two: *PredictionRepository* and *UserRepository* both of which extend the *JpaRepository* interface provided by Spring Data JPA. The *JpaRepository* interface, offers a wealth of predefined methods for interacting with the database, such as querying, saving, updating, and deleting entities. By extending this interface, our repositories inherit these methods, empowering us to perform common data operations without the need for boilerplate code.

We will now go into more detail on the tasks done by the two services, here is relying the logic of the application:

#### **AuthenticationService**

At its essence, the *AuthenticationService* facilitates the authentication of users seeking access to the HeartPrediction system. The primary method, *authenticateUser*, initiates this process upon receiving an *AuthenticationRequestBody*, which encapsulates the user's credentials, namely, their email and password. Leveraging

Spring Security's authentication mechanisms, the `AuthenticationService` verifies the user's credentials against stored records, thereby validating their identity. Central to the authentication process is the integration with Spring Security's `UserDetailsService`. This interface, a fundamental component of the Spring Security framework, abstracts the retrieval of user details from the underlying data source. Beside this, the `AuthenticationService` undertakes the generation of JSON Web Token (JWT) to facilitate subsequent authorization and user session management.

### **PredictionService**

The `PredictionService` comprises methods designed to streamline the retrieval and processing of predictions for heart disease. Leveraging the `PredictionRepository`, the service retrieves predictions associated with a specific user, transforming them into a format suitable for presentation via the API. A central aspect of the `PredictionService` is its ability to interact with external AI models for heart disease prediction. The `predictHeartDisease` method orchestrates this interaction by sending prediction requests to a Python Flask service via RESTful API calls. Upon receiving predictions from the AI model, the service persists the results alongside user-specific data, facilitating ongoing analysis and evaluation. Throughout the predictive analytics flow, the `PredictionService` prioritizes data integrity and reliability. By leveraging Spring's `RestTemplate` for HTTP communication, the service ensures secure and efficient interaction with external AI endpoints. Additionally, logging mechanisms, facilitated by the `Slf4j` library, provide valuable insights into the prediction process, aiding in troubleshooting and performance optimization.

The full diagram of classes and how they are connected can be seen in the Figure 5.3.

## **5.3 User Guide**

We developed a user-friendly interface for our portal. First of all, we have the login step. The login process for the "Heart Disease Prediction Portal" is straightforward and designed to ensure secure access to user-specific features, such as personalized predictions and history tracking. The user needs to navigate to the URL of the "Heart Disease Prediction Portal", after that the user will be prompted with the login page. 5.4.

Here, the user should fill in the fields regarding the email and the password corresponding to this account and then press the *LOGIN* button.

After the successful login, the user is redirected to the home page of the application. Here we have more components, starting with an introduction about our application and a pie-chart with the annual deaths caused by CVDs compared to other causes. Also on the informative side of the portal, there is a box with relevant

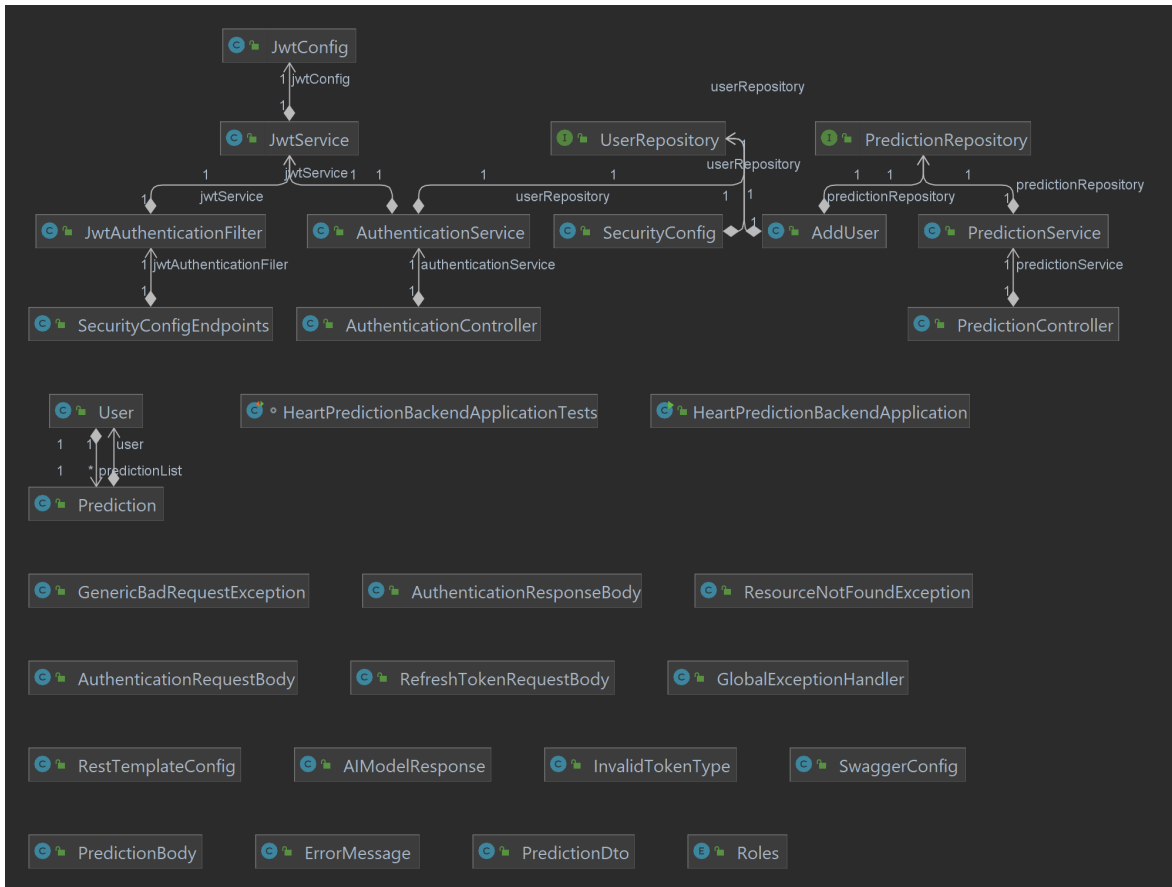


Figure 5.3: Class Diagram

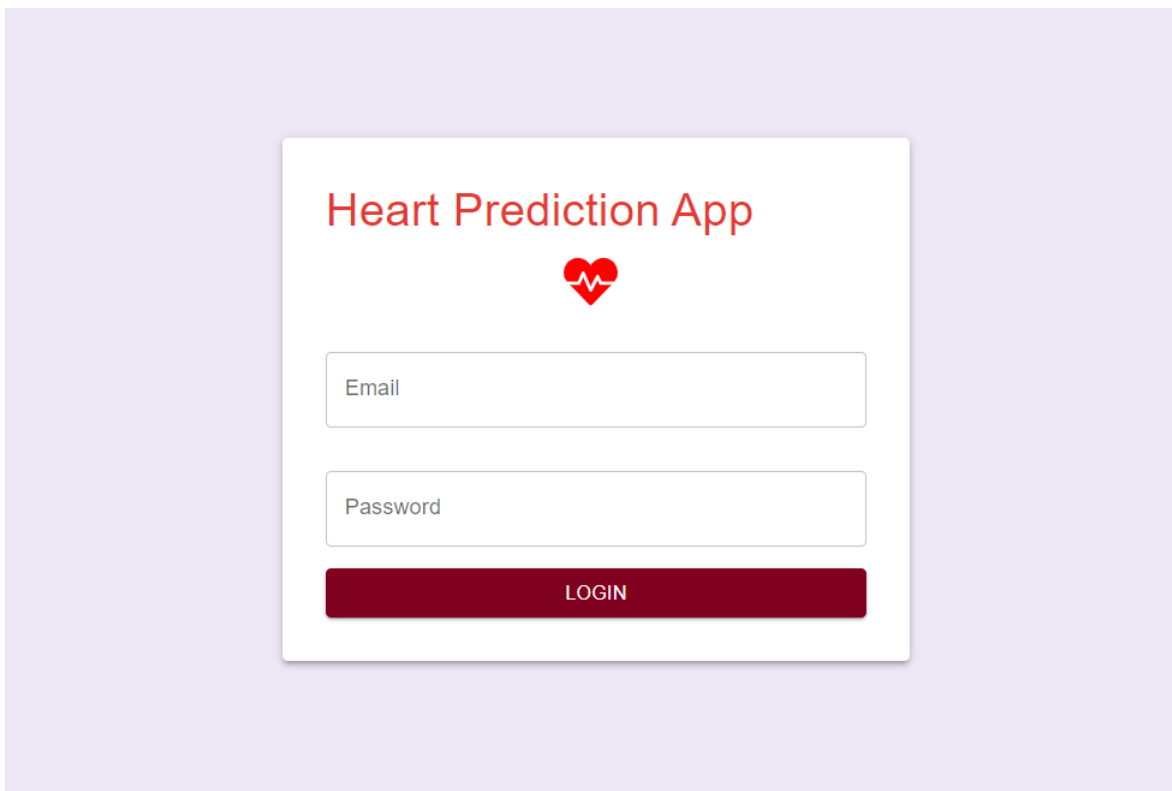


Figure 5.4: Login page

information about CVDs and one more box that presents the form's fields and what values are accepted.


The most interesting part of our application lies in the next two components. The first one being a form in which the user will need to fill in both medical information and other that are relevant and after pressing the *SUBMIT PREDICTION* button, his data will be given to the model and a prediction will be made. Right below the form, there will be a history of the users' previous predictions. Here will be added the new prediction, these predictions being sorted in chronological order. 5.5

## Heart Disease Prediction Portal

Welcome to the Heart Disease Prediction Portal

With advanced AI algorithms, this application provides a prediction of your likelihood to have or develop cardiovascular diseases. By inputting lab results and other health parameters, users can receive an immediate assessment of their cardiovascular health risks.

Our goal is to empower individuals with knowledge about their health, enabling them to make informed decisions and seek appropriate medical advice. This tool is designed to assist in the early detection of potential health issues, leading to better health outcomes and preventive care.



### Enter Prediction Details

Age

Sex

Chest Pain Type

Resting B P

Cholesterol

Fasting B S

Resting E C G

Max H R

Exercise Angina

Old Peak

St Slope

SUBMIT PREDICTION

### Field Descriptions

Age: Age of the individual.  
Sex: Sex of the individual (M for male, F for female).  
Chest Pain Type: Type of chest pain (ATA, NAP, ASY, TA).  
Resting BP: Resting blood pressure in mm Hg.  
Cholesterol: Serum cholesterol in mg/dl.  
Fasting BS: Fasting blood sugar higher than 120 mg/dl (1 if true, 0 if false).  
Resting ECG: Resting electrocardiographic results (Normal, ST, LVH).  
Max HR: Maximum heart rate achieved during exercise.  
Exercise Angina: Exercise-induced angina (Y for yes, N for no).  
Old Peak: ST depression induced by exercise relative to rest, measured in mm.  
ST Slope: The slope of the peak exercise ST segment (Up, Flat, Down).

### Cardiovascular Disease Facts

Cardiovascular diseases (CVDs) are the number one cause of death globally, taking an estimated 17.9 million lives each year.

Major causes include heart attacks and strokes, primarily due to smoking, poor diet, and physical inactivity.

Preventive measures include maintaining a healthy lifestyle, avoiding tobacco, and keeping blood pressure, diabetes, and cholesterol under control.

Nearly 90% of all cardiovascular diseases are preventable.

### Predictions List

Prediction ID: 1

Age: 40 - Sex: M

Chest Pain: ATA - Resting BP: 140

Cholesterol: 289 - Max HR: 172

Model Prediction: 0.89 (Normal), 0.11 (Disease)

LOG OUT

Figure 5.5: Application Home page

# Chapter 6

## Conclusions and future improvements

The development of the "Heart Disease Prediction Portal" has demonstrated the potential of integrating machine learning models with web-based platforms to enhance the early detection and management of cardiovascular diseases (CVDs). By employing advanced machine learning techniques, particularly the Extra Tree Classifier optimized with Optuna, we achieved significant improvements in prediction accuracy compared to existing methods. The use of Python for model development, Java Spring for the back-end, and React with TypeScript for the front-end allowed for seamless integration, providing a user-friendly interface for both patients and healthcare providers.

Our results indicated that the Extra Tree Classifier, when fine-tuned with Optuna, outperformed traditional models in predicting CVDs using the Heart Failure Dataset. This model showed balanced performance across various metrics, including accuracy, F1 score, precision, and recall, making it a reliable tool for early diagnosis.

Moreover, our comparative analysis across different datasets highlighted the importance of data preprocessing in enhancing model performance. Systematic optimization of hyperparameters was crucial in achieving high prediction accuracy.

Future improvements for the "Heart Disease Prediction Portal" include integrating additional and larger datasets from diverse demographics to enhance model generalizability, implementing real-time data processing for integration with wearable devices, and enhancing the user interface for better intuitiveness and interactivity. Strengthening security and privacy measures, ensuring compliance with healthcare data regulations, and improving model accuracy are also critical. These enhancements aim to make the application a more comprehensive, user-friendly, and reliable tool for early cardiovascular disease detection and management.

In conclusion, detecting cardiovascular diseases in the early stages is crucial for both individuals and the healthcare system, and leveraging machine learning models has proven to yield significantly good results.

# Bibliography

- [1] F. S. Alotaibi. Implementation of machine learning model to predict heart failure disease. *International Journal of Advanced Computer Science and Applications*, 10(6), 2019.
- [2] J.-J. Beunza, E. Puertas, E. García-Ovejero, G. Villalba, E. Condes, G. Koleva, C. Hurtado, and M. F. Landecho. Comparison of machine learning algorithms for clinical event prediction (risk of coronary heart disease). *Journal of Biomedical Informatics*, 97:103257, 2019.
- [3] N. Bora, S. Gutta, and A. Hadaegh. *Using machine learning to predict heart disease*. PhD thesis, California State University San Marcos, 2021.
- [4] R. B. D’Agostino, M. J. Pencina, J. M. Massaro, and S. Coady. Cardiovascular disease risk assessment: Insights from framingham. *Global Heart*, 8(1):11–23, 2013. Framingham Legacy Issue.
- [5] H. Denolin, H. Kuhn, H. Krayenbuehl, F. Loogen, and A. Reale. The definition of heart failure. *European heart journal*, 4(7):445–448, 1983.
- [6] Fedesoriano. Heart failure prediction dataset. 2021.
- [7] W. B. KANNEL, M. FEINLEIB, P. M. McNAMARA, R. J. GARRISON, and W. P. CASTELLI. AN INVESTIGATION OF CORONARY HEART DISEASE IN FAMILIES: THE FRAMINGHAM OFFSPRING STUDY. *American Journal of Epidemiology*, 110(3):281–290, 09 1979.
- [8] M. S. Khan, M. S. Arshad, S. J. Greene, H. G. Van Spall, A. Pandey, S. Vemulapalli, E. Perakslis, and J. Butler. Artificial intelligence and heart failure: A state-of-the-art review. *European Journal of Heart Failure*, 25(9):1507–1525, 2023.
- [9] D. Lapp. Heart disease dataset. 2019.
- [10] H. Lin and M. Li. *Practitioner’s Guide to Data Science*. Chapman and Hall/CRC, 2023.

- [11] I. Mahmud, M. M. Kabir, M. Mridha, S. Alfarhood, M. Safran, and D. Che. Cardiac failure forecasting based on clinical data using a lightweight machine learning metamodel. *Diagnostics*, 13(15):2540, 2023.
- [12] N. S. Mansur Huang, Z. Ibrahim, and N. Mat Diah. Machine learning techniques for early heart failure prediction. *Malaysian Journal of Computing (MJoC)*, 6(2):872–884, 2021.
- [13] W. H. Organisation. Cardiovascular diseases (cvds), 2021. Accessed: 2024-04-13.
- [14] A. Rahim, Y. Rasheed, F. Azam, M. W. Anwar, M. A. Rahim, and A. W. Muzaffar. An integrated machine learning framework for effective prediction of cardiovascular diseases. *IEEE Access*, 9:106575–106588, 2021.
- [15] Y.-Y. Song and L. Ying. Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2):130, 2015.
- [16] v. varale. Prediction of Heart Disease Using Machine Learning Algorithm. *Bio-science Biotechnology Research Communications*, 14(13):287–290, 11 2020.