DataScientest

Classification of Arrhythmia

Theresa B., James W., Miran P.

28.03.2024

Cohort: Jan24 DS Bootcamp

Datascientest.com

Project Mentor: Elie Chemouni

Contents

Introduction

Arrhythmia presents a significant challenge in cardiovascular medicine, characterized by irregular heart rhythms that can lead to severe health complications, including stroke and heart failure. Traditional methods of diagnosing arrhythmia rely on manual interpretation of electrocardiogram (ECG) data by skilled professionals. This process, however, is prone to human error and time-consuming due to the complexity of ECG data containing a large number of parameters. On top of that, in the past, a significant portion (up to 90%) of clinical alarms in intensive care units have been classified as false alarms (Lawless, 1994), highlighting the importance for reliable prediction and prevention mechanisms.

Recent advancements in machine learning (ML) offer a promising future for automating arrhythmia detection through the analysis of ECG signals (Mincholé et al., 2019; Huda et al., 2020; Sakib, Fouda & Fadlullah, 2021). ML algorithms can learn patterns and abnormalities in ECG data that may not be easily detectable to the human eye, potentially revolutionizing the way we diagnose and manage cardiac arrhythmias. From a technical standpoint, detecting arrhythmia using machine learning involves complex processes such as feature extraction, model selection, and data processing (Das & Ari, 2014). Key challenges include extracting relevant features from ECG signals that capture important characteristics indicative of arrhythmia, selecting appropriate ML models capable of learning complex patterns in ECG data, ensuring data quality and quantity for training, and enhancing model interpretability for clinical use. Overcoming these challenges requires expertise in signal processing, data science, and machine learning, along with collaboration with domain experts in cardiology.

Understanding the physiological basis of arrhythmias is crucial for the development of effective ML algorithms. Knowledge in cardiac electrophysiology, such as ECG waveforms and mechanisms underlying different types of arrhythmias, inform the selection of relevant features and biomarkers for detection. Additionally, considering the clinical relevance of arrhythmia detection is essential, including its implications for patient risk classification, treatment decisions, and long-term monitoring. By integrating technical expertise with scientific knowledge, machine learning-based approaches have the potential to enhance the accuracy and efficiency of arrhythmia diagnosis. Ultimately improving patient outcomes and advancing cardiovascular care.

Atrial fibrillation is the most common form of arrhythmia and was the original focus of our project. We started with a dataset from the University of Rotterdam containing raw ECG data and cardiology annotations classifying the ECG data concerning atrial fibrillation.

Unfortunately, transforming the raw data proved to be more time-consuming than expected. To adhere to our deadlines, we decided to explore datasets that were already in a readily usable format. This exploration led us to switch data sets not only once but twice in our quest to find suitable data for our machine- and deep learning models.

We first decided to continue with the “Cardiac Arrhythmia Database donated to UCI by Bilkent" and explored several machine learning models. However, after finding out that neural networks did not perform well even with using data augmentation on the UCI Bilkent dataset due to its small size, we pivoted once more and transitioned to the MIT-BIH Arrhythmia Database. This dataset provided a sufficient amount of data for us to implement deep learning techniques effectively. Our transitions marked a shift in our focus from specifically atrial fibrillation to a broader scope of arrhythmia detection. Nonetheless, our core question remained unchanged: "How might we be able to train a machine learning model to detect abnormal heart rhythms in ECG data accurately enough to diagnose arrhythmia?" This journey of dataset exploration underscores the dynamic nature of research and the iterative process of refining methodologies to achieve optimal outcomes.

Objectives

General Application

In this project, our primary objective is to develop a machine learning model capable of accurately distinguishing between the presence and absence of cardiac arrhythmia, initially focused on atrial fibrillation but expanding to encompass a broader detection of arrhythmia types. On top of being able to raise awareness about arrhythmia conditions, we aim to reduce false positives as well as false negatives associated with arrhythmia detection by leveraging machine learning techniques. A good model should contribute to improved accuracy to help healthcare professionals prioritize interventions and reduce unnecessary patient stress. More importantly, early detection of arrhythmias is crucial for timely intervention and treatment (Zhang et al., 2017). Our model aims to detect abnormalities in ECG data promptly, enabling healthcare providers to initiate appropriate interventions at the earliest possible stage (*Tachycardia (Fast Heart Rate) in Children)*. In addition to our primary objective of developing a machine learning model for accurately distinguishing between the presence and absence of cardiac arrhythmia, we recognize the importance of scalability and generalizability. We want to contribute to a framework that offers solutions which can be implemented effectively across various healthcare settings and populations.

With this project we aim to not only create an opportunity for us to apply our technical skills. But we also want to expand on domain knowledge since we all share a keen

interest in the subject and physiological measurements. While some of our group members possess some knowledge about arrhythmia, others have some familiarity with ECG utilization. Despite our team's limited expertise in cardiology, our collective enthusiasm and commitment to learning contribute to our ability to tackle the problem effectively. Our research has primarily relied on online resources and literature to refine our understanding of the problem and develop appropriate models. However, we acknowledge the potential value of consulting with domain experts in cardiology and data science to further refine our approach and validate our findings.

Technical Approach

To achieve our goal of developing a robust machine learning tool capable of accurately predicting arrhythmia from ECG measurement data, we plan to explore various technical approaches. Our methodology will involve several key steps. Initially, we will engage in comprehensive data preprocessing, encompassing standardization to ensure consistency and remove biases or variations in scale. Additionally, we will incorporate feature engineering/dimension reduction techniques such as principal component analysis (PCA) to enhance the quality of ECG signals and capture relevant patterns and characteristics.

After preprocessing the data, we will engage in several rounds of model selection, utilizing techniques like GridSearchCV and Randomized Search to determine the best hyperparameters for each model. To ensure accurate estimation of model performance and mitigate overfitting, we will perform multiple comprehensive rounds of hyperparameter tuning. Throughout this process, it will be crucial to conduct thorough analysis of performance metrics such as precision, recall, F1-score, and receiver operating characteristic (ROC) curves, enabling us to assess the effectiveness of each model.

UCI Bilkent Dataset

Exploratory Data Analysis

The “Cardiac Arrhythmia Database donated to UCI by Bilkent” contains electrocardiogram (ECG) information that can be used for diagnosis of arrhythmia (Guvenir et al., 1997). For example, the QRS interval or the T interval. "QRS" refers to a complex composed of three of the ECG's graphical deflections: Q, R, and S waves (Das & Ari, 2014). These deflections represent the depolarization of the ventricles of the heart. The QRS complex typically occurs within a specific time interval on the ECG tracing and reflects the electrical activity associated with ventricular contraction. Changes in the morphology or

duration of the QRS complex can provide diagnostic information about various cardiac conditions, such as conduction abnormalities, myocardial infarction, and ventricular hypertrophy (Lome).

The T interval, also known as the QT interval, is a segment on an electrocardiogram (ECG) that represents the time from the beginning of the QRS complex to the end of the T wave. It reflects the time it takes for the ventricles of the heart to depolarize and then repolarize. The QT interval is important clinically because abnormalities in its duration can indicate an increased risk of arrhythmias, particularly a type of arrhythmia known as torsades de pointes, which can potentially lead to dangerous ventricular fibrillation and sudden cardiac death. A prolonged QT interval can be congenital or acquired and may be associated with various factors such as certain medications, electrolyte imbalances, structural heart diseases, or genetic conditions (Tan et al., 2000).

The dataset consists of 452 entries (rows) and 280 columns. It contains 279 attributes and one target variable (class). The attributes include:

- Age: Age in years, linear
- Sex: Sex (0 = male; 1 = female), nominal
- Height: Height in centimeters, linear
- Weight: Weight in kilograms, linear
- QRS duration: Average of QRS duration in msec., linear
- P-R interval: Average duration between onset of P and Q waves in msec., linear
- Q-T interval: Ave duration between onset of Q and offset of T waves in msec., linear
- T interval: Average duration of T wave in msec., linear
- P interval: Average duration of P wave in msec., linear
- QRS
- T
- P
- QRST
- J
- Heart rate: Number of heart beats per minute ,linear

The rest of the features are individual signals sent from ECG leads by lead classification and key. The target variable “Class” is divided into 16 classes that are defined as follows:

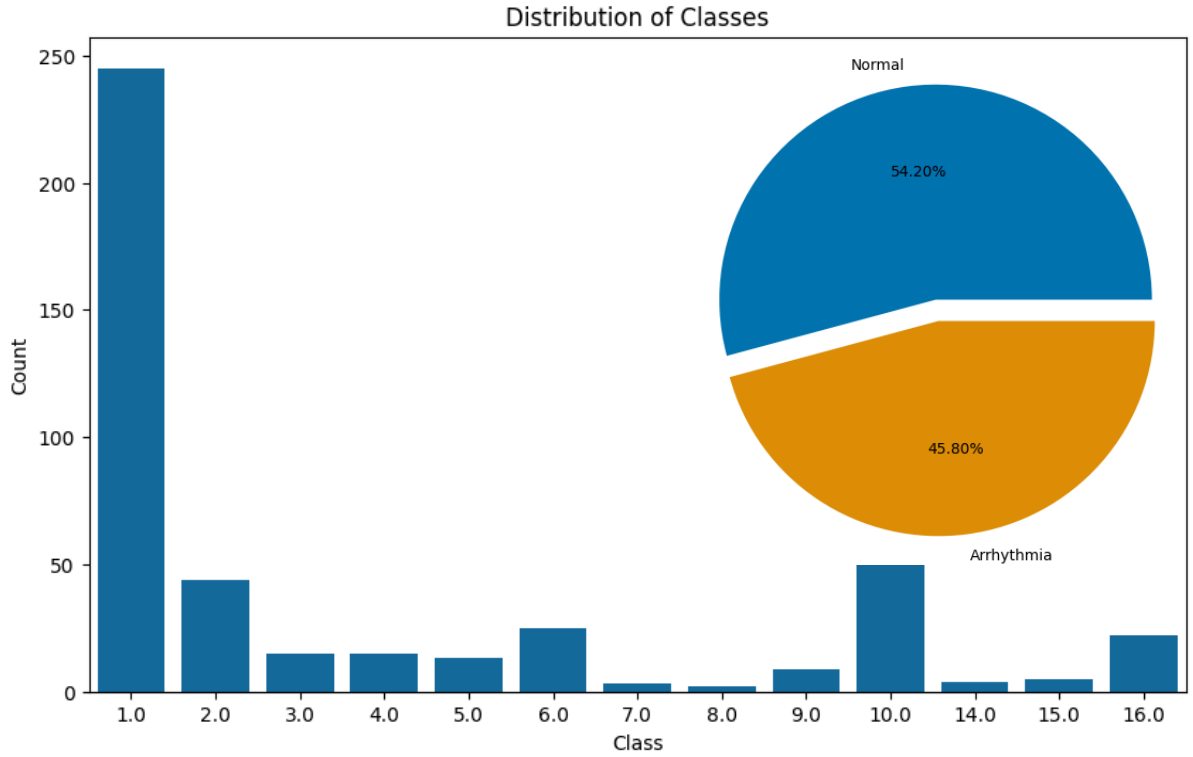
1. Normal
2. Ischemic changes (Coronary Artery Disease

3. Old Anterior Myocardial Infarction
4. Old Inferior Myocardial Infarction
5. Sinus tachycardia
6. Sinus bradycardia
7. Ventricular Premature Contraction (PVC)
8. Supraventricular Premature Contraction
9. Left bundle branch block
10. Right bundle branch block
11. 1. degree AtrioVentricular block
12. 2. degree AV block
13. 3. degree AV block
14. Left ventricular hypertrophy
15. Atrial Fibrillation or Flutter
16. Others

The dataset is available on the UCI database website:
http://archive.ics.uci.edu/dataset/5/arrhythmia
There is no strong imbalance in the binary diagnosis class (Figure 1) or the demographics of the subjects (Figure 2).

Figure 1

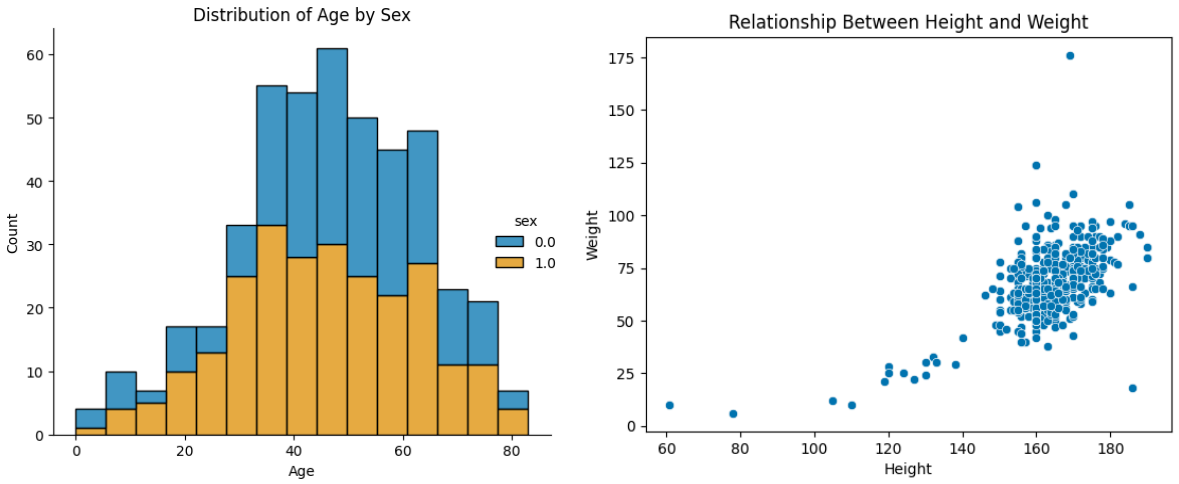
Data Distribution of Classes



Note. The bar graph shows the imbalanced distribution of classes amongst the data alongside normal ECG's (class 1) versus arrhythmia ECG's (all classes except for 1). Classes 11, 12, and 13 have a null-count and are left out. The pie-chart on the right shows the relatively balanced distribution of normal and all summarized arrhythmia ECGs in percentages.

Figure 2

Demographic Data Distribution

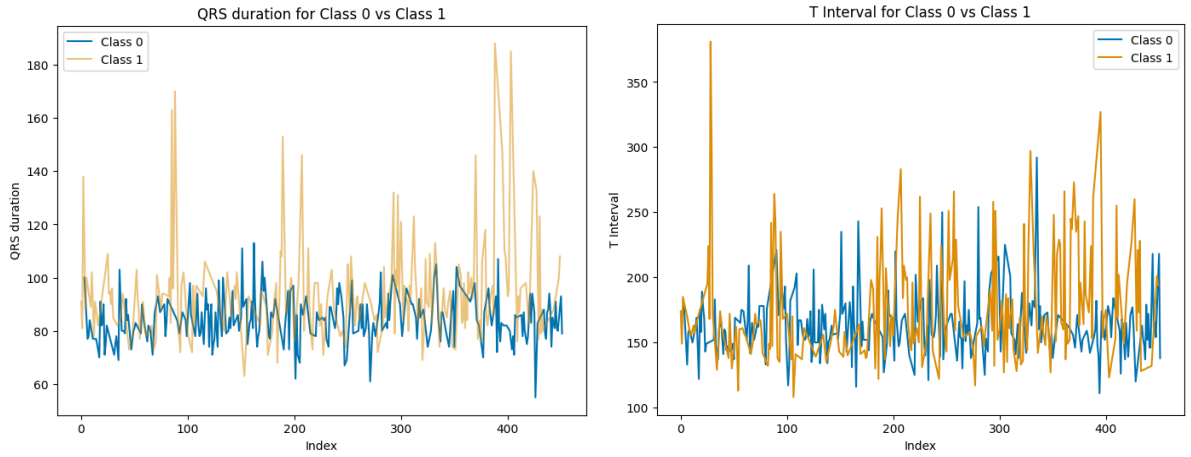


Note. The left Image displays the age distribution of male (blue) and female (orange) subjects. The right image displays the distribution of height versus weight.

According to Shapiro-Wilk Normality Testing, 17 of the features follow a Gausian distribution, while 262 do not. This will be relevant later, for example when deciding how to impute missing values in the columns where they are present. The left-skewed, right-skewed and bi-modally distributed features will not be ideally augmented by mean values. As mentioned above, the QRS duration and T interval are relevant features of an ECG that can give a first indication of an abnormal ECG. Figure 3 visualizes the differences in the QRS duration (left graph) and T intervals (right graph) for normal cases in black and arrhythmia cases in blue.

Figure 3

QRS and T Intervals for Normal and Arrhythmia Cases



Note. The figure depicts the QRS duration (left) and T interval (right) for normal ECG recordings (Class 0) and recordings classified as arrhythmia (Class 1). As highlighted earlier, both the QRS duration and T interval are critical parameters in classifying ECG signals for arrhythmia detection. Visualizing the distinction between normal ECG signals and those with classified arrhythmia reveals noticeable differences. This visualization underscores that certain variables in the dataset exhibit discernible disparities even at a glance, which can be visually detected by analyzing physicians.

Relevance, Data Limitations and Biases

Acknowledging the complexities inherent in our dataset, we recognize several challenges related to feature abundance, limited training data, and significant bias (Ramirez-Santana, 2018). The imbalance between normal cases and individual arrhythmia classes prompted our decision to initially group all arrhythmia cases into one class, ensuring a more balanced training process. Regarding potential biases, we are aware of the selection bias stemming from data collected exclusively from cardiology practices. While this facilitates model development tailored to similar settings, it may restrict generalizability to the broader population, as individuals seeking cardiology care may represent a specific subset with unique health concerns or access to specialized healthcare services. We are unaware of the heritage of the patients, and it being a small dataset supplied by a national university, one concern may be whether models built on it may be biased towards the population of the patients. We assume diagnosis in cardiology is universal, we are however mindful of the fact that there may be cultural nuances that we may not be aware of. Moreover, we are attentive to potential information bias due to missing data, marked by "?" in the dataset, which could

result from measurement instrument failures. In terms of confusion bias, we have identified potential confounding factors such as age, sex, and weight. Navigating these data limitations and biases, we are dedicated to continuously evaluate and refine our methods for a good reliability of our findings.

Given the relatively small size of our dataset, comprising only 452 rows, the utilization of data augmentation becomes a crucial part for enhancing the robustness and efficacy of our machine learning models. Data augmentation involves artificially expanding the dataset by applying various transformations to the existing data samples, such as rotations, translations, or scaling. By generating additional synthetic data points, we can navigate the risk of overfitting and improve the model's generalization capabilities. Essentially, data augmentation helps to diversify the dataset, enabling the model to learn from a more extensive range of scenarios and variations present in real-world data. This approach is particularly important in scenarios where collecting additional labeled data is challenging or resource-intensive. Therefore, incorporating data augmentation techniques is essential to the performance and reliability of our machine learning models, despite the constraints given by the limited size of our dataset.

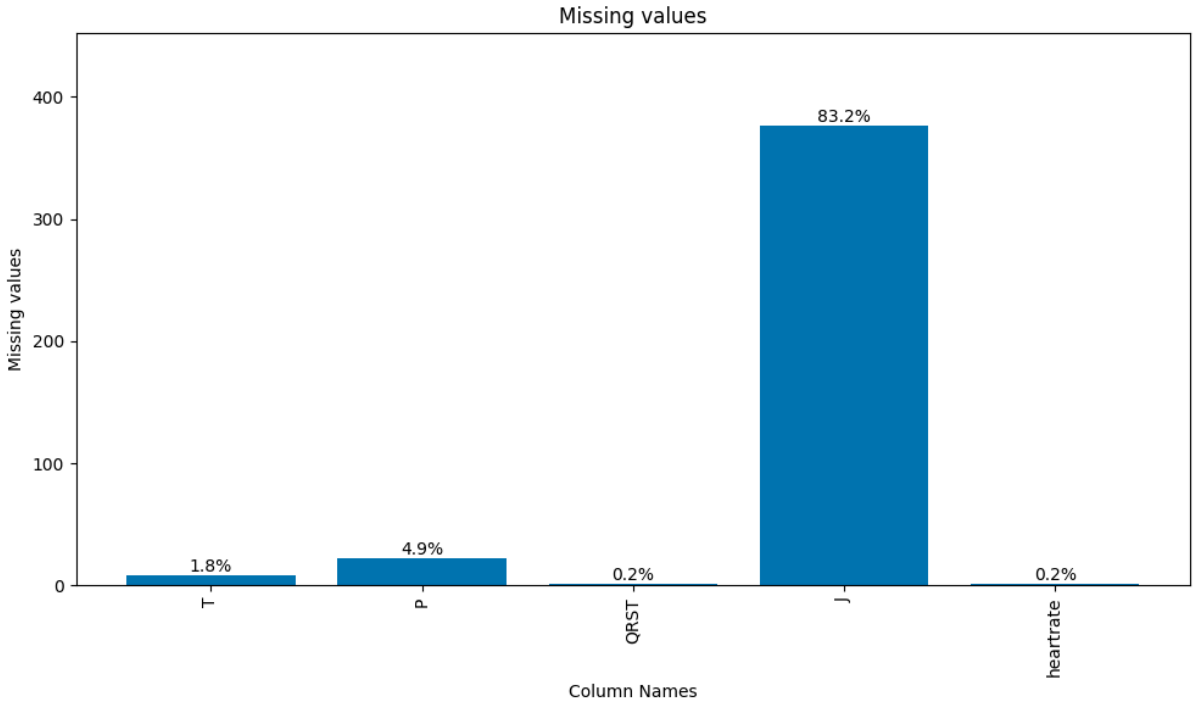
Pre-Processing

Handling Missing Values

In order to clean the dataset, several columns with missing values were identified and the missing values were replaced (Figure 4). For replacing values in the other columns, two approaches were explored: First, Nearest Neighbor approach using KNNImputer from the sklearn.impute model, and second, filling missing values with the median of their respective columns/similar groups. The column “J” was dropped due to a very high number of missing values and because it is represented in the measurement of other columns. The column “heart_rate” contained only one missing value. Here, the heart rate of a patient with the same height, age and weight was selected to replace the missing value. For columns “P”, “QRST” and “T”, the values seemed to be random, so KNN imputation was used to replace missing values.

Figure 4

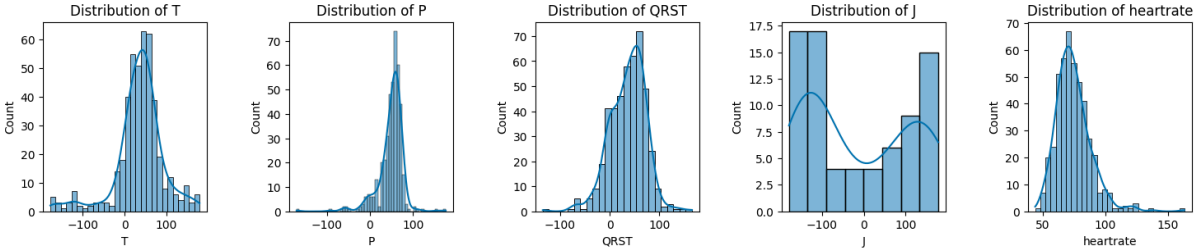
Missing Values



Note. Number of missing values in the dataset, percentages are indicated above the respective bar.

Figure 5

Distribution of Columns with Missing Values



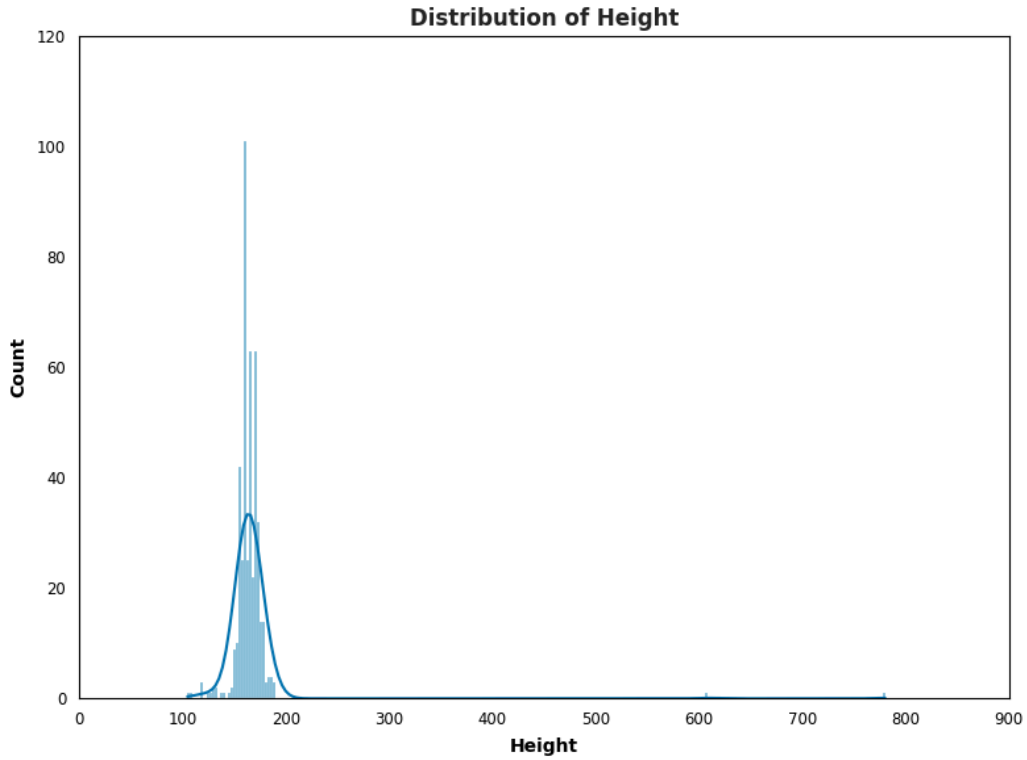
Note. The distribution of the data within the features that had missing data. They do not all follow normal distributions.

Outlier Examination

The column “height” showed two outliers with unrealistic values of 608 cm and 780 cm. Upon closer examination, it shows that the outliers belong to babies (age 1 and age 0), it is thus very likely that the actual heights of them are 60.8 and 78.0 cm. We replaced the outliers respectively with the realistic values.

Figure 6

Height Distribution



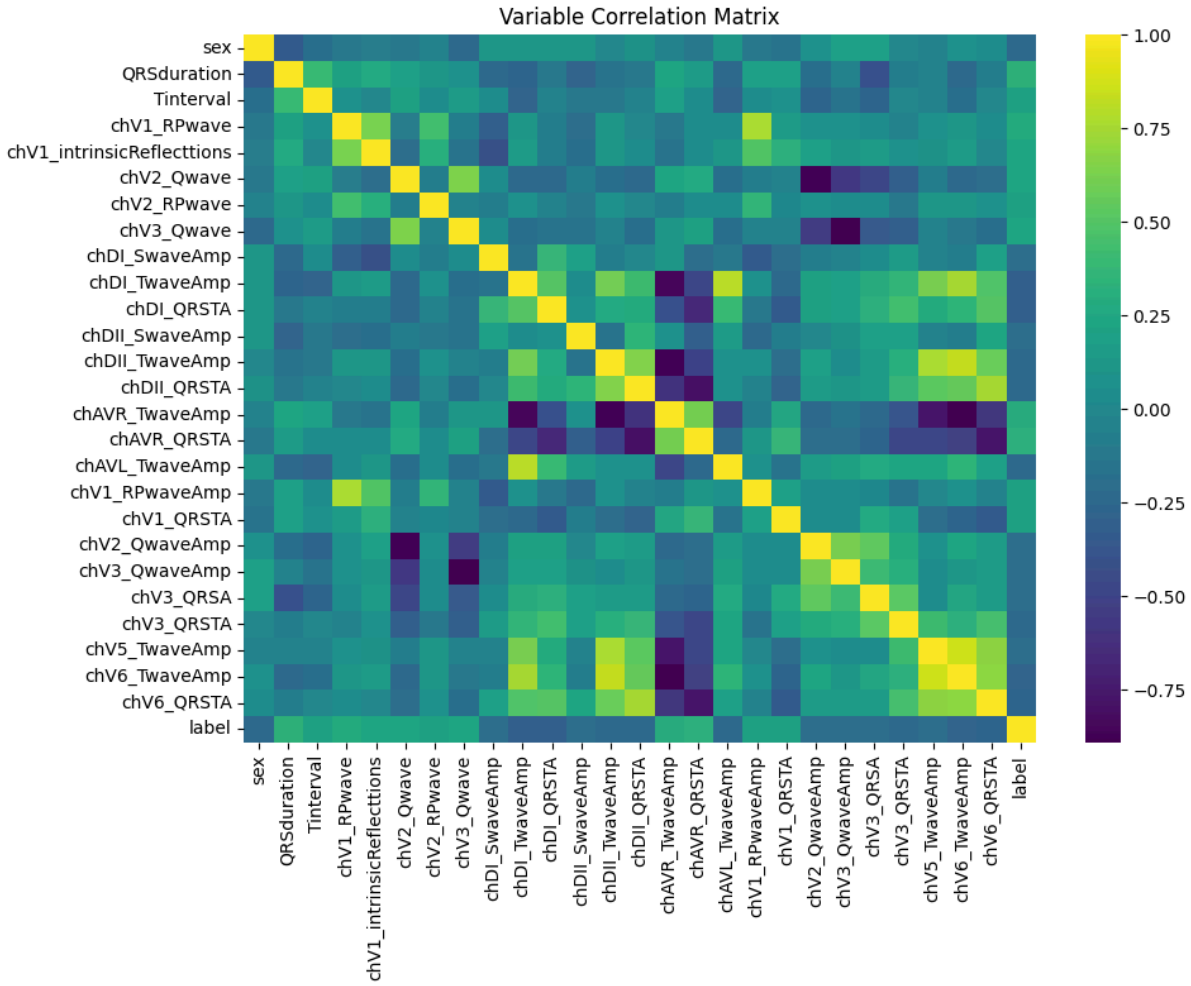
Feature Engineering

Correlations and Dimension Reduction

In order to visualize relevant features, a subset of the dataset was created. Various approaches were conducted to explore potential feature reductions. The first approach was to select only those features with a correlation coefficient greater than 0.2 using the corr function. In Figure 7, the features with a correlation over 0.2 are displayed. Furthermore, a Pearson's Correlation test was conducted, which determined 77 features to be related. The dataset does not contain any qualitative features, so no further data/feature transformation was necessary.

Figure 7

Strongest Correlations to Diagnosis



Note. Correlation Matrix for the features that have a correlation stronger than 0.2 to the feature “label” (Diagnosis). It shows a low-high correlation among features. It shows no single feature with a strong absolute correlation to “label” (Diagnosis) but a few with a medium correlation.

Principal Component Analysis

In addition, we used Principal Component Analysis (PCA) from sklearn.decomposition to reduce the number of features while preserving the most important ones present in the data. In a first trial, we calculated the cumulative variance ratios with different numbers of components on the complete preprocessed dataset, including all features. Explained variance and explained variance ratio are key metrics used in PCA to quantify the amount of information captured by each principal component and to assess the overall effectiveness of dimensionality reduction. Explained variance refers to the amount of variance in the original data that is explained by each principal component. In PCA, the principal components are ordered by the amount of variance they capture. Explained variance ratio is the proportion of the dataset's variance that lies along the axis of each principal component. It is calculated as the ratio of the variance explained by a principal component to the total variance in the dataset. The explained variance ratio provides a normalized measure of the importance of each principal component and can be used to determine how many principal components to retain in order to preserve a certain percentage of the variance in the original data. Our results show that 80 features explain more than 90% of variance (Figure 8) while explained variance ratio for each principal component starts only decreasing minimally after 20 components (Figure 9).

Figure 8

Cumulative Variance Ratio for PCA Trial with different Components

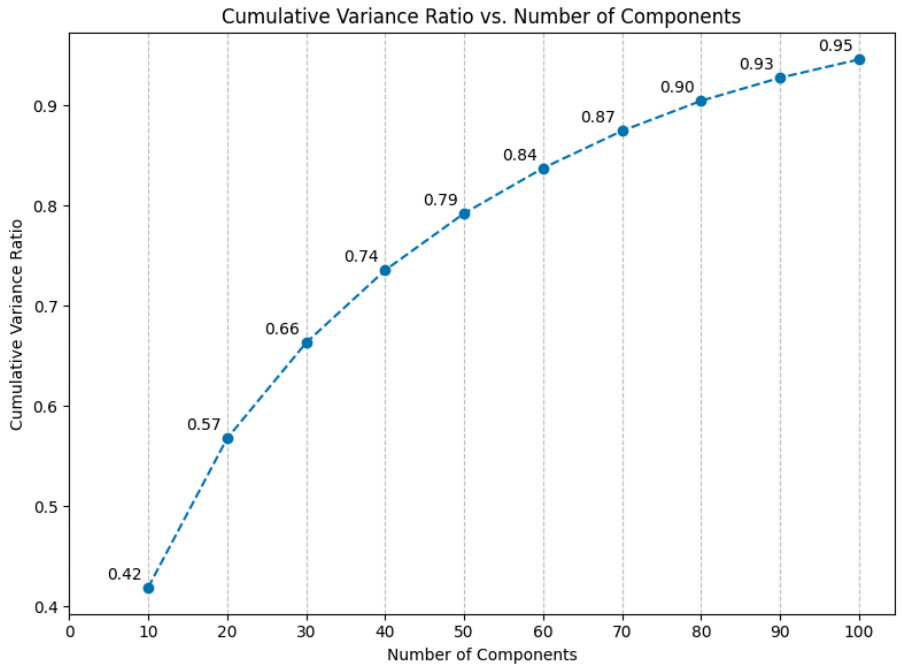
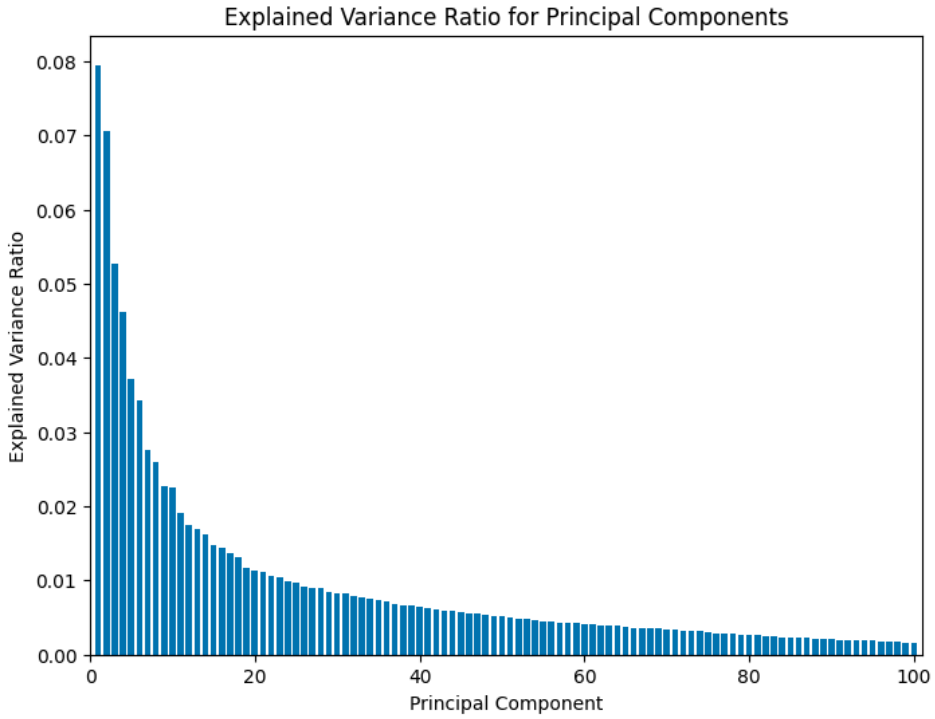


Figure 9

Explained Variance Ratio for individual Principal Components



Data Augmentation

Data augmentation techniques can be particularly useful for improving model performance, especially when dealing with imbalanced datasets or when the available data is limited. The Synthetic Minority Over-sampling Technique (SMOTE) generates synthetic samples for the minority class by interpolating between existing samples and creates new samples along line segments joining k nearest neighbors. Random oversampling is a technique used to balance class distribution in a dataset by randomly duplicating instances from the minority class until it reaches the desired proportion relative to the majority class. We tried the Random Over Sampler as well as SMOTE to create a perfectly balanced dataset. The size of the train data set increased from a shape of [361, 80] to a shape of [402, 80] resampled with Random Over Sampler or SMOTE.

However, since our data set is already sufficiently balanced and those techniques did not improve model performance, we decided not to use them. The main challenge with our data set is its limited size rather than imbalance issues. Thus, we will also look into tools such as Bootstrapping, which involves resampling the dataset with replacement to generate new samples and is commonly used in ensemble methods like bagging. In addition, synthetic data

generation using methods such as Generative Adversarial Networks (GANs) to augment our model training data will be considered.

GAN's are deep learning models that can generate synthetic data samples that closely resemble the real data distribution. The generator network generates new data instances, while the discriminator network evaluates whether the generated data is real or fake. Through iterative training, GAN's learn to generate increasingly realistic data that is indistinguishable from real data, making them a powerful tool for generating synthetic data, image editing, and other creative applications in machine learning and artificial intelligence. However, this dataset proved to be too limited in sample size for the application of GAN's.

Modeling

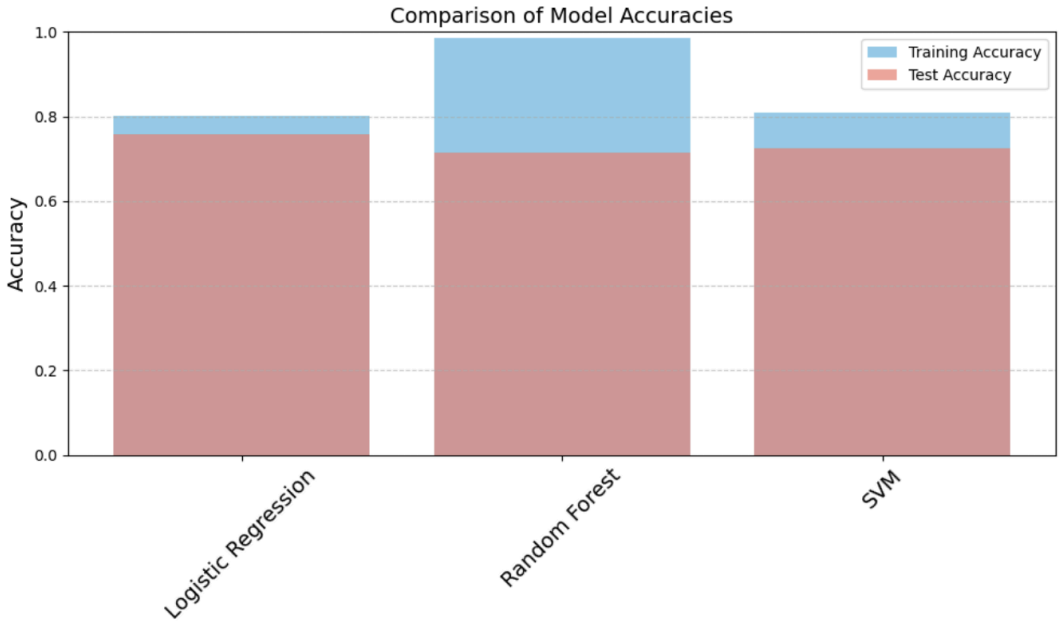
Logistic Regression, Random Forest and Support Vector Machine

In a preliminary analysis, we applied Logistic Regression, Random Forest, and Support Vector Machine (SVM) models to the preprocessed dataset. The approach of initially testing different models (Logistic Regression, Random Forest, SVM) is relevant as it provides a baseline understanding of how different algorithms perform on the dataset. Before modeling, the dataset underwent preprocessing steps including splitting into training and test sets, as well as dimension reduction using Principal Component Analysis (PCA) with 78 components. The hyperparameter selection for each model is listed in Table 1. The best parameters and the performance metrics are given in Table 2.

Logistic Regression and SVM showed similar moderate performance on both training and test datasets, indicating they are not overfitting (Figure 10). Random Forest, however, achieved significantly higher training accuracy compared to test accuracy, suggesting overfitting. This could be due to the complexity of the model or insufficient regularization. Although the model parameters selected via grid search are reasonable, further tuning may be necessary, particularly for Random Forest, to mitigate overfitting. Overall, the approach is relevant for gaining initial insights into model performance, but further refinement and experimentation may be needed to improve accuracy and address potential overfitting issues, particularly with Random Forest.

Figure 10

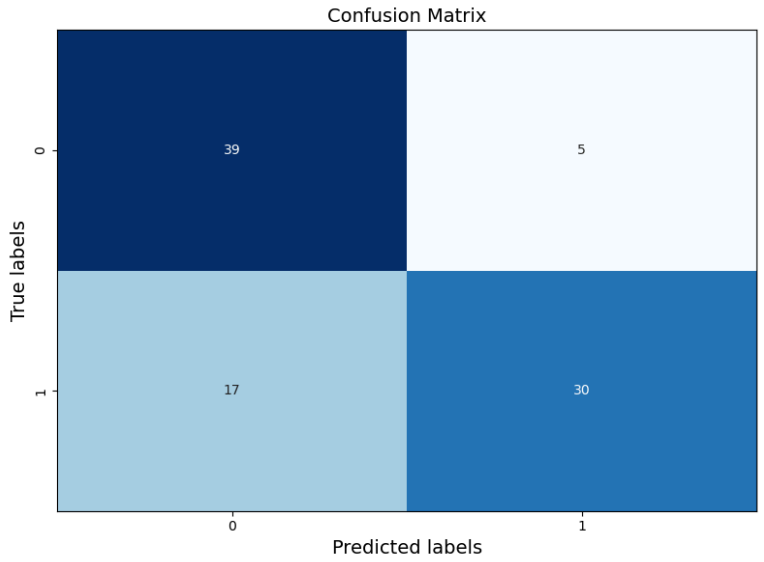
Comparison between Various Models on Reduced Training Dataset



Note. While the Random Forest model showed the highest accuracy score for the training data set, the accuracy with the test data was highest with logistic regression and second highest with SVM.

Figure 11

Confusion Matrix of the Logistic Regression Model with fitted Hyperparameters



Note. The confusion matrix from our Logistic Regression model highlights a concerning trend: a significant number of positive cases (arrhythmia) went undetected. As these cases

hold critical importance for our project, minimizing false negatives becomes paramount. This underscores the necessity for employing advanced modeling techniques and data augmentation strategies.

Adding Complex Models - ElasticNet and Boosting Models

We added additional models including LogisticRegressionCV with ElasticNet, as well as the Boosting Models GradientBoost, AdaBoost and XGBoost. Gradient Boost, XGBoost and ElasticNet were computationally exhaustive and showed no significant improvements compared to displayed models. The hyperparameter grids for GridSearch are listed in Table 1. GradientBoost was clearly overfitting, as it achieved 1.0 score on the training data but a lower score on the test data than several other models. Of the Boosting models, AdaBoost performed best on the test data. The performance metrics are listed in Table 2 and visualized in Figure 12. ROC curves are widely used in binary classification tasks to visualize the trade-off between true positive rate (sensitivity) and false positive rate (1-specificity). As seen in Figure 13, the SVM model maximizes sensitivity while the XGBoost model maximizes specificity. The model with the highest recall with 71 components is XGBClassifier with a recall of 0.66. Using a Pipeline and cross-validation to find the optimal number of PCA with Logistic Regression achieves a recall of 0.72 with 37 components.

Table 1

Hyperparameter Space for each Model

Model	Hyperparameter Space
Logistic Regression	solver: [liblinear, lbfgs]; C: np.logspace(-4, 2, 9)
Random Forest	n_estimators: [10, 50, 100, 250, 500, 1000]; min_samples_leaf: [1, 3, 5]; max_features: [sqrt, log2]
Support Vector	C: np.logspace(-4, 2, 9); kernel: [linear, rbf]
Elastic Net	C: np.logspace(-4, 2, 9); l1_ratio: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]
Gradient Boosting	n_estimators: [50, 100, 200]; learning_rate: [0.01, 0.1, 1.0]; max_depth: [3, 5, 7]
AdaBoost	n_estimators: [50, 100, 200]; learning_rate: [0.01, 0.1, 1.0]
XGBoost	n_estimators: [50, 100, 200]; learning_rate: [0.01, 0.1, 1.0]; max_depth: [3, 5, 7]

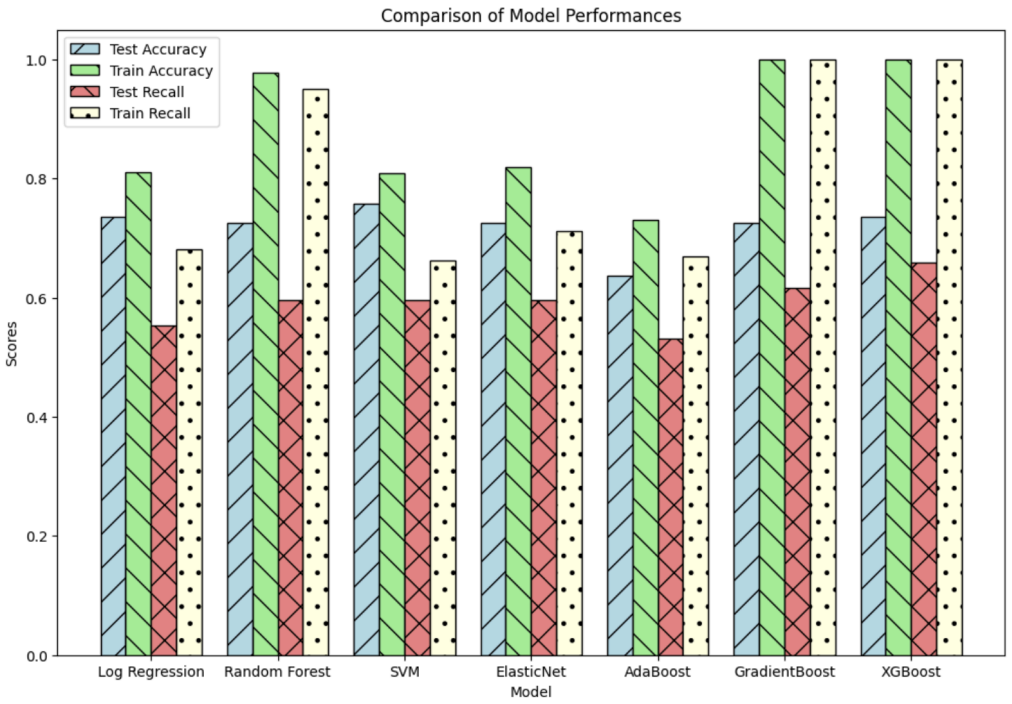
Table 2

Performance Metrics of each Model with the UCI-BILKENT Dataset

Model	Best Hyperparameters	Training Accuracy	Test Accuracy	Test Recall
Logistic Regression	C: 0.0031622776601683794; solver: liblinear	0.80	0.76	0.55
Random Forest	max_features: sqrt; min_samples_leaf: 3; n_estimators: 100	0.99	0.71	0.60
SVM	C: 0.01; kernel: linear	0.81	0.73	0.60
Logistic Regression with ElasticNet	C: 0.1; class_weight: balanced; fit_intercept: True; l1_ratio: 0.9; max_iter: 200; penalty: elasticnet; solver: saga; tol: 0.0001	0.81	0.76	0.60
Gradient Boost	learning_rate: 0.2; max_depth: 3; n_estimators: 100	0.82	0.73	0.62
AdaBoost	base_estimator__max_depth: 2; base_estimator__min_samples_split: 2; learning_rate: 0.05; n_estimators: 150	0.82	0.73	0.53
XGBoost	learning_rate: 0.01; max_depth: 3; n_estimators: 200	0.82	0.73	0.66

Figure 12

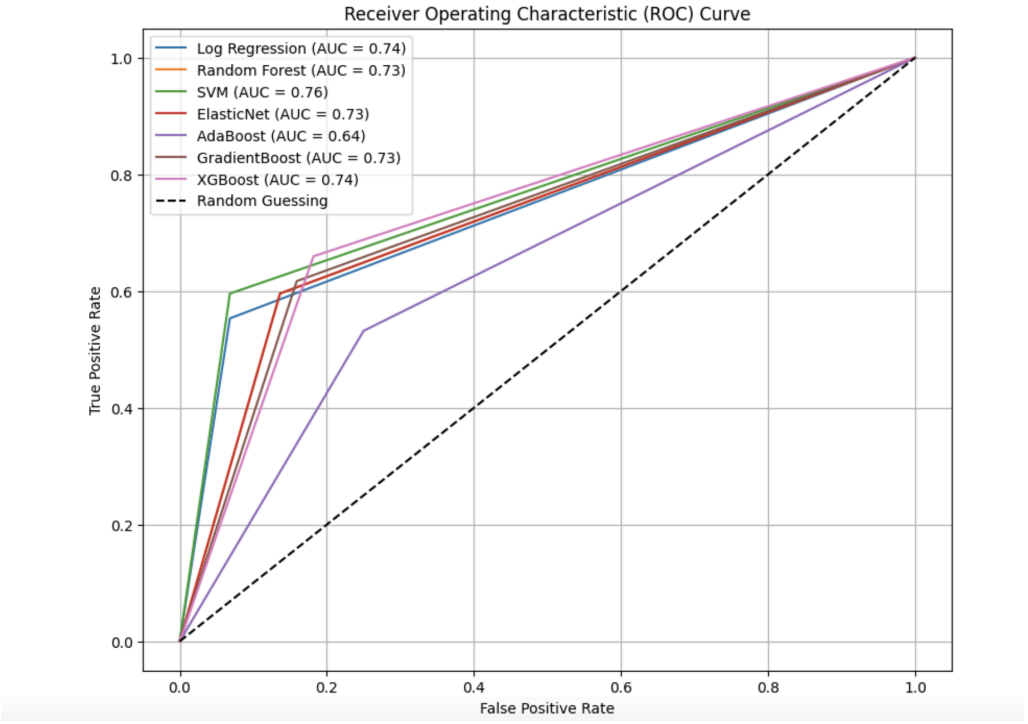
Comparison of Model Performances for Accuracy and Recall



Note. Comparing the performance across models and across the training and testing dataset, we can observe a particularly strong tendency to overfit with the Random Forest, GradientBoost and XGBoost models.

Figure 13

Comparison of ROC Curves for each Model



Note. The ROC shows that all models struggle to increase sensitivity without disproportionately increasing specificity. As our goal is to increase recall, this means that these models will further increase false positives significantly. In addition, the AUC shows us that the overall accuracy across all models is similarly mediocre.

Results

The analysis and modeling conducted on the UCI-Bilkent dataset revealed several key findings regarding the performance of various machine learning algorithms in diagnosing arrhythmia. Among the models evaluated, Logistic Regression with ElasticNet and standard Logistic Regression achieved the highest test accuracy of 76%. These models showed comparable performance to SVM, Gradient Boost, AdaBoost, and XGBoost, all achieving a test accuracy of 73%. On the other hand, Random Forest exhibited the lowest test accuracy at 71%, indicating potential overfitting that requires further parameter tuning. Interestingly, ensemble methods like Gradient Boost, AdaBoost, and XGBoost displayed high training accuracy but did not significantly improve generalization performance compared to simpler models like SVM and Logistic Regression. This suggests that despite their complexity, these

ensemble methods did not provide substantial improvements in model performance on this dataset.

The dimensionality reduction technique using Principal Component Analysis (PCA) with 78 components proved effective in reducing the dataset's dimensionality while preserving more than 90% of the variance. However, further exploration and refinement are necessary to address potential overfitting in models like Random Forest and to optimize hyperparameters for Gradient Boosting models. Data augmentation techniques such as Synthetic Minority Over-sampling Technique (SMOTE) and Random Over Sampler did not yield improvements in model performance due to the dataset's already balanced nature. This highlights the challenge posed by the limited size of the dataset rather than issues related to class imbalance. In conclusion, while Logistic Regression models demonstrated the best overall performance on the UCI-Bilkent dataset, ongoing refinement and experimentation are crucial to optimize accuracy and mitigate overfitting

In the following steps, we will apply the insights gained from analysis and modeling of the UCI-Bilkent dataset to the MIT-BIH dataset. We decided to continue with another dataset to test out deep learning models. The MIT-BIH dataset contains a sufficient number of samples to be used for advanced models such as neural networks. We start by applying machine learning models to the dataset and then continue with deep learning.

MIT-BIH Database

Exploratory Data Analysis

The MIT-BIH Arrhythmia Database consists of 48 half-hour segments of two-channel ambulatory ECG recordings obtained from 47 subjects during the years 1975 to 1979 by the BIH Arrhythmia Laboratory (Moody & Mark, 2001). Among these, 23 segments were chosen randomly from a pool of 4000 24-hour ambulatory ECG recordings at Boston's Beth Israel Hospital, while the remaining 25 were specifically selected to encompass less common yet clinically significant arrhythmias. These recordings were digitized at a rate of 360 samples per second per channel, with an 11-bit resolution covering a range of 10 mV. Each record underwent independent annotation by two or more cardiologists, with any discrepancies resolved to generate computer-readable reference annotations, totaling approximately 110,000 beats. The dataset is freely available on PhysioNet (Goldberger et al., 2000).

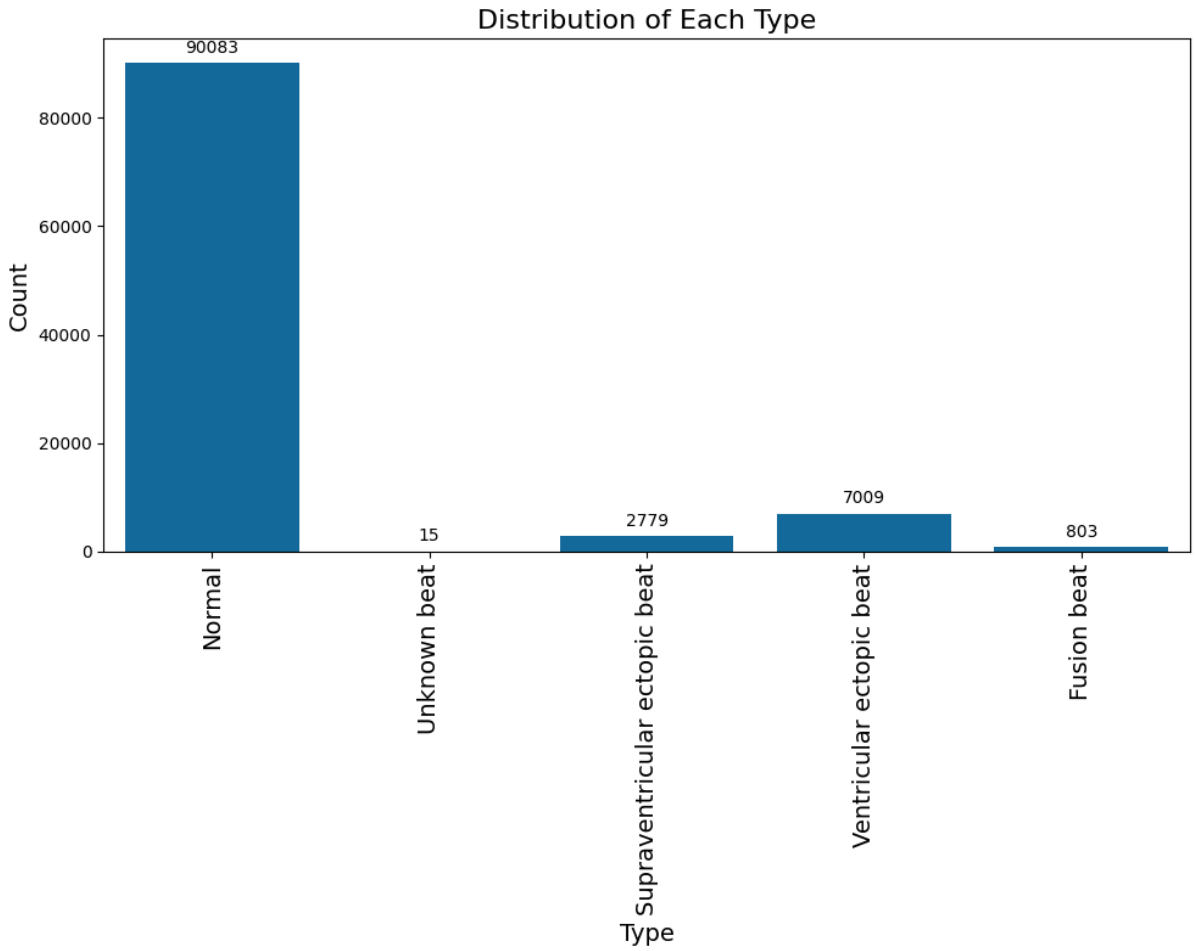
The dataset used in this report already underwent feature engineering (Sakib, Fouda & Fadlullah, 2021). It consists of 100689 samples and 34 columns, with 33 features pertinent to electrocardiogram (ECG) analysis and one target column denoting arrhythmia classifications. The initial column ‘record’ serves as an identifier, denoting the name of the subject or patient. The target column ‘type’ contains several classes, each indicative of distinct cardiac conditions. These classes include N for Normal, SEB for Supraventricular ectopic beat, VEB for Ventricular ectopic beat, F for Fusion beat, and Q for Unknown beat. The allocation of these classes is encapsulated within the "type" column, offering a comprehensive overview of the cardiac conditions represented in the dataset. The other features are split across two leads, namely lead-II and lead-V5, each lead comprises 17 features, collectively contributing to the characterization of cardiac rhythms and morphologies:

- RR Intervals: Encompassing metrics such as Average RR, RR, and Post RR intervals, this category offers insights into the temporal dynamics of cardiac activity.
- Heartbeat Intervals: This section delineates features including PQ Interval, QT Interval, ST Interval, and QRS Duration, offering insights into the temporal and electrical characteristics of heartbeats.
- Heartbeats Amplitude Features: Spanning parameters such as P peak, T peak, R peak, S peak, and Q peak, this category provides valuable information regarding the amplitude of key cardiac signals.
- Morphology Features: Comprising attributes such as QRS morph feature 0 through QRS morph feature 4, this category delves into the nuanced morphological aspects of cardiac waveforms.

After examining the dataset for null values and missing data, we found none. Furthermore, the dataset displayed no noticeable outliers.

Figure 14

Distribution of each Type of Heartbeat



Note. This first visualization of the distribution of each type in the dataset indicates a class imbalance towards normal cases of ECG recordings.

Pre-Processing

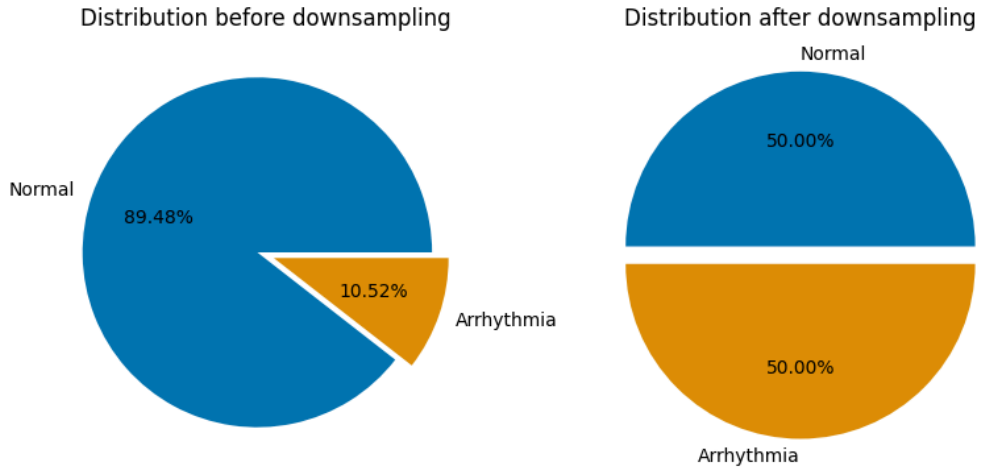
Before proceeding with data augmentation, the dataset underwent several preprocessing steps. Due to class imbalance, a binary target variable was created for normal heartbeat (label 0) and for the types of abnormal heartbeats (label 1), including the unknown beats type. We opted to convert all data, including instances labeled as 'Unknown beat,' into binary format. This decision was made due to the limited sample sizes for certain types and the 'Unknown type' category. We included the unknown beats in the abnormal heartbeat variable in order to decrease the number of false negatives, since we rather want a false alarm then an undetected case of arrhythmia. Our objective is to detect any abnormal ECG signals comprehensively, ensuring that physicians can review all potential anomalies identified by our application. The columns ‘record’ and ‘type’ were then dropped from the dataset.

Subsequent analysis revealed a notable class imbalance in the distribution of types within the dataset (Figure 15).

Subsequently, feature scaling was performed using MinMaxScaler(). Due to the large sample size of the MIT-BIH dataset, we strategically downsampled our data to the minority class of abnormal heartbeat, comprising all heartbeat conditions except normal heartbeat, resulting in 10606 samples. This downsampling technique allowed us to create an evenly balanced dataset with a total of 21212 samples, ensuring that both normal and abnormal heartbeat categories were well represented in our modeling (see Figure 15). After the downsampling, the dataset was partitioned into training and testing sets. For the Deep Learning experiments, we conducted an additional downsampling procedure to further optimize computational efficiency while retaining sufficient data points to a total sample size of 10000.

Figure 15

Data Distribution before and after Downsampling



Modeling

We employed both basic classification models and more sophisticated ones on the MIT-BIH dataset, mirroring the procedures used for the UCI-Bilkent dataset. We found out that RandomSearch is in our case a better choice to GridSearch, since the model performances and best hyperparameters keep relatively stable and computation time is significantly reduced to a few minutes per model. The hyperparameter space for each model was selected based on Table 1, which showed good results in previous modeling. The best hyperparameters and performance metrics are listed in Table 3 and visualized in Figure 16.

All models performed very well, while logistic regression and elastic net performed slightly worse than the other models. The model performances indicate that Logistic Regression achieved high accuracy (0.88) and a balanced precision-recall trade-off (0.90 and 0.87, respectively), with F1 and AUROC scores of 0.88 and 0.88. Random Forest Classifier exhibited exceptional accuracy (0.97) and strong precision-recall metrics (0.98 and 0.98), resulting in F1 and AUROC scores of 0.98 and 0.98. Elastic Net showed relatively lower performance with an accuracy of 0.86, precision of 0.91, and recall of 0.80, resulting in F1 and AUROC scores of 0.85 and 0.86, respectively. Gradient Boosting displayed high accuracy (0.98) and balanced precision-recall (0.98 and 0.98), yielding F1 and AUROC scores of 0.98 and 0.98. AdaBoost and XGBoost achieved the highest accuracies (0.98 and 0.98, respectively) with good precision-recall balances (0.95 and 0.94 for AdaBoost and 0.98 and 0.98 for XGBoost). AdaBoost has an F1 and AUROC score of 0.95 and XGBoost has an F1 and AUROC score of 0.98.

Furthermore, to evaluate the discrimination ability of our models, we generated Receiver Operating Characteristic (ROC) curves for each model. Figure 17 illustrates the ROC curves for the different models evaluated on the MIT-BIH dataset. The curves showcase the varying performance of each model in distinguishing between normal and abnormal heartbeat classes, with Random Forest, Gradient Boosting, and XGBoost exhibiting superior discrimination capabilities compared to Logistic Regression, Elastic Net, and AdaBoost. The area under the ROC curve (AUROC) further quantifies this discrimination ability, where models with AUROC values closer to 1 indicate better classification performance. Our analysis of the ROC curves aligns with the performance metrics listed in Table 3, confirming the effectiveness of Random Forest, Gradient Boosting, and XGBoost as the top-performing models for arrhythmia detection in our study.

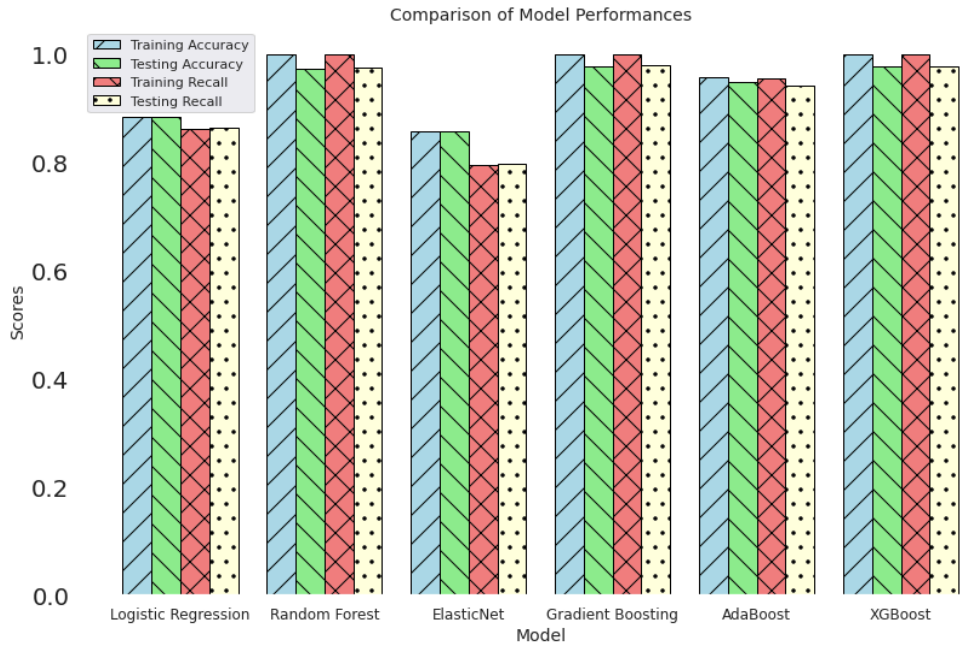
Table 3

Performance Metrics of each Model for the MIT-BIH Dataset

Model	Best Parameters	Training Accuracy	Test Accuracy	Precision	Recall	F1 Score	Auroc Score
Logistic Regression	C: 3.162; penalty: l2	0.88	0.88	0.90	0.87	0.88	0.88
Random Forest	max_features: log2; min_samples_leaf: 1; n_estimators: 250	1.00	0.97	0.98	0.98	0.98	0.98
Elastic Net	C: 0.01; l1_ratio: 0.4; max_iter: 1000; penalty: elasticnet; solver: saga	0.86	0.86	0.91	0.80	0.85	0.86
Gradient Boosting	learning_rate: 0.1; max_depth: 7; n_estimators: 200	1.00	0.98	0.98	0.98	0.98	0.98
Ada Boosting	learning_rate: 1.0; n_estimators: 200	0.96	0.95	0.95	0.94	0.95	0.95
XG Boosting	learning_rate: 1.0; max_depth: 7; n_estimators: 200	1.00	0.98	0.98	0.98	0.98	0.98

Figure 16

Comparison of Model Performances



Note. Random Forest, Gradient Boosting, and XGBoost are the most effective among the models assessed. The corresponding performance metrics are listed in Table 3.

Figure 17

ROC Curves for the different Models

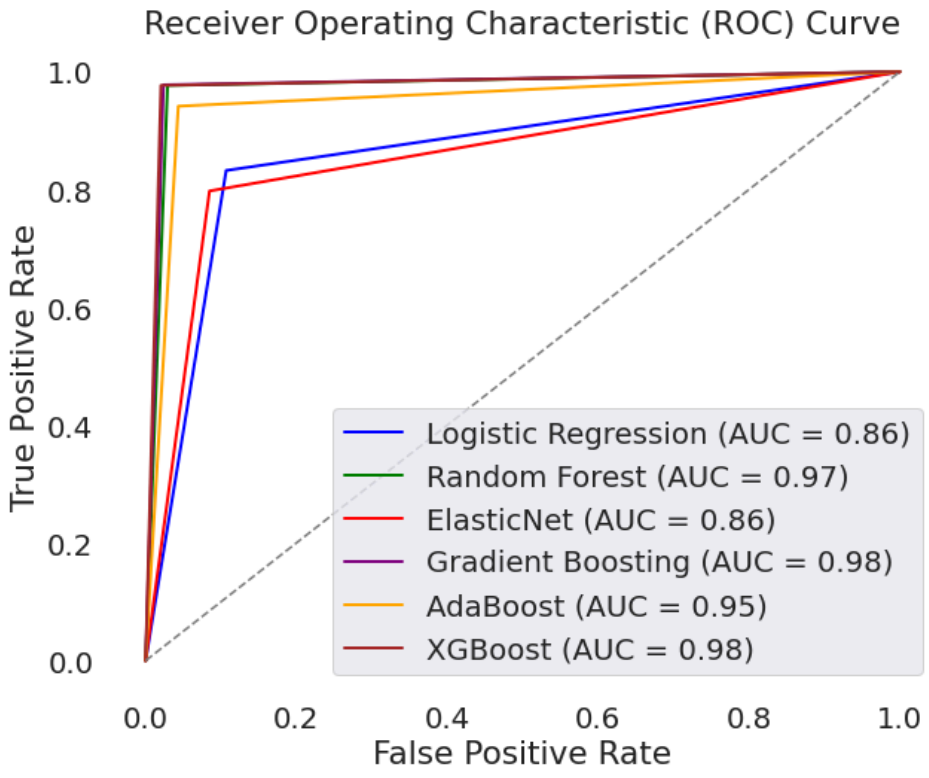
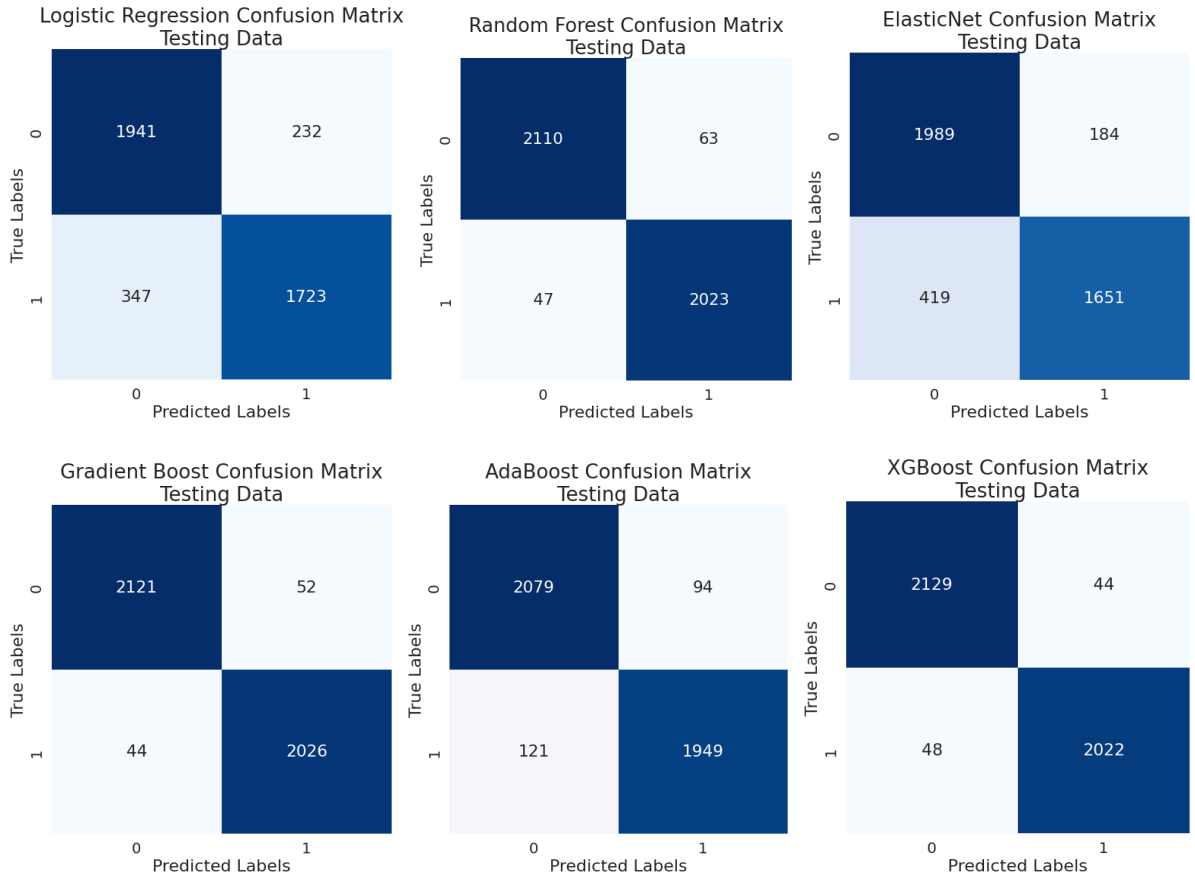


Figure 18

Comparison of Confusion Matrices for each Model



Note. The illustration of confusion matrices reveals a relatively small number of false negatives for Random Forest (47), Gradient Boost (44), and XGBoost (48).

Deep Learning

Transitioning to the deep learning phase, we conducted an additional downsampling procedure to optimize computational efficiency while retaining sufficient data points. Our machine learning models ran relatively fast and performed very well with the sample size of 21212. However, to save computation time for the neural networks, we decided to downsample once more to a total sample size of 10000, with 5000 data points for each condition of normal and abnormal heartbeat, striking a balance between computational cost and model training effectiveness.

Dense Neural Network

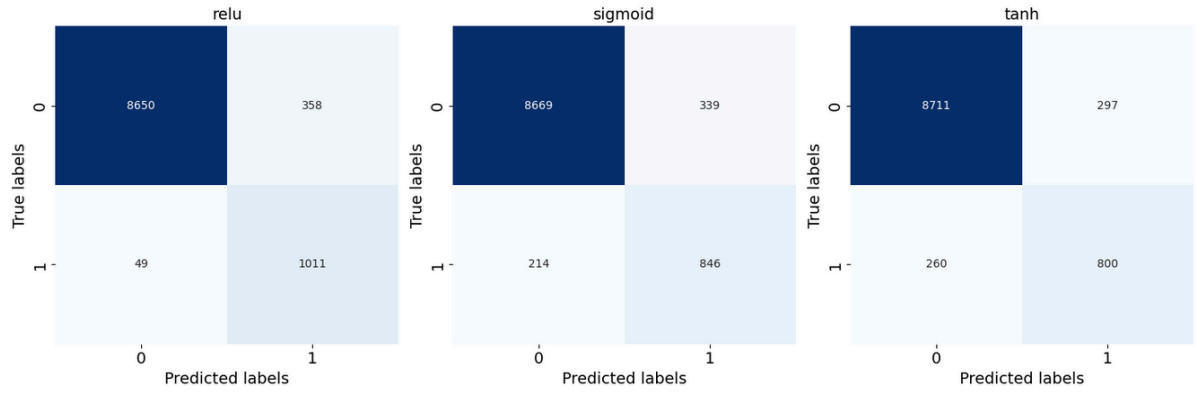
In the initial trial, we employed a neural network architecture comprising four dense layers, where the input layer accommodates inputs with a shape of (None, 32) corresponding to our dataset dimensions. The selected architecture features three hidden dense layers with

10, 8, 6, and 3 neurons, respectively, resulting in a total of 493 trainable parameters in the model. With 500 epochs, the model has ample opportunities to converge and potentially discern intricate data patterns. The learning rate, regulated by the adaptive Adam optimizer, facilitates stable and efficient weight updates throughout training. These hyperparameters are for this first trial through theoretical considerations to optimize model performance for the binary classification task at hand, while guarding against common pitfalls like underfitting and overfitting. We evaluated the model performance using three different activation functions: Tanh, ReLu, and Sigmoid, applied to each dense layer. Activation functions are pivotal in introducing non-linearity to the decision-making process of neural networks. Tanh, or hyperbolic tangent, maps inputs to a range between -1 and 1, suitable for hidden layers due to its capability to capture both positive and negative values. ReLU, or Rectified Linear Unit, is a simple yet effective function that outputs the input directly if positive, facilitating the model's ability to discern complex patterns. Sigmoid, a logistic function, compresses inputs to a range between 0 and 1, often utilized in binary classification tasks to generate probability-like outputs.

The results show that the ReLu activation function yielded the highest overall accuracy of 95.47%, with high precision (0.994) and recall (0.955) for class 0, and moderate precision (0.714) and recall (0.950) for class 1. Conversely, Tanh exhibited lower accuracy at 87.26% and reduced recall metrics, while Sigmoid achieved an accuracy of 95.02% with slightly lower recall compared to ReLu (Table 4). The best performing DNN model had 53 false negatives, representing undetected cases for the downsampled dataset, which accounts for approximately 0.58% of the total cases.

Figure 19

Comparison of Confusion Matrices for DNN with different activation functions



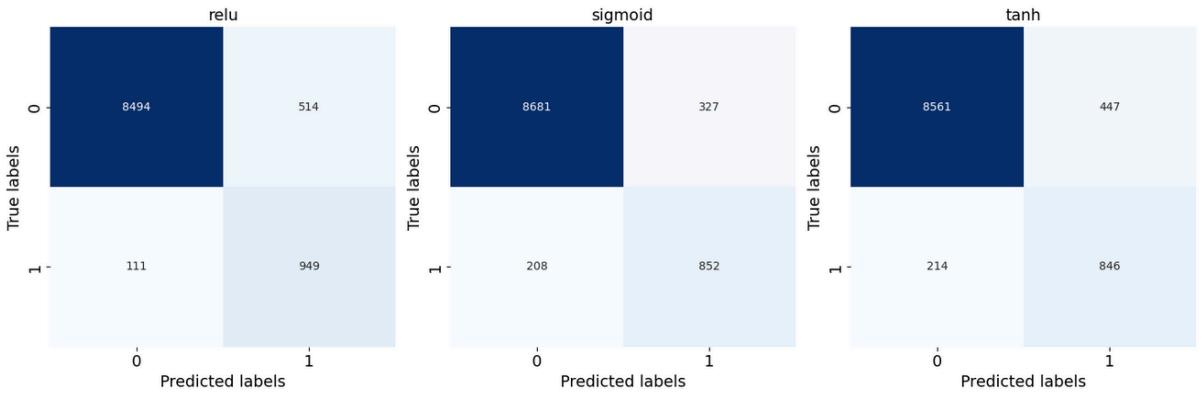
Artificial Neural Network

We implemented an artificial neural network (ANN) architecture that comprises three hidden dense layers, aiming to capture intricate data patterns while mitigating overfitting risks. Each hidden layer consists of four neurons and uses a defined activation function. We compared the performances of the ReLu, Tanh and Sigmoid activation functions. The output layer has one neuron with a Sigmoid activation function, suitable for binary classification tasks. With a training duration of 500 epochs, the model iterates through the dataset multiple times, allowing for comprehensive convergence and learning. The adaptive Adam optimizer dynamically regulates the learning rate, ensuring stable and efficient weight updates throughout training. These hyperparameters are selected through theoretical considerations, seeking an optimal balance between model complexity, training duration, and optimization strategy to achieve robust performance, whilst also considering computational costs.

ANN with the ReLu activation function achieved an accuracy of 93.79% with commendable recall values for both classes, outperforming Tanh which attained an accuracy of 93.43% and moderate recall. The Sigmoid activation function showed the highest accuracy with 94.69% and relatively high recall, particularly for class 1 (Table 4). Overall, the number of false negatives is slightly higher for the best performing ANN in terms of recall than for DNN, with 111 undetected cases, which is equivalent to approximately 10.47%.

Figure 20

Comparison of Confusion Matrices for ANN with different activation functions



Our results indicate that the DNN outperforms the ANN across most metrics. Specifically, DNN models with ReLu and Sigmoid activation functions exhibit higher precision, recall, and F1-scores compared to their ANN counterparts. Additionally, DNN models achieve higher accuracy, macro-average, and weighted-average scores, indicating

better overall performance in classification tasks. When comparing the ROC curves and Precision-Recall curves for both DNNs and ANNs, we see that the DNNs exhibit curves that are closer to the top-left corner, indicating better trade-offs between true positive rate (Recall) and false positive rate (FPR) (Figure 21). Similarly, in the Precision-Recall curves, DNNs results reflect better precision-recall trade-offs (Figure 22). Notably, the tanh activation function yields lower performance in both DNN and ANN models, suggesting its suboptimal suitability for this dataset. Overall, these findings emphasize the effectiveness of DNN architectures, making them a more suitable choice for the classification task in this dataset.

Figure 21

ROC Curves for DNN and ANN Trials with different Activation Functions

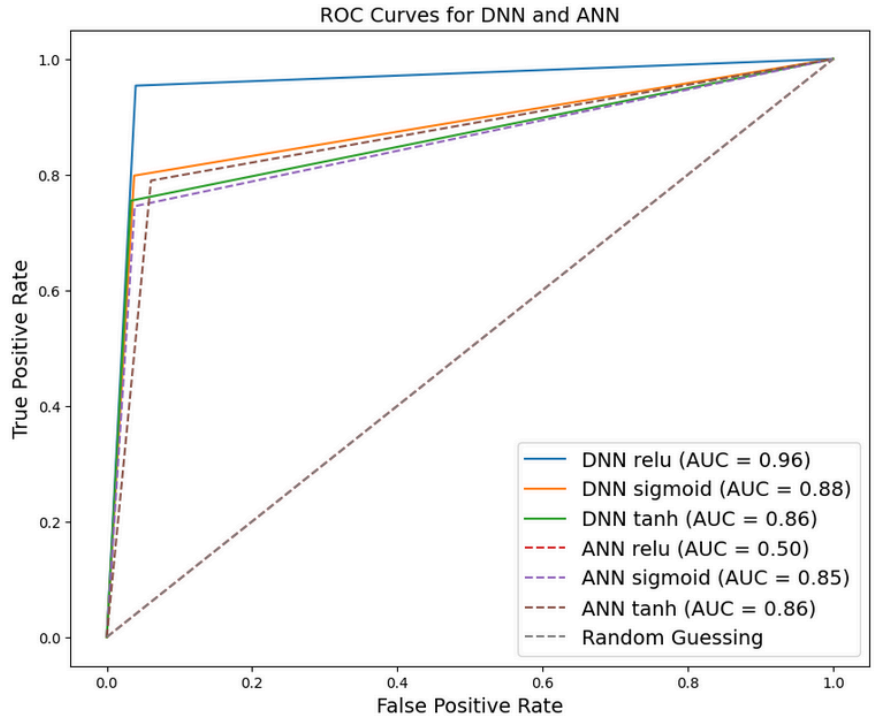


Figure 22

Precision-Recall Curves for DNN and ANN Trials with different Activation Functions

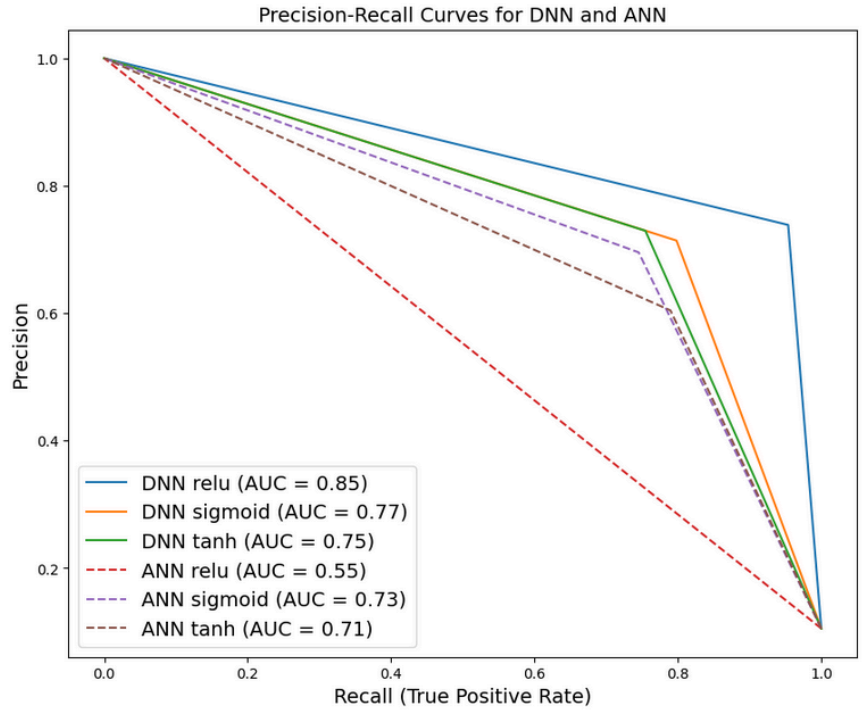


Table 4

Performance Metrics of DNN and ANN with different Activation Functions

Dense Neural Network									
Activation Function	Precision (0)	Precision (1)	Recall (0)	Recall (1)	F1-Score (0)	F1-Score (1)	Accuracy	Macro Avg	Weighted Avg
relu	0.99	0.71	0.96	0.95	0.97	0.82	0.95	0.85	0.96
tanh	0.99	0.45	0.86	0.94	0.92	0.61	0.87	0.72	0.93
sigmoid	0.99	0.69	0.95	0.94	0.97	0.80	0.95	0.84	0.96
Artificial Neural Network									
Activation Function	Precision (0)	Precision (1)	Recall (0)	Recall (1)	F1-Score (0)	F1-Score (1)	Accuracy	Macro Avg	Weighted Avg
relu	0.99	0.65	0.94	0.90	0.96	0.75	0.94	0.82	0.95
tanh	0.98	0.65	0.95	0.80	0.96	0.72	0.93	0.81	0.94
sigmoid	0.98	0.72	0.96	0.80	0.97	0.76	0.95	0.85	0.95

Neural Network Architecture Exploration for Improved Performance

In the subsequent step, we chose the best performing neural network architecture for our application, the DNN with ReLu activation function, for further model refinement and

hyperparameter tuning trials, i.e. by increasing the number of hidden layers and adjusting the units per layer.

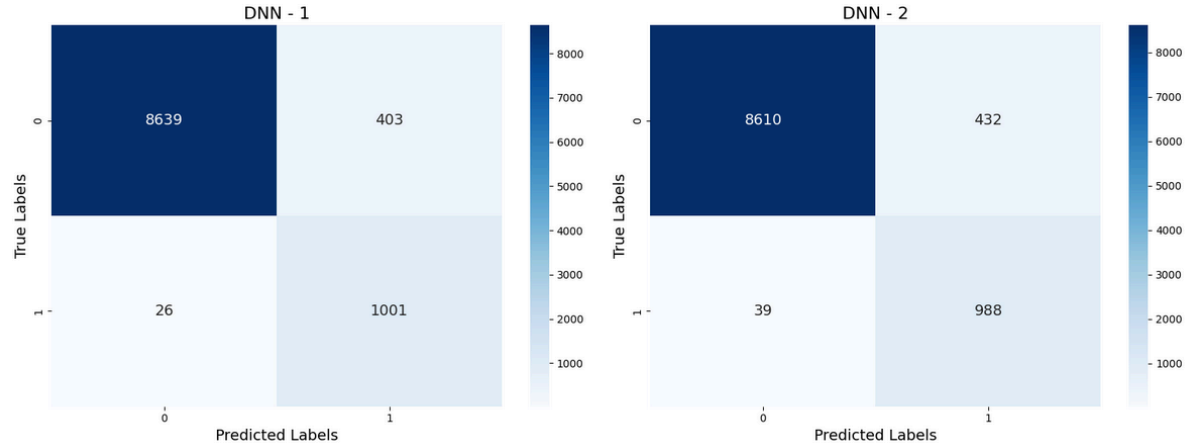
In the first model setup, a sequential deep neural network (DNN) is constructed with an architecture featuring four dense layers. The input layer consists of 64 units with ReLU activation, followed by a hidden layer comprising 32 units also with ReLU activation. To mitigate overfitting, a dropout regularization of 20% is applied before the subsequent hidden layer, which contains 16 units with ReLU activation. The output layer, responsible for binary classification, consists of a single unit with Sigmoid activation. This model is trained using the Adam optimizer with a binary cross-entropy loss function, for 500 epochs with a batch size of 100. Upon evaluation, the first model achieves an overall accuracy of 96%, with a precision of 71% and recall of 97% for the positive class (1), indicating a strong ability to correctly classify instances of this class. However, it also exhibits a non-negligible false positive rate, as evidenced by the precision of 100% and recall of 96% for the negative class (0). This suggests a potential imbalance in the model's ability to discriminate between the two classes (see Figure 23).

In the second model setup, the architecture is expanded to include additional hidden layers, with the aim of further capturing complex patterns in the data. The input layer remains the same, with 128 units and ReLU activation. Following this, two more hidden layers are introduced, each comprising 64 and 32 units, respectively, all with ReLU activation. Dropout regularization is applied with a dropout rate of 30% before the first additional hidden layer. The model is then further augmented with two more hidden layers, containing 16 and 8 units, both utilizing ReLU activation. Finally, the output layer remains unchanged, with a single unit and Sigmoid activation. The training process and evaluation metrics remain consistent with the first model. Upon evaluation, the second model achieves an overall accuracy of 95%, indicating a slight decrease compared to the first model. While the recall for the positive class remains high at 96%, the precision drops to 70%, indicating a higher false positive rate. This could indicate a trade-off between the increased complexity of the model and its ability to generalize effectively to unseen data. Additionally, the increased number of parameters in the second model may contribute to a higher risk of overfitting, as evidenced by the slight decrease in performance compared to the first model. In summary, while the expansion of the model architecture in the second setup offers the potential to capture more intricate patterns in the data, it also introduces challenges such as increased model complexity and potential overfitting. Further experimentation and fine-tuning of hyperparameters may be

necessary to strike a balance between model complexity and performance in this classification task.

Figure 23

Confusion Matrices for the two DNN Architectures (DNN-1 with 2 Hidden Layers and DNN-2 with 4 Hidden Layers)



In addition, we conducted experiments with different learning rates (0.1, 0.01, and 0.001) instead of using the default Adam optimization algorithm. Experimenting with various learning rates allows for finding the optimal balance between convergence speed and stability during training. A learning rate of 0.01 is commonly considered a starting point, as it strikes a balance between convergence speed and stability. However, higher learning rates such as 0.1 may lead to faster convergence but with potential instability due to large updates to model parameters. Conversely, lower learning rates like 0.001 offer better stability but slower convergence. The choice of learning rate depends on the specific characteristics of the dataset and the complexity of the model. In our case, these experiments did not yield better results than using the default Adam optimizer. Although the smallest learning rate, 0.001, typically showed better results than the higher learning rates (0.01 or 0.1), the overall performance did not surpass that of the default Adam optimizer. This suggests that for our dataset and model architecture, the default Adam optimizer provided the most effective optimization strategy, balancing convergence speed and stability.

Results and Conclusions

Based on our results, minimizing false negatives is crucial for our project's success. Gradient Boosting achieved the best performance among the models evaluated concerning the number of false negatives, with a very high accuracy of 98%. Additionally, Random Forest exhibited strong performance with a 97% accuracy, while AdaBoosting and XGBoosting achieved accuracies of 98%. For being the weakest model, Logistic Regression also performed well with an 88% accuracy and balanced precision, recall, and F1 scores.

Comparing deep learning models to traditional machine learning algorithms, DNN and ANN models achieved respectable accuracies ranging from 95% to 96%. While the DNN with two hidden layers in the refinement trials also achieved a very low number of false negatives (lowest among all models), the number of false positives for this model was relatively high. Overall, they were outperformed by simpler models like ensemble methods such as Random Forest and Gradient Boosting. One possible explanation for this discrepancy could be the dataset's size and complexity. Deep learning models often require large datasets to learn complex patterns effectively, while traditional algorithms can perform well even with smaller datasets. Moreover, deep learning models incur higher computational costs and require extensive hyperparameter tuning, as evidenced by our refinement trials with DNN. The refinement trials have illustrated the promise of leveraging deep learning for classification tasks, including the one at hand.

Further refinement through hyperparameter tuning and exploration of advanced deep learning methodologies, such as encoding-decoding techniques, holds potential for optimizing this application, given additional time and resources within a larger project framework. We could also experiment with the use of transfer-learning, leveraging pre-trained neural networks could result in better performance, specifically for smaller datasets. Through this or by expanding the dataset, more diverse heartbeat classifications are possible, such as fusion beats, supraventricular ectopic beats, and ventricular ectopic beats. More granular classification and detection capabilities contribute to a comprehensive arrhythmia diagnosis system. Additionally, exploring different sampling methods, such as oversampling techniques or stratified sampling, could enhance model training and improve the generalization of neural networks.

As for the application of models like ours, they hold significant potential for deployment in clinical settings such as hospitals and healthcare facilities. The robust performance of models like Gradient Boosting, XGBoost, and Random Forest, coupled with

their ability to minimize false negatives, makes them valuable tools for assisting healthcare professionals in arrhythmia diagnosis. Moreover, these models could be integrated into training programs for medical students and healthcare practitioners to improve their understanding and proficiency in identifying cardiac arrhythmias. We can also provide our models to others, so they can leverage transfer learning to train models for similar use cases such as veterinary cardiology. In terms of practical applications, arrhythmia detection models can be integrated into wearable devices for remote monitoring of patients' cardiac health (Huda et al., 2020). These devices are portable ECG monitors, but with advanced technology, even gadgets like smartwatches might be able to play a role in the prevention of cardiac disorders.

References

Das, M. K., & Ari, S. (2014). ECG Beats Classification Using Mixture of Features. *International Scholarly Research Notices*, *2014*, 178436. https://doi.org/10.1155/2014/178436

Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P. C., Mark, R., & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation, 101*(23), 215–220. doi: 10.1161/01.cir.101.23.e215.

Guvenir, H. Altay, Burak Acar, Gulsen Demiroz, Ayhan Cekin. (1997). A Supervised Machine Learning Algorithm for Arrhythmia Analysis. *Proceedings of the Computers in Cardiology*, 433-436.

Huda, N., Khan, S., Abid, R., Shuvo, S. B., Labib, M. M., & Hasan, T. (2020). A Low-cost, Low-energy Wearable ECG System with Cloud-Based Arrhythmia Detection. *MedRxiv.* https://doi.org/10.1101/2020.08.30.20184770

Moody, G. B., & Mark, R. G. (2001). The impact of the MIT-BIH Arrhythmia Database. *IEEE Engineering in Medicine and Biology Magazine, 20*(3), 45-50.

Mincholé, A., Camps, J., Lyon, A., & Rodríguez, B. (2019). Machine learning in the electrocardiogram. *Journal of Electrocardiology*, *57*, 61–64. https://doi.org/10.1016/j.jelectrocard.2019.08.008

Lawless, S. T. (1994). Crying wolf: False alarms in a pediatric intensive care unit. *Crit. Care Med*, *22*, 981–985. PMID: 8205831

Lome, S. *ECG Basics*. Learn the Heart. https://www.healio.com/cardiology/learn-the-heart/ecg-review/ecg-interpretation-tutorial/qrs-complex

Ramirez-Santana, M. (2018). Limitations and Biases in Cohort Studies. *InTech*. doi: 10.5772/intechopen.74324

Sakib, S., Fouda, M. M., & Fadlullah, Z. M. (2021). Harnessing Artificial Intelligence for Secure ECG Analytics at the Edge for Cardiac Arrhythmia Classification. *Secure Edge Computing, 1*, 137-153. CRC Press. ISBN: 9781003028635.

Tachycardia (Fast Heart Rate) in Children. Ann & Robert H. Lurie Children's Hospital of Chicago. https://www.luriechildrens.org/en/specialties-conditions/tachycardia/

Tan, K. F., Chan, K. L., & Choi, K. (2000). Detection of the QRS complex, P wave and T wave in electrocardiogram. In *2000 First International Conference Advances in*

Medical Signal and Information Processing (IEE Conf. Publ. No. 476) (pp. 41-47). IET.

Wang, D., Zhang, L., Wang, Z.-Y., Zhang, Z.-Y., Wang, Y. (2017). The missed diagnosis of aortic dissection in patients with acute myocardial infarction: A disastrous event. *Journal of Thoracic Disease, 9*(7), E636-E639. https://doi.org/10.21037/jtd.2017.06.103