A

STAGE 2 REPORT

ON

DETECTION OF ARRHYTHMIA IN COVID-19 PATIENTS USING SUPERVISED LEARNING METHODS

Submitted in partial fulfillment of the requirement of University of Mumbai
For the Degree of

Master of Engineering
in
Information Technology

by

Mr. Yashodhan Ketkar

Under the Guidance of

Dr. Sushopti Gawade

DEPARTMENT OF INFORMATION TECHNOLOGY (PG)

**PILLAI COLLEGE OF ENGINEERING
NEW PANVEL - 410206**

UNIVERSITY OF MUMBAI

Academic Year 2020-21

DISSERTATION APPROVAL CERTIFICATE

This is to certify that the dissertation work entitled **"Detection of Arrhythmia in COVID-19 Patients Using Supervised Learning Methods"**, for **M.E. (Information Technology)** submitted to University of Mumbai by Student name, a bonafide student of Pillai College of Engineering, New Panvel has been approved for the award of Master of Engineering Degree in Computer Engineering.

Examiners:

Internal Examiner   **External Examiner**

(Signature)   (Signature)

Name: Name:

Date: Date:

DEPARTMENT OF INFORMATION TECHNOLOGY (PG)

PILLAI COLLEGE OF ENGINEERING
NEW PANVEL - 410206
UNIVERSITY OF MUMBAI

Academic Year 2020-21

DEPARTMENT OF INFORMATION TECHNOLOGY (PG)
PILLAI COLLEGE OF ENGINEERING
NEW PANVEL - 410206
UNIVERSITY OF MUMBAI
Academic Year 2020-21

Certificate

This is to certify that **Mr. Yashodhan Prakash Ketkar** has satisfactorily carried out the dissertation work entitled **"Detection of Arrhythmia in COVID-19 Patients Using Supervised Learning Methods"** for the degree of **Master of Engineering** in **Information Technology** of **University of Mumbai**

Dr. Sushopti Gawade   **Dr. Satishkumar Verma**
Project Guide   Head of Department
Computer Engineering   Information Technology
Pillai College of Engineering   Pillai College of Engineering

Dr. Sandeep M. Joshi
PRINCIPAL
Pillai College of Engineering, New Panvel

SYNOPSIS OF PROJECT WORK

Name of the Dissertation:	Detection of Arrhythmia in Covid-19 Patients Using Supervised Learning Methods
Student's Name:	Mr. Yashodhan Prakash Ketkar
Class:	M.E. (Information Technology)
College:	Pillai College of Engineering, New Panvel
Semester:	IV
University Registration Number:	2019016402240447
Date of Registration:	26-07-2019
Exam Fee Receipt No.:	CF 2019-2020/2021
	CF2019-2020/2022
Name of the Guide:	Dr. Sushopti Gawade

Semester	Exam Seat Number	Result
1st	4021718	7.81
2nd	6041462	9.52

Student	**Project Guide**
(Mr. Yashodhan Ketkar)	(Dr. Sushopti Gawade)

Table of Contents

List of Figures

List of Tables

List of Equations

Abstract

COVID-19 disease has become a pandemic over the past year. This disease is highly contagious and spread throughout many countries. This disease with combination of comorbidities can lead to serious consequences for its victims. Cardiovascular comorbidities are leading to serious illness and death in COVID-19 infection. These cardiovascular comorbidities can also manifest after COVID-19 infection. These Cardiovascular problems can be detected using supervised learning methods, faster and more efficiently than traditional methods. Method used in this report uses supervised learning methods for detection of these problems. The method automatically trains and selects the best model for data provided and stores this data for future predictions. This will allow medical staff to focus on patient care and develop solutions faster. The implemented system uses K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest Algorithm (RF), Multi-layer Perceptron (MLP) and Support Vector Machine (SVM). The proposed system measures performance of this algorithm with respect to accuracy, F1, recall, precision, ROC and prediction time for each item.

Chapter 1

Introduction

Chapter 1

Introduction

1.1 Introduction

In recent years COVID-19 has become extremely widespread. It attacks the human respiratory system and leads to complicated respiratory problems. This disease can be very severe depending on a person's health and presence of comorbidities. Cardiovascular comorbidities are very common in severe cases of COVID-19 disease. Cardiovascular comorbidities are also harder to detect without proper equipment. One detection method is to check the presence of arrhythmia in patients. Arrhythmia is improper beating of the heart, whether irregular, too fast or too slow. This can be detected by reading ECG signals. As COVID-19 has put a lot of stress on the medical staff so detection is taking more time than usual. Increased access to the Internet has led various industries to implement machine learning and AI for service. This leads to the growth of research in the field of machine learning and will affect machine learning in various fields. The medical and healthcare industries are one of them. Machine learning is used to quickly detect a patient's illness and classify viruses and other microbes. Machine learning algorithms have already proven to be very effective in medical applications.

The reading of these ECG signals and detection can be done with help of the machine learning system. The supervised learning algorithms can be used for detection of these symptoms in a much faster and more efficient way. The supervised algorithms are already proven faster in such precise classification tasks than unsupervised algorithms. Once trained this algorithm can be used for future predictions too.

There are multiple supervised algorithms available, so we can choose algorithms that suit our needs. In case of the general populace this step can be automated. The preset of a few algorithms can be provided to the system, then the machine can train and select the best model selected for the provided dataset. This will allow medical staff to focus on patient care and develop solutions to medical problems.

1.2 Motivation

COVID-19 disease is a global pandemic which caused more than 250 million infections worldwide and 5 million deaths. Followed by respiratory disorders, the cardiovascular diseases and cardiovascular complications lead to the most severe cases and are one of most common causes

of deaths in COVID-19 patients. Due to the nature of cardiovascular disease they often go undetected and untreated. This further leads to risks of severe COVID-19 in those individuals.

Earlier diagnosis of cardiovascular disease can lead to lower severity of disease and reduce the mortality rate. Arrhythmia is one of the symptoms of cardiovascular disease and cardiovascular complications. ECG signals are used in detection of cardiovascular disease. But this process is very time consuming and can lead to a lot of human errors. Due to strain on the healthcare system this has become an incredibly difficult task. The implementation of machine learning can lead to faster detection of arrhythmia. This will help in early diagnosis and treatment of cardiovascular problems in COVID-19 patients and will lead to good outcomes.

1.3 Beneficiaries

This system is helpful for medical personnels handling covid-19 cases. This system will also help other patients that suffer from cardiovascular problems or other problems.

The system can be used for a wide range of applications, such as detection of epilepsy from frequency charts obtained during testing, also detection of other disorders. This system will reduce workload from medical staff and allow for better understanding of problems present in patients' cases.

This system can be trained once for generalized predictions, or used for unique purposes depending on users needs. The system does not need machine learning experts to operate, it can be trained and used by end users directly, hence providing an additional layer of security.

1.4 Problem Statement

In current COVID-19 pandemic life threatening complications can be avoided with earlier diagnosis. One of such complications is arrhythmia which can be detected by reading ECG signals. This task is carried out by medical personnel. This repetitive and mundane task puts strain on medical staff.

Supervised learning models can be used for such tasks. These models have their own set advantages and disadvantages. Hence automating selection of the best model for required tasks will allow medical personnel to customise applications based on need. This will reduce strain from medical staff and allow them to focus on patient care.

1.5 Scope of the project

To make the system more efficient, data used for training is provided in csv format. And obtained results are stored in json format. The system handles imputed data with the help of the pandas library. The dataset provided needs to be specific format regardless of length and parameters present in the dataset. The datasets used in this projects are available on kaggle and are converted in required format, but if needed the raw datasets obtained directly from machines can be used for operational purposes.

1.6 Organization of Report

The report is structure into seven chapters as follows:

Chapter 1 - Introduction: This chapter gives an overview of the report. It gives a general overview of the current situation of COVID-19 pandemic and the stress it puts on the medical system. It also describes the proposed solution to tackle this problem.

Chapter 2 - Literature Survey: This chapter provides review of relevant literature of supervised learning systems and automation approaches for the model training.

Chapter 3 - System Architecture: This chapter describes the architecture of the system. It discusses the models used in the system, the automated training and selection of models.

Chapter 4 - System Interface: This chapter describes the user interface of the system. This chapter will discuss the pages accessible by the user and the information about the development server.

Chapter 5 - Result And Analysis: This chapter will describe the performance metrics used by the system for the selection process. The chapter also gives details about the dataset used for training and testing. It also provides recommended system specification required by the application.

Chapter 6 - Applications: This chapter gives various applications of the developed system.

Chapter 7 - Conclusion And Future Scope: This chapter concludes the report by providing a brief summary of the project. This chapter also describes the future scope of the system.

Chapter 2

Literature Survey

Chapter 2

Literature Survey

2.1 Introduction

Arrhythmia is one of the most common symptoms in COVID-19 patients [6]. Arrhythmia was present in 7% of total COVID-19 cases in Wuhan and 14.8% of patients with poor outcome had arrhythmia. Data from 17 studies consisting of 5815 patients suggested that up to 9.3% of confirmed patients had been detected with arrhythmia [1, 11]. Up to 94.4% of total patients died had arrhythmia and 95.8% of patients with serious infection had arrhythmia [9, 10]. Only 8% of patients with arrhythmia had prior cardiovascular diseases while in 56% arrhythmic symptoms were new onset after contracting COVID-19 disease according to study [6, 11, 3].

2.2 Methodologies Used By Authors

In paper [12] the author utilized ensemble classifiers to detect anomalies in ECG signals. This approach of combining multiple classifiers with each other for prediction proved effective, as accuracy of ensemble classifiers was significantly greater than single classifiers. This approach is also used by other researchers to improve prediction accuracy of supervised learning models [4, 5, 13]. The studies also presented greater accuracy and validation score for anomaly detection [4, 13], while one study showed that accurate predictions can be obtained without any prior information about the same class [5]. The standard classifiers can be used to obtain accurate and precise predictions for incoming data streams automatically [2].

For detection presence of epileptic seizure in patients, the authors preprocessed data by extracting suitable features and applied supervised and unsupervised machine learning classifiers to it. The predictions from test data suggest that supervised learning models are more effective than unstructured learning models [7]. The standard classifiers can be automated for accurate and precise prediction [2]. Author of study [8], also presented that the off-the-shelf classifier SVM is very efficient for detection of arrhythmia in ECG signals.

The ensemble classifiers approach i.e. combination of multiple classifiers together for predictions can be used for anomaly detection. The ensemble classifiers can be more effective and accurate than individual supervised learning models [12, 13, 4]. The author of paper [5] discovered that accurate predictions can be obtained without prior information about past target values of the same class.

The artificial neural networks are also very effective and accurate for arrhythmia detection from ECG signals. The neural networks can process raw ECG signals and make predictions [14]. Smaller scale neural networks are also very efficient in detection processes [15]. The neural network predictions are comparatively more accurate and precise than well known classifiers.

The lightweight classifiers such as LDA classifier showed higher accuracy than SVM classifier in low energy system [17]. Self adapting learning algorithms are also efficient for low energy lightweight systems [20, 21]. The self learning algorithms make the system more dynamic and adapt to incoming signals. This dynamic system takes very little time for identification tasks, whereas classification tasks are more difficult to manage and are very time consuming.

2.3 Automated model selection approaches

In the paper [19], the author used 2 years of dataset collected from glumo lake, the use of expert knowledge is used for model training and selection. The mixed approach of data driven and knowledge driven modeling is used for successful application. While author of the paper [22], used loo rate and stopping criteria for model selection. The author considered 8 different problems and discovered that larger loo rate is more desirable. Author also suggested that modeling difficulties can only be found with careful numerical calculations.

In paper [16], novel kernels are used to obtain explanations of the dataset. This approach leads to discovery of unseen models with minimal human interaction. The author of the paper [18], used the glmulti package to build new models, these models are unique and flexible. The models are optimized automatically and provide a multi-model interface. With this approach a large set of models can be considered for selection purposes at a rapid rate.

2.4 Inference of Literature Review

The following inference can be drawn from the basis of literature review

- Most papers introduced here use supervised learning algorithms for the prediction. This technique needs a well labeled dataset for training and evaluation. Therefore, expertknowledge is needed during the training and evaluation process.

- Some papers concluded that it is possible to select suitable models with minimum human interactions automatically. The knowledge driven and data driven approaches are used for automated model suggestions.

- Most papers also showed that on-shelf models can be used for accurate and efficient use

cases, but few authors used novel kernels and models to get greater accuracy and efficiency for predictions.

- The recent paper showed that there is great need for models that can be setup faster for required work. To solve this problem we can use limited data for training which is possible with supervised learning algorithms according to few studies.

- Finally it can be inferred that a very negligible amount of work has been done for the automated model selection for general prediction purposes. Moreover, a limited amount of study is done with the general population as end users. Hence, this system is focusing on training, evaluation and selection of models for users with limited knowledge of data science.

Chapter 3

System Architecture

Chapter 3

System Architecture

The trainer module is responsible for generating and training supervised learning models. This module sends trained models to the selection module for performance evaluation. The selection module generates performance files and ranks the models according to its performance with respect to parameters if sent by the user. Rank obtained by model is used for selection of the best model. This best model is stored for future predictions. This model can be accessed by the prediction module. This module takes a user provided dataset and makes predictions with help of the best model. Figure 3.1 shows the whole process.

Figure 3.1: System architecture

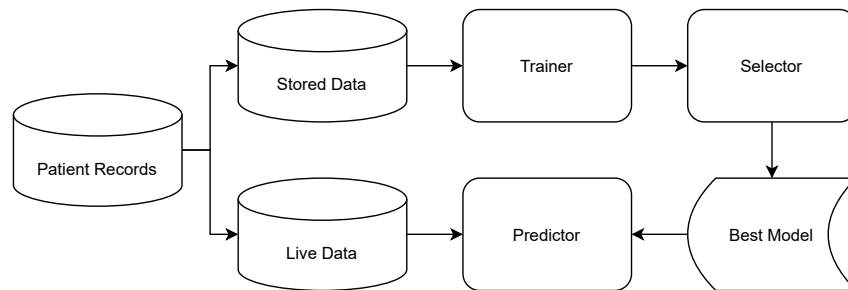## 3.1 Data Flow in system

The data used in this process goes through this sequential process shown in fig. 3.2
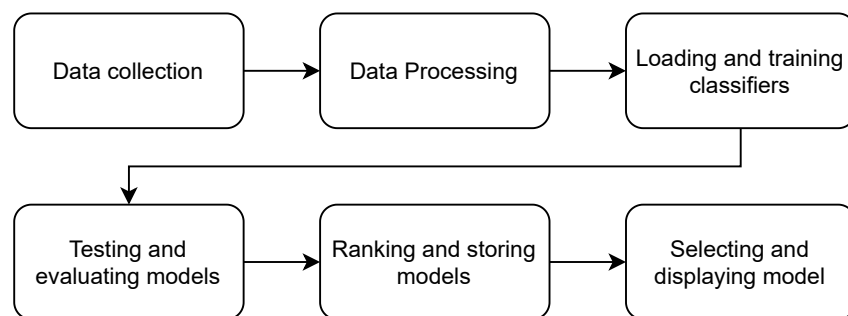
Figure 3.2: Training and Selection Process

Data collection: Collection of data is the first step in this process. The data is collected from the user. The data is stored in a local directory for further processing. The data is then divided into training and testing sets.

Data processing: In this step data labels are converted to 1 or 0, with respect to previous values. In case of data with more than 2 labels, labels with min value are converted to 0 and other values are converted into 1.

Loading and training classifiers: In this step model templates are used for generation of models. These models are trained with the help of processed training data.

Testing and evaluating models: The trained models are evaluated with help of processed testing datasets. The performance metrics are stored for ranking models.

Ranking and storing models: Performance metrics obtained in previous steps are used to rank the models, the models are stored in a local directory.

Selecting and displaying model: In this step rank of models are used for selection of best model, this model is stored into best models directory for future predictions. The name of this selected model is displayed to the user.

3.2 Training and selection process

As shown in fig. 3.3, the system is provided with a dataset, in this case records of ECG reports of patients. This data is already provided with labels. These labels can be boolean values i.e., 1 for True, and 0 for False, or they can have a series of values starting from 0. This data is further divided into training dataset and testing dataset. These datasets are forwarded to the extraction process. In the extraction process, datasets labels are converted into 0 and 1, this will eliminate extra classes present in labels by changing their value to 1. The processed training dataset is sent for the model training. As soon as this process takes place, the training module accesses premade models templates, and generates the model for training purposes. These models are trained with processed training data and stored in a directory for evaluation purposes.
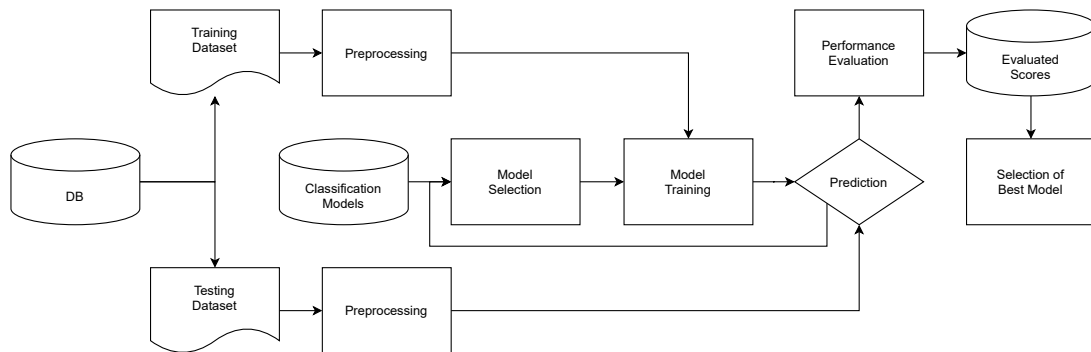

Figure 3.3: Training and Selection Process

Selector module accesses the stored models and processed testing dataset to evaluate the performance of models. The performance metrics used for evaluation are accuracy score, F1 score, precision score, recall score, roc score and prediction time of model. These parameters are multiplied with default or user provided weightage provided to generate scores of the models. These scores are stored and used to select the best suited model.

3.3 Methodology

Figure 3.4 shows the approach to the selection of a suitable model. The system can handle an infinite number of models. For each model, six performance parameters are used. These performance metrics are accuracy P_1, F1 score P_2, precision P_3, recall P_4, area under the ROC P_5, and prediction time P_6. From these, the first five parameters are grouped and given a weightage range between 0.2 and 1.0, whereas prediction time is given a weightage range between 0.25 to 0.75. Equation (3.1) shows the mathematical formula used to calculate the Vscore of the models.

For models M_1, M_2,..., M_n, the Vscores V_1, V_2,..., V_n are obtained. These obtained Vscores are compared with each other. The model with the highest Vscore is selected as the most-suited model.

$$V_{score} = \left(\sum_{x=1}^{5} w_x P_x \right) - w_6^2 P6 \tag{3.1}$$

where,

V_{score} Vscore of the model

w_x The weightage generated by the system for the xth parameter.

P_x Performance of xth parameter

3.4 Classifiers Used

The classifiers are algorithms that classify data into two or more classes based on rules set out by users. The models trained with these classifiers are used for classification purposes. In this project we are using classifiers to train models, these models will output data into two classes.

By default this system uses five different classifiers, they are K-Nearest Neighbors, Decision Tree, Random Forest, Multilayer Perceptron and Support vector machines. These classifiers are
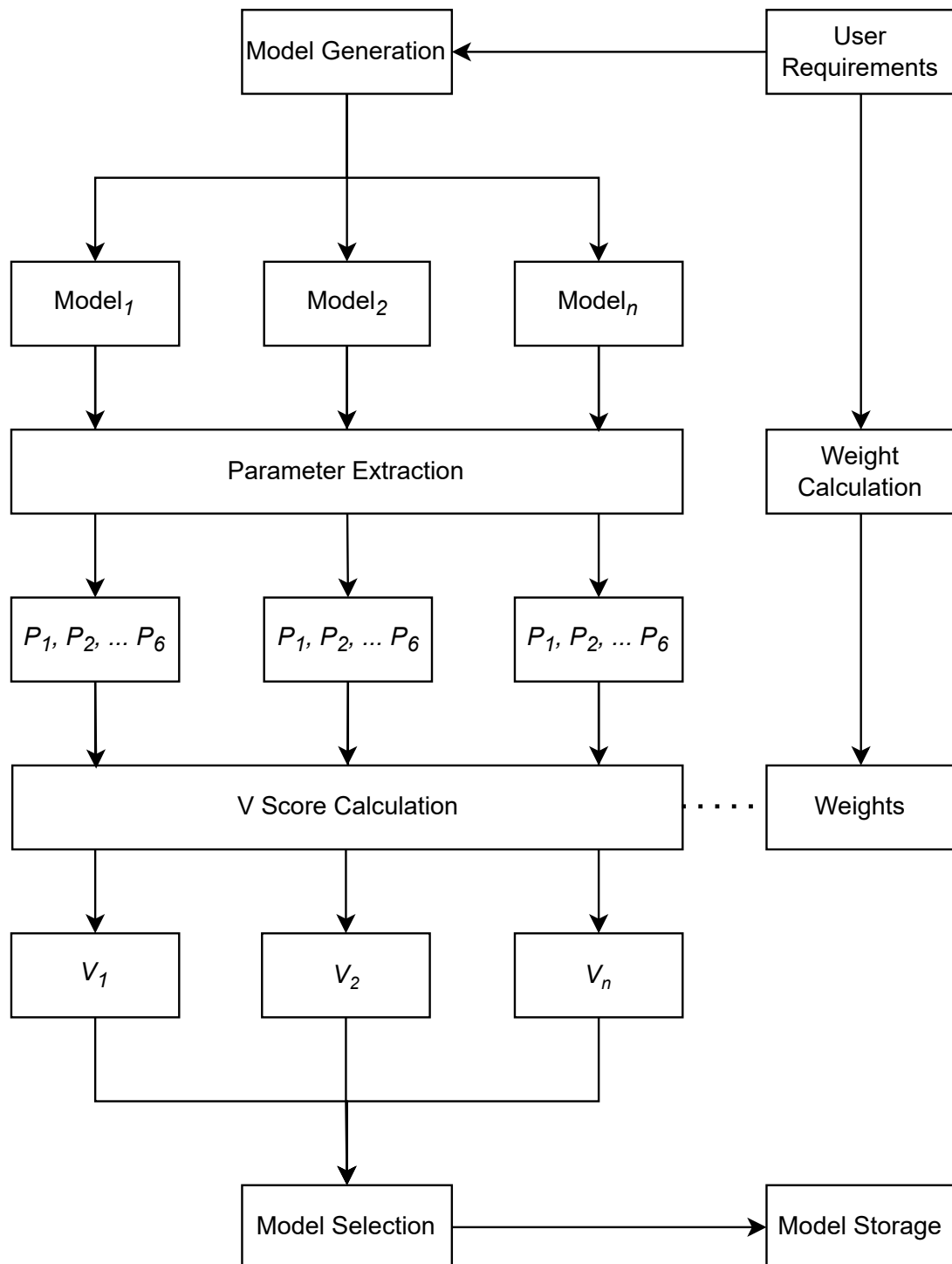
Figure 3.4: Model Selection Approach

categorized as supervised learning algorithms or classifiers. These classifiers will predict the classes of data provided by the user.

3.4.1 K-Nearest Neighbors

K-nearest neighbor (KNN) algorithm is a supervised machine learning algorithm. Classification and regression problems such as multiclass problems are solved using this algorithm. KNN is lazy learning and non-parametric algorithm, which can be advantages or disadvantages depending on the user's requirements. These properties make this algorithm very effective and uncomplicated in general tasks. The drawback of the simplicity is the slow prediction time.

Algorithm for K-Nearest Neighbors:

Step 1: Determine the number of K neighbors.

Step 2: Determine the Euclidean distance between the K neighbors.

Step 3: Using the estimated Euclidean distance, find the K nearest neighbors.

Step 4: Count the number of data points in each category among these K neighbors.

Step 5: Assign new data points to the category with the greatest number of neighbors.

Step 6: The model is now complete.

3.4.2 Decision Tree

Decision trees are the most popular and effective tool for classification and prediction. The decision tree is a flow chart-like tree structure, where each internal node specifies a test for the attribute, each branch represents the result of the test, and each leaf node contains a class label. Decision trees are the most popular high-performance tool for classification and prediction.

Construction of Decision Trees:

The trees are trained by subdividing the features into subsets based on the attribute values. This process is run recursively on derived subsets. The recursion completes when all subsets of nodes have the same value for the target variable or if the split doesn't add any additional value to the prediction. Building a decision tree classifier is suitable for exploratory knowledge discovery as it does not require domain knowledge or parameter adjustment. Decision trees can handle high-dimensional data. In general, decision tree classifiers are more accurate. Decision tree induction is a typical inductive approach to learning classification knowledge.

Alogrithm for Decision Trees:

Step 1: Start the tree on the root node that contains the complete dataset.

Step 2: Use the Attribute Selection Scale (ASM) to find the best attribute in the dataset.

Step 3: Divide S into a subset containing the possible values of the best attributes.

Step 4: Create a decision tree node with the best attributes.

Step 5: Recursively build a new decision tree using the subset of the dataset created in step 3. Continue this process until you reach a stage where you can no longer classify the node, and you can call the last node a leaf node.

3.4.3 Random Forest

Random forest is a supervised learning algorithm based on a decision tree algorithm. This algorithm generates multiple decision trees from the provided sample. These decision trees are combined, and a majority vote is taken to solve the classification and regression problem. The Random Forest classifier offers more accurate and stable predictions at a slightly slower prediction time when compared with a decision tree.

Bagging

The random forest classifier is a type of ensemble classifier which uses the bagging technique. Random samples from data sets are collected. These samples generate independent models. Random forest uses decision tree classifiers for model generation. The final output is generated based on majority voting after combining the results of all models. This process of combination and majority voting is known as the aggregation process.

Classification algorithm for Random Forest:

Step 1: N number of random records are taken from the dataset with k number of reports.

Step 2: Each sample generates a separate decision tree.

Step 3: Each decision tree will generate an output/prediction unrelated to other trees.

Step 4: The final output is determined based on majority voting in classification problems.

3.4.4 Multi-layer Perceptron

The multilayer perceptron is a neural network-based supervised learning algorithm. In this network, input and output are not linearly mapped. It is a feedforward algorithm hence more suitable for classification and regression problems. MLP uses a backpropagating algorithm for training, making it faster and usable for general predictions.

Construction of MLP

The neural network is made up of three primary components, neuron, activation function, and layers.

Neuron: A neuron is the smallest unit of a neural network. It is based on neurons in the brain, which take one or more inputs to produce an output. The inputs are weighted, and each neuron has biases which are also weighted similar to inputs.

Activation functions: The weighted sum of inputs and bias is passed through activation functions. This function serves as a mapping between two layers. These functions are usually non-linear. Activation functions control the firing of a neuron. We are using the sigmoid as an activation function. The sigmoid function returns either a zero or postitive value.

Layers: Layers are rows of neurons in the network. The input layer is the entry point for the neural network. The output layer serves as the exit point for the neural network. A hidden layer or series of hidden layers connects input and output layers. In MLP, input layers take multiple features as data. The output layer returns either 1 or 0.

Algorithm for Multilayer Perceptron:

Step 1: N number of epochs is calculated by dividing sample size S by batch size B.

Step 2: Input layers accept the data features.

Step 3: Input layer forward data to hidden layers, which forward data to the output layer.

Step 4: Errors are calculated by comparing expected output and network output.

Step 5: The error propagated backward to adjust the weights of the network.

Step 6: All weights in the network are adjusted.

Step 7: Process in steps 3-6 is known as an epoch. The process is run for N number of epochs.

3.4.5 Support Vector Machine

Support Vector Machines or SVM is one of the most commonly used supervised learning algorithms for data classification and regression. This algorithm is primarily used to solve classification problems in machine learning. The SVM algorithm created decision boundaries to divide n-dimensional spaces into classes. The new data points are categorized using these classes. The best boundary line which compartmentalizes most data accurately is called a hyperplane. The SVM algorithm uses vectors to create the hyperplane. Support vectors handle extreme cases giving the algorithm its name Support Vector Machine.

Hyperplane: A decision boundary separates new data points into different classes. The decision boundary that produces the most accurate categorization is called the hyperplane. The hyperplane

adjusts with new data points by repositioning or introducing new dimensions. Hyperplanes can be linear or non-linear, depending on user requirements.

Support Vectors: These are extreme data points present in data sets. Support vectors have very close proximity to the hyperplane. Support vectors can directly affect the position and dimensions of the hyperplane.

Tuning parameters of SVM

Tuning parameters used in this projects are:

Kernal: SVM uses the kernel function to solve problems. These functions allow smoother operations on high dimensions. With kernels, SVM hyperplanes can theoretically reach infinite dimensions. Kernels also reduce complexity at higher dimensions. The primary purpose of the kernel is to provide a hyperplane way to accommodate non-linear datasets. In this project, we are providing an RBF kernel. This kernel is suitable for both small and large quantities of data.

C parameter: This is also known as the regularization parameter. C parameter suggests SVM optimization amount of margin used by a hyperplane. A higher value of the C parameter produces accurate results. Conversely, a lower value results in fast but less accurate predictions.

Gamma: The Gamma parameter defines the amount of influence of a model over the dataset. Gamma is inversely proportional to the curve of influence of the model. Lower gamma results in higher accuracy, but it is prone to overfitting. Higher gamma is constraining but can't handle complexity.

Chapter 4

System Interface

Chapter 4

System Interface
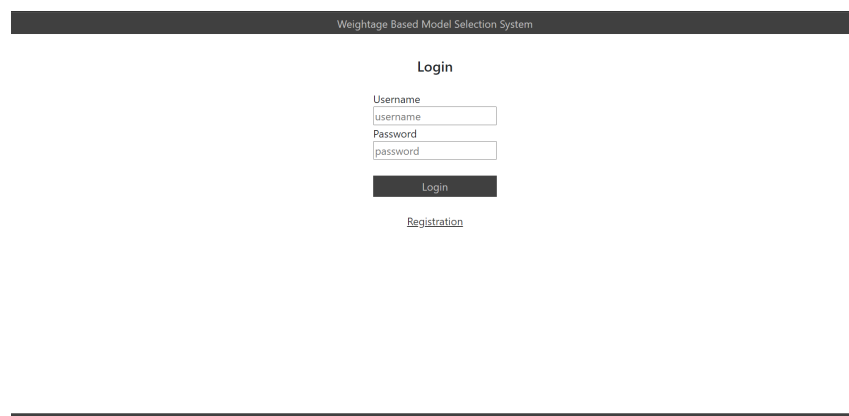
The system takes data in CSV format. A web-based GUI will be offered to users. We are using a flask-based web app for easier integration of the main system. We have implemented separate pages for users depending on tasks. These pages are the selection page, prediction page, performance page, and cross-performance page. We will see a detailed explanation of the pages and the process carried out in the background.

4.1 System Design

The server is accessed with the help of a port number. The site is accessed by URL in the form of `http://ip_address:port_number` URL syntax. In this development server we are using `http` protocol and `127.0.0.1` or `localhost` as `ip_address`. The `port_number` 5000 is used as per flask recommendation. Hence, the URL used by the application to access the website is `http://localhost:5000`. In the next subsections, we will go through each page available to the user.

4.1.1 Login Page

This page is the first page encountered by a user when logging on to the website. This page will take a valid username and password from the user and start the session for that username. After starting the session, the user will be redirected to the Home page. Figure 4.1 displays the login page.

Figure 4.1: Login Page

4.1.2 Home Page

The home page is the landing page for the users after the session is started. It provides information about the website. The side navigation is provided for users for easier access to the rest of the website. This side navigation is present on all the pages except the login page.

4.1.3 Selector Page

This page takes file input from the user. Currently, only CSV file format is supported. The files are copied into the directory of the application. The training and selection module will process the copied file. These modules will execute model training, performance evaluation, and selection process. After process completion, the user will receive confirmation. Figure 4.2 displays the selector page.

Figure 4.2: Selector Page

4.1.4 Patients Page

This consist of two subpages 1. registration and 2. reports.

4.1.5 Selector-Upload Page

This page is a subpage to the selector page. This page is used to display the name of the selected classifier to the user. This page has a similar layout to the selector page with a small difference, and it acts similarly to the selector page. Figure 4.3 displays the selector-upload subpage.
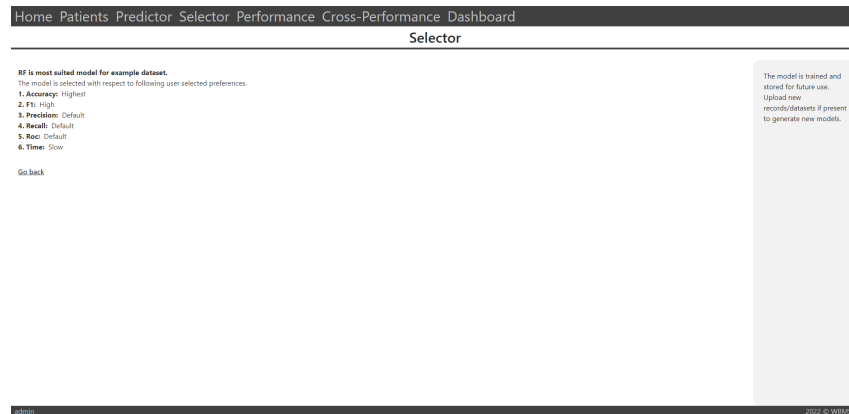
Figure 4.3: Selector-Selected Page

4.1.6 Predictor Page

This page is used to make predictions based on the dataset provided by the user. This page generates a list of the best models for users to select models for predictions. The page also generated a list of datasets provided by the user for predictions. If the dataset is not available, the user is allowed to upload the dataset from the page. After submitting the selected model and dataset page will run the prediction process on the datasets. After completion of this process, the user is forwarded to the subpage. Figure 4.4 displays the predictor page.
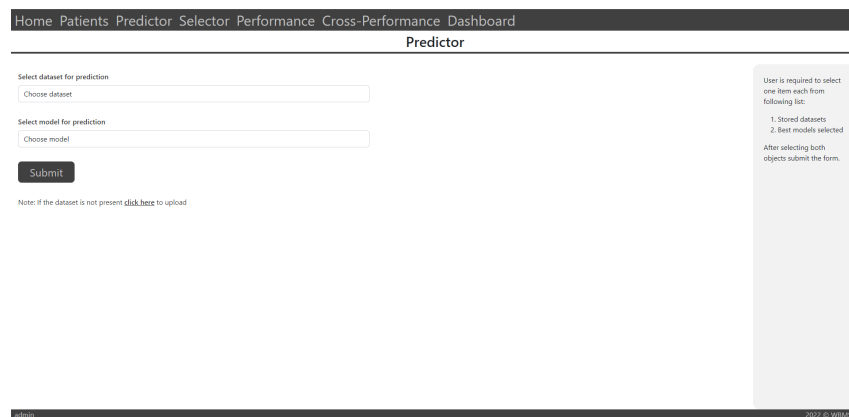
Figure 4.4: Predictor Page

Predictor-Results Page

This page will display the results of predictions obtained from the model and dataset submitted by the user with the predictor page. The result is displayed in tabular format with ID and prediction results as columns. Figure 4.5 displays the predictor-results subpage.

Figure 4.5: Predictor-Results Page

4.1.7 Performance Page

This page allows users to select models from a list of models created from previous datasets. After selecting the model, the page will get performance data stored in the local directory and redirect the user to the subpage. Figure 4.6 displays the performance page.

Figure 4.6: Performance Page

4.1.8 Performance-Display Page

This page will display the performance obtained from the performance page. This page has a similar base layout to the performance page and has an option to display the graph generated from performance. This page also acts similar to the performance page. It allows the user to select another model to display its performance. Section 4.1.8 displays the performance-display page.

Figure 4.7: Performance-Display Page

4.1.9 Cross-Performance Page

This page allows users to select models from a list of models created from previous datasets. The page also provides a list of datasets uploaded by the user for predictions. The user is required to select a dataset from this list. The user-selected dataset and model will be sent to the server for cross-performance analysis. After completion of the cross-performance process, the user is redirected to the subpage. Figure 4.8 displays cross-performance page.
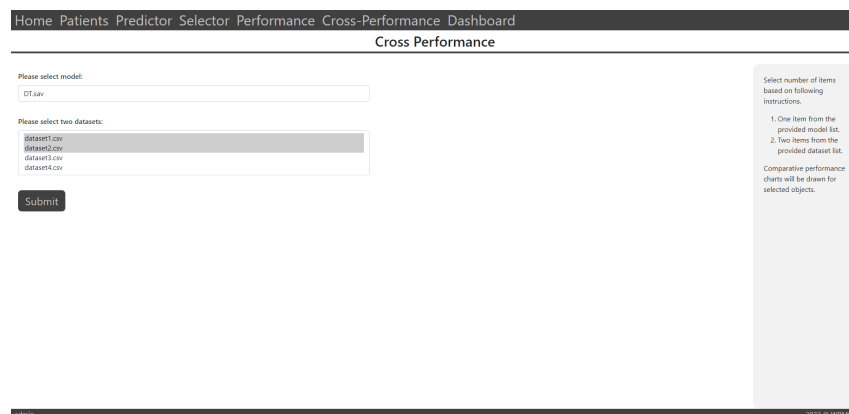
Figure 4.8: Cross-Performance Page

4.1.10 Cross-Performance-Display Page

This page will receive the data obtained from the cross-performance process and display the data in chart format. This page has a similar layout and acts similar to a cross-performance page, with a display place for charts. Figure 4.9 displays the cross-performance-display page.

Figure 4.9: Cross-Performance-Display Page

4.1.11 Helper Pages

These pages act as helper pages for users as well as handles errors. In the instance of unauthorized access or invalid information, users are redirected to these pages. Figure 4.10 shows the error pages.

(a) Unauthorized Access Error Page

(b) Not Found Error Page

(c) Method Not Allowed Page
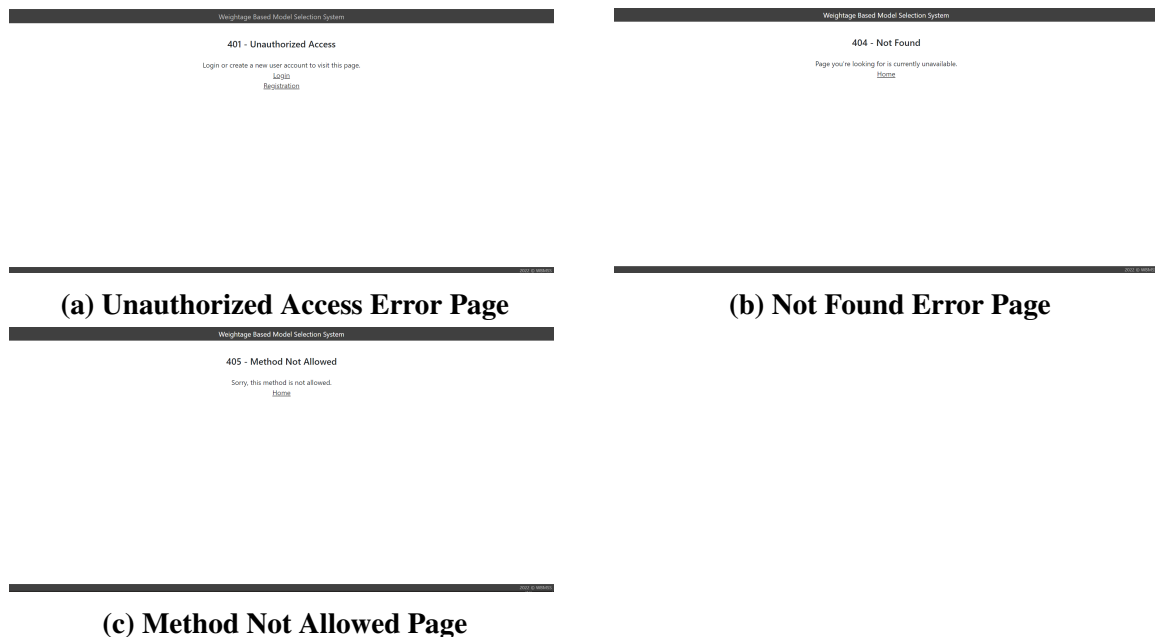
Figure 4.10: Error Pages

4.2 Hardware Details

The hardware used for the program is given below. The system needs at least a moderate amount of memory for operations. The amount of system memory used during development is 16 GB.

The system also needs a high-performance CPU for the model training. The number of cores available for calculation is important too. The development system uses a 3.6 GHz processor with 6 Cores. The data uploaded by users and system-generated data need storage space. The recommended size of the storage is 250 GB.

Standard input and output devices are required to take data and commands from users and display them.

Table 4.1: Hardware Details

Hardware	Details
Installed memory (RAM)	16.00 GB
Storage	250 GB SSD
Processor	AMD Ryzen 5 3500 6-Core Processor 3.6 GHz
GPU	NVIDIA GeForce GTX 1650 Super
Input device	Standard Keyboard and Mouse

4.3 Software Details

The software requirements of the application are described below. The application is developed on Windows 10 system, but it is OS independent application. The technologies used are python as a base programming language, sklearn library to make model templates, and flask to develop the web application. Pandas library is used for data handling. HTML, CSS, and JavaScript are used for web pages.

Table 4.2: Software Requirement

Software	Details
Operating System	Windows 10
Technology	Python, Scikit-learn, Pandas, Flask, HTML, JavaScript

Chapter 5

Result And Analysis

Chapter 5

Result And Analysis

5.1 Performance Measures

There are a lot of different types of parameters to evaluate the performance of machine learning models. In case of the classification models Accuracy score, F1 score, Recall score, Precision score and total area under ROC curve used for performance evaluation.

5.1.1 Accuracy Score

The accuracy score is a fraction of the correct prediction by model with respect to total predictions by model. It can be represented by following formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (5.1)$$

5.1.2 Precision score

The precision score is a fraction of the correct positive predictions with respect to all positive predictions of the model. Higher precision scores result in fewer false positive predictions. It can be represented by following formula:

$$Precision = \frac{TP}{TP + FP} \qquad (5.2)$$

5.1.3 Recall score

The recall score is the fraction of correct positive predictions with respect to all predictions of the class. It can be represented by following formula:

$$Recall = \frac{TP}{TP + FN} \qquad (5.3)$$

5.1.4 F1 score

The F1 score is the weighted average of the recall score and precision score of the model. F1 scores are more reliable than accuracy scores in case of biased or uneven dataset. It can be represented by following formula:

$$F1 = 2 \cdot \frac{Recall \cdot Precision}{Recall + Precision} \tag{5.4}$$

F1 score can also be represented as $F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$

5.1.5 AUC ROC score

The ROC is a classifier's predictive quality that compares and visualizes the trade-off between the model's sensitivity and specificity. In graphical format, the area under it gives a relationship between false positives and true positives. The higher these areas are, the better the predictive quality of the model.

5.1.6 Prediction Time

Prediction times are nothing but the amount of time required by the classifier to make predictions for certain testing datasets. A model with a lower prediction time is desirable.

5.1.7 Performance Evaluation Methodology

Performance scores are calculated with performance evaluation metrics and average prediction time. The models are ranked based on performance scores.

Weighted sums of the ranks are stored for the selection process. Weightage is predefined. It lies between 0 to 1, and it is directly proportional to the significance of the metric. For example, in situations where speed is more important than accuracy, prediction time is given higher weightage than accuracy metrics.

In the proposed system lowest scoring model is selected as the ideal model.

5.2 Dataset Description

The goal of the project is to detect the presence of arrhythmia from ECG signals accurately and faster than the traditional approach. For this task, a well-known open-access database MIT-BIH Arrhythmia Training and Testing Dataset are used. Both datasets are divided into four equal parts randomly and used for training and testing respectively. Each training set contains 21888 signals and the testing set contains 5473 signals.

5.3 Performance Evaluation

The datasets obtained from the MIT-BIH training dataset showed significant similarities in performance evaluation. All four datasets showed higher performance metrics for SVM classifiers. The prediction time required by SVM and KNN was significantly higher than other classifiers. The desired result can be tweaking ranking parameters. The performance of models is shown in fig. 5.1.

Table 5.1 shows the Random Forest model have highest Vscore for dataset 1, hence it is selected as the most-suited model for this dataset. While tables 5.2 to 5.4 shows the Support Vector Machine model scored higher Vscore for other three datasets, hence it has been chosen as most-suited model for those datasets. Table 5.5 shows the V_{score} of all models.

Table 5.1: Performance of models trained on Dataset 1

Metric	KNN	DT	MLP	RF	SVM
Accuracy	96.72	94.46	96.89	**97.33**	97.40
F1	89.71	83.43	90.05	**91.30**	91.69
Precision	92.53	81.86	94.71	**97.95**	96.43
Recall	87.06	85.06	85.84	**85.50**	87.40
ROC	92.84	90.68	92.45	**92.57**	93.38
Time(s)	0.457	0.001	0.002	**0.015**	0.297
V_{score}	2.747	2.643	2.782	**2.807**	2.482

Table 5.2: Performance of models trained on Dataset 2

Metric	KNN	DT	MLP	RF	SVM
Accuracy	96.83	95.04	96.69	97.09	**97.46**
F1	90.28	85.50	89.73	90.75	**92.15**
Precision	94.25	84.90	94.73	98.60	**96.79**
Recall	86.63	86.09	85.23	84.05	**87.93**
ROC	92.77	91.48	92.13	91.90	**93.66**
Time(s)	0.435	0.001	0.003	0.014	**0.295**
V$_{score}$	2.758	2.684	2.772	2.795	**2.808**

Table 5.3: Performance of models trained on Dataset 3

Metric	KNN	DT	MLP	RF	SVM
Accuracy	97.07	94.64	96.41	97.22	**97.44**
F1	90.93	84.30	89.34	91.21	**92.00**
Precision	95.82	83.81	90.13	98.37	**97.81**
Recall	86.53	84.80	88.57	85.02	**86.85**
ROC	92.87	90.73	93.29	92.36	**93.22**
Time(s)	0.404	0.001	0.002	0.017	**0.293**
V$_{score}$	2.774	2.658	2.766	2.803	**2.803**

Table 5.4: Performance of models trained on Dataset 4

Metric	KNN	DT	MLP	RF	SVM
Accuracy	96.84	94.99	95.34	96.88	**97.15**
F1	90.52	85.73	85.01	90.37	**91.39**
Precision	95.71	85.91	97.83	98.65	**97.41**
Recall	85.86	85.55	75.15	83.37	**86.07**
ROC	92.52	91.28	87.40	91.56	**92.79**
Time(s)	0.452	0.001	0.002	0.014	**0.294**
V$_{score}$	2.760	2.687	2.674	2.786	**2.790**

Table 5.5: V$_{score}$ of all models

Models	Training Dataset			
	Dataset 1	**Dataset 2**	**Dataset 3**	**Dataset 4**
KNN	2.747	2.758	2.774	2.760
DT	2.643	2.684	2.658	2.687
MLP	2.782	2771	2.766	2.674
RF	**2.807**	2.795	2.803	2.786
SVM	2.482	**2.808**	**2.803**	**2.790**
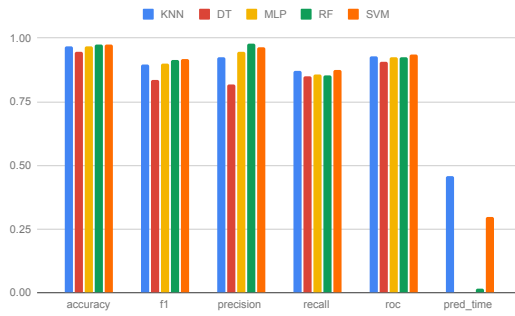
(a) Dataset 1
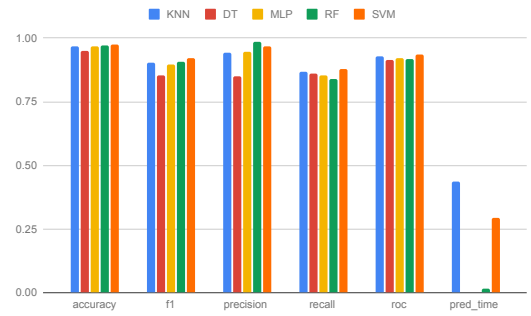
(b) Dataset 2
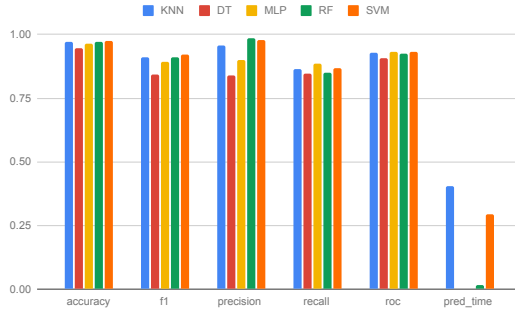
(c) Dataset 3
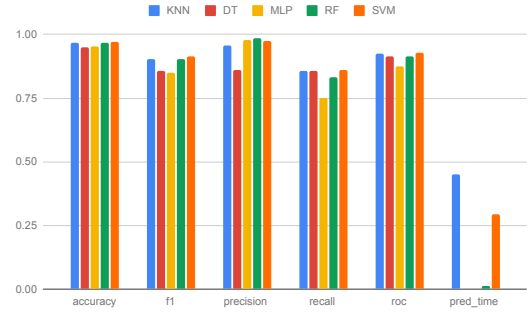
(d) Dataset 4

Figure 5.1: Perfromance Results

5.4 Cross Performance Evaluation

Models tested against other datasets show a slight difference in performance. This difference suggests that the models are capable of performing general predictions. These general tasks can be performed with a small efficiency cost. Figure 5.2 shows the performance of models tested on all training datasets.

(a)

(b)

(c)

(d)

(e)
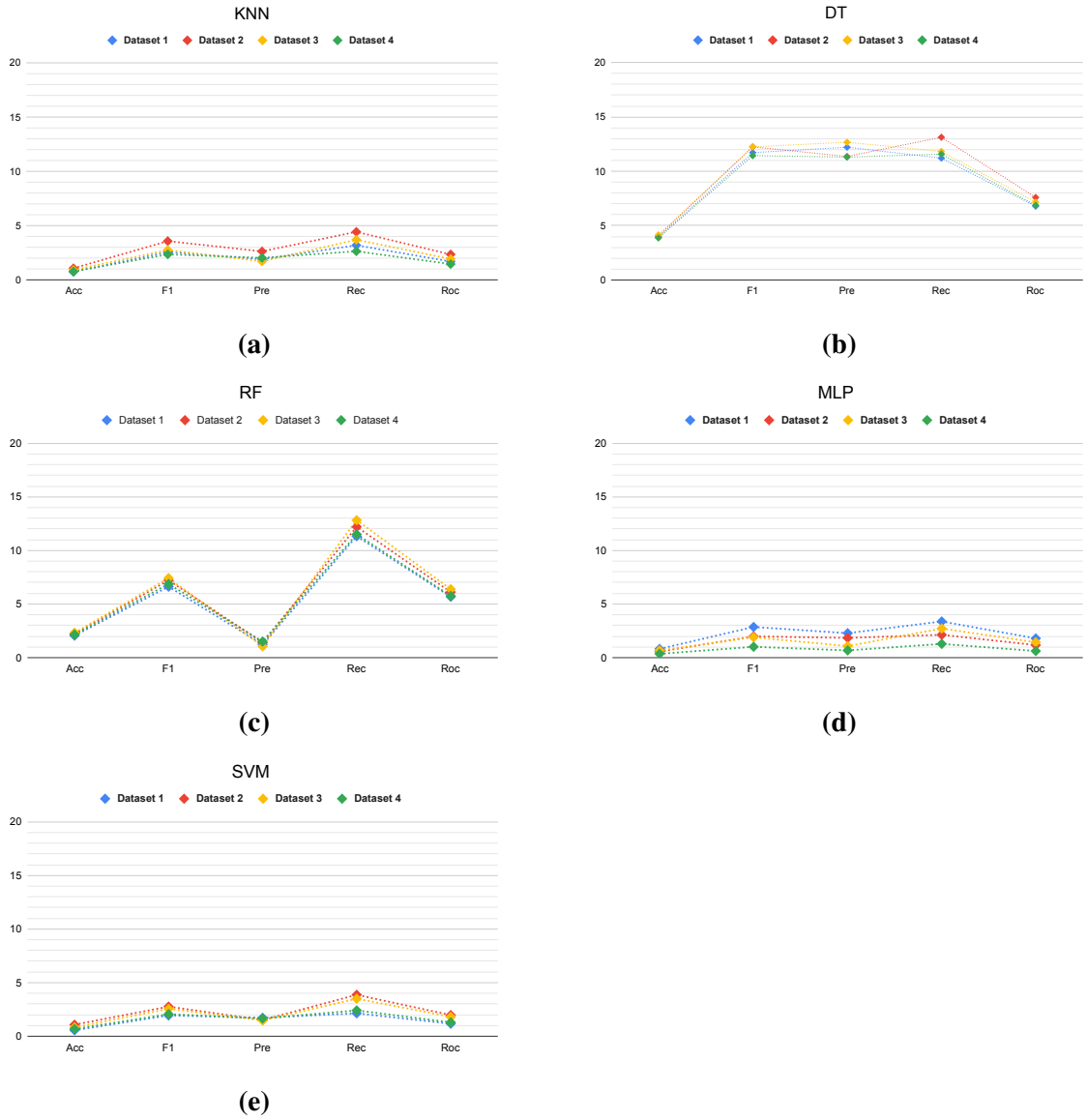
Figure 5.2: Average Perfromance drop

The fig. 5.2 shows the average performance difference of models concerning training datasets. From fig. 5.2 (b) and fig. 5.2 (c), we can see that the difference in performance metrics of the Decision Tree and Random Forest algorithm is almost similar to all training sets. It indicates that these models can be used as solutions to similar problems. Figure 5.2 (d) shows the maximum difference in performance metrics when tested against other datasets. It indicates that the model is overfitting and is not suitable to be used for other datasets. While fig. 5.2 (a) and fig. 5.2 (e) shows good performance metrics for some datasets. This shows models can be tuned for specific requirements. This will produce more satisfactory results in general predictions. Tables 5.6 to 5.10 shows the cross-performance evaluation of models.

Table 5.6: DT model cross-performance results

Dataset	1	2	3	4
Accuracy	0.98	0.94	0.94	0.94
F1	0.96	0.85	0.85	0.84
Precision	0.95	0.83	0.84	0.83
Recall	0.96	0.86	0.85	0.85
ROC	0.97	0.91	0.91	0.90

(a) Dataset 1 - DT Model

Dataset	1	2	3	4
Accuracy	0.94	0.98	0.94	0.94
F1	0.84	0.96	0.84	0.85
Precision	0.85	0.96	0.85	0.85
Recall	0.82	0.96	0.84	0.84
ROC	0.89	0.97	0.90	0.90

(b) Dataset 2 - DT Model

Dataset	1	2	3	4
Accuracy	0.94	0.94	0.98	0.94
F1	0.84	0.84	0.96	0.83
Precision	0.84	0.83	0.95	0.83
Recall	0.84	0.85	0.96	0.84
ROC	0.90	0.90	0.97	0.90

(c) Dataset 3 - DT Model

Dataset	1	2	3	4
Accuracy	0.94	0.94	0.95	0.98
F1	0.85	0.85	0.85	0.96
Precision	0.85	0.85	0.85	0.96
Recall	0.85	0.84	0.85	0.96
ROC	0.91	0.90	0.91	0.97

(d) Dataset 4 - DT Model

Table 5.7: KNN model cross-performance results

Dataset	1	2	3	4
Accuracy	0.97	0.97	0.97	0.96
F1	0.93	0.91	0.90	0.90
Precision	0.96	0.95	0.94	0.94
Recall	0.90	0.87	0.87	0.87
ROC	0.94	0.93	0.93	0.93

(a) Dataset 1 - KNN Model

Dataset	1	2	3	4
Accuracy	0.96	0.97	0.96	0.96
F1	0.90	0.93	0.90	0.89
Precision	0.94	0.96	0.94	0.94
Recall	0.86	0.90	0.86	0.85
ROC	0.92	0.94	0.92	0.92

(b) Dataset 2 - KNN Model

Dataset	1	2	3	4
Accuracy	0.96	0.96	0.97	0.96
F1	0.90	0.90	0.93	0.90
Precision	0.95	0.95	0.96	0.94
Recall	0.86	0.86	0.89	0.86
ROC	0.92	0.92	0.94	0.92

(c) Dataset 3 - KNN Model

Dataset	1	2	3	4
Accuracy	0.96	0.96	0.96	0.97
F1	0.90	0.90	0.90	0.92
Precision	0.94	0.95	0.94	0.96
Recall	0.86	0.86	0.87	0.89
ROC	0.92	0.92	0.93	0.94

(d) Dataset 4 - KNN Model

Table 5.8: MLP model cross-performance results

Dataset	1	2	3	4
Accuracy	0.97	0.96	0.96	0.96
F1	0.92	0.89	0.89	0.89
Precision	0.97	0.95	0.94	0.95
Recall	0.88	0.85	0.85	0.85
ROC	0.93	0.92	0.92	0.92

(a) Dataset 1 - MLP Model

Dataset	1	2	3	4
Accuracy	0.96	0.97	0.96	0.96
F1	0.90	0.91	0.90	0.90
Precision	0.95	0.97	0.94	0.95
Recall	0.85	0.87	0.85	0.85
ROC	0.92	0.93	0.92	0.92

(b) Dataset 2 - MLP Model

Dataset	1	2	3	4
Accuracy	0.96	0.96	0.96	0.96
F1	0.89	0.89	0.90	0.88
Precision	0.89	0.89	0.90	0.89
Recall	0.88	0.88	0.91	0.88
ROC	0.93	0.93	0.94	0.93

(c) Dataset 3 - MLP Model

Dataset	1	2	3	4
Accuracy	0.95	0.95	0.95	0.95
F1	0.85	0.85	0.85	0.86
Precision	0.97	0.97	0.97	0.98
Recall	0.75	0.75	0.75	0.76
ROC	0.87	0.87	0.87	0.88

(d) Dataset 4 - MLP Model

Table 5.9: RF model cross-performance results

Dataset	1	2	3	4
Accuracy	0.99	0.97	0.97	0.97
F1	0.98	0.91	0.91	0.91
Precision	0.99	0.98	0.97	0.98
Recall	0.96	0.85	0.85	0.85
ROC	0.98	0.92	0.92	0.92

(a) Dataset 1 - RF Model

Dataset	1	2	3	4
Accuracy	0.96	0.99	0.97	0.97
F1	0.90	0.97	0.91	0.90
Precision	0.98	0.99	0.97	0.98
Recall	0.83	0.96	0.85	0.84
ROC	0.91	0.98	0.92	0.91

(b) Dataset 2 - RF Model

Dataset	1	2	3	4
Accuracy	0.96	0.97	0.99	0.97
F1	0.90	0.90	0.97	0.90
Precision	0.98	0.98	0.99	0.98
Recall	0.83	0.84	0.96	0.83
ROC	0.91	0.91	0.98	0.91

(c) Dataset 3 - RF Model

Dataset	1	2	3	4
Accuracy	0.96	0.97	0.97	0.99
F1	0.90	0.91	0.90	0.97
Precision	0.98	0.98	0.97	0.99
Recall	0.84	0.84	0.85	0.95
ROC	0.91	0.92	0.92	0.97

(d) Dataset 4 - RF Model

Table 5.10: SVM model cross-performance results

Dataset	1	2	3	4
Accuracy	0.97	0.97	0.97	0.97
F1	0.93	0.92	0.91	0.91
Precision	0.98	0.97	0.96	0.96
Recall	0.89	0.87	0.87	0.87
ROC	0.94	0.93	0.93	0.93

(a) Dataset 1 - SVM Model

Dataset	1	2	3	4
Accuracy	0.97	0.98	0.97	0.97
F1	0.91	0.94	0.91	0.91
Precision	0.97	0.98	0.96	0.96
Recall	0.86	0.89	0.86	0.86
ROC	0.92	0.94	0.93	0.92

(b) Dataset 2 - SVM Model

Dataset	1	2	3	4
Accuracy	0.97	0.97	0.98	0.97
F1	0.91	0.92	0.94	0.91
Precision	0.97	0.97	0.98	0.97
Recall	0.85	0.87	0.89	0.87
ROC	0.92	0.93	0.94	0.93

(c) Dataset 3 - SVM Model

Dataset	1	2	3	4
Accuracy	0.97	0.97	0.97	0.97
F1	0.91	0.91	0.91	0.93
Precision	0.97	0.96	0.96	0.98
Recall	0.85	0.86	0.86	0.88
ROC	0.92	0.93	0.93	0.94

(d) Dataset 4 - SVM Model

Chapter 6

Applications

Chapter 6

Applications

The application is intended to be used by medical staff in COVID-19 wards. It is primarily built to detect arrhythmia in COVID-19 patients.

6.1 Detection of Arrhythmia

In the early phase of the COVID-19 pandemic, the mortality rate was extremely high. It was partly due to undetected comorbidities in patients. Undetected conditions were the primary cause of severe cases of COVID-19 infection. These conditions were respiratory disorders, cardiovascular disorders, diabetic disorders, etc. Earlier detection of these conditions will help in better case management. The project's application is to detect cardiovascular disorders. For this purpose, the presence of an arrhythmia can be used as a primary sign.

Arrhythmia can be detected from the patient's ECG reports. This process is automated with the application of a supervised learning system. This system allows medical staff to train and select the best-suited model for arrhythmia detection, and help them diagnose such patients. While the primary goal of this project is to be used in COVID-19 cases, it can also be used for regular patients.

6.2 Detection of anomalies in medical field

Machine learning is widely used for various tasks in the medical field. Supervised learning is used as diagnosing tool in a few cases. For example, epilepsy is diagnosed by studying scans obtained from the patient's brain. The proposed system can perform an initial screening to detect any anomalies. If such an anomaly is present, it will allow medical personnel to diagnose the patient efficiently.

The application of the system is not limited to diagnosis only, as it can be used for research purposes. These uses are filtering out unwanted data and detecting anomalies. This anomaly detection can be used in research data for discovery purposes. This will increase the productivity of research and lead to rapid development.

6.3 Model Training and Predictions

The system is capable of training models and selecting models. This system only needs a well-labeled dataset, and it will automatically select the best model for the task. The system can be used by a person without any technical or data science knowledge. The training process can be fine-tuned by adjusting performance weightage. The stored model can be used on similar datasets for prediction purposes.

Chapter 7

Conclusion And Future Scope

Chapter 7

Conclusion And Future Scope

7.1 Conclusion

As the number of COVID-19 are still increasing, the mortality rate is still rising. The mortality rate is higher in cases with comorbidities. Cardiovascular diseases are few of the comorbidities that drive this mortality rate. These diseases can be diagnosed earlier with a machine learning system. In this project we built an automated model training and selection system. This system can be used for predicting the presence of arrhythmia, which is the most common symptom in cardiovascular diseases.

The system uses supervised learning algorithms to generate models. This leads to efficient training and higher accuracy in predictions. The system can be fine tuned with by users, or it can be directly used without any formal training. The system showed good performance for similar datasets, so it can be used for general prediction purposes too.

7.2 Future Scope

The current system only provides solutions to binary classification problems, but it can be used for multiclass classification problems. The provision of user created model templates can be done, with user provided performance modification. The system can be connected directly to the hospital servers to train models from patient records directly with the help of the RPA system.

References

[1] E. P. B. Mulia, I. Maghfirah, D. A. Rachmi, and R. Julario, "Atrial arrhythmia and its association with covid-19 outcome: a pooled analysis," *Diagnosis*, 2021.

[2] A.-I. Imbrea, "Automated machine learning techniques for data streams," *arXiv preprint arXiv:2106.07317*, 2021.

[3] H. Yarmohammadi, J. P. Morrow, J. Dizon, A. Biviano, F. Ehlert, D. Saluja, M. Waase, P. Elias, T. J. Poterucha, J. Berman *et al.*, "Frequency of atrial arrhythmia in hospitalized patients with covid-19," *The American Journal of Cardiology*, vol. 147, pp. 52–57, 2021.

[4] J.-S. Huang, B.-Q. Chen, N.-Y. Zeng, X.-C. Cao, and Y. Li, "Accurate classification of ecg arrhythmia using mowpt enhanced fast compression deep learning networks," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–18, 2020.

[5] A. Rajak, A. K. Shrivastava, and Vidushi, "Applying and comparing machine learning classification algorithms for predicting the results of students," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 23, no. 2, pp. 419–427, 2020.

[6] S. Babapoor-Farrokhran, R. T. Rasekhi, D. Gill, S. Babapoor, and A. Amanullah, "Arrhythmia in covid-19," *SN Comprehensive Clinical Medicine*, pp. 1–6, 2020.

[7] M. K. Siddiqui, R. Morales-Menendez, X. Huang, and N. Hussain, "A review of epileptic seizure detection using machine learning classifiers," *Brain informatics*, vol. 7, pp. 1–18, 2020.

[8] C. K. Jha and M. H. Kolekar, "Cardiac arrhythmia classification using tunable q-wavelet transform based features and support vector machine classifier," *Biomedical Signal Processing and Control*, vol. 59, p. 101875, 2020.

[9] A. Beri and K. Kotak, "Cardiac injury, arrhythmia, and sudden death in a covid-19 patient," *HeartRhythm case reports*, vol. 6, no. 7, pp. 367–369, 2020.

[10] H. G. Ren, X. Guo, L. Tu, Q. Hu, K. Blighe, L. B. Safdar, J. Stebbing, S. D. Weiner, M. S. Willis, F. R. Rosendaal *et al.*, "Clinical characteristics and risk factors for myocardial injury and arrhythmia in covid-19 patients," *medRxiv*, 2020.

[11] Q. Liu, H. Chen, and Q. Zeng, "Clinical characteristics of covid-19 patients with complication of cardiac arrhythmia," *The Journal of Infection*, vol. 81, no. 3, p. e6, 2020.

[12] Z. Sun, C. Wang, Y. Zhao, and C. Yan, "Multi-label ecg signal classification based on ensemble classifier," *IEEE Access*, vol. 8, pp. 117 986–117 996, 2020.

[13] Y. Liu, Y. Lou, and S. Huang, "Parallel algorithm of flow data anomaly detection based on isolated forest," in *2020 International Conference on Artificial Intelligence and Electromechanical Automation (AIEA)*. IEEE, 2020, pp. 132–135.

[14] A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, G. H. Tison, C. Bourn, M. P. Turakhia, and A. Y. Ng, "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network," *Nature medicine*, vol. 25, no. 1, pp. 65–69, 2019.

[15] G. Sannino and G. De Pietro, "A deep learning approach for ecg-based heartbeat classification for arrhythmia detection," *Future Generation Computer Systems*, vol. 86, pp. 446–455, 2018.

[16] G. Malkomes, C. Schaff, and R. Garnett, "Bayesian optimization for automated model selection," in *Workshop on Automatic Machine Learning*. PMLR, 2016, pp. 41–47.

[17] T. Chen, E. B. Mazomenos, K. Maharatna, S. Dasmahapatra, and M. Niranjan, "Design of a low-power on-body ecg classifier for remote cardiovascular monitoring systems," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 3, no. 1, pp. 75–85, 2013.

[18] V. Calcagno and C. de Mazancourt, "glmulti: An r package for easy automated model selection with (generalized) linear models," *Journal of Statistical Software*, vol. 34, no. 12, p. 1–29, 2010. [Online]. Available: https://www.jstatsoft.org/index.php/jss/article/view/v034i12

[19] N. Atanasova, L. Todorovski, S. Džeroski, and B. Kompare, "Application of automated model discovery from data and expert knowledge to a real-world domain: Lake glumsø," *Ecological Modelling*, vol. 212, pp. 92–98, 03 2008.

[20] W. K. Lei, B. N. Li, M. C. Dong, and M. I. Vai, "Afc-ecg: An adaptive fuzzy ecg classifier," in *Soft computing in industrial applications*. Springer, 2007, pp. 189–199.

[21] M. I. Owis, A. H. Abou-Zied, A.-B. Youssef, and Y. M. Kadah, "Study of features based on nonlinear dynamical modeling in ecg arrhythmia detection and classification," *IEEE transactions on Biomedical Engineering*, vol. 49, no. 7, pp. 733–736, 2002.

[22] J.-H. Lee and C.-J. Lin, "Automatic model selection for support vector machines," *online*, 2000.

[23] S. Fazeli, "ECG heartbeat categorization dataset," accessed on 15 Dec 2020. [Online]. Available: https://www.kaggle.com/shayanfazeli/heartbeat

List of Publications

[1]   Y. Ketkar and S. Gawade, "Effectiveness of Robotic Process Automation for data mining using UiPath", in *2021 International Conference on Artificial Intelligence and Smart Systems(ICAIS)*. IEEE, 2021,pp. 864–867.

ACKNOWLEDGEMENT

I remain immensely obliged to **Dr. Sushopti Gawade**, for providing me with the idea of this topic, and for her invaluable support in garnering resource for me either by way of information or computers also her guidance and supervision which made this seminar happen.

I would like to thank my college Pillai's Institute of Information Technology, New Panvel, our principal **Dr. Sandeep Joshi**, M.E. Coordinator **Dr. Prashant Nitnaware** and the H.O.D. of Information Technology Department **Dr. Satishkumar Varma** for their invaluable support.

I would like to say that it has indeed been a fulfilling experience for working on this seminar topic.

<div style="text-align:right">

Yashodhan Prakash Ketkar
</div>