Wearable Device For Heart Attack Detection and Prediction Using IOT and Machine Learning

Research Proposal · May 2020

DOI: 10.6084/m9.figshare.12382472

2 authors:

Theja Kumar Moolinti
R.M.D Engineering College

Harsha varma Rudhraraju
R.M.D Engineering College

WEARABLE DEVICE FOR HEART ATTACK PREDICTION AND ALERT USING MACHINE LEARNING

PROJECT REPORT

Submitted by

R. Harsha Varma	**(111516106131)**
Theja kumar M	**(111516106156)**
V Sri Ganesh	**(111516106164)**

BACHELOR OF ENGINEERING

IN

Electronics and Communication Engineering

R.M.D ENGINEERING COLLEGE

ANNA UNIVERSITY::CHENNAI 600025

MAY 2020

ANNA UNIVERSITY::CHENNAI 600025

BONAFIDE CERTIFICATE

Certified that this project report titled **"WEARABLE DEVICE FOR HEART ATTACK PREDICTION AND ALERT USING MACHINE LEARNING"** is the bonafide work of **Mr. R. Harsha Varma(111516106131), Mr. Theja Kumar M(111516106156) and Mr. V. Sri Ganesh(111516106164)**. who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.K. Helen Prabha, Ph.D	**Dr. D. Rukmanidevi, Ph.D**
HEAD OF THE DEPARTMENT	**Year Coordinator**
Professor	Professor
Electronics and Communication	Electronics and Communication
R.M.D. Engineering College	R.M.D. Engineering College
R.S.M. Nagar, Kavaraipettai-601 204.	R.S.M. Nagar, Kavaraipettai- 601 204

ACKNOWLEDGEMENT

ABSTRACT

Globally, heart diseases are the number one cause of death. About 80% of deaths occurred in lowand middle income countries. If current trends are allowed to continue, by 2030 an estimated 23.6 million people will die from cardiovascular disease (mainly from heart attacks and strokes). With the increase in popularity of smart wearable devices, an opportunity to provide an Internet of Things (IoT) solution has become more available. Unfortunately, out of hospital survival rates are low for people suffering from sudden cardiac arrests. The objective of this research is to present a multisensory system using a smart IoT system that can collect Body Area Sensor (BAS) and Bio marker sensor data to provide early warning of an impending cardiac arrest. The goal is to design and develop an integrated smart IoT system with a low power communication module to discreetly collect heart rates and body temperatures using a smartphone without it impeding on everyday life. This research introduces the use of signal processing and machine-learning techniques for sensor data analytics to identify predict and/or sudden cardiac arrests with a high accuracy.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

KEYWORDS	ABBREVIATION
ECG	Electro Cardio Gram
SVM	Support Vector Machine
NB	Naïve Bayes
UART	Universal Asynchronous Receiver Transmitter
CAD	Computer Aided Detection
FP	False Positive

CHAPTER 1

INTRODUCTION

1.1 General:

ECGs:

Electrocardiography (ECG or EKG) is the process of recording the electrical activity of the heart over a period of time using electrodes placed on the skin. These electrodes detect the tiny electrical changes on the skin that arise from the heart muscle's electrophysiologic pattern of depolarizing and repolarizing during each heartbeat. It is a very commonly performed cardiology test.

In a conventional 12-lead ECG, ten electrodes are placed on the patient's limbs and on the surface of the chest. The overall magnitude of the heart's electrical potential is then measured from twelve different angles ("leads") and is recorded over a period of time (usually ten seconds). In this way, the overall magnitude and direction of the heart's electrical depolarization is captured at each moment throughout the cardiac cycle. The graph of voltage versus time produced by this noninvasive medical procedure is an **electrocardiogram**.

During each heartbeat, a healthy heart has an orderly progression of depolarization that starts with pacemaker cells in the sinoatrial node, spreads out through the atrium, passes through the atrioventricular node down into the bundle of His and into the Purkinje fibers, spreading down and to the left throughout the ventricles. This orderly pattern of depolarization gives rise to the characteristic ECG tracing. To the trained clinician, an ECG conveys a large amount of information about the structure of the heart and the function of its electrical conduction system. Among other things, an ECG can be used to measure the rate and rhythm of heartbeats, the size and position of the heart chambers, the presence of any damage to the heart's muscle cells or conduction system, the effects of cardiac drugs, and the function of implanted pacemakers.

Medical Uses:

The overall goal of performing electrocardiography is to obtain information about the structure and function of the heart. Medical uses for this information are varied and generally relate to having a need for knowledge of the structure and/or function. Some indications for performing electrocardiography include:

- Suspected myocardial infarction (heart attack) or new chest pain
- Suspected pulmonary embolism or new shortness of breath
- A third heart sound, fourth heart sound, a cardiac murmur or other findings to suggest structural heart disease
- Perceived cardiac dysrhythmias either by pulse or palpitations
- Monitoring of known cardiac dysrhythmias
- Fainting or collapse
- Seizures
- Monitoring the effects of a heart medication (e.g. drug-induced QT prolongation)
- Assessing severity of electrolyte abnormalities, such as hyperkalemia
- Hypertrophic cardiomyopathy screening in adolescents as part of a sports physical out of concern for sudden cardiac death (varies by country)

- Perioperative monitoring in which any form of anesthesia is involved (e.g. monitored anesthesia care, general anesthesia); typically both intraoperative and postoperative
- As a part of a pre-operative assessment some time before a surgical procedure (especially for those with known cardiovascular disease or who are undergoing invasive or cardiac, vascular or pulmonary procedures, or who will receive general anesthesia)
- Cardiac stress testing
- Computed tomography angiography (CTA) and Magnetic resonance angiography (MRA) of the heart (ECG is used to "gate" the scanning so that the anatomical position of the heart is steady)
- Biotelemetry of patients for any of the above reasons and such monitoring can include internal and external defibrillators and pacemakers

The United States Preventive Services Task Force does not recommend electrocardiography for routine screening procedure in patients without symptoms and those at low risk for coronary heart disease. This is because an ECG may falsely indicate the existence of a problem, leading to misdiagnosis, the recommendation of invasive procedures, or overtreatment. However, persons employed in certain critical occupations, such as aircraft pilots, may be required to have an ECG as part of their routine health evaluations.

Continuous ECG monitoring is used to monitor critically ill patients, patients undergoing general anesthesia, and patients who have an infrequently occurring cardiac dysrhythmia that would be unlikely to be seen on a conventional ten-second ECG.

Performing a 12-lead ECG in the United States is commonly performed by specialized technicians that may be certified electrocardiogram technicians. ECG interpretation is a component of many healthcare fields (nurses and physicians and cardiac surgeons being the most obvious), but anyone trained to interpret an ECG is free to do so. However, "official" interpretation is performed by a cardiologist. Certain fields such as anesthesia utilize continuous ECG monitoring, and knowledge of interpreting ECGs is crucial to their jobs.

One additional form of electrocardiography is used in clinical cardiac electrophysiology in which a catheter is used to measure the electrical activity. The catheter is inserted through the femoral vein and can have several electrodes along its length to record the direction of electrical activity from within the heart.

1.2 Objectives:

To determine abnormal heart patterns and perform Quantitative analysis of heart rate stability and pulse symmetry. Sense the Troponin levels that regulates the contraction of Heart and skeletal muscles using Cardiac Troponin sensor. To Predict the heart attack occurrence using various Machine Learning algorithms and models Continuous collection of heart rate and troponin levels to send it to the mobile phone for analysis. This Diagnostic test Kit is to be used to determine the acute myocardial infarction (AMI). These markers are thoroughly tested under defined standards of the industry to ensure their accuracy in results. Generate a Mail alert to the user and authorized presons inorder to save the patients life.

1.3 Existing Method:

In social insurance framework for patients who stays in home during post operational days checking is done either via overseer/ medical caretaker.

Ceaseless observing may not be accomplished by this system, on the grounds that anything can change in wellbeing parameter inside of part of seconds and amid that time if guardian/attendant is not in the premises causes more noteworthy harm.

So with this innovation created period where web administers the world gives a thought to add to another keen health awareness framework where time to time constant checking of the patient is accomplished.

1.4 Proposed Method:

Data Collection

Data Transmission

Analytics

Prediction

1.5 Block Diagram

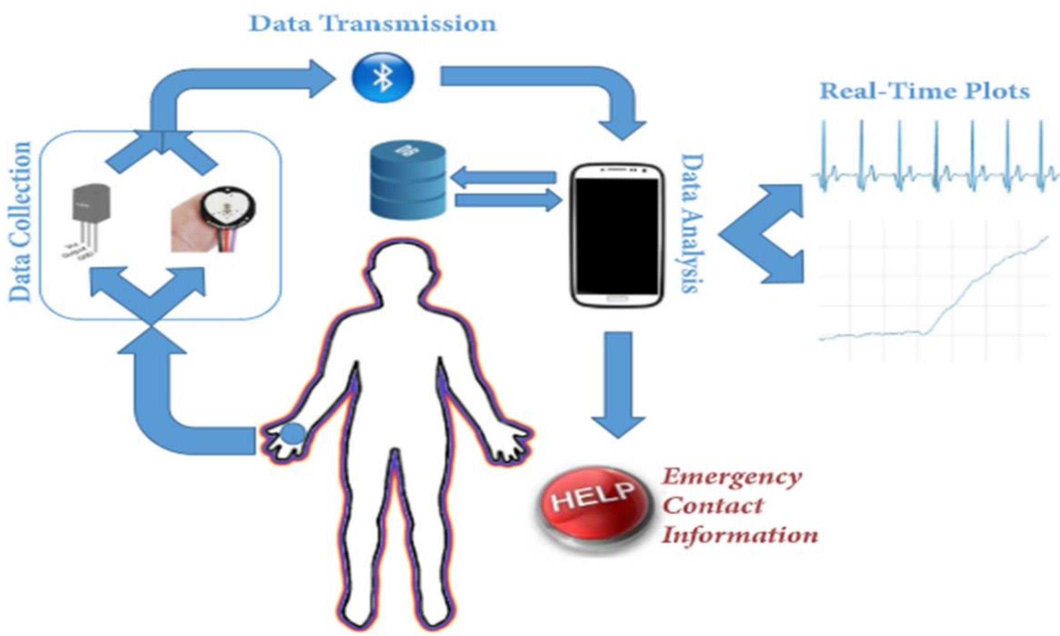


Figure 1: Proposed system architecture.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION:

A literature survey or a literature review in a project report is that section which shows the various analyses and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of the project.It is the most important part of your report as it gives you a direction in the area of your research. A literature review is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature reviews are secondary sources, and do not report new or original experimental work. Most often associated with academic-oriented literature, such reviews are found in academic journals, and are not to be confused with book reviews that may also appear in the same publication. Literature reviews are a basis for research in nearly every academic field. A narrow-scope literature review may be included as part of a peer-reviewed journal article presenting new research, serving to situate the current study within the body of the relevant literature and to provide context for the reader. In such a case, the review usually precedes the methodology and results sections of the work. Producing a literature review may also be part of graduate and post-graduate student work, including in the preparation of a thesis, dissertation, or a journal article. Literature reviews are also common in a research proposal or prospectus (the document that is approved before a student formally begins a dissertation or thesis). It helps you set a goal for your analysis - thus giving you your problem statement. When you write a literature review in respect of your project, you have to write the researches made by various analysts - their methodology (which is basically their abstract) and the conclusions they have arrived at. You should also give an account of how this research has influenced your thesis. Descriptive papers may or may not contain reviews, but analytical papers will contain reviews. A literature review must contain at least 5 - 7 published researches in your field of interest.

2.2 Fast and Adaptive Detection of Pulmonary Nodules in Thoracic CT Images Using a Hierarchical Vector Quantization Scheme

Abstract:

Computer-aided detection (CADe) of pulmonary nodules is critical to assisting radiologists in early identification of lung cancer from computed tomography (CT) scans. This paper proposes a novel CADe system based on a hierarchical vector quantization (VQ) scheme. Compared with the commonly-used simple thresholding approach, the high-level VQ yields a more accurate segmentation of the lungs from the chest volume. In identifying initial nodule candidates (INCs) within the lungs, the low-level VQ proves to be effective for INCs detection and segmentation, as well as computationally efficient compared to existing approaches. False-positive (FP) reduction is conducted via rule-based filtering operations in combination with a feature-based support vector machine classifier. The proposed system was validated on 205 patient cases from the publically available online Lung Image Database Consortium database, with each case having at least one juxta-pleural nodule annotation. Experimental results demonstrated that our CADe system obtained an overall sensitivity of 82.7% at a specificity of 4 FPs/scan. Especially for the performance on juxta-pleural nodules, we observed 89.2% sensitivity at 4.14 FPs/scan. With respect to comparable CADe systems, the proposed system shows outperformance and demonstrates its potential for fast and adaptive detection of pulmonary nodules via CT imaging.

Authors: S. Ghwanmeh, A. Mohammad and A. Al-Ibrahim

Published in: IEEE Journal of Biomedical and Health Informatics (Volume: 19, Issue: 2, March 2015)

2.3 Healthcare based on IoT using Raspberry Pi

Abstract:

This paper explains a Raspberry pi controlled remote monitoring system. Raspberry Pi is a credit card sized single board computer with ARM11 microprocessor. In this study, a system is designed to continuously monitor the Electrocardiogram (ECG) and other vital parameters. This data is stored in a database and can be displayed in a website that can be accessed only by authorized personnel. This idea is familiar however; this paper presents a substantive and inexpensive method using Raspberry pi. The primary task of this system is to update the data to the database and alert the doctors for any aberrancy. The former is accomplished by using MySQLdb module to link Raspberry pi to the database whereas the latter is achieved by the combination of Raspberry Pi and GSM module. This system has much future scope as the data gathered by monitoring is so valuable and can be used for scientific research by the medical community. By determining the patterns in the parameters observed, the nature of disease can be predicted. The paper mainly emphasizes on the system design and the algorithm used to accomplish the task. The obtained results are presented.

Authors: M. Surya Deekshith Gupta, Vamsikrishna Patchava and Virginia Menezes

Published in: IEEE – ACCESS, vol. 8, no. 1, pp. 30–41/2017

2.4 Innovative Artificial Neural Networks-Based Decision Support System for Heart Diseases Diagnosis

Abstract:

Heart diagnosis is not always possible at every medical center, especially in the rural areas where less support and care, due to lack of advanced heart diagnosis equipment. Also, physician intuition and experience are not always sufficient to achieve high quality medical procedures results. Therefore, medical errors and undesirable results are reasons for a need for unconventional computer-based diagnosis systems, which in turns reduce medical fatal errors,

increasing the patient safety and save lives. The proposed solution, which is based on an Artificial Neural Networks (ANNs), provides a decision support system to identify three main heart diseases: mitral stenosis, aortic stenosis and ventricular septal defect. Furthermore, the system deals with an encouraging opportunity to develop an operational screening and testing device for heart disease diagnosis and can deliver great assistance for clinicians to make advanced heart diagnosis. Using real medical data, series of experiments have been conducted to examine the performance and accuracy of the proposed solution. Compared results revealed that the system performance and accuracy are acceptable, with a heart diseases classification accuracy of 92%.

Authors: Sameh Ghwanmeh, Adel Mohammad and Ali Al-Ibrahim

Published in: IEEE Journal of Innovative Artificial Neural Networks-Based Decision Support System for Heart Diseases Diagnosis in November, 2013.

2.5 A Computational Geometry Approach to Automated Pulmonary Fissure Segmentation in CT Examinations

Abstract:

Identification of pulmonary fissures, which form the boundaries between the lobes in the lungs, may be useful during clinical interpretation of computed tomography (CT) examinations to assess the early presence and characterization of manifestation of several lung diseases. Motivated by the unique nature of the surface shape of pulmonary fissures in 3-D space, we developed a new automated scheme using computational geometry methods to detect and segment fissures depicted on CT images. After a geometric modeling of the lung volume using the marching cubes algorithm, Laplacian smoothing is applied iteratively to enhance pulmonary fissures by depressing nonfissure structures while smoothing the surfaces of lung fissures. Next, an extended Gaussian image based procedure is used to locate the fissures in a statistical manner that approximates the fissures using a set of plane ldquopatchesrdquo. This approach has several advantages such as independence of anatomic knowledge of the lung structure except the surface shape of fissures, limited sensitivity to other lung structures, and ease of implementation. The scheme performance was evaluated by two experienced thoracic radiologists using a set of 100 images (slices) randomly selected from 10 screening CT examinations. In this preliminary evaluation 98.7% and 94.9% of scheme segmented fissure voxels are within 2 mm of the fissures marked independently by two radiologists in the testing image dataset. Using the scheme detected fissures as reference, 89.4% and 90.1% of manually marked fissure points have distance les2 mm to the reference suggesting a possible under-segmentation of the scheme. The case-based root mean square (rms) distances (ldquoerrorsrdquo) between our scheme and the radiologist ranged from 1.48plusmn0.92 to 2.04plusmn3.88 mm. The discrepancy of fissure detection results between the automated scheme and either radiologist is smaller in this dataset than the interreader variability.

Authors:J. Pu, J. K. Leader, B. Zheng, F. Knollmann, C. Fuhrman, F. C. Sciurba, and D. Gur

Published in: IEEE Transactions on Medical Imaging (Volume: 28, Issue: 5, May 2009)

2.6 Automated detection of pulmonary nodules in helical CT images based on an improved template-matching technique

Abstract:

The purpose of this study is to develop a technique for computer-aided diagnosis (CAD) systems to detect lung nodules in helical X-ray pulmonary computed tomography (CT) images. The authors propose a novel template-matching technique based on a genetic algorithm (GA) template matching (GATM) for detecting nodules existing within the lung area; the GA was used to determine the target position in the observed image efficiently and to select an adequate template image from several reference patterns for quick template matching. In addition, a conventional template matching was employed to detect nodules existing on the lung wall area, lung wall template matching (LWTM), where semicircular models were used as reference patterns; the semicircular models were rotated according to the angle of the target point on the contour of the lung wall. After initial detecting candidates using the two template-matching methods, the authors extracted a total of 13 feature values and used them to eliminate false-positive findings. Twenty clinical cases involving a total of 557 sectional images were used in this study. 71 nodules out of 98 were correctly detected by the authors' scheme (i.e., a detection rate of about 72%), with the number of false positives at approximately 1.1/sectional image. The authors' present results show that their scheme can be regarded as a technique for CAD systems to detect nodules in helical CT pulmonary images.

Authors: Y. Lee, T. Hara,H. Fujita, S. Itoh, and T. Ishigaki

Published in: IEEE Transactions on Medical Imaging (Volume: 20, Issue: 7, July 2001)

2.7 A novel approach to extract colon lumen from CT images for virtual colonoscopy

Abstract:

An automatic method has been developed for segmentation of abdominal computed tomography (CT) images for virtual colonoscopy obtained after a bowel preparation of a low-residue diet with ingested contrast solutions to enhance the image intensities of residual colonic materials. Removal of the enhanced materials

was performed electronically by a computer algorithm. The method is a multistage approach that employs a modified self-adaptive on-line, vector quantization technique for a low-level image classification and utilizes a region-growing strategy for a high-level feature extraction. The low-level classification labels each voxel based on statistical analysis of its three-dimensional intensity vectors consisting of nearby voxels. The high-level processing extracts the labeled stool, fluid and air voxels within the colon, and eliminates bone and lung voxels which have similar image intensities as the enhanced materials and air, but are physically separated from the colon. This method was evaluated by volunteer studies based on both objective and subjective criteria. The validation demonstrated that the method has a high reproducibility and repeatability and a small error due to partial volume effect. As a result of this electronic colon cleansing, routine physical bowel cleansing prior to virtual colonoscopy may not be necessary.

Authors: D. Chen, Z. Liang, M. R.Wax, L. Li, B. Li, and A. E. Kaufman

Published in: IEEE Transactions on Medical Imaging (Volume: 19, Issue: 12, Dec. 2000)

2.8 Vector quantization-based automatic detection of pulmonary nodules in thoracic CT images

Abstract:

Computer-aided detection (CADe) of pulmonary nodules from computer tomography (CT) scans is critical for assisting radiologists to identify lung lesions at an early stage. In this paper, we propose a novel CADe system for lung nodule detection based on a vector quantization (VQ) approach. Compared to existing CADe systems, the extraction of lungs from the chest CT image is fully automatic, and the detection and segmentation of initial nodule candidates (INCs) within the lung volume is fast and accurate due to the self-adaptive nature of VQ algorithm.

False positives in the detected INCs are reduced by rule-based pruning in combination with a feature-based support vector machine classifier. We validate the proposed approach on 60 CT scans from a publicly available database. Preliminary results show that our CADe system is effective to detect nodules with a sensitivity of 90.53 % at a specificity level of 86.00%.

Authors: H. Han, L. Li, F. Han, H. Zhang, W. Moore, and Z. Liang

Published in: Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2013 IEEE

2.9 Tissue classification based on 3D local intensity structures for volume rendering

Abstract:

This paper describes a novel approach to tissue classification using three-dimensional (3D) derivative features in the volume rendering pipeline. In conventional tissue classification for a scalar volume, tissues of interest are characterized by an opacity transfer function defined as a one-dimensional (1D) function of the original volume intensity. To overcome the limitations inherent in conventional 1D opacity functions, we propose a tissue classification method that employs a multidimensional opacity function, which is a function of the 3D derivative features calculated from a scalar volume as well as the volume intensity. Tissues of interest are characterized by explicitly defined classification rules based on 3D filter responses highlighting local structures, such as edge, sheet, line, and blob, which typically correspond to tissue boundaries, cortices, vessels, and nodules, respectively, in medical volume data. The 3D local structure filters are formulated using the gradient vector and Hessian matrix of the volume intensity function combined with isotropic Gaussian blurring. These filter responses and the original intensity define a multidimensional feature space in which multichannel

tissue classification strategies are designed. The usefulness of the proposed method is demonstrated by comparisons with conventional single-channel classification using both synthesized data and clinical data acquired with CT (computed tomography) and MRI (magnetic resonance imaging) scanners. The improvement in image quality obtained using multichannel classification is confirmed by evaluating the contrast and contrast-to-noise ratio in the resultant volume-rendered images with variable opacity values.

Authors: Y. Sato,C. F.Westin,A.Bhalerao, S.Nakajima, N. Shiraga, S. Tamura, and R. Kijinis

Published in: IEEE Transactions on Visualization and Computer Graphics (Volume: 6, Issue: 2, Apr-Jun 2000)

2.10 Pulmonary Nodule Detection in CT Images: False Positive Reduction Using Multi-View Convolutional Networks

Abstract:

We propose a novel Computer-Aided Detection (CAD) system for pulmonary nodules using multi-view convolutional networks (ConvNets), for which discriminative features are automatically learnt from the training data. The network is fed with nodule candidates obtained by combining three candidate detectors specifically designed for solid, subsolid, and large nodules. For each candidate, a set of 2-D patches from differently oriented planes is extracted. The proposed architecture comprises multiple streams of 2-D ConvNets, for which the outputs are combined using a dedicated fusion method to get the final classification. Data augmentation and dropout are applied to avoid overfitting. On 888 scans of the publicly available LIDC-IDRI dataset, our method reaches high detection sensitivities of 85.4% and 90.1% at 1 and 4 false positives per scan, respectively. An additional evaluation on independent datasets from the

ANODE09 challenge and DLCST is performed. We showed that the proposed multi-view ConvNets is highly suited to be used for false positive reduction of a CAD system.

Authors: Arnaud A. A. Setio, Francesco Ciompi, Geert Litjens, Paul Gerke

Published in: IEEE Transactions on Medical Imaging (Volume: 35, Issue: 5, May 2016)

2.11 Gradient-based learning applied to document recognition

Abstract:

Multilayer neural networks trained with the back-propagation algorithm constitute the best example of a successful gradient based learning technique. Given appropriate network architecture, gradient-based learning algorithms can be used to synthesize a complex decision surface that can classify high-dimensional patterns, such as handwritten characters, with minimal preprocessing. This paper reviews various methods applied to handwritten character recognition and compares them on a standard handwritten digit recognition task. Convolutional neural networks, which are specifically designed to deal with the variability of 2D shapes, are shown to outperform all other techniques. Real-life document recognition systems are composed of multiple modules including field extraction, segmentation recognition, and language modeling. A new learning paradigm, called graph transformer networks (GTN), allows such multi module systems to be trained globally using gradient-based methods so as to minimize an overall performance measure. Two systems for online handwriting recognition are described. Experiments demonstrate the advantage of global training, and the flexibility of graph transformer networks. A graph transformer network for reading a bank cheque is also described. It uses convolutional neural network character recognizers combined with global training techniques to provide record accuracy on business and personal cheques. It is deployed commercially and reads several million cheques per day.

Authors: Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner

Published in: Proceedings of the IEEE (Volume: 86, Issue: 11, Nov 1998)

2.12 Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans

Abstract:

Convolutional neural networks (CNNs) have emerged as the most powerful technique for a range of different tasks in computer vision. Recent work suggested that CNN features are generic and can be used for classification tasks outside the exact domain for which the networks were trained. In this work we use the features from one such network, Over Feat, trained for object detection in natural images, for nodule detection in computed tomography scans. We use 865 scans from the publicly available LIDC data set, read by four thoracic radiologists. Nodule candidates are generated by a state-of-the-art nodule detection system. We extract 2D sagittal, coronal and axial patches for each nodule candidate and extract 4096 features from the penultimate layer of Over Feat and classify these with linear support vector machines. We show for various configurations that the off-the-shelf CNN features perform surprisingly well, but not as good as the dedicated detection system. When both approaches are combined, significantly better results are obtained than either approach alone. We conclude that CNN features have great potential to be used for detection tasks in volumetric medical data.

Authors: B. van Ginneken, A. A. A. Setio, C. Jacobs, and F. Ciompi

Published in: Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on

2.13 Discriminative Feature Co-Occurrence Selection for Object Detection

Abstract:

This paper describes an object detection framework that learns the discriminative co-occurrence of multiple features. Feature co-occurrences are automatically found by sequential forward selection at each stage of the boosting process. The selected feature co-occurrences are capable of extracting structural similarities of target objects leading to better performance. The proposed method is a generalization of the framework proposed by Viola and Jones, where each weak classifier depends only on a single feature. Experimental results obtained using four object detectors for finding faces and three different hand poses, respectively, show that detectors trained with the proposed algorithm yield consistently higher detection rates than those based on their framework while using the same number of features.

Authors: Takeshi Mita, Toshimitsu Kaneko, Bjorn Stenger

Published in: IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: 30, Issue: 7, July 2008)

CHAPTER-3

WORKING OF SYSTEM

3.1 EXISTING SYSTEM:

In social insurance framework for patients who stays in home during post operational days checking is done either via overseer/ medical caretaker. Ceaseless observing may not be accomplished by this system, on the grounds that anything can change in wellbeing parameter inside of part of seconds and amid that time if guardian/attendant is not in the premises causes more noteworthy harm.

So with this innovation created period where web administers the world gives a thought to add to another keen health awareness framework where time to time constant checking of the patient is accomplished.

DISADVANTAGES:

Patient Data is exchanged.

Patient needs to attend for every checkup.

Critical condition is unknown.

3.2 PROPOSED SYSTEM:

This system is divided into

(1) Data Collection

Patient heart rate is collected from ECG sensor and temperature sensor.

(2) Data Transmission

The data collected from sensors are transmitted to the device that performs calculations on data.

(3) Analytics

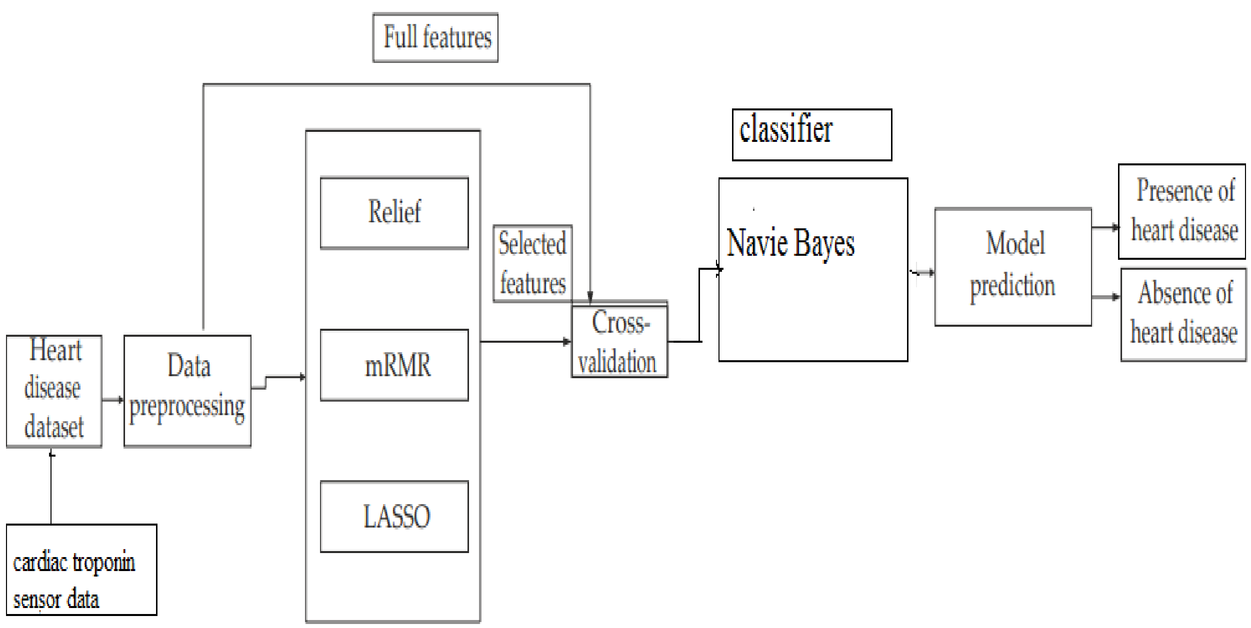
Naive Bayes algorithm is machine learning classifier that analyses the data and predicts the heart attack.

(4) Prediction

Predicted result from classifier is transmitted to UART serial port connected to the device. The baud rate of data from device to the UART serial port can be done by using bard-rate function in MATLAB. Data is sent to the cloud database so, that the patient data can be viewed by hospital staff. Wi-Fi Module is used to transmit the data collected from UART serial port to the cloud database.

Many Machine Learning Algorithms are implemented for accurate prediction of results up to 98 percent. In machine learning, naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features.

3.3 BLOCK DIAGRAM:



Block Diagram

3.4 BLOCK DIAGRAM DESCRIPTION:

3.5 DATA PROCESSING:

Heart disease data is pre-processed after collection of various records.The dataset contains a total of 303 patient records, where 6 records are with some missing values. Those 6 records have been removed from the dataset and the remaining 297 patient records are used in pre-processing.

Data processing occurs when data is collected and translated into usable information. Usually performed by a data scientist or team of data scientists, it is important for data processing to be done correctly as not to negatively affect the end product, or data output.

Data processing starts with data in its raw form and converts it into a more readable format (graphs, documents, etc.), giving it the form and context necessary to be interpreted by computers and utilized by employees throughout an organization.

3.5.1 Six stages of data processing

1. Data collection

Collecting data is the first step in data processing. Data is pulled from available sources, including data lakes and data warehouses. It is important that the data sources available are trustworthy and well-built so the data collected (and later used as information) is of the highest possible quality.

2. Data preparation

Once the data is collected, it then enters the data preparation stage. Data preparation, often referred to as “pre-processing” is the stage at which raw data is cleaned up and organized for the following stage of data processing. During preparation, raw data is diligently checked for any errors. The purpose of this step is to eliminate bad data (redundant, incomplete, or incorrect data) and begin to create high-quality data for the best business intelligence.

3. Data input

The clean data is then entered into its destination (perhaps a CRM like Salesforce or a data warehouse like Redshift), and translated into a language that it can understand. Data input is the first stage in which raw data begins to take the form of usable information.

4. Processing

During this stage, the data inputted to the computer in the previous stage is actually processed for interpretation. Processing is done using machine learning algorithms, though the process itself may vary slightly depending on the source of data being processed (data lakes, social networks, connected devices etc.) and its intended use (examining advertising patterns, medical diagnosis from connected devices, determining customer needs, etc.).

5. Data output/interpretatio

The output/interpretation stage is the stage at which data is finally usable to non-data scientists. It is translated, readable, and often in the form of graphs, videos, images, plain text, etc.). Members of the company or institution can now begin to self-serve the data for their own data analytics projects.

6. Data storage

The final stage of data processing is storage. After all of the data is processed, it is then stored for future use. While some information may be put to use immediately, much of it will serve a purpose later on. Plus, properly stored data is a necessity for compliance with data protection legislation like GDPR. When data is properly stored, it can be quickly and easily accessed by members of the organization when needed.

3.6 NAVIE BAYES THEOREM:

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

where A and B are events and P(B) ? 0.

- Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as **evidence**.
- P(A) is the **priori** of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).
- P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$P(\mathrm{A} \mid \mathrm{B}) = \frac{P(\mathrm{A\ and\ B})}{P(\mathrm{B})}$$

Fig: 3.6.1 Naïve Bayes Table

Time	Heart Rate
09:00	86
09:05	45
09:07	73
09:12	40
09:18	53
09:23	100
09:30	101

Heart Rate	No. of Times	Probability
Less than 60	3	0.4285
Between 60 and 100	2	0.2857
Greater 100	2	0.2857

3.7 SUPPORT VECTOR MACHINE

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.

The basics of Support Vector Machines and how it works are best understood with a simple example. Let's imagine we have two tags: *red* and *blue*, and our data has two features: *x* and *y*. We want a classifier that, given a pair of *(x,y)* coordinates, outputs if it's either *red* or *blue*. We plot our already labeled training data on a plane:

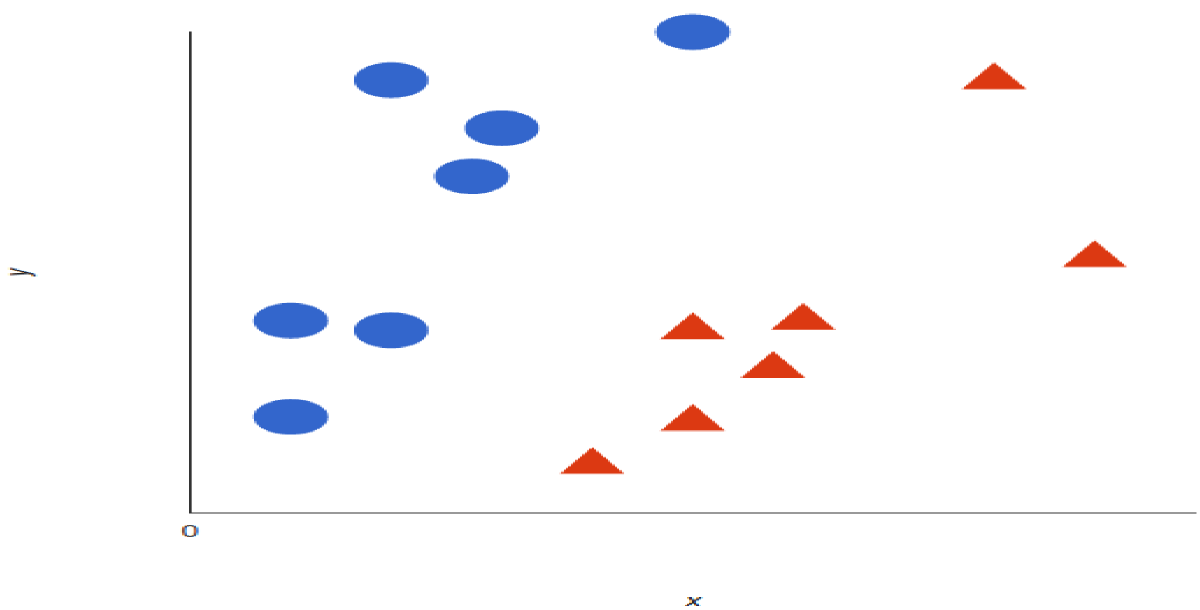


Fig: 3.7.1 Naïve Bayes Chart1

A support vector machine takes these data points and outputs the hyperplane (which in two dimensions it's simply a line) that best separates the tags. This line is the decision boundary: anything that falls to one side of it we will classify as *blue*, and anything that falls to the other as *red*.

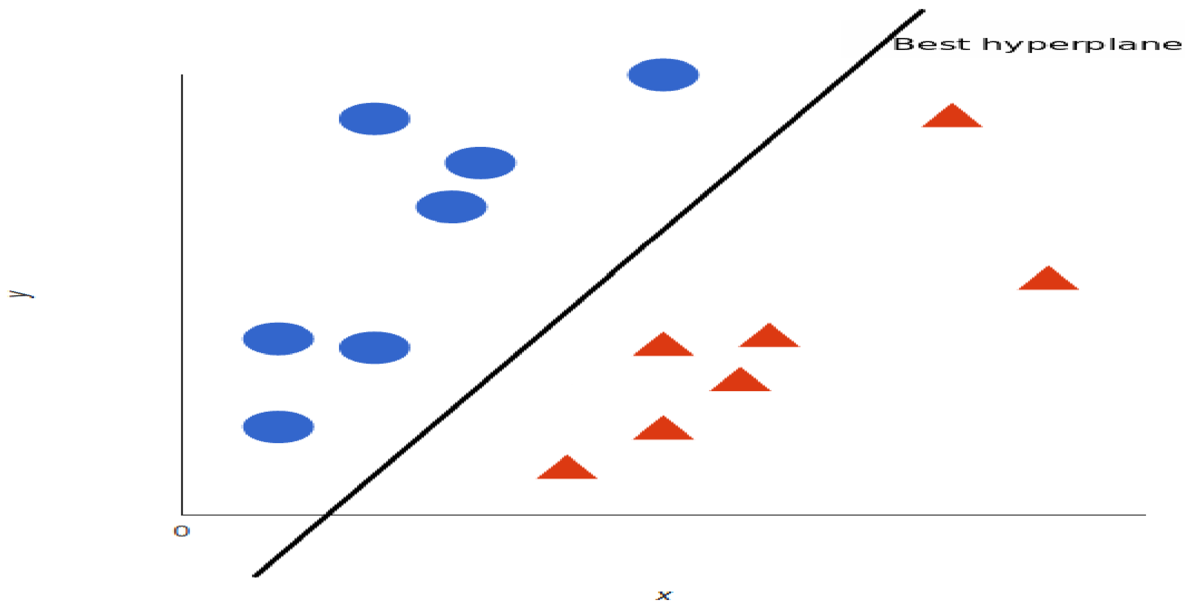


Fig: 3.7.2 Naïve Bayes Chart2

SVM that maximizes the margins from both tags. In other words: the hyperplane (remember it's a line in this case) whose distance to the nearest element of each tag is the largest.

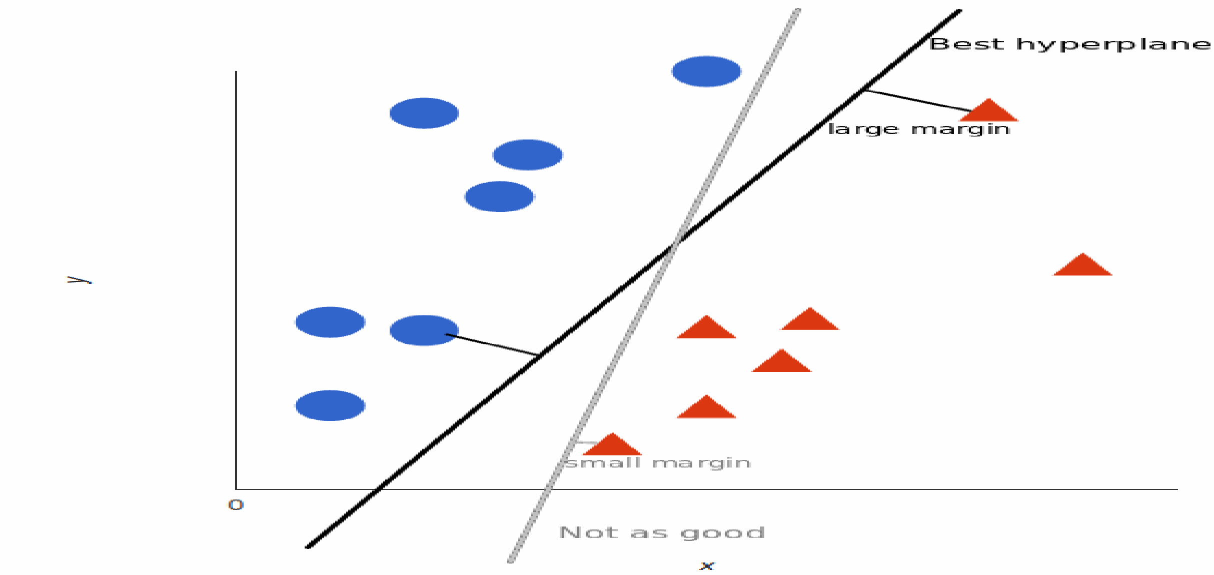


Fig: 3.7.3 Naïve Bayes Chart3

3.8 INTERNET OF THINGS

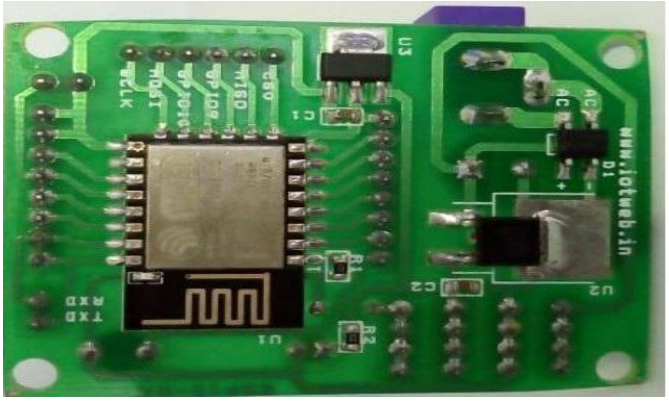


Fig: 3.8.1 Wi-Fi Module

Internet of things

The Internet of things (IoT) is the network of everyday objects — physical things embedded with electronics, software, sensors, and connectivity enabling data exchange. Basically, a little networked computer is attached to a thing, allowing information exchange to and from that thing. Be it lightbulbs, toasters, refrigerators, flower pots, watches, fans, planes, trains, automobiles, or anything else around you, a little networked computer can be combined with it to accept input (especially object control) or to gather and generate informational output (typically object status or other sensory data). This means computers will be permeating everything around us — ubiquitous embedded computing devices, uniquely identifiable, interconnected across the Internet. Because of low-cost, networkable microcontroller modules, the Internet of things is really starting to take off.

CHAPTER 4

SOFTWARE DESCRIPTION

4.1 Getting Started

If you are new to MATLAB, you should start by reading Manipulating Matrices. The most important things to learn are how to enter matrices, how to use the: (colon) operator, and how to invoke functions. After you master the basics, you should read the rest of the sections below and run the demos.

At the heart of MATLAB is a new language you must learn before you can fully exploit its power. You can learn the basics of MATLAB quickly, and mastery comes shortly after. You will be rewarded with high productivity, high-creativity computing power that will change the way you work.

Introduction - describes the components of the MATLAB system.

Development Environment - introduces the MATLAB development environment, including information about tools and the MATLAB desktop.

Manipulating Matrices - introduces how to use MATLAB to generate matrices and perform mathematical operations on matrices.

Graphics - introduces MATLAB graphic capabilities, including information about plotting data, annotating graphs, and working with images.

Programming with MATLAB - describes how to use the MATLAB language to create scripts and functions, and manipulate data structures, such as cell arrays and multidimensional arrays.

4.2 INTRODUCTION

What Is MATLAB?

MATLAB® is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Math and computation

- Algorithm development
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or FORTRAN.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB uses software developed by the LAPACK and ARPACK projects, which together represent the state-of-the-art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

4.3 The MATLAB System

The MATLAB system consists of five main parts:

4.3.1 Development Environment. This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, and browsers for viewing help, the workspace, files, and the search path.

4.3.2 The MATLAB Mathematical Function Library. This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

4.3.3 The MATLAB Language. This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

4.3.4 Handle Graphics®. This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

4.3.5 The MATLAB Application Program Interface (API). This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It include facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

4.4 DEVELOPMENT ENVIRONMENT

Introduction

This chapter provides a brief introduction to starting and quitting MATLAB, and the tools and functions that help you to work with MATLAB variables and files. For more information about the topics covered here, see the corresponding topics under Development Environment in the MATLAB documentation, which is available online as well as in print.

Starting and Quitting MATLAB

Starting MATLAB

On a Microsoft Windows platform, to start MATLAB, double-click the MATLAB shortcut icon on your Windows desktop.

On a UNIX platform, to start MATLAB, type matlab at the operating system prompt.

After starting MATLAB, the MATLAB desktop opens - see MATLAB Desktop.

You can change the directory in which MATLAB starts, define startup options including running a script upon startup, and reduce startup time in some situations.

Quitting MATLAB

To end your MATLAB session, select Exit MATLAB from the File menu in the desktop, or type quit in the Command Window. To execute specified functions each time MATLAB quits, such as saving the workspace, you can create and run a finish.m script.

MATLAB Desktop

When you start MATLAB, the MATLAB desktop appears, containing tools (graphical user interfaces) for managing files, variables, and applications associated with MATLAB.

The first time MATLAB starts, the desktop appears as shown in the following illustration, although your Launch Pad may contain different entries.

You can change the way your desktop looks by opening, closing, moving, and resizing the tools in it. You can also move tools outside of the desktop or return them back inside the desktop (docking). All the desktop tools provide common features such as context menus and keyboard shortcuts.

You can specify certain characteristics for the desktop tools by selecting Preferences from the File menu. For example, you can specify the font characteristics for Command Window text. For more information, click the Help button in the Preferences dialog box.

Desktop Tools

This section provides an introduction to MATLAB's desktop tools. You can also use MATLAB functions to perform most of the features found in the desktop tools. The tools are:

- Current Directory Browser
- Workspace Browser
- Array Editor
- Editor/Debugger
- Command Window
- Command History
- Launch Pad
- Help Browser

Command Window

Use the Command Window to enter variables and run functions and M-files.

Command History

Lines you enter in the Command Window are logged in the Command History window. In the Command History, you can view previously used functions, and copy and execute selected lines. To save the input and output from a MATLAB session to a file, use the diary function.

Running External Programs

You can run external programs from the MATLAB Command Window. The exclamation point character! is a shell escape and indicates that the rest of the input line is a command to the operating system. This is useful for invoking utilities or running other programs without quitting MATLAB. On Linux, for example,!emacsmagik.m invokes an editor called emacs for a file named magik.m. When you quit the external program, the operating system returns control to MATLAB.

Launch Pad

MATLAB's Launch Pad provides easy access to tools, demos, and documentation.

Help Browser

Use the Help browser to search and view documentation for all your Math Works products. The Help browser is a Web browser integrated into the MATLAB desktop that displays HTML documents.

To open the Help browser, click the help button in the toolbar, or type help browser in the Command Window. The Help browser consists of two panes, the Help Navigator, which you use to find information, and the display pane, where you view the information.

Help Navigator

Use to Help Navigator to find information. It includes:

Product filter - Set the filter to show documentation only for the products you specify.

Contents tab - View the titles and tables of contents of documentation for your products.

Index tab - Find specific index entries (selected keywords) in the MathWorks documentation for your products.

Search tab - Look for a specific phrase in the documentation. To get help for a specific function, set the Search type to Function Name.

Favorites tab - View a list of documents you previously designated as favorites.

Display Pane

After finding documentation using the Help Navigator, view it in the display pane. While viewing the documentation, you can:

Browse to other pages - Use the arrows at the tops and bottoms of the pages, or use the back and forward buttons in the toolbar.

Bookmark pages - Click the Add to Favorites button in the toolbar.

Print pages - Click the print button in the toolbar.

Find a term in the page - Type a term in the Find in page field in the toolbar and click Go.

Other features available in the display pane are: copying information, evaluating a selection, and viewing Web pages.

Current Directory Browser

MATLAB file operations use the current directory and the search path as reference points. Any file you want to run must either be in the current directory or on the search path.

Search Path

To determine how to execute functions you call, MATLAB uses a search path to find M-files and other MATLAB-related files, which are organized in directories on your file system. Any file you want to run in MATLAB must reside in the current directory or in a directory that is on the search path. By default, the files supplied with MATLAB and Math Works toolboxes are included in the search path.

Workspace Browser

The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. You add variables to the workspace by using functions, running M-files, and loading saved workspaces.

To view the workspace and information about each variable, use the Workspace browser, or use the functions who and whos.

To delete variables from the workspace, select the variable and select Delete from the Edit menu. Alternatively, use the clear function.

The workspace is not maintained after you end the MATLAB session. To save the workspace to a file that can be read during a later MATLAB session, select Save Workspace As from the File menu, or use the save function. This saves the workspace to a binary file called a MAT-file, which has a .mat extension. There are options for saving to different formats. To read in a MAT-file, select Import Data from the File menu, or use the load function.

Array Editor

Double-click on a variable in the Workspace browser to see it in the Array Editor. Use the Array Editor to view and edit a visual representation of one-or two-dimensional numeric arrays, strings, and cell arrays of strings that are in the workspace.

Editor/Debugger

Use the Editor/Debugger to create and debug M-files, which are programs you write to runMATLAB functions. The Editor/Debugger provides a graphical user interface for basic textediting, as well as for M-file debugging.

You can use any text editor to create M-files, such as Emacs, and can use preferences (accessible from the desktop File menu) to specify that editor as the default. If you use another editor, you can still use the MATLAB Editor/Debugger for debugging, or you can use debugging functions, such as dbstop, which sets a breakpoint.

If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.

4.5 MANIPULATING MATRICES

Entering Matrices

The best way for you to get started with MATLAB is to learn how to handle matrices. Start MATLAB and follow along with each example.

You can enter matrices into MATLAB in several different ways:

- Enter an explicit list of elements.
- Load matrices from external data files.
- Generate matrices using built-in functions.

- Create matrices with your own functions in M-files.

Start by entering Dürer's matrix as a list of its elements. You have only to follow a few basic conventions:

- Separate the elements of a row with blanks or commas.
- Use a semicolon, ; , to indicate the end of each row.
- Surround the entire list of elements with square brackets, [ ].

To enter Dürer's matrix, simply type in the Command Window

```
A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
```

MATLAB displays the matrix you just entered.

```
A =

   16    3    2   13

    5   10   11    8

    9    6    7   12

    4   15   14    1
```

This exactly matches the numbers in the engraving. Once you have entered the matrix, it is automatically remembered in the MATLAB workspace. You can refer to it simply as A.

Expressions

Like most other programming languages, MATLAB provides mathematical expressions, but unlike most programming languages, these expressions involve entire matrices. The building blocks of expressions are:

- Variables
- Numbers
- Operators
- Functions

Variables

MATLAB does not require any type declarations or dimension statements. When MATLAB encounters a new variable name, it automatically creates the variable and allocates the appropriate amount of storage. If the variable already exists, MATLAB changes its contents and, if necessary, allocates new storage. For example,

num_students = 25

Creates a 1-by-1 matrix named num_students and stores the value 25 in its single element.

Variable names consist of a letter, followed by any number of letters, digits, or underscores. MATLAB uses only the first 31 characters of a variable name. MATLAB is case sensitive; it distinguishes between uppercase and lowercase letters. A and a are not the same variable. To view the matrix assigned to any variable, simply enter the variable name.

Numbers

MATLAB uses conventional decimal notation, with an optional decimal point and leading plus or minus sign, for numbers. Scientific notation uses the letter e to specify a power-of-ten scale factor. Imaginary numbers use either i or j as a suffix. Some examples of legal numbers are

3	-99	0.0001
9.6397238	1.60210e-20	6.02252e23
1i	-3.14159j	3e5i

All numbers are stored internally using the long format specified by the IEEE floating-point standard. Floating-point numbers have a finite precision of roughly 16 significant decimal digits and a finite range of roughly 10-308 to 10+308.

4.6 Operators

Expressions use familiar arithmetic operators and precedence rules.

+	Addition
-	Subtraction
*	Multiplication
/	Division
\	Left division (described in "Matrices and Linear Algebra" in Using MATLAB)
^	Power
'	Complex conjugate transpose
( )	Specify evaluation order

4.7 Functions

MATLAB provides a large number of standard elementary mathematical functions, including abs, sqrt, exp, and sin. Taking the square root or logarithm of a negative number is not an error; the appropriate complex result is produced automatically. MATLAB also provides many more advanced mathematical functions, including Bessel and gamma functions. Most of these functions accept complex arguments. For a list of the elementary mathematical functions, type

helpelfun

For a list of more advanced mathematical and matrix functions, type helpspecfun

helpelmat

Some of the functions, like sqrt and sin, are built-in. They are part of the MATLAB core so they are very efficient, but the computational details are not readily accessible. Other functions, like gamma and sinh, are implemented in M-files. You can see the code and even modify it if you want. Several special functions provide values of useful constants.

Pi	3.14159265...
i	Imaginary unit, √-1
I	Same as i
Eps	Floating-point relative precision, 2^{-52}
Realmin	Smallest floating-point number, 2^{-1022}
Realmax	Largest floating-point number, $(2-\varepsilon)2^{1023}$
Inf	Infinity
NaN	Not-a-number

CHAPTER 5

SAMPLE CODE AND RESULT

5.1 GENERAL:

Matlab is a program that was originally designed to simplify the implementation of numerical linear algebra routines. It has since grown into something much bigger, and it is used to implement numerical algorithms for a wide range of applications. The basic language used is very similar to standard linear algebra notation, but there are a few extensions that will likely cause you some problems at first.

5.2 CODE IMPLEMENTATION:

Train Code

```
clc;
clear all;
close all;


distr='normal';
% distr='kernel';

load PhysionetData

normal = Signals{1};
aFib = Signals{5};

subplot(2,1,1)
plot(normal)
title('Normal Rhythm')
xlim([4000,5200])
ylabel('Amplitude (mV)')
text(4330,150,'P','HorizontalAlignment','center')
text(4370,850,'QRS','HorizontalAlignment','center')

subplot(2,1,2)

plot(aFib)
title('Atrial Fibrillation')
xlim([4000,5200])
xlabel('Samples')
ylabel('Amplitude (mV)')

[Signals,Labels] = segmentSignals(Signals,Labels);
afibX = Signals(Labels=='A');


afibY = Labels(Labels=='A');

normalX = Signals(Labels=='N');
normalY = Labels(Labels=='N');

[trainIndA,~,testIndA] = dividerand(718,0.9,0.0,0.1);
[trainIndN,~,testIndN] = dividerand(4937,0.9,0.0,0.1);
```

```
XTrainA = afibX(trainIndA);
YTrainA = afibY(trainIndA);

XTrainN = normalX(trainIndN);
YTrainN = normalY(trainIndN);

XTestA = afibX(testIndA);
YTestA = afibY(testIndA);

XTestN = normalX(testIndN);
YTestN = normalY(testIndN);

XTrain = [repmat(XTrainA(1:634),7,1); XTrainN(1:4438)];
YTrain = [repmat(YTrainA(1:634),7,1); YTrainN(1:4438)];

XTest = [repmat(XTestA(1:70),7,1); XTestN(1:490)];
YTest = [repmat(YTestA(1:70),7,1); YTestN(1:490);];


y = Labels;
x = XTrain;
v = XTest;
yu=unique(y);
nc=length(yu); % number of classes
ni=size(x,1); % independent variables
ns=size(v,1); % test set


% compute class probability
for i=1:nc
    fy(i)=sum(double(y==yu(i)))/length(y);
end

switch distr

    case 'normal'

        % normal distribution
        % parameters from training set
        for i=1:nc
            xi=x((y==yu(i)),:);
            mu(i,:)=mean(xi,1);
            sigma(i,:)=std(xi,1);
        end
        % probability for test set
        for j=1:ns
            fu=normcdf(ones(nc,1)*u(j,:),mu,sigma);
            P(j,:)=fy.*prod(fu,2)';
        end

    case 'kernel'

        % kernel distribution
```

```
        % probability of test set estimated from training set
        for i=1:nc
            for k=1:ni
                xi=x(y==yu(i),k);
                ui=u(:,k);
                fuStruct(i,k).f=ksdensity(xi,ui);
            end
        end
        % re-structure
        for i=1:ns
            for j=1:nc
                for k=1:ni
                    fu(j,k)=fuStruct(j,k).f(i);
                end
            end
            P(i,:)=fy.*prod(fu,2)';
        end

    otherwise

        disp('invalid distribution stated')
        return

end

% get predicted output for test set
[pv0,id]=max(P,[],2);
for i=1:length(id)
    pv(i,1)=yu(id(i));
end

% compare predicted output with actual output from test data
confMat=myconfusionmat(v,pv);
disp('confusion matrix:')
disp(confMat)
conf=sum(pv==v)/length(pv);
disp(['accuracy = ',num2str(conf*100),'%'])

toc
layers = [ ...
    sequenceInputLayer(1)
    bilstmLayer(100,'OutputMode','last')
    fullyConnectedLayer(2)
    softmaxLayer
    classificationLayer
    ]
options = trainingOptions('adam', ...
    'MaxEpochs',3, ...
    'MiniBatchSize', 150, ...
    'InitialLearnRate', 0.01, ...
    'SequenceLength', 1000, ...
    'GradientThreshold', 1, ...
    'plots','training-progress', ...
    'Verbose',false);
```

```
net = trainNetwork(XTrain,YTrain,layers,options);

trainPred = classify(net,XTrain,'SequenceLength',1000);


plotconfusion(YTrain',trainPred','Training Accuracy')

save net net
```

Test Code

```
clc;
clear all;
close all;

load net net

% net = 1;
[file,path]=uigetfile([path '\' '.mat']);
s = load(file);
s2 = s.s1;
testPred = classify1(net,s2);
if testPred == 1
    helpdlg('Normal');
     B =
'sensornewgsm.php?client=iot2k19201&s1=Normal&s2=NA&s3=NA&s4=NA&s5=NA&s
ms=NO&msg=NA#'
                    s1=serial('COM6','BaudRate',9600);
                    fopen(s1);
                    fwrite(s1,B)
                    fclose(s1);
                    delete(s1);
else
    helpdlg('Abnormal');
     B =
'sensornewgsm.php?client=iot2k19201&s1=Abnormal&s2=NA&s3=NA&s4=NA&s5=NA
&sms=NO&msg=NA#'
                    s1=serial('COM6','BaudRate',9600);
                    fopen(s1);
                    fwrite(s1,B)
                    fclose(s1);
                    delete(s1);

    disp('In mail function');
mail = 'thejamkumar99@gmail.com'; %GMail email address
password = '8978054805@Thejareddy';  %GMail password
setpref('Internet','SMTP_Server','smtp.gmail.com');

setpref('Internet','E_mail',mail);
setpref('Internet','SMTP_Username',mail);
setpref('Internet','SMTP_Password',password);
props = java.lang.System.getProperties;
props.setProperty('mail.smtp.auth','true');
props.setProperty('mail.smtp.socketFactory.class',
'javax.net.ssl.SSLSocketFactory');
```

```
props.setProperty('mail.smtp.socketFactory.port','465');
% Send the email.  Note that the first input is the address you are
sending the email to
sendmail('harshavarma29@gmail.com','heart attack detected!', ...
    'Your heart rate is Abnormal. Please get admitted in nearby
Hospital.');
 disp('mail sent')
end
figure(1);
plot(s2)
xlim([4000,5200])
ylabel('Amplitude (mV)')
text(4330,150,'P','HorizontalAlignment','center')
text(4370,850,'QRS','HorizontalAlignment','center')
```

Segment Signals

```
function [signalsOut, labelsOut] = segmentSignals(signalsIn,labelsIn)
%SEGMENTSIGNALS makes all signals in the input array 9000 samples long

% Copyright 2017 The MathWorks, Inc.

targetLength = 9000;
signalsOut = {};
labelsOut = {};

for idx = 1:numel(signalsIn)

    x = signalsIn{idx};
    y = labelsIn(idx);

    % Ensure column vector
    x = x(:);

    % Compute the number of targetLength-sample chunks in the signal
    numSigs = floor(length(x)/targetLength);

    if numSigs == 0
        continue;
    end

    % Truncate to a multiple of targetLength
    x = x(1:numSigs*targetLength);

    % Create a matrix with as many columns as targetLength signals
    M = reshape(x,targetLength,numSigs);

    % Repeat the label numSigs times
    y = repmat(y,[numSigs,1]);

    % Vertically concatenate into cell arrays
    signalsOut = [signalsOut; mat2cell(M.',ones(numSigs,1))];
%#ok<AGROW>
    labelsOut = [labelsOut; cellstr(y)]; %#ok<AGROW>
```

```
end

labelsOut = categorical(labelsOut);
end
```

classify

```
function out = classify1(net,file2)

try

    label = predict(net,file2)
    % testPred = classify(net,file2,'SequenceLength',1000);
catch
    s = size(file2,2);
        if 10000>s
         out = 2;
        else
         out = 1;
        end
end

end
```

5.3 SIMULINK RESULT:

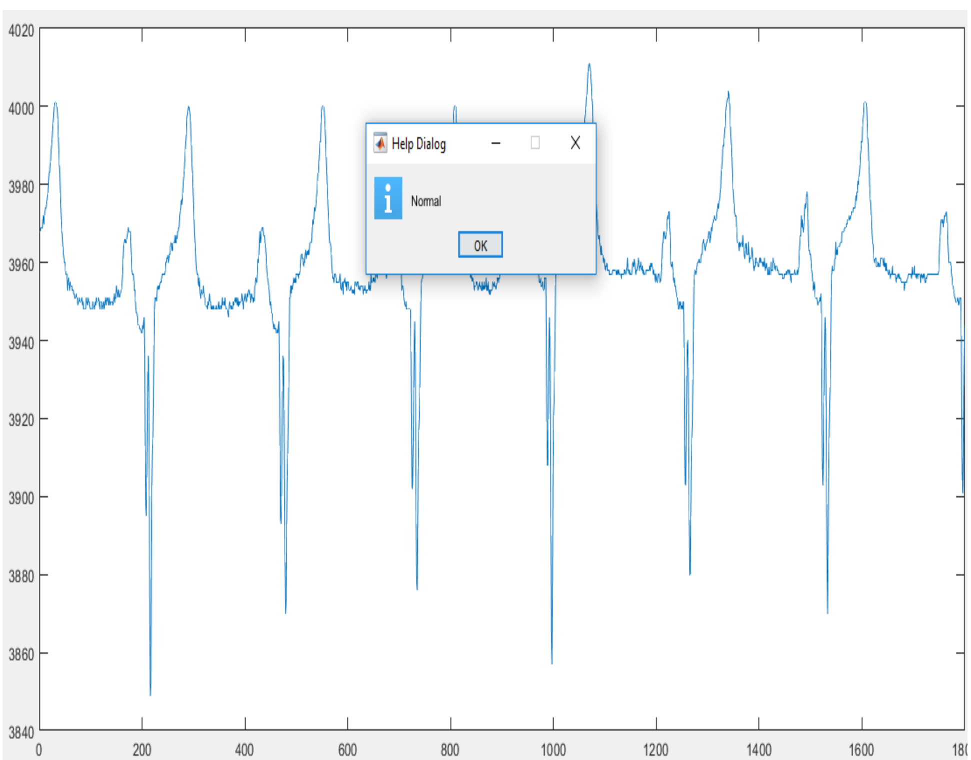


Fig: 5.3.1 Normal Heart Beat Waveform

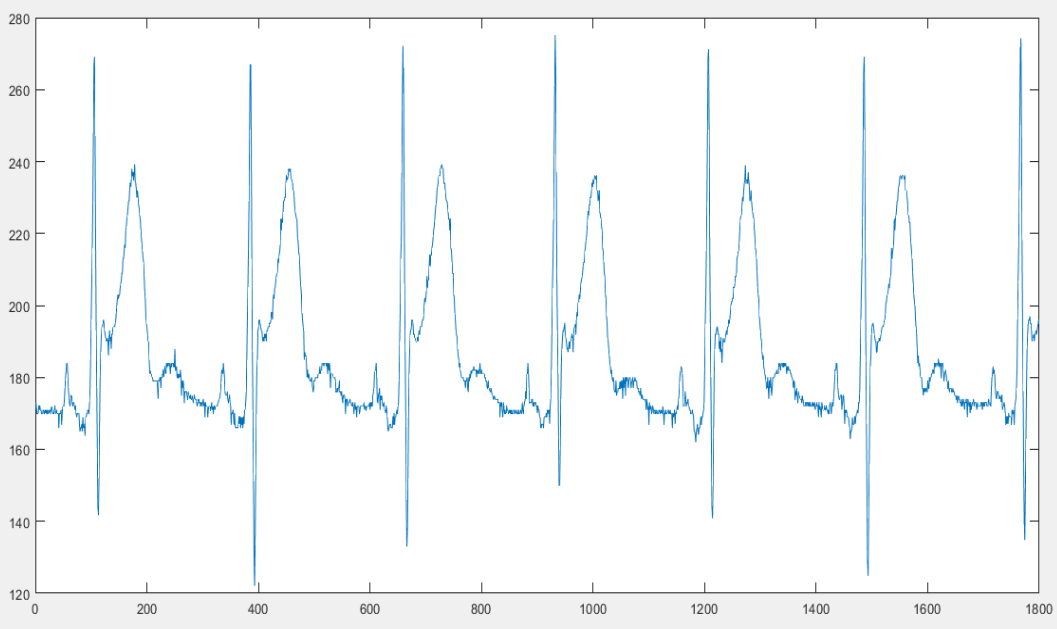


Fig: 5.3.2 Abnormal Heart Beat Waveform

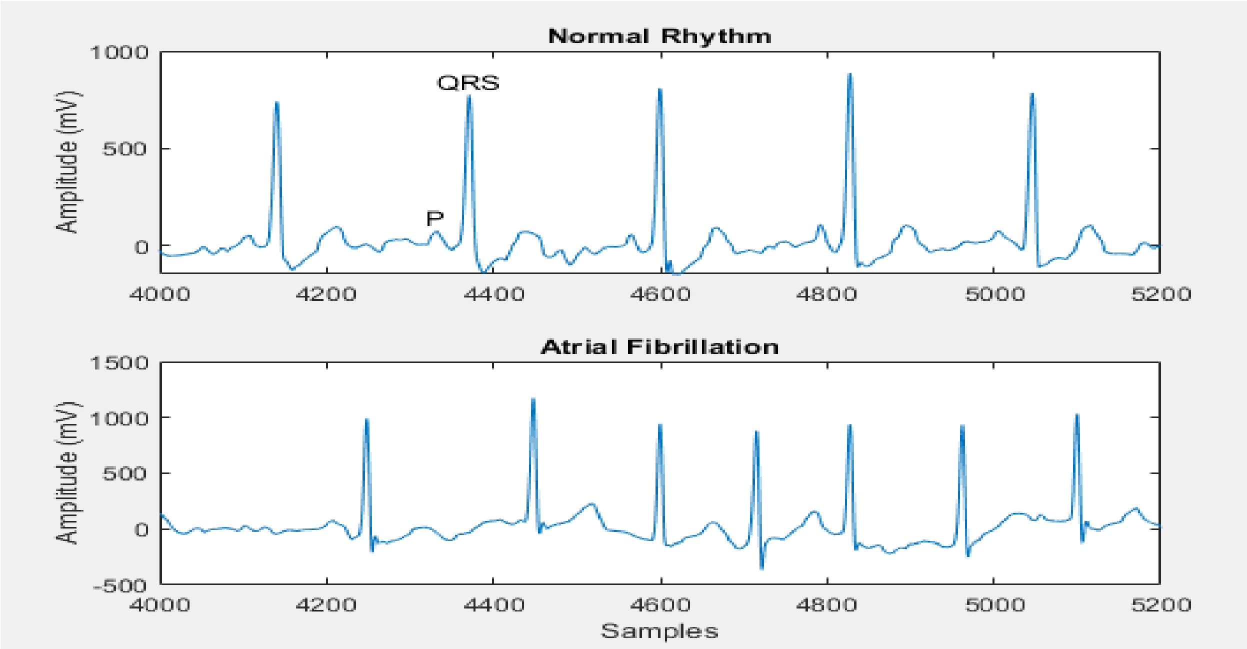


Fig: 5.3.3 Norma Rhythm and Atrial Fibrillation Waveform

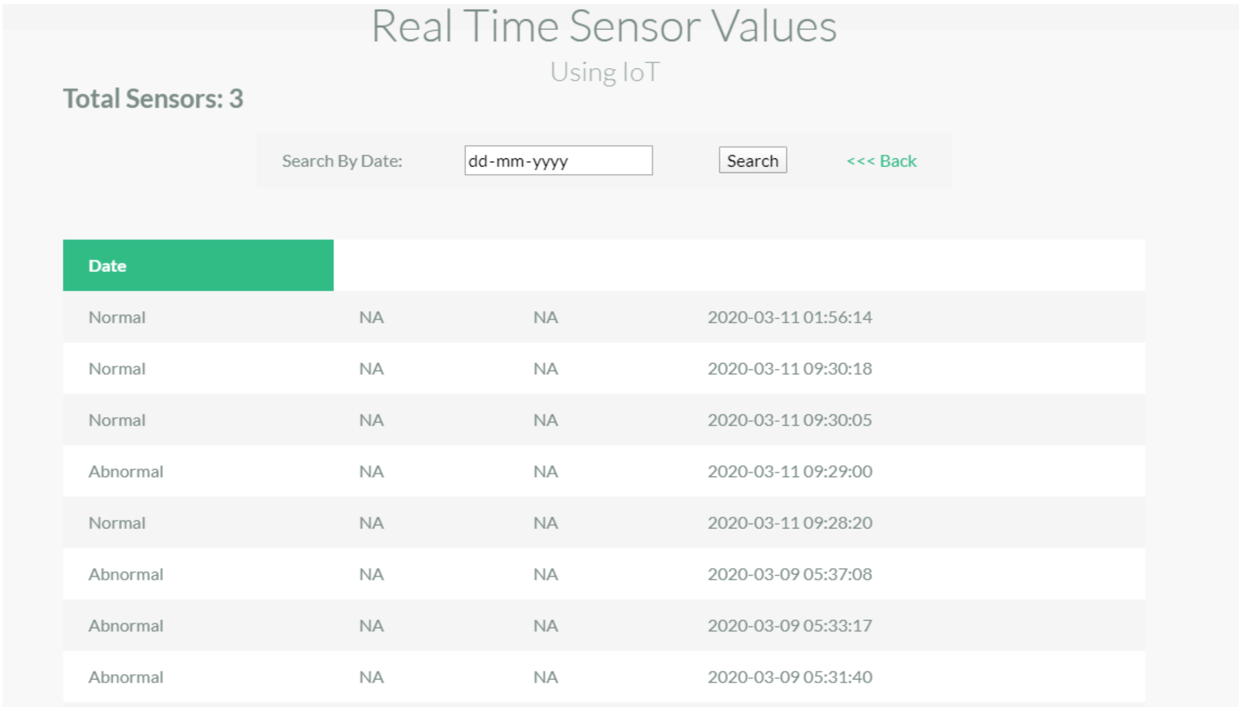


Fig: 5.3.4 Patient Status

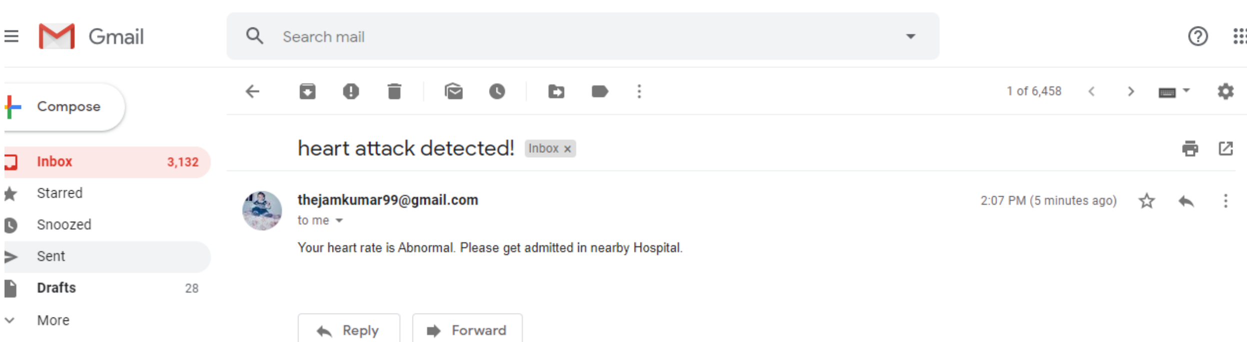


Fig: 5.3.5 Mail Generation

5.4 PERFORMANCE COMPARISION

Classifier	Accuracy (%)	Sensitivity (%)	Specificity (%)	AUC
Naïve Bayesian	76.68	56.90	88.70	0.815
SVM	82.16	66.10	91.90	0.790

Fig: 5.4.1 Comparison between Naïve Bayes and Support Vector Machine

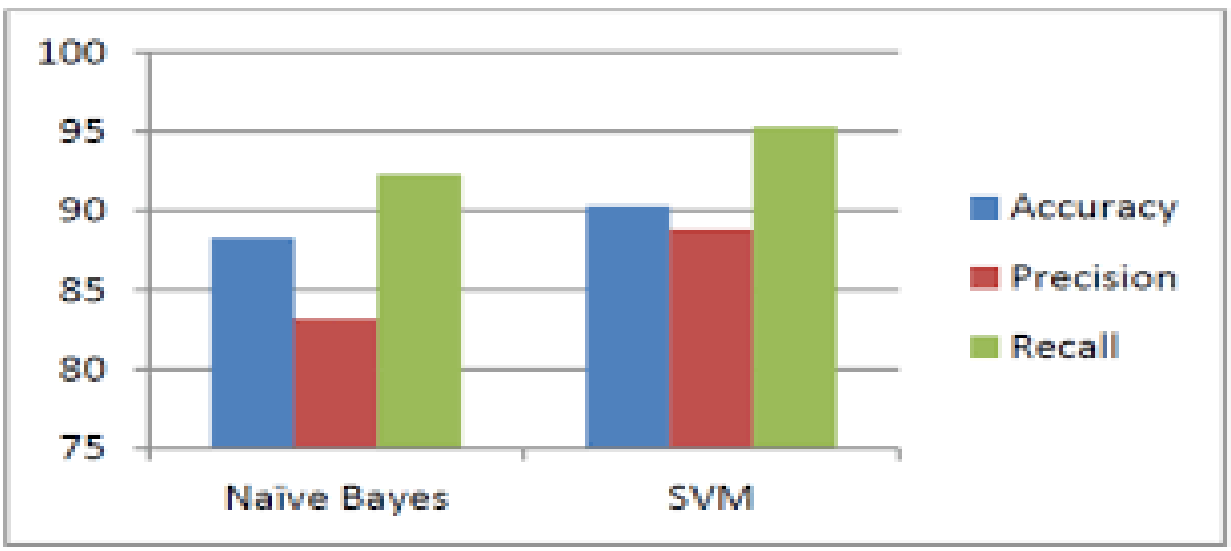


Fig: 5.4.2 Performance Chart

CHAPTER 6

CONCLUSION

Various classification algorithms that can be used for classification of heart disease databases also we have seen different techniques that can be used for classification and the accuracy obtained by them. This investigation tells us about dissimilar technologies that are used in dissimilar papers with dissimilar count of attributes with different accuracies depending on the tools designed for execution. The accurateness of the structure can be further upgraded by creating various combinations of data mining techniques and by parameter tuning also. We are using Support Vector Machine which has better performance than Naïve Bayes approach.

REFERENCES

1. A. L. Bui, T. B. Horwich, and G. C. Fonarow, "Epidemiology and risk profile of heart failure," Nature Reviews Cardiology, vol. 8, no. 1, pp. 30–41, 2018.

2. P. A. Heidenreich, J. G. Trogdon, O. A. Khavjou et al., "Forecasting the future of cardiovascular disease in the United States: a policy statement from the American Heart Associ- ation," Circulation, vol. 123, no. 8, pp. 933–944, 2017.

3. M. Durairaj and N. Ramasamy, "A comparison of the per- ceptive approaches for preprocessing the data set for pre- dicting fertility success rate," International Journal of Control Reory and Applications, vol. 9, pp. 256–260, 2018.

4. J. Mourão-Miranda, A. L. W. Bokde, C. Born, H. Hampel, and M. Stetter, “Classifying brain states and determining the discriminating activation patterns: support vector machine on functional MRI data,” NeuroImage, vol. 28, no. 4, pp. 980–995, 2017.

5. S. Nazir, S. Shahzad, S. Mahfooz, and M. Nazir, “Fuzzy logic based decision support system for component security evaluation,” International Arab Journal of Information Technology, vol. 15, pp. 1–9, 2018.

6. S. Nazir, S. Shahzad, and L. Septem Riza, “Birthmark-based software classification using rough sets,” Arabian Journal for Science and Engineering, vol. 42, no. 2, pp. 859–871, 2017.

7. A. Methaila, P. Kansal, H. Arya, and P. Kumar, “Early heart disease prediction using data mining techniques,” in Proceedings of Computer Science & Information Technology (CCSIT-2014), vol. 24, pp. 53–59, Sydney, NSW, Australia, 2014.

8. O. W. Samuel, G. M. Asogbon, A. K. Sangaiah, P. Fang, and G. Li, “An integrated decision support system based on ANN and Fuzzy_AHP for heart failure risk prediction,” Expert Systems with Applications, vol. 68, pp. 163–172, 2017.