A Project report on

PATIENT STABILITY PREDICTION USING MACHINE LEARNING

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

G. ARCHITH KUMAR

(19H51A05D5)

MOHD IFTEQUARUDDIN

(19H51A05E0)

R. ROHITH REDDY

(19H51A05E8)

Under the esteemed guidance
Of

MAJOR Dr. V.A. NARAYANA

(PROFESSOR OF CSE & PRINCIPAL)

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC Autonomous)

*Approved by AICTE *Affiliated to JNTUH *NAAC Accredited with A$^+$ Grade

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2019- 2023

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Major Project report entitled **"PATIENT STABILITY PREDICTION USING MACHINE LEARNING"** being submitted by **G. ARCHITH KUMAR (19H51A05D5)**, **MOHD IFTEQUARUDDIN (19H51A05E0)**, **R. ROHITH REDDY (19H51A05E8)** in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

MAJOR Dr. V.A. NARAYANA
- PROFESSOR, PRINCIPAL
Dept. of CSE

Dr. SIVA SKANDHA SANAGALA
-ASSOCIATE PROFESSOR, HOD
Dept. of CSE

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Major Dr. V A Narayana,** Professor, Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala,** Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary & Correspondent, CMR Group of Institutions, for his continuous care.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

G. ARCHITH KUMAR (19H51A05D5)
MOHD IFTEQUARUDDIN (19H51A05E0)
R. ROHITH REDDY (19H51A05E8)

TABLE OF CONTENTS

List of Figures

List of Tables

ABSTRACT

These years, with artificial intelligence and machine learning becoming the hotspot of research, several applications have emerged in each of these areas. It exists not only as a kind of academic frontier but also something close to our life. In this trend, the combination of medical care and machine learning becomes more and more tighter. The proposal of its main idea also greatly alleviated the existing situation of unbalanced medical distribution and resources strain. This paper summarizes some application of machine learning and auxiliary tumor treatment in the process of medical resource allocation, and puts forward some new methods of application to realize it closer to human life in the era of artificial intelligence and the explores a good situation of mutual combination of medical industry and computer industry, which is benefit both.

Machine learning techniques in healthcare use the increasing amount of health data provided by the Internet of Things to improve patient outcomes. These techniques provide promising applications as well as significant challenges. The three main areas machine learning is applied to include medical imaging, natural language processing of medical documents, and genetic information. Many of these areas focus on diagnosis, detection, and prediction. A large infrastructure of medical devices currently generates data but a supporting infrastructure is oftentimes not in place to effectively utilize such data. The many different forms medical information exist in also creates some challenges in data formatting and can increase noise. We examine a brief history of machine learning, some basic knowledge regarding the techniques, and the current state of this technology in healthcare.

CHAPTER 1
INTRODUCTION

CHAPTER 1
INTRODUCTION

Greenspan H, et al [3] has observed that the majority of the current publications are using some form of CNNs when it comes to object detection in medical images. Graphic-processing unit (GPU) acceleration has made the building of deep CNNs more efficient, however significant challenges in creating a competent model still exist. The biggest issue is the need for a large amount of annotated medical image data. The cost to aggregate and create such databases is often prohibitive since it requires trained physicians' time to annotate the images. Additionally, concerns involving patient privacy often hinders the ability to make such databases open-source.

Many studies only use around 100–1000 samples in training CNNs. Giger ML [4] has observed that this limited sample size increases the risk of overfitting and reduces the accuracy of the predictions.

Foster KR, et al [5] has concerns regarding the implementation of machine learning into clinical diagnosis have been raised regarding proper validation of models. The main fears entail properly scoping the intended goals of a machine learning model, reducing dimensionality of the data, and reproducibility of training such models on real-world and new clinical data. Validating results on other datasets can be difficult due to the lack of larger datasets for niche diseases, where the aggregation of this data can take more work than the actual training of the model. Medical imaging data is inherently more difficult to acquire and is more difficult to store and process. The infrastructure to handle the data has simply not kept up with the increase in the amount of data.

Lundervold AS, et al [6] has observed that the most common use of current machine learning technologies in medicine is for computer automated detection (CAD) specifically in the detection of lesions such as those commonly found in mammograms, brain scans, and other body scans.

These methods use CNNs to arrive at the probability that a candidate lesion is in fact a lesion, often utilizing several 2D slices of 3D rotational scans of either CAT or MRI images.

Ultrasound images are also used in training and a variety of methods such as randomized rotation of the images or centering candidate lesions in the center of the image. Especially in mammography, CAD techniques have reached a level where they are used as a "second opinion" for most radiologists, greatly improving the accuracy of screenings without doubling the cost associated with using a human as the "second opinion".

M.P. Sendak, et al [7] has observed that the healthcare industry has always been a strong supporter of cutting-edge technologies. AI and ML have found several applications in the healthcare industry, just as they have in business and e-commerce. There are virtually limitless possibilities with this technology. ML is assisting in transforming the healthcare industry for the better with its cutting-edge applications. Healthcare systems have already used Big Data tools for next-generation data analytics because of mandatory procedures like Electronic Medical Records (EMR). ML tools are set to offer even more value to this process. These improve the quality of automation and intelligent decision-making in primary/tertiary patient care and public healthcare systems. This could be the most significant impact of ML tools, as it can improve the quality of life for billions of people worldwide.

G. Manogaran, et al [8] has observed that ML has many potential applications for research and clinical trials. Using ML-based predictive research to identify latent clinical trial participants can help researchers move with a supply from many data points, such as previous doctor visits, social media, etc. It also ensures that data is accessed in real-time and manages the trial associates, thereby supporting the most appropriate sample size to be investigated and utilising the energy of electronics work, which aids in the reduction of data-based errors.

R. Bhardwaj, et al [9] has observed that electronically recorded medical imaging data is abundant these days, and various algorithms may be used to find and discover patterns and anomalies using this dataset. ML algorithms can analyse imaging data in the same way a highly-skilled radiologist can, detecting abnormal skin patches, lesions, tumours, and brain bleeding. As a result, the use of these platforms to aid radiologists is expected to skyrocket.

What is Machine Learning?

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Features of Machine learning

- Machine learning is data driven technology. Large amount of data generated by organizations on daily bases. So, by notable relationships in data, organizations make better decisions.
- Machine can learn itself from past data and automatically improve.
- From the given dataset it detects various patterns on data.
- For the big organizations branding is important and it will become easier to target relatable customer base.
- It is similar to data mining because it is also deals with the huge amount of data.

1.1 PROBLEM STATEMENT:

- In the medical image analysis, although the error caused by the subjective condition of the doctor is avoided, it is also limited by the objective conditions, such as noises, and other errors are still easy to occur.
- The current research has made achievements in pathological analysis, but it is still not applicable to the illness requiring human resources, such as analgesia and fever, which are more common.
- Although machine learning has already invested in many researches and applications in assisting tumor treatment, it still requires more financial and personnel requirements to make relevant research and development and to put into large-scale use.

1.2 RESEARCH OBJECTIVE:

- The adoption of the machine learning method means that simple work can be directly replaced by machines. It means that, at the same time, the employment situation of the relevant personnel and the education level of the medical profession need to be improved.
- If we want to make sure that the patient's condition is as correct as possible, you must first ensure that we have a database that covers as much relevant information as possible, create a target data set, and then preprocess it (this process may cost 60 About % of the energy), and then through the data conversion to find useful data features, and then data mining.

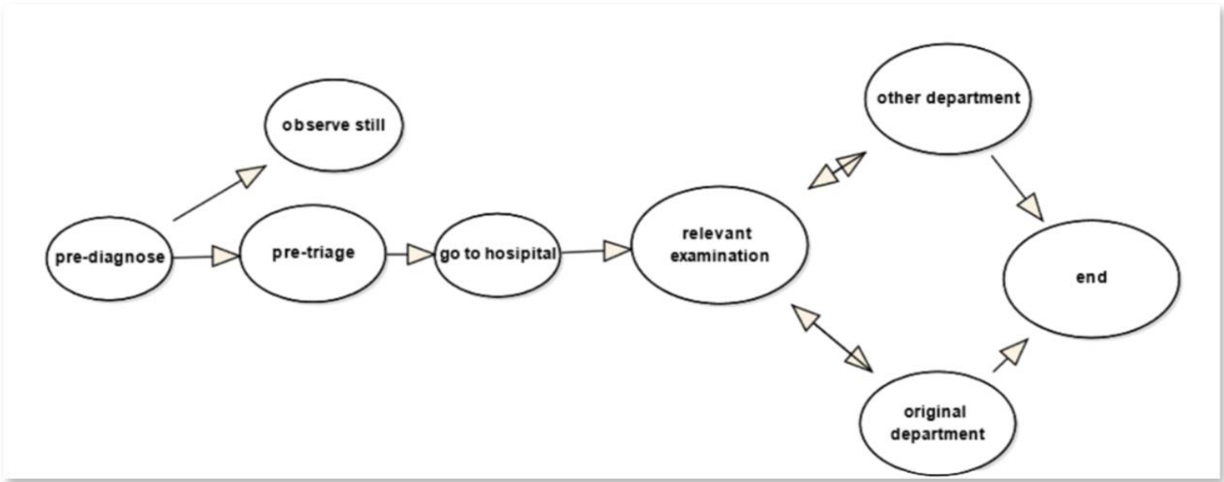


Fig. 1.2.1 - Traditional and New Model of Medical Care

1.3 THE MAIN ALGORITHM OF MACHINE LEARNING:

(A) Decision Tree Based Methods

The algorithm of decision tree is a method, which creates a decision tree by existing data and inputs the training set. According to the growth direction of decision tree, the test data can be classified. The main idea of decision tree is which feature is the best, how many branches can be generated and the time when to stop splitting. During this procession, it can be determined by the variable which is called impurity and some other mathematical method. However, due to the fact that it is a greedy approach, decision tree may disable to find the best tree sometimes.

(B) Naïve Bayes and Bayesian Belief Networks

Naïve Bayes is a ML method based on probability theory. It assumes that vents are

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)}$$

independent and calculates through prior probability and posterior probability of the target object. The formula is as follows. Because this principle of this method is relatively simple, and the prediction efficiency is high although, the conditions 177 are strict, it is still widely used in the field of natural language recognition and so on.

(C) <u>K-means</u>

In this method, the variable k is chosen by the actual situation. After choosing k objects as the primary center of clusters randomly, it calculates the distance between every object and the center of clusters and then assigns the object into the closest cluster. Until all objects have been assigned, the centroid of every cluster will be calculated again. The process will be repeated until the centroid doesn't change. The algorithm is an iterative and programming is less difficult.

(D) <u>Artificial Neural Network (ANN)</u>

Artificial Neural Network (ANN) is an algorithm that imitates the learning process of human brain, consisting of many nodes which is called neurons connected to each other. Each node represents a special function called activation function. There is a value of weight between two nodes. As a kind of computing model, neural network is divided into forward network and feedback network. Through the input of training set the neural network is trained and different weight values are modified, the nonlinear data will be processed to achieve the purpose of learning.

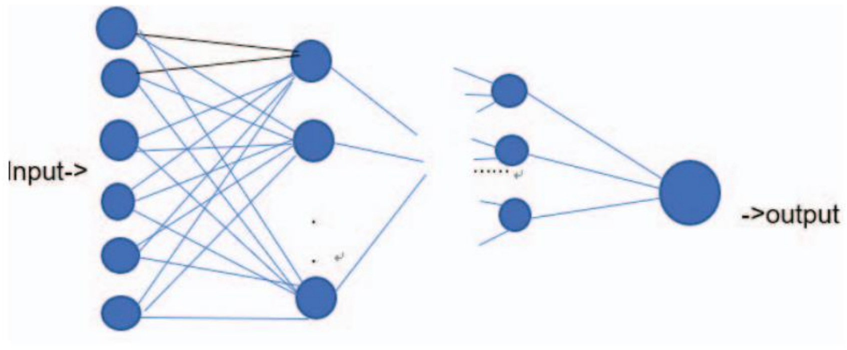


Fig. 1.3.1 ANN model

(E) <u>Support Vector Machine (SVM)</u>

SVM is an important part of statistical learning theory, which by transforming input space into high-dimensional space. In the linear classification, the hyperplane and loss function are constructed to solve the minimum of the loss of agent; for the linear indivisible problem, the method can be applied and the method is used to segment the hypersurface with feature space. SVM is often used in the analysis of medical conditions and the judgement of benign and malignant tumors, but it is difficult to implement in largescale training samples because it may involve the calculation of high-order matrices.

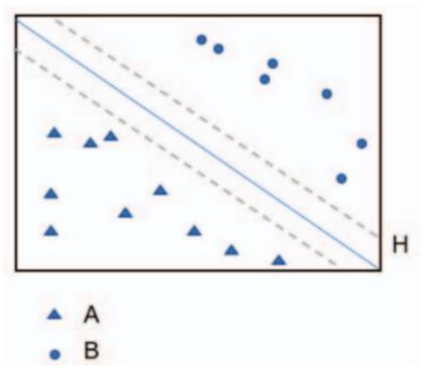


Fig. 1.3.2 SVM model

CHAPTER 2
BACKGROUND WORK

CHAPTER 2

BACKGROUND WORK

2.1 Liver Segmentation From CT Images Using A Sparse Priori Statistical Shape Model (SP-SSM)

2.1.1 INTRODUCTION:

Xuehu Wang, et al [1] has proposed a new liver segmentation method based on a sparse a priori statistical shape model (SP-SSM). First, mark points are selected in the liver a priori model and the original image. Then, the a priori shape and its mark points are used to obtain a dictionary for the liver boundary information. Second, the sparse coefficient is calculated based on the correspondence between mark points in the original image and those in the a priori model, and then the sparse statistical model is established by combining the sparse coefficients and the dictionary. Finally, the intensity energy and boundary energy models are built based on the intensity information and the specific boundary information of the original image. Then, the sparse matching constraint model is established based on the sparse coding theory. The SP-SSM can achieve a mean overlap error of 4.8% and a mean volume difference of 1.8%, whereas the average symmetric surface distance and the root mean square symmetric surface distance can reach 0.8 mm and 1.4 mm, respectively.

2.1.2 MERITS, DEMERITS AND CHALLENGES:

MERITS:

- ❖ This new liver segmentation method achieving at some level in medical field.

DEMERITS:

- ❖ Can achieve a mean overlap error of 4.8%.
- ❖ Time taking process.

CHALLENGES:

- ❖ One should overcome the mean overlap error and Time-consuming process.
- ❖ And there are some un predictable results one should overcome that.

2.1.3. IMPLEMENTATION

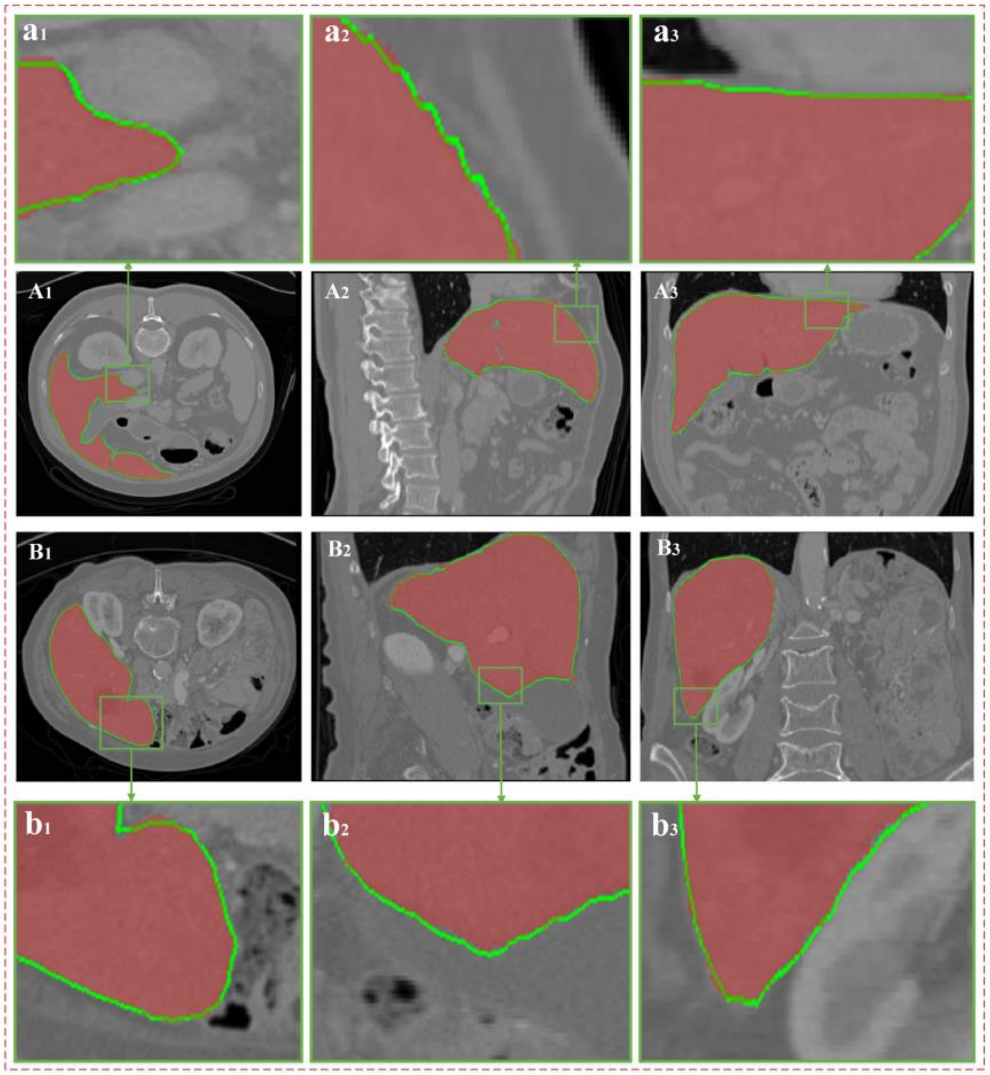


Fig. 2.1.3.1 - Segmentation Results.

(A) and (B) show the two groups of CT data; (A1), (A2) and (A3) express the same groups of data on the cross section, vertical plane, and coronal plane, respectively; and (a1), (a2) and (a3) show the partially enlarged images corresponding to the green areas in (A1), (A2) and (A3), respectively. The red area in every image indicates the real liver boundaries, and the green line indicates the liver segmentation results based on the proposed algorithm.

2.2 Automatic Detection Of Cerebral Microbleeds From MR Images Via 3D Convolutional Neural Networks

2.2.1 INTRODUCTION:

Qi Dou, et al [2] has proposed Cerebral microbleeds (CMBs) are small hemorrhages nearby blood vessels. They have been recognized as important diagnostic biomarkers for many cerebrovascular diseases and cognitive dysfunctions. Compared with previous methods that employed either low-level hand-crafted descriptors or 2D CNNs, our method can take full advantage of spatial contextual information in MR volumes to extract more representative high-level features for CMBs, and hence achieve a much better detection accuracy. To further improve the detection performance while reducing the computational cost, we propose a cascaded framework under 3D CNNs for the task of CMB detection. We first exploit a 3D fully convolutional network (FCN) strategy to retrieve the candidates with high probabilities of being CMBs, and then apply a well-trained 3D CNN discrimination model to distinguish CMBs from hard mimics. Compared with traditional sliding window strategy, the proposed 3D FCN strategy can remove massive redundant computations and dramatically speed up the detection process. We constructed a large dataset with 320 volumetric MR scans and performed extensive experiments to validate the proposed method, which achieved a high sensitivity of 93.16% with an average number of 2.74 false positives per subject, outperforming previous methods using low-level descriptors or 2D CNNs by a significant margin.

2.2.2 MERITS, DEMERITS AND CHALLENGES:

MERITS:

- ❖ This Automatic detection method achieving at some level in medical field.

DEMERITS:

- ❖ This procedure is laborious, time-consuming, high cost and error prone.

CHALLENGES:

- ❖ One should overcome Time-consuming process problem and reduce the cost.
- ❖ And the procedure also little complicated.

2.2.3. IMPLEMENTATION

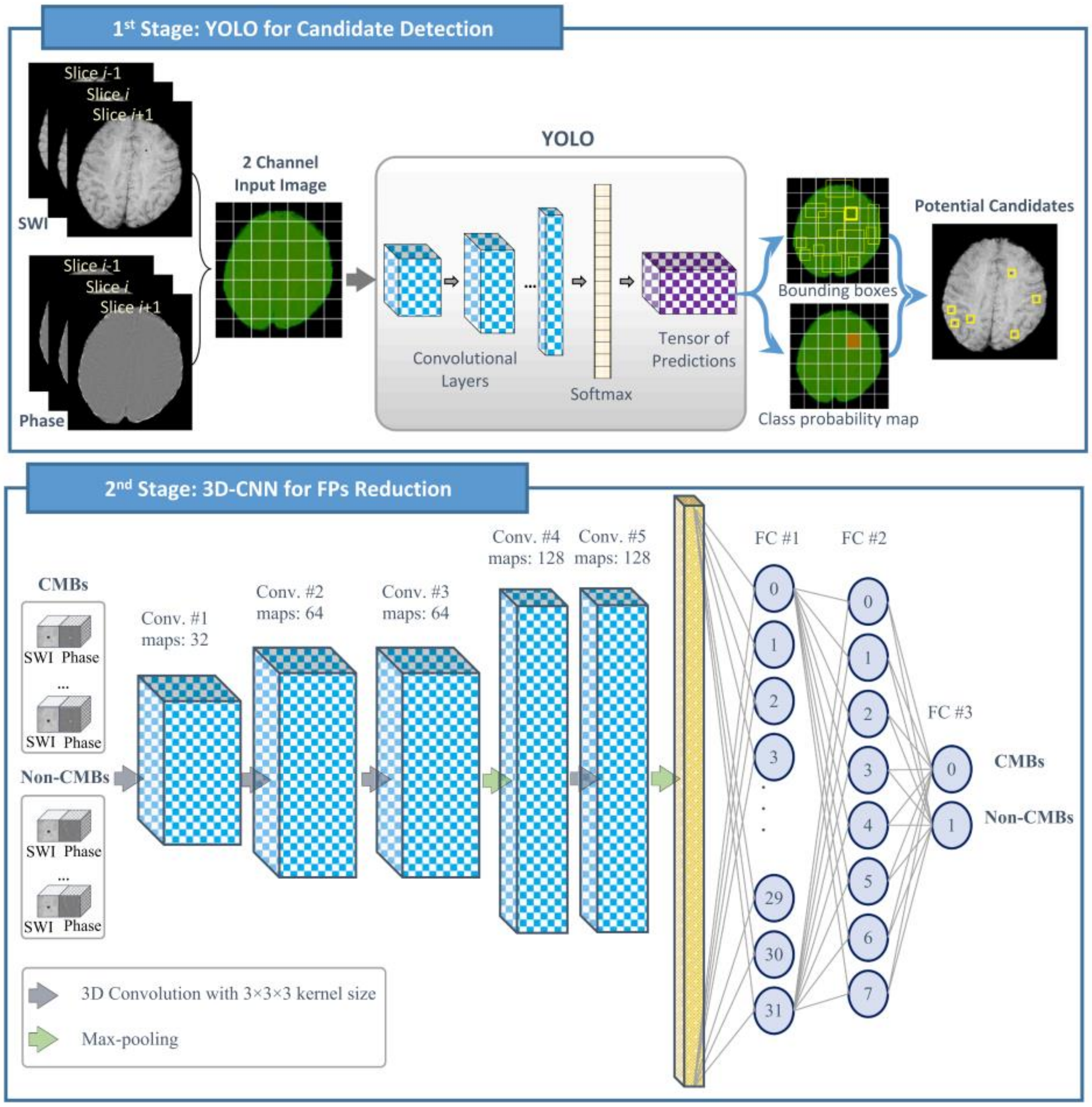


Fig. 2.2.3.1 - Overview Diagram of The Two-Stage Deep Learning Approach.

CHAPTER 3
PROPOSED SYSTEM

3.1. Objective of Proposed Model

The adoption of the machine learning method means that simple work can be directly replaced by machines. It means that, at the same time, the employment situation of the relevant personnel and the education level of the medical profession need to be improved.

If we want to make sure that the patient's condition is as correct as possible, you must first ensure that we have a database that covers as much relevant information as possible, create a target data set, and then preprocess it (this process may cost 60 About % of the energy), and then through the data conversion to find useful data features, and then data mining.

3.2. Algorithms Used for Proposed Model

Decision Tree Based Methods:

The algorithm of decision tree is a method, which creates a decision tree by existing data and inputs the training set. According to the growth direction of decision tree, the test data can be classified. The main idea of decision tree is which feature is the best, how many branches can be generated and the time when to stop splitting. During this procession, it can be determined by the variable which is called impurity and some other mathematical method. However, due to the fact that it is a greedy approach, decision tree may disable to find the best tree sometimes.

Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**

In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset.

It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.** A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Pruning:** Pruning is the process of removing the unwanted branches from the tree.
- **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node.

Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:

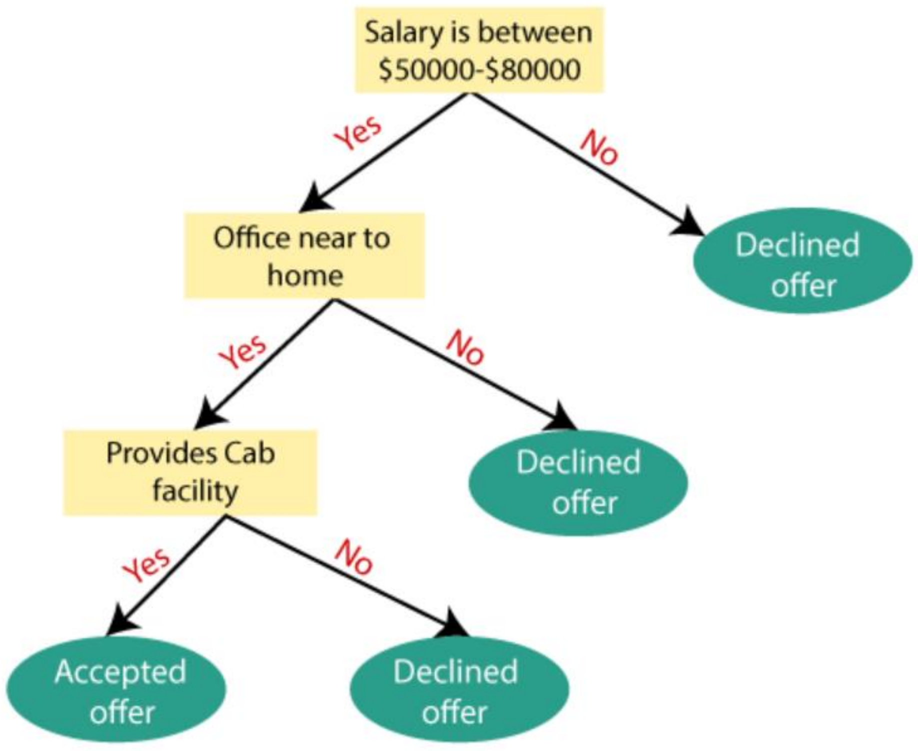


Fig. 3.2.1 – Example of decision tree.

Python Implementation of Decision Tree

Now we will implement the Decision tree using Python. For this, we will use the dataset "**user_data.csv**," which we have used in previous classification models. By using the same dataset, we can compare the Decision tree classifier with other classification models such as KNN SVM, Logistic Regression, etc. which are given below:

- Data Pre-processing step
- Fitting a Decision-Tree algorithm to the Training set
- Predicting the test result
- Test accuracy of the result (Creation of Confusion matrix)
- Visualizing the test set result.

Support Vector Machine (SVM):

SVM is an important part of statistical learning theory, which by transforming input space into high-dimensional space. In the linear classification, the hyperplane and loss function are constructed to solve the minimum of the loss of agent; for the linear indivisible problem, the method can be applied and the method is used to segment the hypersurface with feature space. SVM is often used in the analysis of medical conditions and the judgement of benign and malignant tumors, but it is difficult to implement in largescale training samples because it may involve the calculation of high-order matrices.

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog.

On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:

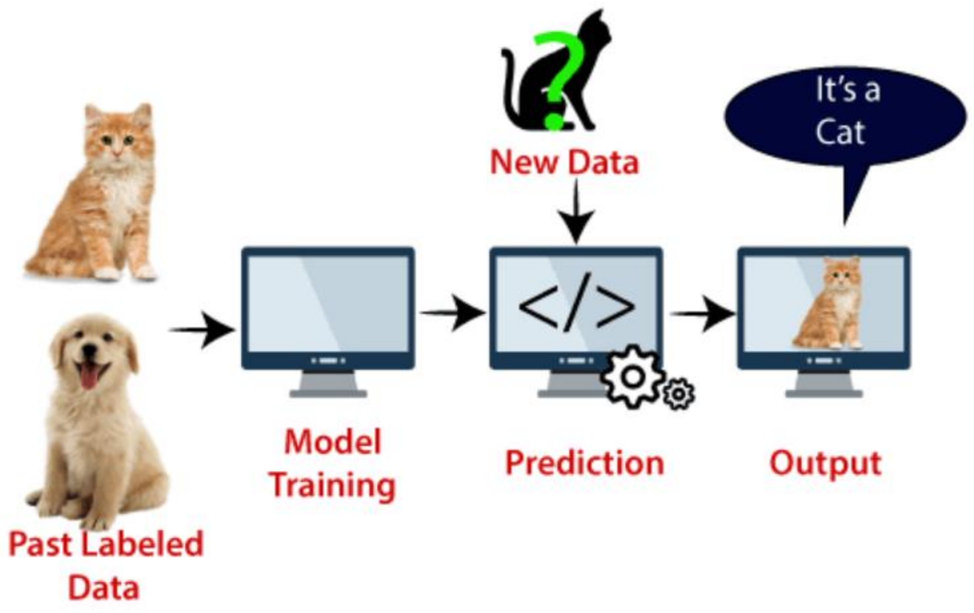


Fig. 3.2.2 – Example of SVM model.

SVM algorithm can be used for **Face detection, image classification, text categorization,** etc.

Types of SVM

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

Python Implementation of Support Vector Machine

- Data Pre-processing step
- Predicting the test set result
- Creating the confusion matrix
- Visualizing the training set result
- Visualizing the test set result

3.3. Designing

3.3.1. UML & ER Diagrams

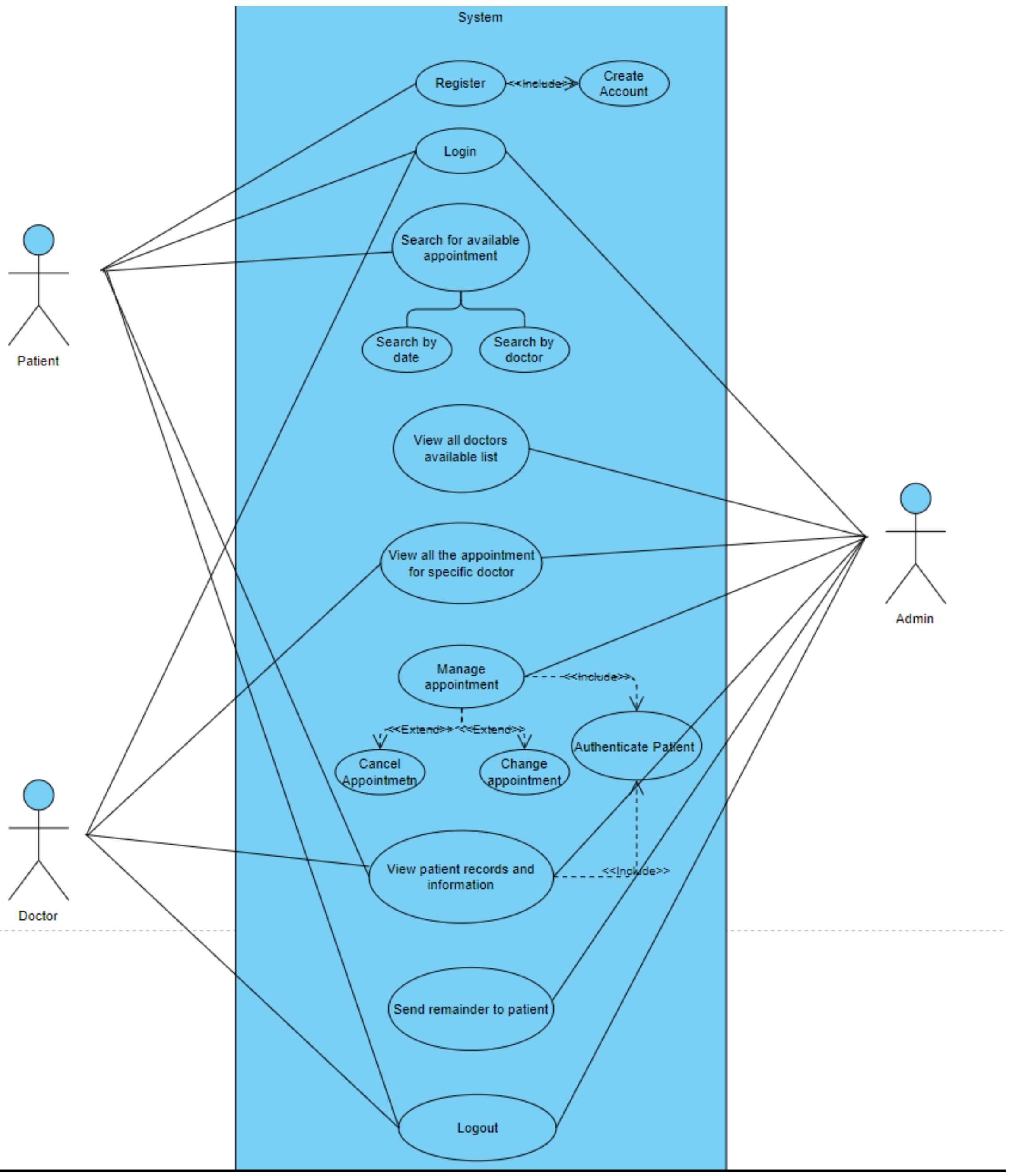


Fig. 3.3.1.1 – Use Case Diagram.

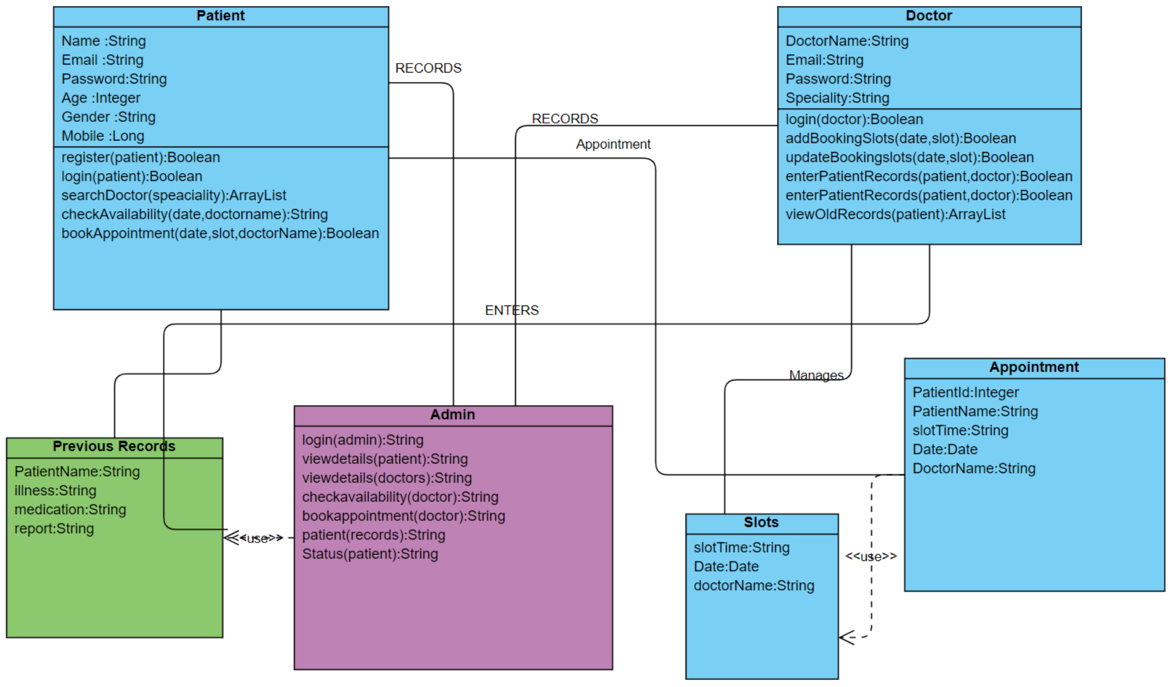


Fig. 3.3.1.2 – Class Diagram.

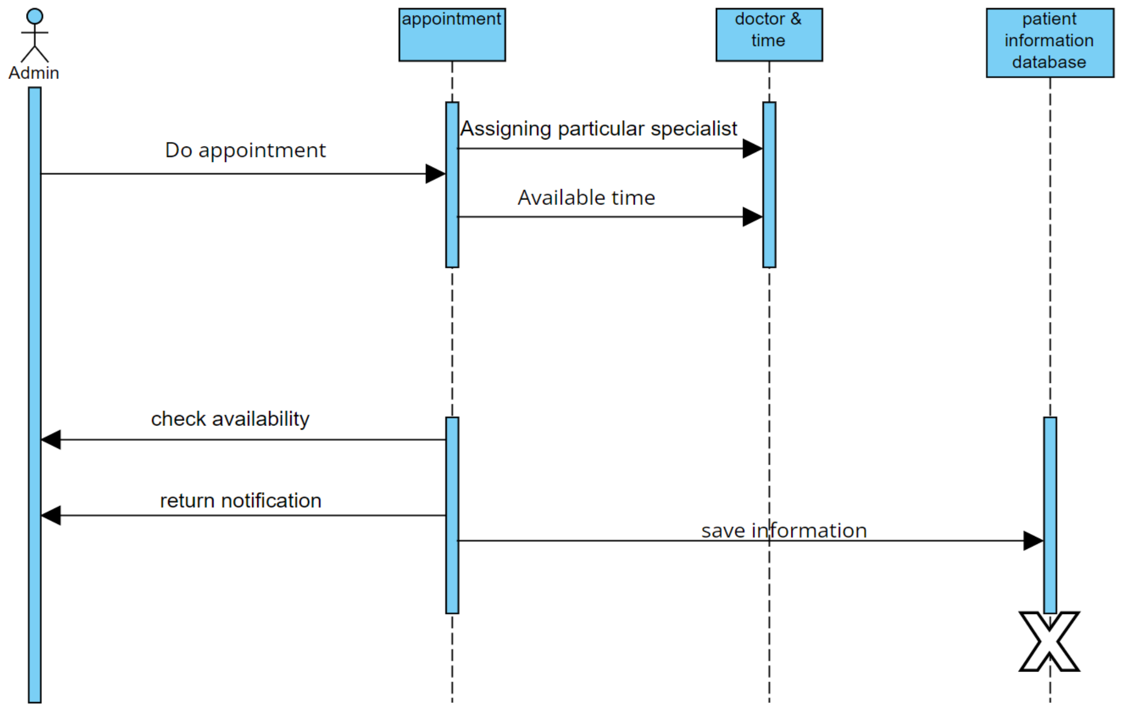


Fig. 3.3.1.3 – Sequence Diagram.

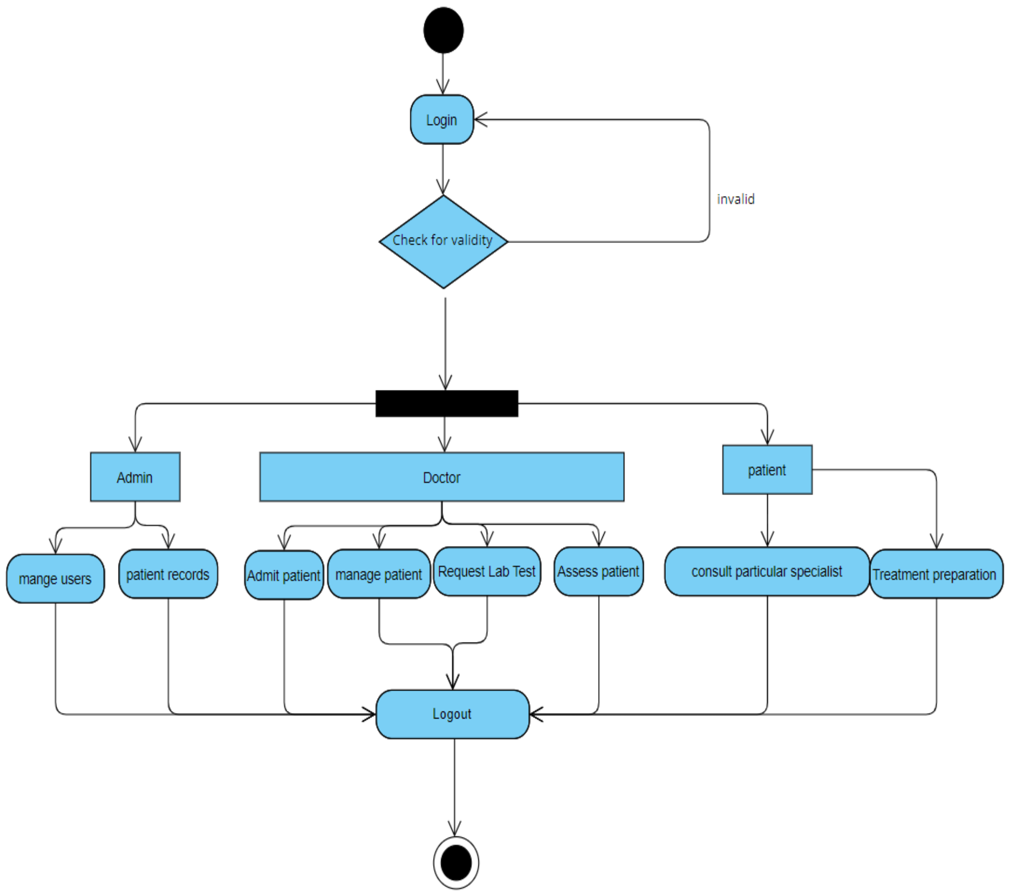


Fig. 3.3.1.4 – Activity Diagram.

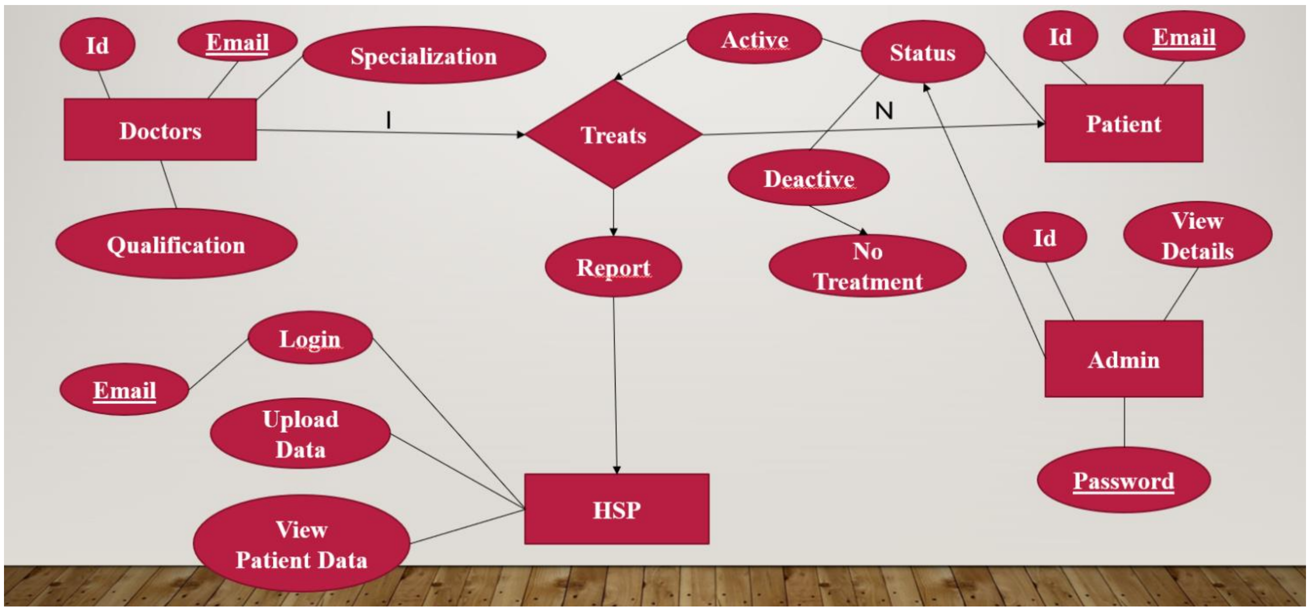


Fig. 3.3.1.5 – ER Diagram

3.4. Stepwise Implementation and Code

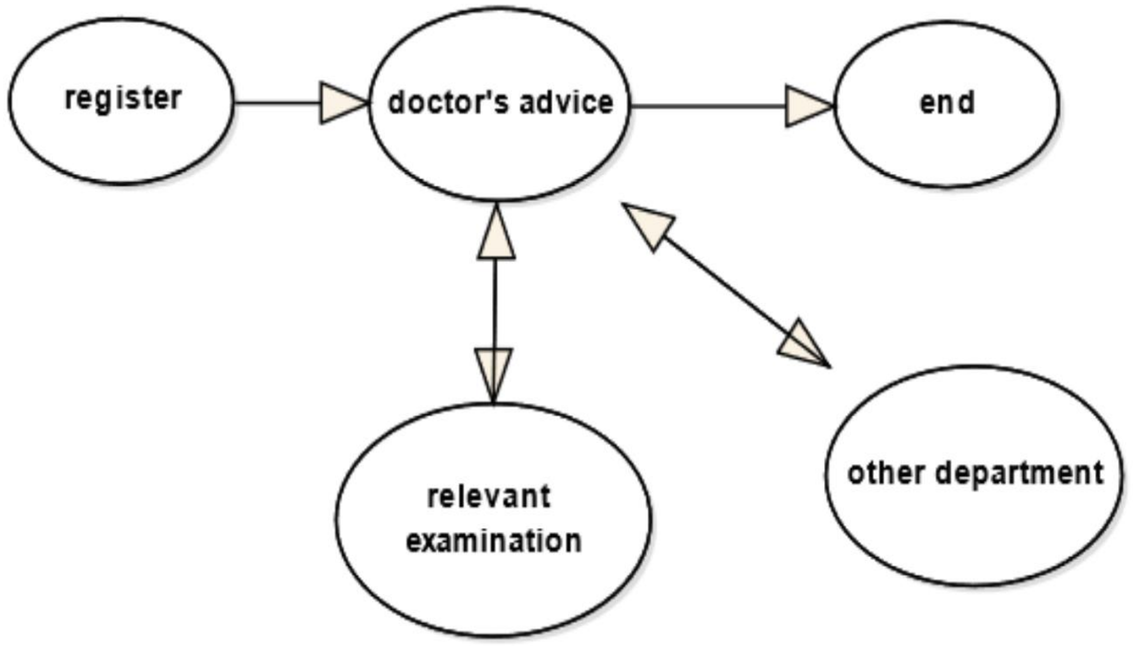


Fig. 3.4.1 – Implementation Process.

Data Collection

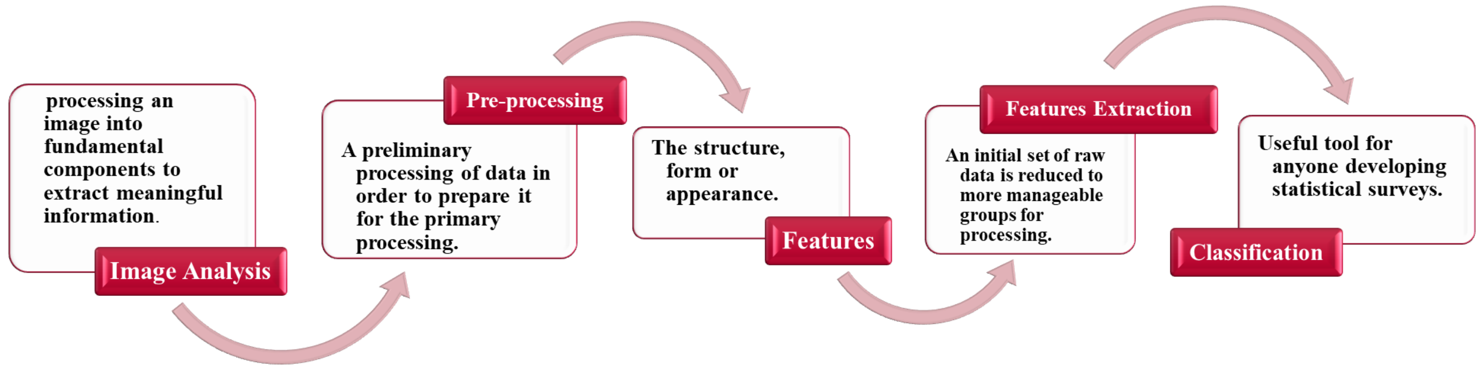

processing an image into fundamental components to extract meaningful information.
Image Analysis
Pre-processing
A preliminary processing of data in order to prepare it for the primary processing.
The structure, form or appearance.
Features
Features Extraction
An initial set of raw data is reduced to more manageable groups for processing.
Useful tool for anyone developing statistical surveys.
Classification

Fig. 3.4.2 – Collection of Data.

3.4.1. IMPLEMENTATION:

MODULES:

- Doctor.
- Patient.
- Admin.
- Machine learning.

MODULES DESCRIPTION:

Doctor:

The Doctor can register the first. While registering he required a valid doctor email and mobile for further communications. Once the doctor registers, then the admin can activate the customer. Once the admin activates the customer then the customer can login into our system. After login he can see the view-patient data. based on patient symptoms, the doctor will give the precautions and he will give the doctor treatment.

Patient:

The Doctor can register the first. While registering he required a valid patient email and mobile for further communications. Once the patient registers, then the admin can activate the patient. Once the admin activates the patient then the patient can login into our system. After login he can provide symptoms. based on patient symptoms, the doctor will give the precautions and he will give the doctor treatment.

Admin:

Admin can login with his credentials. Once he logs in, he can activate the doctors. The activated user only login in our applications. Once he logs in, he can activate the patients. The admin can add new data to the dataset. So, this data user can perform the testing process. admin can get predictions svm algorithm and also get the prediction from the decision tree.

Machine learning:

Machine learning refers to the computer's acquisition of a kind of ability to make predictive judgments and make the best decisions by analyzing and learning a large number of existing data. The representation algorithms include deep learning, artificial neural networks, decision trees, enhancement algorithms and so on. The key way for computers to acquire artificial intelligence is machine learning. Nowadays, machine learning plays an important role in various fields of artificial intelligence. Whether in aspects of internet search, biometric identification, auto driving, Mars robot, or in American presidential election, military decision assistants and so on, basically, as long as there is a need for data analysis, machine learning can be used to play a role**.**

3.4.2. INPUT AND OUTPUT DESIGN:

<u>INPUT DESIGN</u>

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

url.py:

```
from django.contrib import admin
from django.urls import path
from django.conf import settings
from django.conf.urls.static import static
from doctor import views as doctor
from patient import views as patient
from Medicalcare import views as medical
from hsp import views as hsp

urlpatterns = [
    path('admin/', admin.site.urls),
    path('index/', doctor.index, name='index'),
    path('logout/',medical.logout,name='logout'),
    path('admin1/', medical.adminlogin, name='admin1'),
    path('adminloginentered/', medical.adminloginentered, name='adminloginentered'),
    path('doctordetails/', medical.doctordetails, name='doctordetails'),
    path('patientdetails/', medical.patientdetails, name='patientdetails'),
    path('activatedoctor/',medical.activatedoctor,name='activatedoctor'),
    path('activatepatient/',medical.activatepatient,name='activatepatient'),
    path('storecsvdata/',medical.storecsvdata,name='storecsvdata'),
    path('svm/',medical.svm,name='svm'),
    path('decision/',medical.decision,name='decision'),



    path('hsplogin/', hsp.hsplogin, name='hsplogin'),
    path('hsploginentered/', hsp.hsploginentered, name='hsploginentered'),
    path('uploadpatientdata/', hsp.uploadpatientdata,name='uploadpatientdata'),
    path('viewpatientdata/', hsp.viewpatientdata,name='viewpatientdata'),
    path('doctorlogin/',doctor.doctorlogin,name='doctorlogin'),
    path('doctorregister/',doctor.doctorregister,name='doctorregister'),
    path('doctorlogincheck/',doctor.doctorlogincheck,name='doctorlogincheck'),
    path('doctorviewpatientdata/',doctor.doctorviewpatientdata,name='doctorviewpatientdata'),
    path('addtreatment/',doctor.addtreatment,name='addtreatment'),


    path('patientlogin/',patient.patientlogin,name='patientlogin'),
    path('patientregister/',patient.patientregister,name='patientregister'),
    path('patientlogincheck/',patient.patientlogincheck,name='patientlogincheck'),
    path('treatment/',patient.treatment,name='treatment'),
    path('patienttreatmentresult/',patient.patienttreatmentresult,name='patienttreatmentresult'),
]
if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)
```

models.py:

```
from django.db import models
class doctorModel(models.Model):
    name = models.CharField(max_length=50)
    email = models.EmailField()
    passwd = models.CharField(max_length=40)
    cwpasswd = models.CharField(max_length=40)
    mobileno = models.CharField(max_length=50, default="", editable=True)
    specialization = models.CharField(max_length=50, default="", editable=True)
    status = models.CharField(max_length=40, default="", editable=True)
```

```
 def  __str__(self):
        return self.email
    class Meta:
        db_table='doctorregister'

    class storedatamodel(models.Model):

    Pregnancies = models.CharField(max_length=500)
    Glucose = models.CharField(max_length=300)
    BloodPressure = models.CharField(max_length=300)
    SkinThickness = models.CharField(max_length=300)
    Insulin = models.CharField(max_length=255)
    BMI = models.CharField(max_length=255)
    DiabetesPedigreeFunction = models.CharField(max_length=255)
    Age = models.CharField(max_length=255)
    Outcome = models.CharField(max_length=255)

    def __str__(self):
        return
self.Pregnancies,self.Glucose,self.BloodPressure,self.SkinThickness,self.Insulin,self.BMI,self.
DiabetesPedigreeFunction,self.Age,self.Outcome
```

forms.py:

```
from django import forms
from doctor.models import *
from django.core import validators
class doctorForm(forms.ModelForm):
    name = forms.CharField(widget=forms.TextInput(), required=True, max_length=100,)
```

```
passwd = forms.CharField(widget=forms.PasswordInput(), required=True, max_length=100)
    cwpasswd = forms.CharField(widget=forms.PasswordInput(), required=True,
max_length=100)
    email = forms.CharField(widget=forms.TextInput(),required=True)
    mobileno= forms.CharField(widget=forms.TextInput(), required=True,
max_length=10,validators=[validators.MaxLengthValidator(10),validators.MinLengthValidat
or(10)])
    specialization= forms.CharField(widget=forms.TextInput(), required=True,
max_length=100)
    status = forms.CharField(widget=forms.HiddenInput(), initial='waiting', max_length=100)

    def __str__(self):
        return self.email

    class Meta:
        model=doctorModel
        fields=['name','passwd','cwpasswd','email','mobileno','specialization','status']
```

doctors.html:

```
{% extends 'base.html' %}
{% load static %}

{% block contents %}
<div class="main-banner header-text" id="top">
        <div class="Modern-Slider">
          <!-- Item -->
          <div class="item">
            <!--<div class="img-fill">-->
```

```
<img src="{% static 'images/home.jpg' %}" alt="">
        <div class="text-content">
<h3 style="color:blue"><center>Doctor REGISTRATION</center></h3>
             <center> <h4><form action="/doctorregister/" method="POST">
                  {% csrf_token %}
                  <table>
                     <tr><td style="color:blue">doctor
name</td><td>{{form.name}}</td></tr>
              <tr><td style="color:blue">Password</td><td>{{form.passwd}}</td></tr>
              <tr><td
style="color:blue">confirmPassword</td><td>{{form.cwpasswd}}</td></tr>
              <tr><td style="color:blue">Email</td><td>{{form.email}}</td></tr>
              <tr><td style="color:blue">Mobile</td><td>{{form.mobileno}}</td></tr>
              <tr><td
style="color:blue">specialization</td><td>{{form.specialization}}</td></tr>
              <tr><td></td><td>{{form.status}}</td></tr>
                  </table>
                <input type="submit" value="submit">


             </form></h4></center>


              {% if messages %}
                {% for message in messages %}
                  <font color='darkred'> {{message}} </font>
                {% endfor %}
              {% endif %}
        </div>
      </div>
     </div>
{% endblock %}
```

Views.py

```
from collections import defaultdict
from io import TextIOWrapper
import csv
from django.shortcuts import render
from django.http import HttpResponse
from patient.models import patientModel
from patient.forms import patientForm
from doctor.models import doctorModel, storedatamodel
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix
import pandas as pd
from django_pandas.io import read_frame
from sklearn import metrics
from sklearn import datasetsdef logout(request):
    return render(request, "index.html")


def adminlogin(request):
    return render(request, "admin/adminlogin.html")


def adminloginentered(request):
    if request.method == 'POST':
        uname=request.POST['uname']
        passwd=request.POST['upasswd']
        if uname =='admin' and passwd=='admin':
            return render(request,"admin/adminloginentered.html")
```

```
else:
            return HttpResponse("invalied credentials")
    return render(request, "admin/adminloginentered.html")

def doctordetails(request):
    qs=doctorModel.objects.all()
    return render(request,'admin/doctordetails.html',{"object":qs})

def patientdetails(request):
    qs=patientModel.objects.all()
    return render(request,'admin/patientdetails.html',{"object":qs})

def activatedoctor(request):
    if request.method == 'GET':
        uname = request.GET.get('pid')
        print(uname)
        status = 'Activated'
        print("pid=", uname, "status=", status)
        doctorModel.objects.filter(id=uname).update(status=status)
        qs = doctorModel.objects.all()
        return render(request,"admin/doctordetails.html", {"object": qs})

def activatepatient(request):
    if request.method == 'GET':
        uname = request.GET.get('pid')
        print(uname)
        status = 'Activated'
        print("pid=", uname, "status=", status)
        patientModel.objects.filter(id=uname).update(status=status)
```

```
qs = patientModel.objects.all()
        return render(request, 'admin/patientdetails.html', {"object": qs})
def storecsvdata(request):
    if request.method == 'POST':
        name = request.POST.get('name')
        csvfile =TextIOWrapper( request.FILES['file'])
        # columns = defaultdict(list)
        storecsvdata =csv.DictReader(csvfile)
for row1 in storecsvdata:
                Pregnancies = row1["Pregnancies"]
                Glucose = row1["Glucose"]
                BloodPressure = row1["BloodPressure"]
                SkinThickness = row1["SkinThickness"]
                Insulin = row1["Insulin"]
                DiabetesPedigreeFunction = row1["DiabetesPedigreeFunction"]
                Age = row1["Age"]
                BMI = row1["BMI"]
                Outcome = row1["Outcome"]

storedatamodel.objects.create(Pregnancies=Pregnancies, Glucose=Glucose,
BloodPressure=BloodPressure, SkinThickness=SkinThickness,
Insulin=Insulin,BMI=BMI,DiabetesPedigreeFunction=DiabetesPedigreeFunction,Age=Age,O
utcome=Outcome)
        print("Name is ",csvfile)
        return HttpResponse('CSV file successful uploaded')
    else:
        return render(request, 'admin/storecsvdata.html', {})
```

Support Vector Algorithm:

```
def svm(request):
    qs = storedatamodel.objects.all()
    data = read_frame(qs)
    data = data.fillna(data.mean())
    #data[0:label]
    data.info()
    print(data.head())
    print(data.describe())
    #print("data-label:",data.label)
    dataset = data.iloc[:,[6,7]].values
    print("x",dataset)
    dataset1 = data.iloc[:,-1].values
    print("y",dataset1)
    print("shape",dataset.shape)
    X = dataset
    y = dataset1
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1/3,random_state=0)
    svclassifier = SVC(kernel='linear')
    svclassifier.fit(X_train, y_train)
    #print(svclassifier.predict([0.58, 0.76]))
    y_pred = svclassifier.predict(X_test)
    m = confusion_matrix(y_test, y_pred)
    accurancy = classification_report(y_test, y_pred)

    print(accurancy)
    print("Confusion Matrix")
```

```
 print(" ")
    print(m)
    x = accurancy.split()
    print(" ")
    print("Toctal splits: ", len(x))
dict = {

        "m": m,
        "accurancy": accurancy,
        'len0': x[0],
        'len1': x[1],
        'len2': x[2],
        'len3': x[3],
        'len4': x[4],
        'len5': x[5],
        'len6': x[6],
        'len7': x[7],
        'len8': x[8],
        'len9': x[9],
        'len10': x[10],
        'len11': x[11],
        'len12': x[12],
        'len13': x[13],
        'len14': x[14],
        'len15': x[15],
        'len16': x[16],
        'len17': x[17],
        'len18': x[18],
 'len19': x[19],
        'len20': x[20],
        'len21': x[21],
        'len22': x[22],
        'len23': x[23],
        'len24': x[24],
        'len25': x[25],
        'len26': x[26],
        'len27': x[27],
        'len28': x[28],
        }
return render(request, 'admin/accuracy.html', dict)
```

Decision Tree Algorithm:

```
def decision(request):
    qs = storedatamodel.objects.all()
    data = read_frame(qs)
    data = data.fillna(data.mean())
    #data[0:label]
    data.info()
    print(data.head())
    print(data.describe())
    #print("data-label:",data.label)
    dataset = data.iloc[:,[6,7]].values
    print("x",dataset)
    dataset1 = data.iloc[:,-1].values
    print("y",dataset1)
    print("shape",dataset.shape)
    X = dataset
    y = dataset1
```

```
 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1/3,random_state=0)
   from sklearn.tree import DecisionTreeClassifier
   dt =DecisionTreeClassifier()
   dt.fit(X_train, y_train)
   #print(svclassifier.predict([0.58, 0.76]))
   y_pred = dt.predict(X_test)
   m = confusion_matrix(y_test, y_pred)
   accurancy = classification_report(y_test, y_pred)

print(accurancy)
   print("Confusion Matrix")
   print(" ")
   print(m)
   x = accurancy.split()
   print(" ")
   print("Toctal splits: ", len(x))

dict = {


       "m": m,
       "accurancy": accurancy,
       'len0': x[0],
       'len1': x[1],
       'len2': x[2],
       'len3': x[3],
       'len4': x[4],
       'len5': x[5],
       'len6': x[6],
       'len7': x[7],
       'len8': x[8],
       'len9': x[9],
       'len10': x[10],
       'len11': x[11],
       'len12': x[12],
       'len13': x[13],
       'len14': x[14],
       'len15': x[15],
       'len16': x[16],
       'len17': x[17],
       'len18': x[18],
       'len19': x[19],
       'len20': x[20],
       'len21': x[21],
       'len22': x[22],
       'len23': x[23],
       'len24': x[24],
       'len25': x[25],
       'len26': x[26],
       'len27': x[27],
       'len28': x[28],
       # 'len29': x[29],
       # 'len30': x[30],
       # 'len31': x[31],
       # 'len32': x[32],
       # 'len33': x[33],

   }
   return render(request, 'admin/navieaccuracy.html', dict)
```

CHAPTER 4
RESULTS AND DISCUSSION

4.1. Performance metrics

1. Precision: Precision is defined as the ratio of correctly classified positive samples (True Positive) to a total number of classified positive samples (either correctly or incorrectly).
Precision Formula = True Positive/True Positive + False Positive

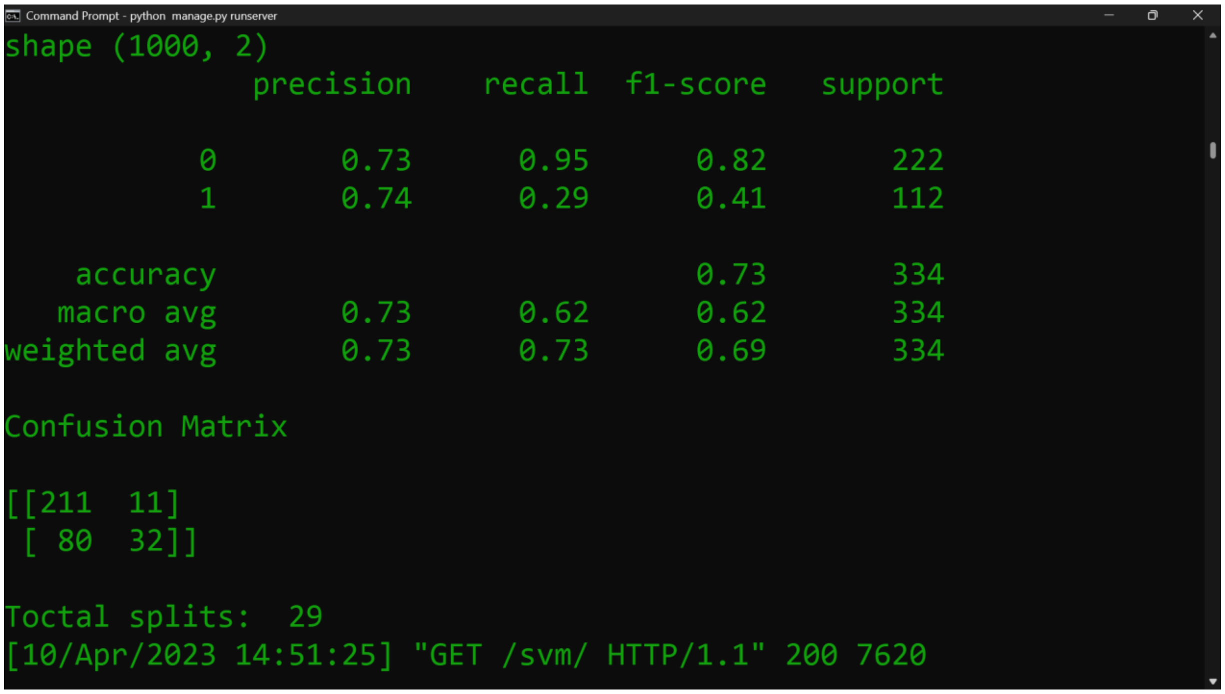
2. Recall: The recall is calculated as the ratio between the numbers of Positive samples correctly classified as Positive to the total number of Positive samples. The recall measures the model's ability to detect positive samples. The higher the recall, the more positive samples detected.
Recall Formula = True Positive/True Positive + False Negative

3. F1 Score: F1 score is a machine learning evaluation metric that combines precision and recall scores.
F1 Score Formula = $\frac{2*Precision*Recall}{Precision+Recall}$

4. Support: Support defined as the number of samples of the true response that lies in each class of target values.
Support (0) Formula = True Positive + False Positive
Support (1) Formula = False Negative + True Negative

5. Accuracy: Accuracy is the percentage of correct classifications that a trained machine learning model achieves, i.e., the number of correct predictions divided by the total number of predictions across all classes.

Accuracy Formula: $\frac{TP+TN}{TP+TN+FP+FN}$



```
shape (1000, 2)
              precision    recall  f1-score   support

           0       0.73      0.95      0.82       222
           1       0.74      0.29      0.41       112

    accuracy                           0.73       334
   macro avg       0.73      0.62      0.62       334
weighted avg       0.73      0.73      0.69       334

Confusion Matrix

[[211  11]
 [ 80  32]]

Toctal splits:  29
[10/Apr/2023 14:51:25] "GET /svm/ HTTP/1.1" 200 7620
```

Fig. 4.1.1 – Screenshot of SVM Algorithm Result.

Confusion Matrix = $\begin{bmatrix} \boldsymbol{TP} & \boldsymbol{FP} \\ \boldsymbol{FN} & \boldsymbol{TN} \end{bmatrix}$

Calculations:

<u>SVM Algorithm (0)</u>

Precision: $\frac{TP}{TP+FP}$ = 0.73

Recall: $\frac{TP}{TP+FN}$ = 0.95

F1 Score: $\frac{2*Precision*Recall}{Precision+Recall}$ = 0.82

Support: TP + FP => 211 + 11 = 222

SVM Algorithm (1)

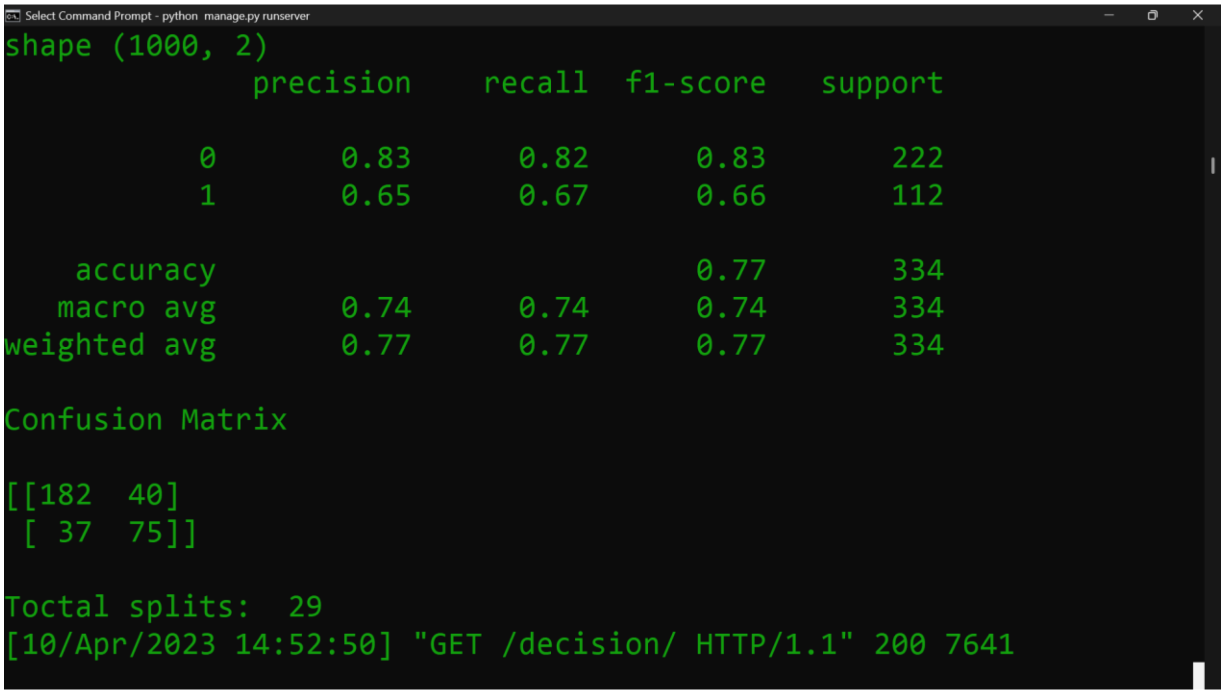
Precision: $\frac{TP}{TP+FP}$ = 0.74

Recall: $\frac{TP}{TP+FN}$ = 0.29

F1 Score: $\frac{2*Precision*Recall}{Precision+Recall}$ = 0.41

Support: FN + TN => 80 + 32 = 112

Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$ = 0.73



```
shape (1000, 2)
              precision    recall  f1-score   support

           0       0.83      0.82      0.83       222
           1       0.65      0.67      0.66       112

    accuracy                           0.77       334
   macro avg       0.74      0.74      0.74       334
weighted avg       0.77      0.77      0.77       334

Confusion Matrix

[[182  40]
 [ 37  75]]

Toctal splits:  29
[10/Apr/2023 14:52:50] "GET /decision/ HTTP/1.1" 200 7641
```

Fig. 4.1.2 – Screenshot of DECISION TREE Algorithm Result.

Calculations:

<u>Decision Tree Algorithm (0)</u>

Precision: $\frac{TP}{TP+FP}$ = 0.83

Recall: $\frac{TP}{TP+FN}$ = 0.82

F1 Score: $\frac{2*Precision*Recall}{Precision+Recall}$ = 0.83

Support: TP + FP => 182 + 40 = 222

<u>Decision Tree Algorithm (1)</u>

Precision: $\frac{TP}{TP+FP}$ = 0.65

Recall: $\frac{TP}{TP+FN}$ = 0.67

F1 Score: $\frac{2*Precision*Recall}{Precision+Recall}$ = 0.66

Support: FN + TN => 37 + 75 = 112

Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$ = 0.77

Comparison between both the algorithms:

Algorithms	**Precision**	**Recall**	**F1 Score**	**Support**	**Accuracy**
SVM (0)	0.73	0.95	0.82	222	
SVM (1)	0.74	0.29	0.41	112	0.73
Decision Tree (0)	0.83	0.82	0.83	222	
Decision Tree (1)	0.65	0.67	0.66	112	0.77

Table 4.1.1 – Comparison of SVM vs DECITION TREE Algorithms.

From the above table we can observe that Decision Tree Algorithm performed better compare to SVM Algorithm.

In all the aspects that is Precision, Recall, F1 Score, Support and Accuracy Decision Tree Algorithm performance is clearly better than SVM Algorithm.

SCREENSHOTS OF THE VARIOUS LOGINS AND TABLES

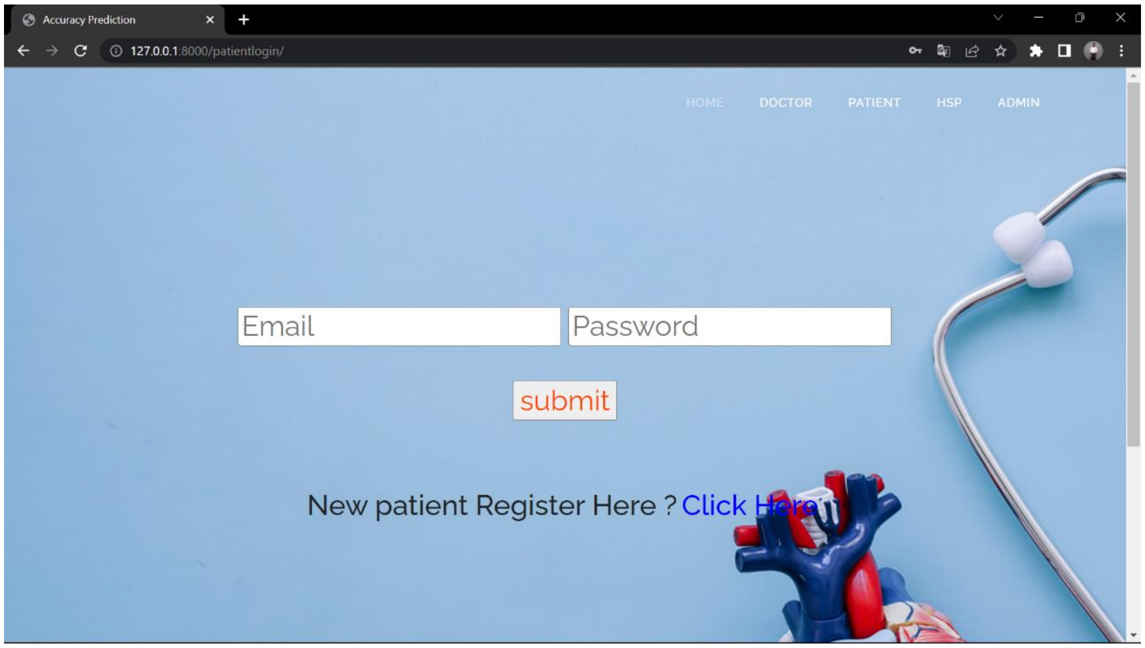

Accuracy Prediction
127.0.0.1:8000/patientlogin/
HOME
DOCTOR
PATIENT
HSP
ADMIN
Email
Password
submit
New patient Register Here ? Click Here

Fig. 4.1.3 – Screenshot of Patient Login.

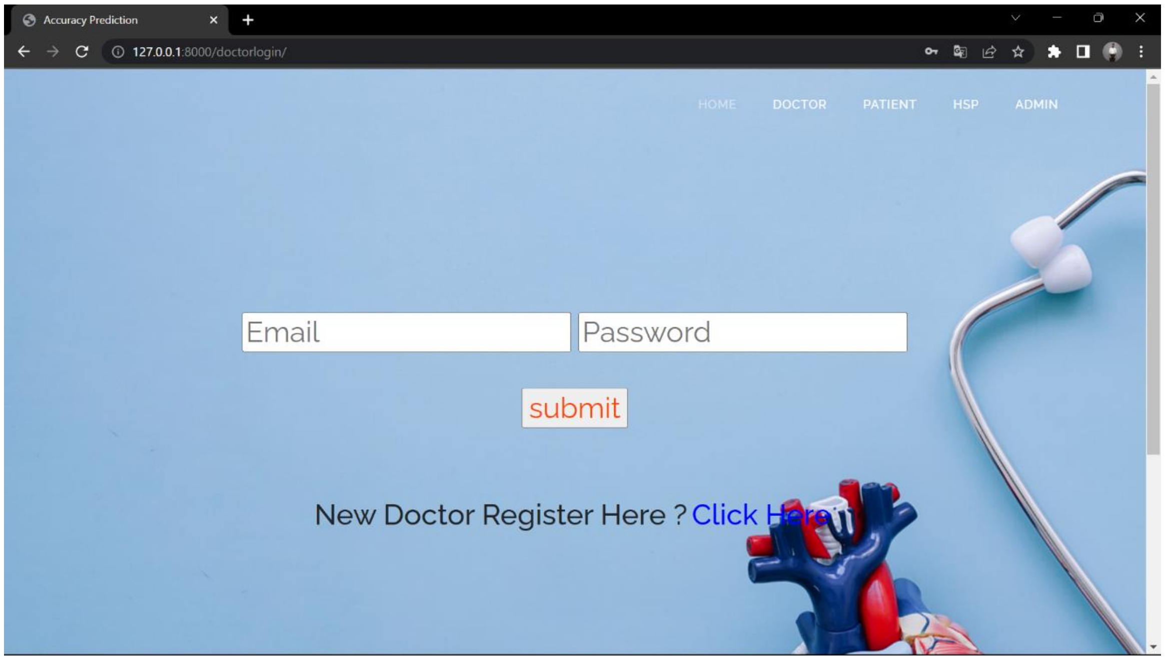

Accuracy Prediction
127.0.0.1:8000/doctorlogin/
HOME
DOCTOR
PATIENT
HSP
ADMIN
Email
Password
submit
New Doctor Register Here ? Click Here

Fig. 4.1.4 – Screenshot of Doctor Login.

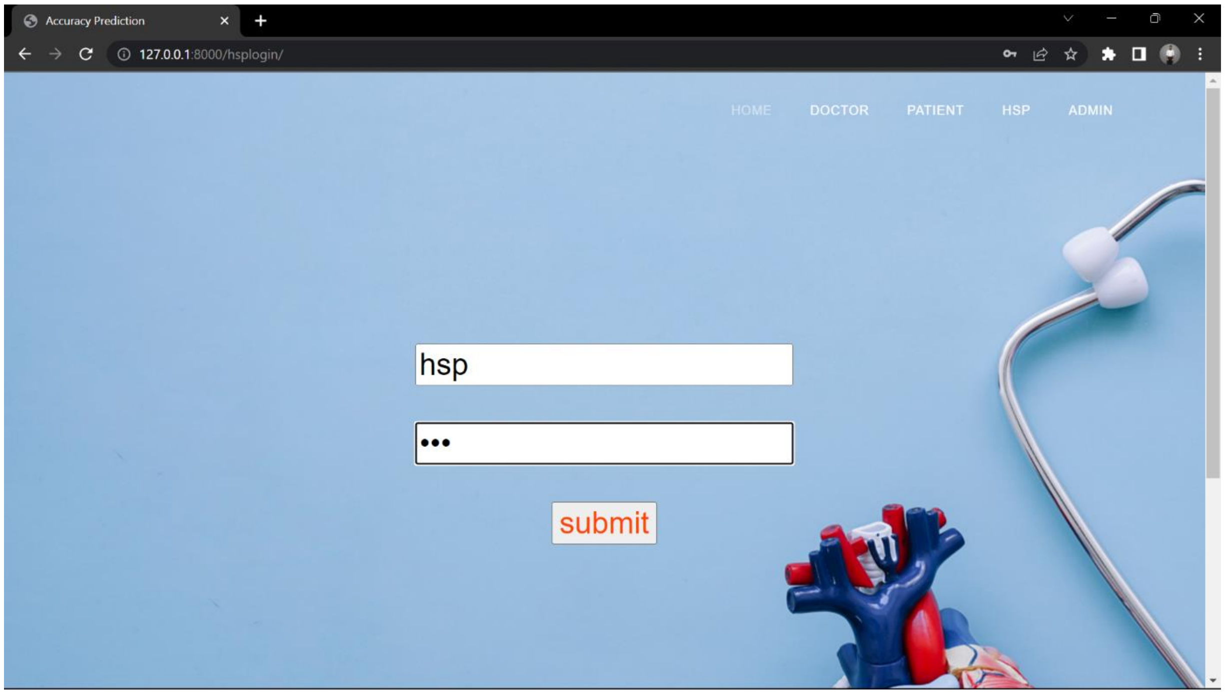


Fig. 4.1.5 – Screenshot of HSP Page.

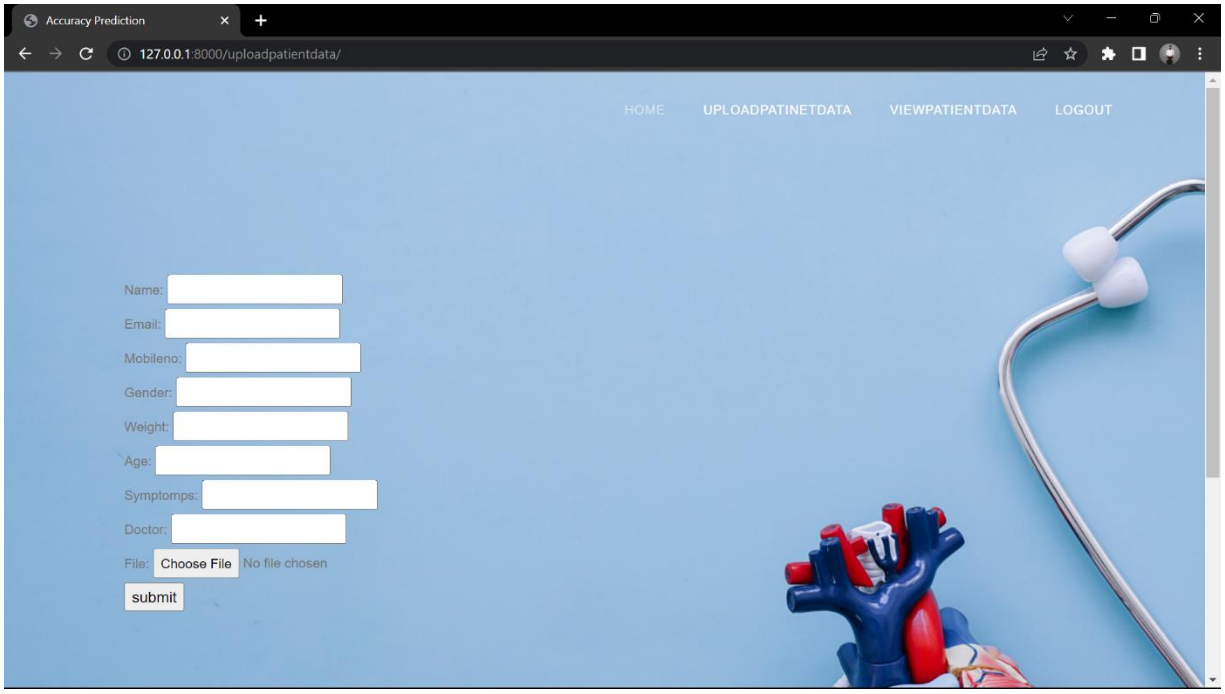
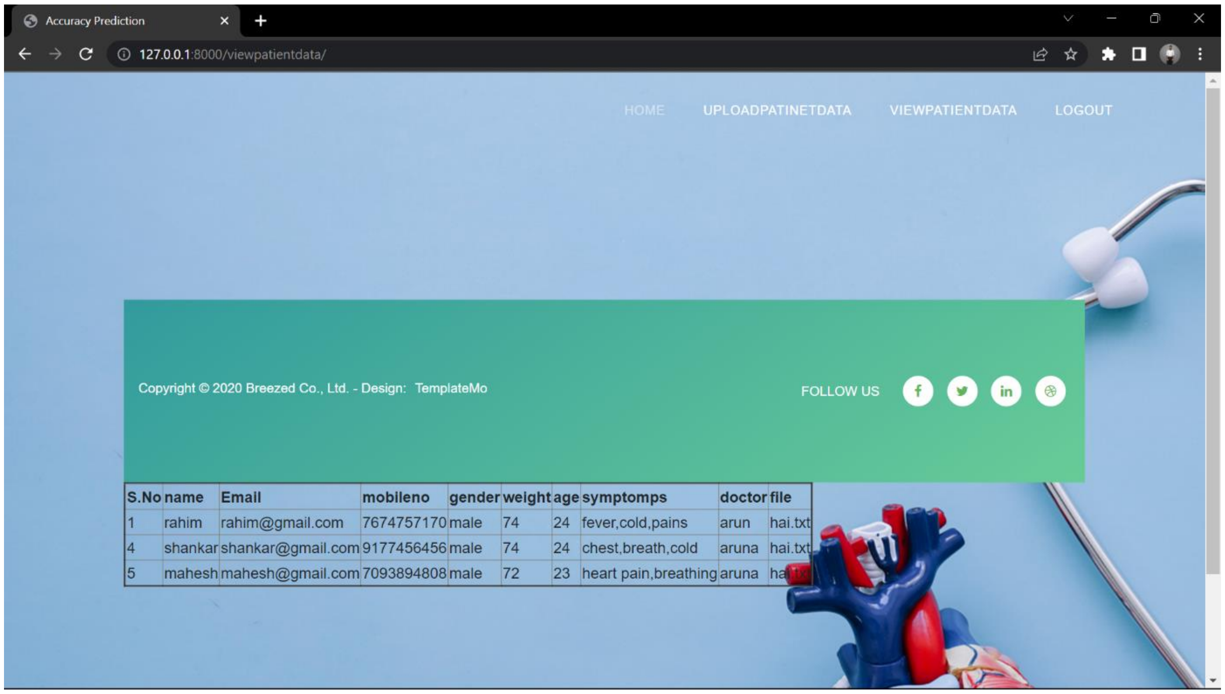
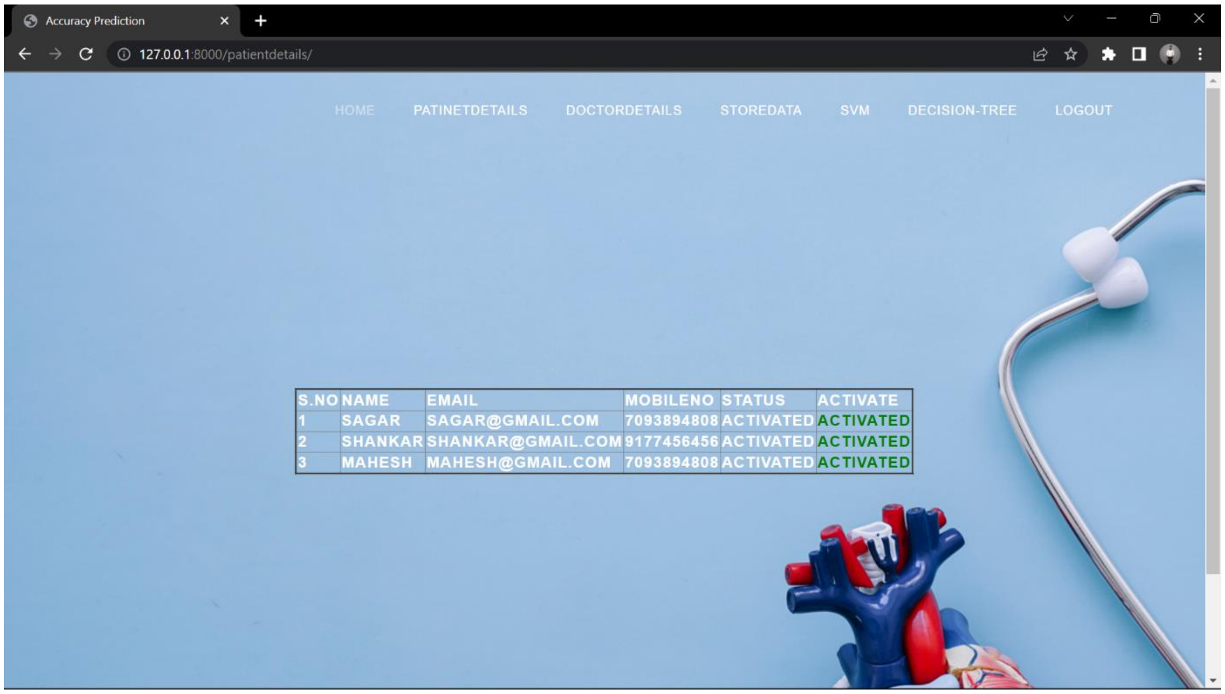


Fig. 4.1.6 – Screenshot of Patient Data Upload.


Accuracy Prediction
127.0.0.1:8000/viewpatientdata/
HOME
UPLOADPATINETDATA
VIEWPATIENTDATA
LOGOUT
Copyright © 2020 Breezed Co., Ltd. - Design: TemplateMo
FOLLOW US

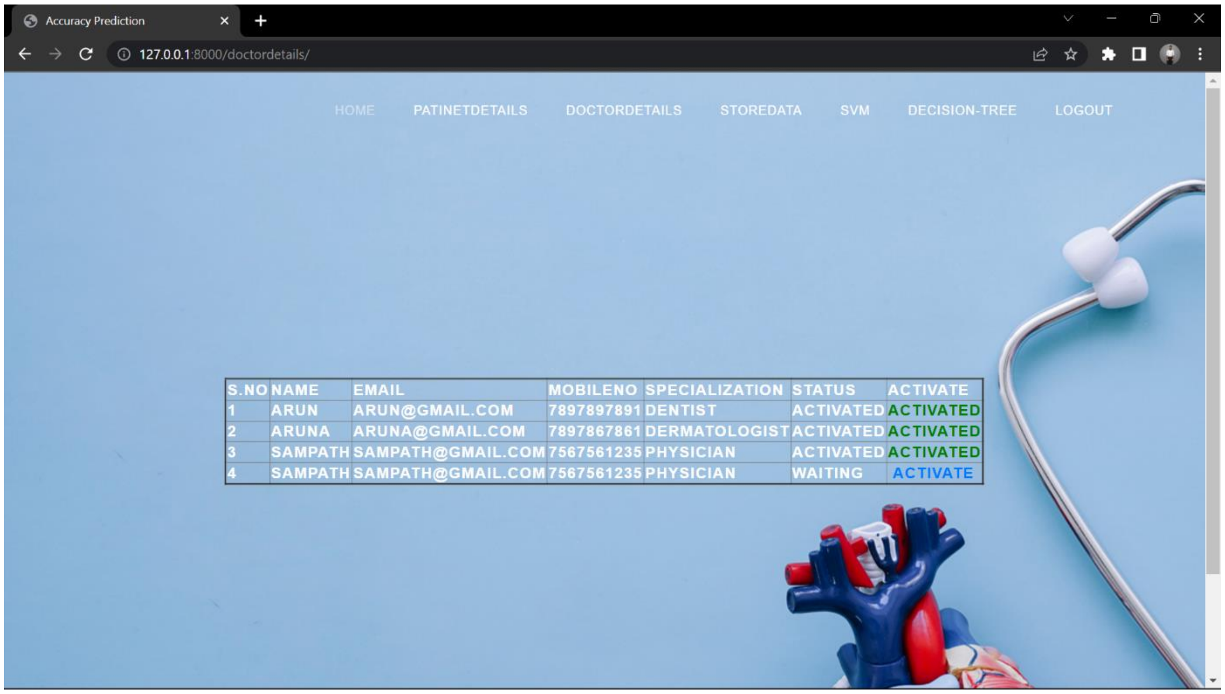
S.No	name	Email	mobileno	gender	weight	age	symptomps	doctor	file
1	rahim	rahim@gmail.com	7674757170	male	74	24	fever,cold,pains	arun	hai.txt
4	shankar	shankar@gmail.com	9177456456	male	74	24	chest,breath,cold	aruna	hai.txt
5	mahesh	mahesh@gmail.com	7093894808	male	72	23	heart pain,breathing	aruna	ha

Fig. 4.1.7 – Screenshot of Patient Data View.



S.NO	NAME	EMAIL	MOBILENO	STATUS	ACTIVATE
1	SAGAR	SAGAR@GMAIL.COM	7093894808	ACTIVATED	ACTIVATED
2	SHANKAR	SHANKAR@GMAIL.COM	9177456456	ACTIVATED	ACTIVATED
3	MAHESH	MAHESH@GMAIL.COM	7093894808	ACTIVATED	ACTIVATED

Fig. 4.1.8 – Screenshot of Patient Details.



S.NO	NAME	EMAIL	MOBILENO	SPECIALIZATION	STATUS	ACTIVATE
1	ARUN	ARUN@GMAIL.COM	7897897891	DENTIST	ACTIVATED	ACTIVATED
2	ARUNA	ARUNA@GMAIL.COM	7897867861	DERMATOLOGIST	ACTIVATED	ACTIVATED
3	SAMPATH	SAMPATH@GMAIL.COM	7567561235	PHYSICIAN	ACTIVATED	ACTIVATED
4	SAMPATH	SAMPATH@GMAIL.COM	7567561235	PHYSICIAN	WAITING	ACTIVATE

Fig. 4.1.9 – Screenshot of Doctor Details.

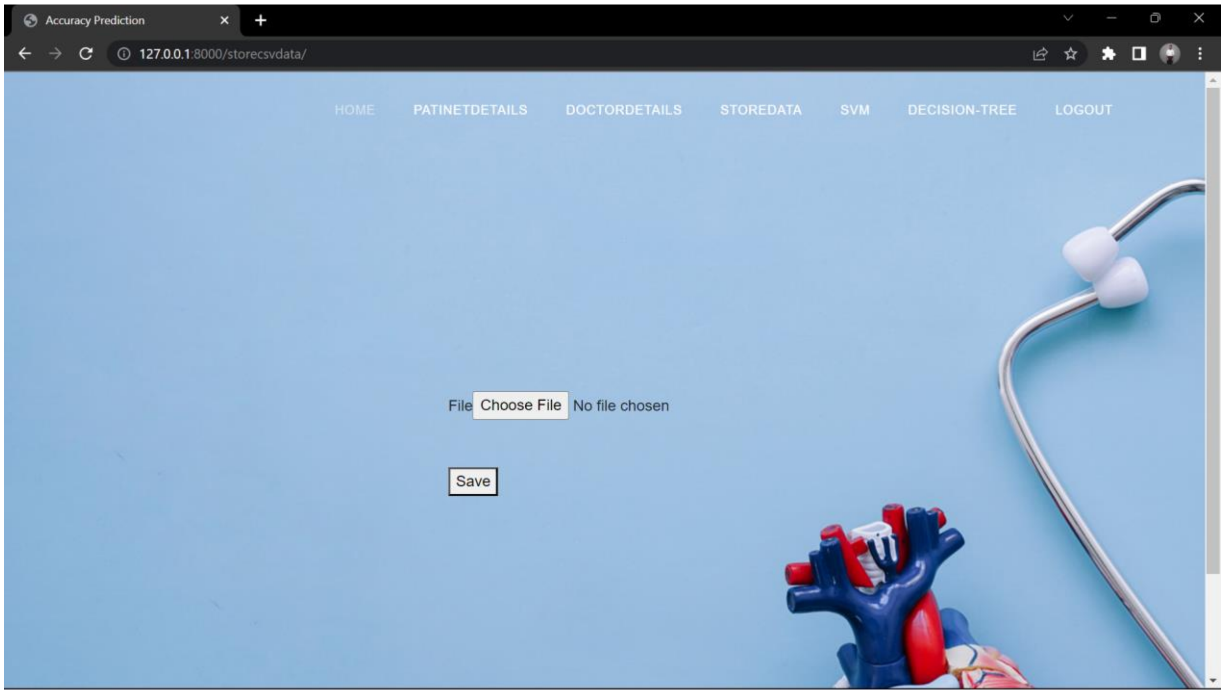
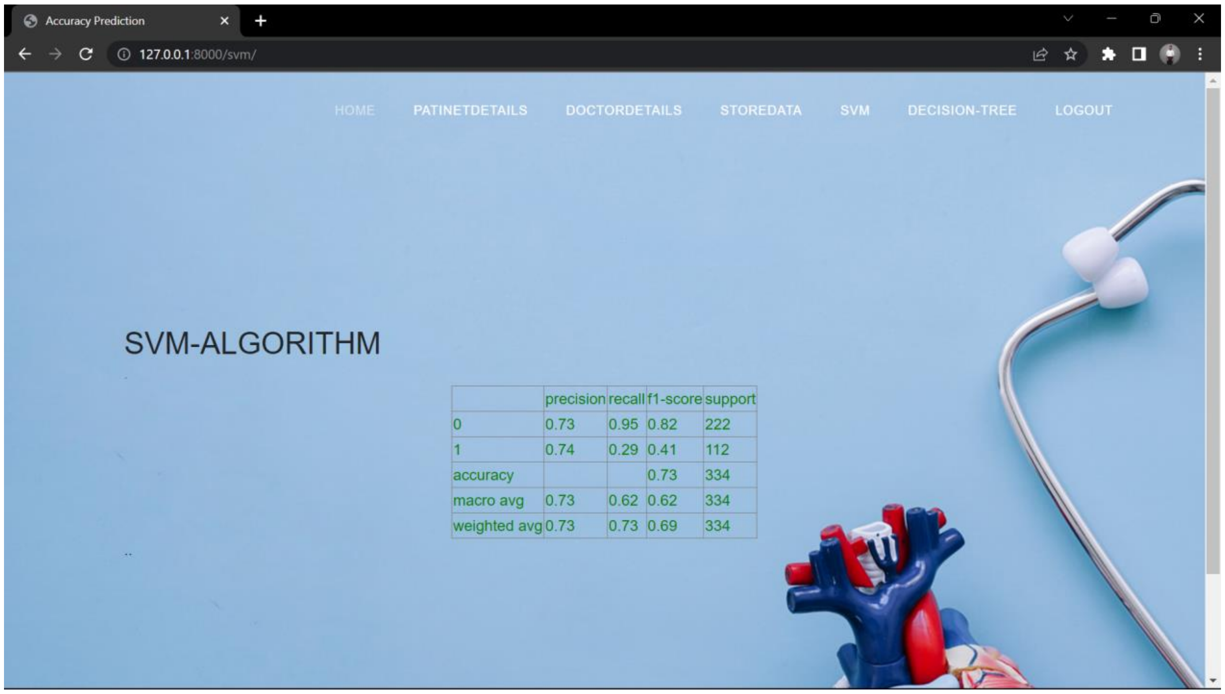


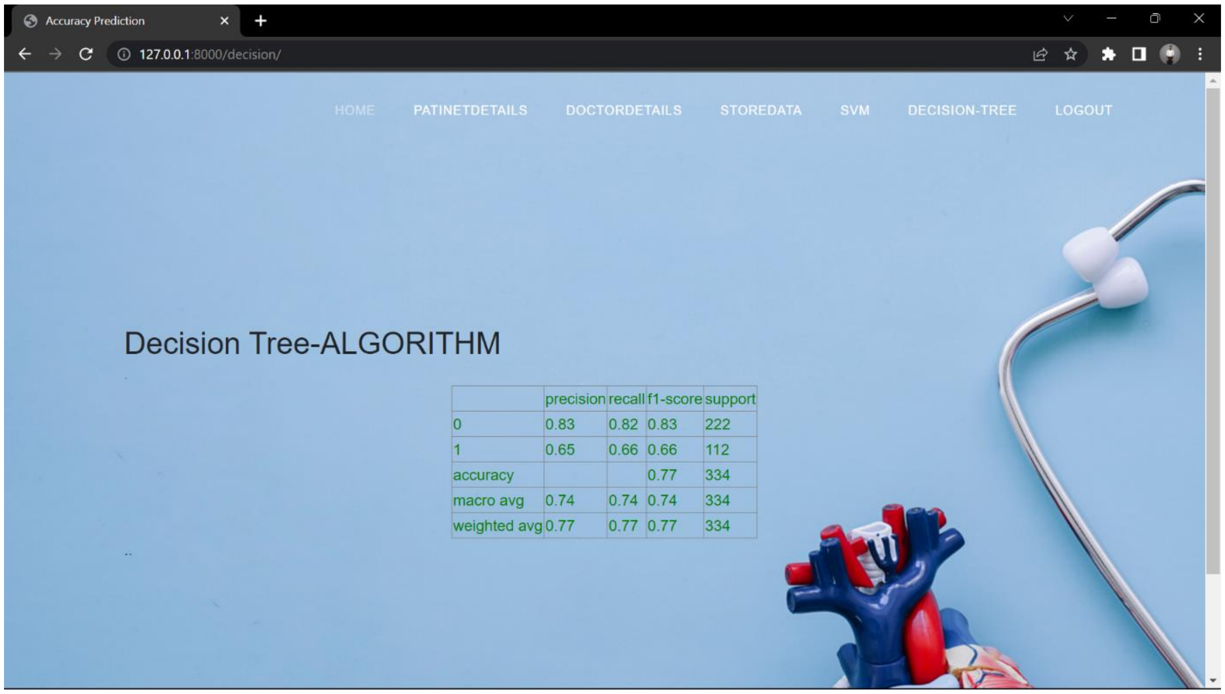
Fig. 4.1.10 – Screenshot of Dataset Upload.



SVM-ALGORITHM

	precision	recall	f1-score	support
0	0.73	0.95	0.82	222
1	0.74	0.29	0.41	112
accuracy			0.73	334
macro avg	0.73	0.62	0.62	334
weighted avg	0.73	0.73	0.69	334

Fig. 4.1.11 – Screenshot of SVM Table.



Decision Tree-ALGORITHM

	precision	recall	f1-score	support
0	0.83	0.82	0.83	222
1	0.65	0.66	0.66	112
accuracy			0.77	334
macro avg	0.74	0.74	0.74	334
weighted avg	0.77	0.77	0.77	334

Fig. 4.1.12 – Screenshot of DECISION TREE Table.

CHAPTER 5
CONCLUSION

CHAPTER 5

5.1 Conclusion:

Compare to present system proposed system performance is better. Out of the two Algorithms that is Decision Tree and SVM, Decision Tree Algorithm performing better. These reviews the main methods of machine learning, and summarizes several representative applications after understanding the history of machine learning in the medical field and its current application. The typical ideas and algorithms are summarized. At the same time, the improvement method based on machine learning in the process of visiting is proposed. However, this does not mean that ML is perfect. Whether in terms of technology, ethic or law, it has certain problems. The solution of these problems requires technicians and legal personnel. Working together, and how to strike a balance between manpower and machine is also a problem that everyone of us must face.

5.2 Future Scope:

Improving diagnosis: ML in healthcare can be used to develop better diagnostic tools to analyze medical images.
For example, a machine learning algorithm can be used in medical imaging using pattern recognition to look for patterns that indicate a particular disease.

Reducing costs: ML can be used to improve the efficiency of healthcare, which could lead to cost savings.
For example, machine learning in healthcare could be used to develop better algorithms for managing patient records or scheduling appointments. This could potentially help to reduce the amount of time and resources that are wasted on repetitive tasks in the healthcare system.

Improving care: ML in healthcare can also be used to improve the quality of patient care.
For example, deep learning algorithms could be used to develop systems that proactively monitor patients and provide alerts to medical devices or electronic health records when there are changes in their condition.

CHAPTER 6
REFERENCES

CHAPTER 6

REFERENCES

[1] Xuehu Wang, Yongchang Zheng, Lan Gan, Xuan Wang, Xinting Sang, Xiangfeng Kong, Jie Zhao, Study Liver Segmtation Method from CT Images based on Deformation Optimization and Sparse Statistics[D]. Beijing Institute of Technology, 2015.

[2] Qi Dou, Hao Chen, Lequan Yu, Lei Zhao, Jing Qin, Defeng Wang, Vincent CT Mok, Lin Shi, Pheng-Ann Heng. Study Automatic Detection of Cerebral Microbleeds from MR Images via 3D Convolutional Neural Networks[J]. IEEE Transactions on Medical Imaging, 2016:1-1.

[3] Greenspan H, Van Ginneken B, Summers RM. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. IEEE Transactions on Medical Imaging. 2016.

[4] Giger ML. Machine learning in medical imaging. Journal of the American College of Radiology. 2018;15(3):512-520

[5] Foster KR, Koprowski R, Skufca JD. Machine learning, medical diagnosis, and biomedical engineering research—Commentary. BioMedical Engineering Online. 2014. Online: Published 5 July 2014. Article number: 94

[6] Lundervold AS, Lundervold A. An overview of deep learning in medical imaging focusing on MRI. Zeitschrift fur Medizinische Physik. 2019;29(2):102-127

[7] M.P. Sendak, J. D'Arcy, S. Kashyap, M. Gao, M. Nichols, K. Corey, S. Balu A path for translation of machine learning products into healthcare delivery EMJ Innov, 10 (2020) 19-172

[8] G. Manogaran, D. Lopez A survey of big data architectures and machine learning algorithms in healthcare Int. J. Biomed. Eng. Technol., 25 (2–4) (2017), pp. 182-211

[9] R. Bhardwaj, A.R. Nambiar, D. Dutta A study of machine learning in healthcare 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC), vol. 2, IEEE (2017, July), pp. 236-241

GitHub Link -

https://github.com/rohithreddy2001/majorProject.io