# INTRODUCTION

**1.1. ABOUT THE PROJECT**

Machine Learning is used across many spheres around the world. The healthcare industry is no exception. Machine Learning can play an essential role in predicting presence/absence of Locomotor disorders, Heart diseases and more. Such information, if predicted well in advance, can provide important insights to doctors who can then adapt their diagnosis and treatment per patient basis. Machine Learning is a system of computer algorithms that can learn from example through self-improvement without being explicitly coded by a programmer. Machine learning is a part of artificial Intelligence which combines data with statistical tools to predict an output which can be used to make actionable insights. Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict. The core objective of machine learning is the learning and inference. First of all, the machine learns through the discovery of patterns. One crucial part of the data scientist is to choose carefully which data to provide to the machine.

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data. This project aims to predict future Heart Disease by analyzing data of patients which classifies whether they have heart disease or not using machine-learning algorithms.

**1.2. PROJECT DESCRIPTION**

The proposed framework of the research work is conducted in three different modules as in Dataset and its Features The first module includes data collection and preprocessing of data. A large sample of patient records is collected from the Kaggle website. The data set consists of over 1000 records. It includes six features. Each data set is in the Comma Separated Values (CSV) file format and available as supplement. In the second step, data are pre-processed to remove noisy data, cleansing of data, normalization. In this project, various ML algorithms are used for predicting heart disease, and then the accuracies of different algorithms are compared using bar plot as shown in the Screen 5.

**1.3. MODULES**

**1.3.1 Data Collection**

This is the first real step towards the real development of a machine learning model, collecting data. This is a critical step that will cascade in how good the model will be, the more and better data that we get, the better our model will perform.

There are several techniques to collect the data, like web scraping, manual interventions and etc.

Heart Disease dataset taken from Kaggle (<https://www.kaggle.com/ronitf/heart-disease-uci>)

**1.3.2. Dataset**

The dataset consists of 1024 individual data. There are 14 columns in the dataset, which are described below.

1. ***Age***: displays the age of the individual.
2. ***Sex***: displays the gender of the individual using the following format :  
   1 = male  
   0 = female
3. ***Chest-pain type(*cp*)***: displays the type of chest-pain experienced by the individual using the following format :

1 = typical angina  
2 = atypical angina  
3 = non — anginal pain  
4 = asymptotic

1. ***Resting Blood Pressure(*trestbps*)***: displays the resting blood pressure value of an individual in mmHg (unit)
2. ***Serum Cholestrol(*chol*)***: displays the serum cholesterol in mg/dl (unit)
3. ***Fasting Blood Sugar(*fbs*)***: compares the fasting blood sugar value of an individual with 120mg/dl.  
   If fasting blood sugar > 120mg/dl then : 1 (true) else : 0 (false)
4. ***Resting ECG (*restecg*)***: displays resting electrocardiographic results  
   0 = normal  
   1 = having ST-T wave abnormality  
   2 = left ventricular hyperthrophy
5. ***Max heart rate achieved***: displays the max heart rate achieved by an individual.
6. ***Exercise induced angina*** :  
   1 = yes  
   0 = no
7. ***ST depression induced by exercise relative to rest***: displays the value which is an integer or float.
8. ***Peak exercise ST segment*** :  
   1 = upsloping  
   2 = flat  
   3 = downsloping
9. ***Number of major vessels (0–3) colored by flourosopy*** : displays the value as integer or float.
10. ***Thal*** : displays the thalassemia :  
    3 = normal  
    6 = fixed defect  
    7 = reversible defect
11. ***Diagnosis of heart disease*** : Displays whether the individual is suffering from heart disease or not :

0 = absence  
1 = present.

## 1.3.3. Data Preparation

### Wrangle data and prepare it for training. Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.). Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data. Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis. Split into training and evaluation sets

### 1.3.4. Model Selection

### We used Naïve Bayes, SVM and Decision Tree Classifier machine learning algorithms, We got a accuracy of 96.7% on test set when Decision Tree classifier is used, so we implemented this algorithm.

### 1.3.5. Analyze and Prediction

### In the actual dataset, we chose only few features : Age, Sex, Chest-pain type(cp), Resting blood pressure(trest bps), Serum Cholestrol(chol), Fasting Blood Sugar(fbs), Resting ECG(restecg), Max Heart Rate achieved, Exercise Induced Angina, ST depression induced by exercise relative to rest, Peak exercise ST segment.

### 1.3.6. Saving the Trained Model

Once you’re confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or .pkl file using a library like pickle.

Make sure you have pickle installed in your environment.

Next, import the module and dump the model into.pkl file.

**2. SYSTEM ANALYSIS**

Analysis is a logical process. The objective of this phase is to determine exactly what must be done to solve the problem. Tools such as Class Diagrams, Sequence Diagrams, Data Flow Diagrams and data dictionary are used in developing a logical model of system.

**2.1. DOMAIN ANALYSIS**

Domain analysis is the process by which a software engineer learns background information, which helps to understand the problem. The word ‘domain’ in the case means the general field of business or technology in which the customers expect to be using the software. For this project, personal experiences of the team members with competing software were observed to understand the domain.

**2.2. REQUIREMENT ANALYSIS**

A requirement is a relatively short and concise piece of information, expressed as a fact. It can be written as a sentence or can be expressed using some kind of diagram.

**2.2.1. Functional Requirements**

Functional requirements describe what the system should do. The functional requirements can be further categorized as follows:

* What inputs the system should accept?
* What outputs the system should produce?
* What data the system must store?
* What are the computations to be done?

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and the steps are necessary to put transaction data in to a usable form for processing that 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:

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

**2.2.2. Non-Functional Requirements**

Non-functional requirements are the constraints that must be adhered during development. They limit what resources can be used and set bounds on aspects of the software’s quality.

**2.2.3. User Interfaces**

The User Interface is a GUI developed using Flask Framework in Python.

**2.2.4. Software Interfaces**

The main processing is done in Python and console application.

**2.2.5. Manpower Requirements**

5 members can complete the project in 2-4 months if they work fulltime on it.

**2.3. EXISTING SYSTEM**

There are many prediction systems available in modern days. Although they use Machine Learning techniques to predict, some of the algorithms have low accuracy rates. There are many existing algorithms like KNN, NB, SVM to predict heart disease using machine learning. Though all these algorithms can be used for prediction, sometimes they can predict wrongly. Due to this there will be many problems for predicting incorrectly.

**2.3.1. Disadvantages**

* Low Accuracy rates.
* Accuracy depends on the quality of the data.
* Higher time required to train datasets.
* Doesn’t work well with larger datasets.

**2.4. PROPOSED SYSTEM**

To overcome this we are implementing Decision Tree Classifier in order to achieve accurate results in less time. Our aim is to build an application of heart disease prediction system using robust Machine Learning algorithm which is Decision Tree Classifier. A CSV file is given as input. After the successful completion of operation the result is predicted and displayed.

**2.4.1. Advantages**

* Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.
* A decision tree does not require normalization and scaling of data.
* Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.
* The advantages of proposed model are High performance and accuracy rate.

3. FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

# Three key considerations involved in the feasibility analysis are

# ECONOMICAL FEASIBILITY

# TECHNICAL FEASIBILITY

# SOCIAL FEASIBILITY

# 3.1. ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available.

# 3.2. TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system. This application in going to be used in an internet environment called WWW (World Wide Web). So, it is necessary to use a technology that is capable of providing the networking facility to the application. One major advantage in application is platform neutral. We can deploy and used it in any operating system.

GUI is developed using HTML to capture the information from the customer. HTML is used to display the content on the browser. It uses TCP/IP protocol. It is an interpreted language.

**3.3. SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**4. SYSTEM REQUIREMENTS**

**4.1. SOFTWARE REQUIREMENTS**

Operating system : Windows 10.

Coding Language : Python.

Web Framework : Flask

**4.2. HARDWARE REQUIREMENTS**

Processor : Intel Core i3 or above

RAM : 4 GB or above

Hard Disk : 500 GB or above

Input Devices : Keyboard, Mouse

**5. SYSTEM DESIGN**

5.1. UML DIAGRAMS INTRODUCTION

The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntax, semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective.

UML is specifically constructed through two different domains they are:

* UML Analysis modeling, this focuses on the user model and structural model views of the system.
* UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

**5.2. SYSTEM DESIGN ASPECTS**

Once the analysis stage is completed, the next stage is to determine in broad outline form how the problem might be solved. During system design, we are beginning to move from the logical to physical level.

System design involves architectural and detailed design of the system. Architectural design involves identifying software components, decomposing them into processing modules and conceptual data structures, and specifying the interconnections among components.

Detailed design is concerned with how to package processing modules and how to implement the processing algorithms, data structures and interconnections of standard algorithms, invention of new algorithms, and design of data representations and packaging of software products. Two kinds of approaches are available:

* Top down approach
* Bottom up approach

5.2.1. Design of Code

Since information systems projects are designed with space, time and cost saving in mind, coding methods in which conditions, words, ideas or control errors and speed the entire process. The purpose of the code is to facilitate the identification and retrieval of the information. A code is an ordered collection of symbols designed to provide unique identification of an entity or an attribute.

**5.2.2. Design of Input**

Design of input involves the following decisions

* Input data
* Input medium
* The way data should be arranged or coded
* Validation needed to detect every step to follow when error occurs

The input controls provide ways to ensure that only authorized users access the system guarantee the valid transactions, validate the data for accuracy and determine whether any necessary data has been omitted. The primary input medium chosen is display. Screens have been developed for input of data using HTML. The validations for all important inputs are taken care of through various events using JSP control.

**5.2.3. Design of Output**

Design of output involves the following decisions

* Information to present
* Output medium
* Output layout

Output of this system is given in easily understandable, user-friendly manner.

**5.2.4 Design of Control**

The system should offer the means of detecting and handling errors.

Input controls provides ways per

* Valid transactions are only acceptable
* Validates the accuracy of data
* Ensures that all mandatory data have been captured

**5.3. UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

In this project, the following UML diagrams have been explained among the following list:

* Class Diagram
* Use Case Diagram
* Sequence Diagram
* Activity Diagram
* Data Flow Diagram

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices

**5.3.1. Class Diagram**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

Fig.5.1: Class Diagram

**5.3.2. Use Case Diagram**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

User

Input data

Preprocessing

Training

Classification

Fig.5.2: Use Case Diagram

**5.3.3. Sequence Diagram**

A Sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called Event-trace diagrams, event scenarios, and timing diagrams.

Data collection

Training

Testing

Collect the data from the user feature on cnn

Send the data to the training stage

Perform Preprocessing

Train the data

Extracted feature with images sending to the testing stage

Give input

Predict the type using proposed algorithm

Fig.5.3: Sequence Diagram

**5.3.4. Activity Diagram**

Activity diagrams are a loosely defined diagram technique for showing workflows of stepwise activities and actions, with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

Input Data

Preprocessing

Training

Prediction using proposed algorithm (decision tree classifier)

Predicted Label As positive or negative

Fig.5.4: Activity Diagram

**5.3.5. DATA FLOW DIAGRAM**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

Fig.5.5: Data Flow Diagram

6. OVERVIEW OF TECHNOLOGIES

## 6.1. PYTHON

## Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. The biggest strength of Python is huge collection of standard library which can be used for the following –

## Machine Learning

## GUI Applications (like Kivy, Tkinter, PyQt etc. )

## Web frameworks like Django (used by YouTube, Instagram, Dropbox)

## Image processing (like Opencv, Pillow)

## Web scraping (like Scrapy, BeautifulSoup, Selenium)

## Test frameworks

## Multimedia

**6.1.1. FEATURES OF PYTHON**

Python's features include −

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
* **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases** − Python provides interfaces to all major commercial databases.
* **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows, Macintosh, and the X Window system of Unix.
* **Scalable** − Python provides a better structure and support for large programs than shell scripting.

## 6.1.2. HISTORY OF PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL). Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

**6.2. HTML**

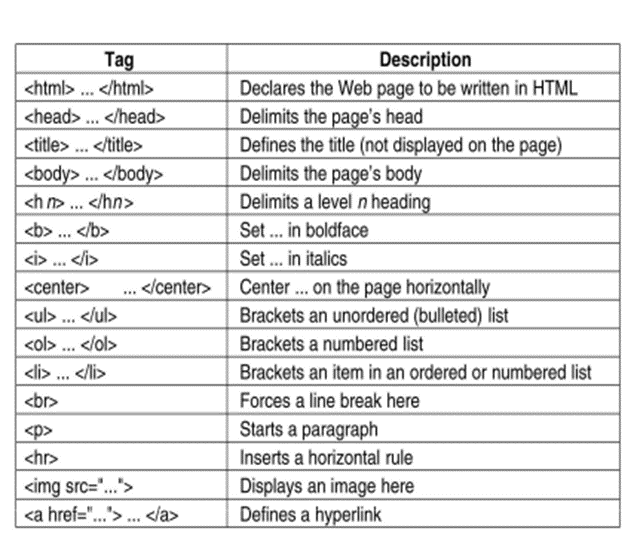
HTML stands for Hyper Text Markup Language, which is the most widely used language on Web to develop web pages. Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers. Now, HTML is being widely used to format web pages with the help of different tags available in HTML language. HTML is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. some of the key advantages of learning HTML are:

* Create Web site - You can create a website or customize an existing web template if you know HTML well.
* Become a web designer - If you want to start a carrer as a professional web designer, HTML and CSS designing is a must skill.
* Understand web - If you want to optimize your website, to boost its speed and performance, it is good to know HTML to yield best results.
* Learn other languages - Once you understands the basic of HTML then other related technologies like javascript, php, or angular are become easier to understand.

**6.2.1. HTML Tags**

HTML tags are like keywords which defines that how web browser will format and display the content. With the help of tags, a web browser can distinguish between an HTML content and a simple content. HTML tags contain three main parts: opening tag, content and closing tag. But some HTML tags are unclosed tags.

1. All HTML tags must enclosed within < > these brackets.

**Table 6.1. HTML Tags**

.

**6.3. CSS**

CSS stands for Cascading Style Sheets. It describes how HTML elements are to be displayed on screen, paper, or in other media and saves a lot of work. It can control the layout of multiple web pages all at once.

**6.3.1. Types of CSS**

Diagram

Description automatically generated

Fig. 6.1. Types of CSS

**Inline CSS**

Inline CSS allows you to apply style rules to specific HTML elements. Inlining CSS means putting CSS into an HTML file instead of an external CSS. Since inline CSS allows the application of a unique style to one HTML element, its usage is limited but is beneficial for creating unique attributes.

**Internal CSS**

Internal or embedded CSS requires you to add <style> tag in the <head> section of your HTML document. This CSS style is an effective method of styling a single page. However, using this style for multiple pages is time-consuming as you need to put CSS rules on every page of your website.

**External CSS**

With external CSS, you’ll link your web pages to an external **.css** file, which can be created by any text editor in your device (e.g., [Notepad++](https://notepad-plus-plus.org/)).]This CSS type is a more efficient method, especially for styling a large website. By editing one **.css** file, you can change your entire site at once.

**6.4. FLASK WEB FRAMEWORK**

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, developed it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods –

**Table 6.2. HTTP Methods**

|  |  |
| --- | --- |
| **Sl.No.** | **Methods & Description** |
| 1 | **GET**  Sends data in unencrypted form to the server. Most common method. |
| 2 | **HEAD**  Same as GET, but without response body |
| 3 | **POST**  Used to send HTML form data to server. Data received by POST method is not cached by server. |
| 4 | **PUT**  Replaces all current representations of the target resource with the uploaded content. |
| 5 | **DELETE**  Removes all current representations of the target resource given by a URL |

**6.5. MACHINE LEARNING**

Machine Learning is a system of computer algorithms that can learn from example through self-improvement without being explicitly coded by a programmer. Machine learning is a part of artificial Intelligence which combines data with statistical tools to predict an output which can be used to make actionable insights.

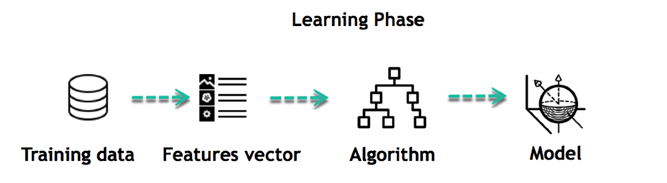
The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input and uses an algorithm to formulate answers.

A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

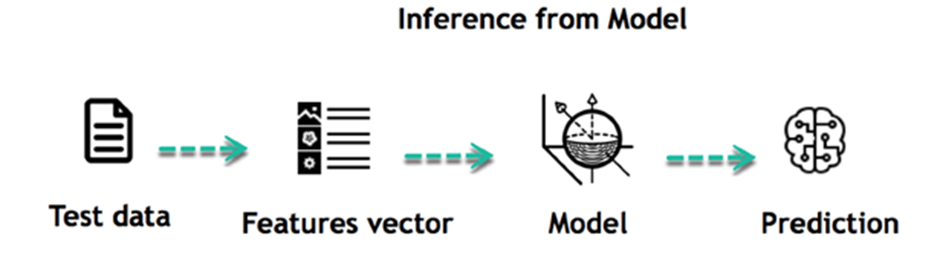
Machine learning is also used for a variety of tasks like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

**6.5.1. How does Machine Learning Work?**

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it’s feed a previously unseen example, the machine has difficulties to predict. The core objective of machine learning is the learning and inference. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a feature vector. You can think of a feature vector as a subset of data that is used to tackle a problem. Therefore, the learning stage is used to describe the data and summarize it into a model.



**Fig 6.2. Learning Phase of Model**

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

**Fig 6.3. Prediction Phase of Model**

**6.5.2. TYPES OF MACHINE LEARNING**

**Supervised learning**

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans.

You can use supervised learning when the output data is known. The algorithm will predict new data.

**Unsupervised learning**

In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns)

You can use it when you do not know how to classify the data, and you want the algorithm to find patterns and classify the data for you.

**Reinforcement learning.**

A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). As it navigates its problem space, the program is provided feedback that's analogous to rewards, which it tries to maximize.

**6.5.3. MACHINE LEARNING MODELS**

**Artificial Neural Networks**

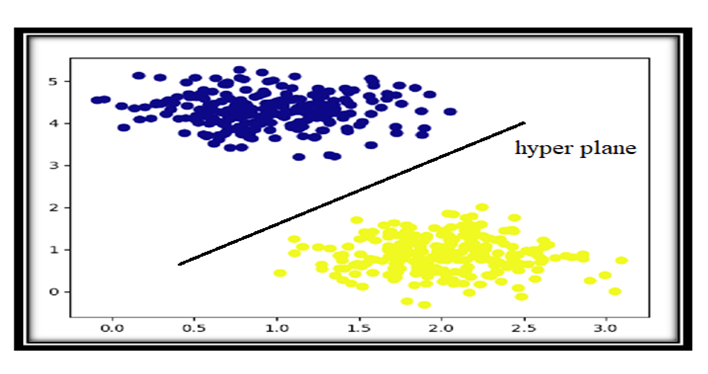
An artificial neural network is an interconnected group of nodes, akin to the vast network of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one artificial neuron to the input of another.

An ANN is a model based on a collection of connected units or nodes called "artificial neurons", which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit information, a "signal", from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called "edges". Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the layers multiple times.

**Decision Trees**

Decision tree learning uses a decision tree as a predictive model to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modeling approaches used in statistics, data mining, and machine learning. Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data, but the resulting classification tree can be an input for decision making.

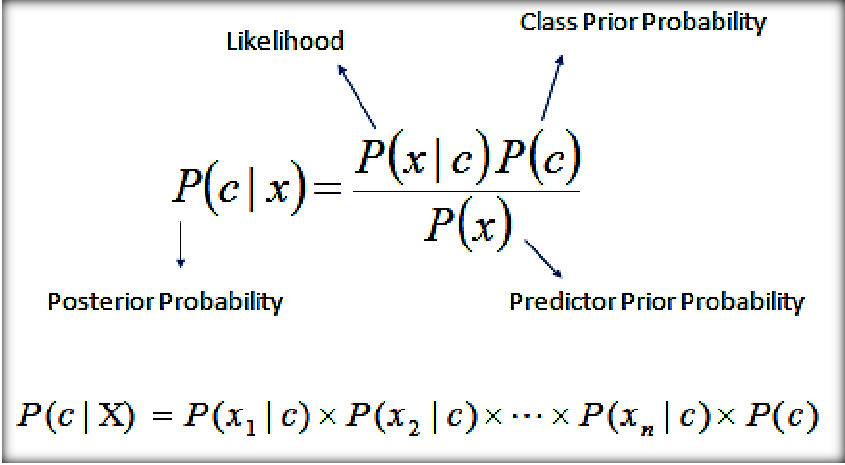
**Support Vector Machine**

Support vector machine (SVM), also known as support vector network, is a set of related supervised learning method used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. An SVM training algorithm is a non-probabilistic, binary, linear classifier, although methods such as Platt scaling exist to use SVM in a probabilistic classification setting. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

**Fig. 6.4. Support Vector Machine**

## Naïve Bayes Algorithm

Naive Bayes is a classification method which is based on Bayes’ theorem. This assumes independence between predictors. A Naive Bayes classifier will assume that a feature in a class is unrelated to any other. Consider a fruit. This is an apple if it is round, red, and 2.5 inches in diameter. A Naive Bayes classifier will say these characteristics independently contribute to the probability of the fruit being an apple. This is even if features depend on each other. For very large data sets, it is easy to build a Naive Bayesian model. Not only is this model very simple, it performs better than many highly sophisticated classification methods. Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. [Maximum-likelihood](https://en.wikipedia.org/wiki/Maximum-likelihood_estimation) training can be done by evaluating a [closed-form expression,](https://en.wikipedia.org/wiki/Closed-form_expression) which takes [linear time,](https://en.wikipedia.org/wiki/Linear_time) rather than by expensive [iterative](https://en.wikipedia.org/wiki/Iterative_method) [approximation](https://en.wikipedia.org/wiki/Iterative_method) as used for many other types of classifiers.



**Fig. 6.5. Naïve Bayes Classification**

**7. IMPLEMENTATION**

In the implementation phase software development is concerned with translating design specifications into source code. The primary goal of implementation is to write the source code internal documentation so that conformance of the code to its specification can be easily verified, and so that debugging, testing and modifications are erased. This goal is achieved by making the source code as clear and straightforward as possible. Simplicity, clarity and elegance are the hallmarks of good programs. Obscurity, cleverness and complexity are indications of inadequate design and misdirected thinking.

Source code clarity is enhanced by strutted techniques, by good coding style, by appropriate documents, by go internal comments, and by the features provided in the modern programming languages.

**Data Preparation:**

Wrangle data and prepare it for training. Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.)

Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data

Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis

Split into training and evaluation sets

**Model Selection:**

We used Decision Tree Classifier machine learning algorithm, We got a accuracy of 96.7% on test set so we implemented this algorithm.

**Decision Tree Classification Algorithm:**

Decision Tree is a Supervised learning techniquethat can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, whereinternal 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.

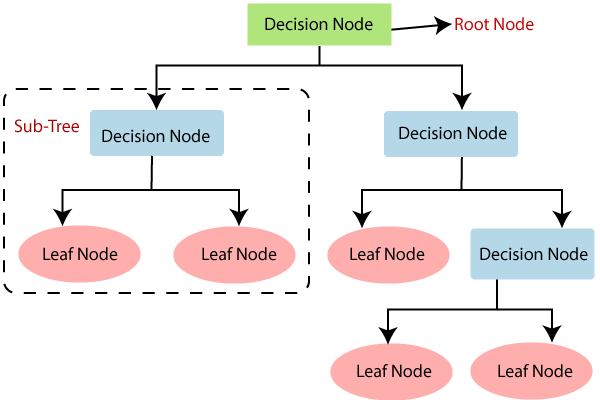
It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

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.

Below diagram explains the general structure of a decision tree:



**Fig 7. Structure of Decision Tree**

**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.

**Optimizing Decision Tree**

A few things should be considered when improving the accuracy of the decision tree classifier. The following are some possible optimizations to consider when looking to make sure the decision tree model produced makes the correct decision or classification.

**Increasing the number of levels of the tree**

The accuracy of the decision tree can change based on the depth of the decision tree. In many cases, the tree’s leaves are pure nodes. When a node is pure it means that all the data in that node belongs to a single class. Occasionally, going deeper in the tree can cause an accuracy decrease in general, so it is very important to test modifying the depth of the decision tree and selecting the depth that produces the best results.

**The choice of node splitting functions**

The node splitting function used can have an impact on improving the accuracy of the decision tree. For example, using the Information gain function may yield better results than using the phi function. The phi function is known as a measure of “goodness” of a candidate split at a node in the decision tree. The information gain function is known as measure of the “reduction in entropy”.

The Decision Tree model is trained using the dataset after pre-processing the data. From sklearn library the DecisionTreeClassifier is imported and given the dataset file to train. After successful training the model is given input data for prediction purpose.

After implementing the Decision Tree Classifier, the accuracy of the prediction has been highest and most accurate results.

Our predicting system has provided better results than other existing systems. Here, we have used Decision Tree (ID3 Algorithm) for predicting output values. ID3 is algorithm which is used to build decision trees. ID3 has some features like removing outliers, handling missing values.

**Accuracy on test set:**

We got an accuracy of 96.7% on test set.

**Saving the Trained Model:**

Once you’re confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a h5 or .pkl file using a library like pickle.

Make sure you have pickle installed in your environment.

Next, import the module and dump the model into .pkl file

Now, run the python file using command **python app.py** in the command prompt at the location where the file is location.

**8. SOURCE CODE**

**app.py**

import numpy as np

import pandas as pd

from flask import Flask, request, jsonify, render\_template, redirect, flash, send\_file

from sklearn.preprocessing import MinMaxScaler

from werkzeug.utils import secure\_filename

import pickle

import numpy as np

import pandas as pd

from flask import Flask, request, jsonify, render\_template, redirect, flash, send\_file

from sklearn.preprocessing import MinMaxScaler

from werkzeug.utils import secure\_filename

import pickle

import numpy as np

import pandas as pd

from sklearn.tree import DecisionTreeClassifier

from sklearn.svm import SVC

app = Flask(\_\_name\_\_) #Initialize the flask App

heart = pickle.load(open('heartd.pkl','rb'))

@app.route('/')

@app.route('/index')

def index():

    return render\_template('index.html')

@app.route('/chart')

def chart():

    return render\_template('chart.html')

@app.route('/login')

def login():

    return render\_template('login.html')

@app.route('/upload')

def upload():

    return render\_template('upload.html')

@app.route('/preview',methods=["POST"])

def preview():

    if request.method == 'POST':

        dataset = request.files['datasetfile']

        df = pd.read\_csv(dataset,encoding = 'unicode\_escape')

        df.set\_index('Id', inplace=True)

        return render\_template("preview.html",df\_view = df)

@app.route('/prediction', methods = ['GET', 'POST'])

def prediction():

    return render\_template('prediction.html')

@app.route('/predict',methods=['POST'])

def predict():

    int\_feature = [x for x in request.form.values()]

    final\_features = [np.array(int\_feature)]

    result=heart.predict(final\_features)

    if result == 1:

        result = "Positive"

    else:

        result = 'Negative'

    return render\_template('prediction.html', prediction\_text= result)

@app.route('/performance')

def performance():

    return render\_template('performance.html')

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(debug=True)

**Jupyter Source File (heart.ipynb)**

#importing libraries

import pandas as pd

import matplotlib.pyplot as plt

from functools import reduce

#reading dataset

df = pd.read\_csv('hearts.csv')

print(df)

#split into X and Y datasets

X = df.drop(['target','ca','thal','Id'], axis = 1)

y = df['target']

print(X)

print(y)

#splitting into training and testing sets

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=9)

print(X\_train.shape)

print(X\_test.shape)

**Decision Tree Classifier**

from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier(random\_state=0)

clf.fit(X\_train, y\_train)

from sklearn.metrics import confusion\_matrix, accuracy\_score

y\_pred = clf.predict(X\_test)

cm = confusion\_matrix(y\_test,y\_pred)

print(cm)

dt = accuracy\_score(y\_test,y\_pred)

print(dt)

#heatmap with confusion matrix

import seaborn as sns

import matplotlib.pyplot as plt

f, ax=plt.subplots(figsize=(5,5))

sns.heatmap(cm,annot=True,linewidths=0.5,linecolor="red",fmt=".0f",ax=ax)

plt.xlabel("y\_pred")

plt.ylabel("y\_true")

plt.show()

**Support Vector Machine**

from sklearn.svm import SVC

clf2 = SVC(kernel='linear' , random\_state = 0)

clf2.fit(X\_train,y\_train)

y\_svm = clf2.predict(X\_test)

from sklearn.metrics import confusion\_matrix,accuracy\_score

cm = confusion\_matrix(y\_test,y\_svm)

svm = accuracy\_score(y\_test,y\_svm)

print(cm)

print(svm)

**Naïve Bayes Classification**

from sklearn.naive\_bayes import GaussianNB

clf3 = GaussianNB()

clf3.fit(X\_train,y\_train)

y\_nb = clf3.predict(X\_test)

from sklearn.metrics import confusion\_matrix,accuracy\_score

cm = confusion\_matrix(y\_test,y\_nb)

nb = accuracy\_score(y\_test,y\_nb)

print(cm)

print(nb)

**# Comparing Accuracies of ML Algorithms**

sc=[]

sc.append(dt)

sc.append(svm)

sc.append(nb)

print(sc)

y1=["Decision Tree","SVM","Naive Bayes"]

y1

df1=pd.DataFrame()

df1['Algorithm'] = y1

df1['Accuracies'] = sc

print(df1)

**# Visualising Accuracies**

#barplot

import seaborn as sns

sns.barplot(x='Algorithm',y='Accuracies',data=df1)

#lineplot

sns.lineplot(x='Algorithm',y='Accuracies',data=df1)

**# Generating Decision Tree**

pip install graphviz

pip install pydotplus

pip install six

pip install mlrose

conda install graphviz

conda install -c conda-forge pydotplus

import sklearn.externals

import six

import sys

sys.modules['sklearn.externals.six'] = six

import mlrose

from sklearn.tree import export\_graphviz

from sklearn.externals.six import StringIO

from IPython.display import Image

import pydotplus

feature\_cols = ['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang','oldpeak','slope']

dot\_data = StringIO()

export\_graphviz(clf, out\_file=dot\_data,filled=True,rounded=True,special\_characters=True,feature\_names = feature\_cols,class\_names=['0','1'])

graph = pydotplus.graph\_from\_dot\_data(dot\_data.getvalue())

graph.write\_png('tree.png')

Image(graph.create\_png())

**index.html**

<!doctype html>

<html class="no-js" lang="en">

<head>

    <meta charset="utf-8">

    <meta http-equiv="x-ua-compatible" content="ie=edge">

    <title>heart</title>

    <meta name="description" content="">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- <link rel="manifest" href="site.webmanifest"> -->

    <link rel="shortcut icon" type="image/x-icon" href="img/favicon.png">

    <!-- Place favicon.ico in the root directory -->

    <!-- CSS here -->

    <link rel="stylesheet" href="../static/css/bootstrap.min.css">

    <link rel="stylesheet" href="../static/css/style.css">

</head>

<body>

    <!-- header-start -->

    <header>

        <div class="header-area ">

            <div id="sticky-header" class="main-header-area">

                <div class="container">

                    <div class="row align-items-center">

                        <div class="col-xl-3 col-lg-3">

                            <div class="logo-img">

                                <a href="index.html">

                                    <img src="img/logo.png" alt="">

                                </a>

                            </div>

                        </div>

                        <div class="col-xl-9 col-lg-9">

                            <div class="menu\_wrap d-none d-lg-block">

                                <div class="menu\_wrap\_inner d-flex align-items-center justify-content-end">

                                    <div class="main-menu">

                                        <nav>

                                            <ul id="navigation">

                                                <li><a href="{{url\_for('index')}}">Home</a></li>

                                                <li><a href="{{url\_for('login')}}">login</a></li>

                                            </ul>

                                        </nav>

                                    </div>

                                </div>

                            </div>

                        </div>

                        <div class="col-12">

                      <div class="mobile\_menu d-block d-lg-none"></div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </header>

    <!-- header-end -->

    <!-- slider\_area\_start -->

    <div class="slider\_area">

        <div class="slider\_active owl-carousel">

            <div class="single\_slider  d-flex align-items-center slider\_bg\_1 overlay">

                <div class="container">

                    <div class="row">

                        <div class="col-xl-12">

                            <div class="slider\_text ">

<h3> <span>HEART DISEASE<br> PREDICTION   </span>USING <br>

                              <span>MACHINE LEARNING</span></h3>

                                </a>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <!-- slider\_area\_end -->

    <!-- welcome\_clicnic\_area\_start -->

    <footer class="footer">

        <div class="copy-right\_text">

            <div class="container">

                <div class="row">

                    <div class="bordered\_1px "></div>

                    <div class="col-xl-12">

                    </div>

                </div>

            </div>

        </div>

    </footer>

    <!-- JS here -->

    <script src="../static/js/vendor/modernizr-3.5.0.min.js"></script>

    <script src="../static/js/vendor/jquery-1.12.4.min.js"></script>

    <script src="../static/js/gijgo.min.js"></script>

    <!--contact js-->

    <script src="../static/js/contact.js"></script>

    <script src="../static/js/jquery.form.js"></script>

    <script src="../static/js/jquery.validate.min.js"></script>

    <script src="../static/js/mail-script.js"></script>

    <script src="../static/js/main.js"></script>

    <script>

        $('.datepicker').datepicker({

            iconsLibrary: 'fontawesome',

            icons: {

                rightIcon: '<span class="fa fa-calendar"></span>'

            }

        });

        $('.timepicker').timepicker({

            iconsLibrary: 'fontawesome',

            icons: {

                rightIcon: '<span class="fa fa-clock-o"></span>'

            }

        });

    $(document).ready(function() {

    $('.js-example-basic-multiple').select2();

});

    </script>

</body>

</html>

**upload.html**

<!doctype html>

<html class="no-js" lang="zxx">

<head>

    <meta charset="utf-8">

    <meta http-equiv="x-ua-compatible" content="ie=edge">

    <title>Docmed</title>

    <meta name="description" content="">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="shortcut icon" type="image/x-icon" href="img/favicon.png">

    <!-- Place favicon.ico in the root directory -->

    <!-- CSS here -->

    <link rel="stylesheet" href="../static/css/bootstrap.min.css">

    <link rel="stylesheet" href="../static/css/style.css">

</head>

<body>

    <!-- header-start -->

    <header>

        <div class="header-area ">

            <div id="sticky-header" class="main-header-area">

                <div class="container">

                    <div class="row align-items-center">

                        <div class="col-xl-3 col-lg-3">

                            <div class="logo-img">

                                <a href="index.html">

                                    <img src="img/logo.png" alt="">

                                </a>

                            </div>

                        </div>

                        <div class="col-xl-9 col-lg-9">

                            <div class="menu\_wrap d-none d-lg-block">

                                <div class="menu\_wrap\_inner d-flex align-items-center justify-content-end">

                                    <div class="main-menu">

                                        <nav>

                                            <ul id="navigation">

                                                <li><a href="{{url\_for('index')}}">Home</a></li>

                                                <li><a href="{{url\_for('login')}}">Login</a></li>

                                                <li><a href="{{url\_for('upload')}}">Upload</a></li>

                                            </ul>

                                        </nav>

                                    </div>

                                </div>

                            </div>

                        </div>

                        <div class="col-12">

                            <div class="mobile\_menu d-block d-lg-none"></div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </header>

    <!-- header-end -->

    <!-- bradcam\_area\_start  -->

    <div class="bradcam\_area breadcam\_bg\_1">

        <div class="container">

            <div class="row">

                <div class="col-xl-12">

                    <div class="bradcam\_text">

                        <center><h3>Upload the dataset for Training</h3></center>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <!-- bradcam\_area\_end  -->

    <!-- ================ contact section start ================= -->

    </br>

    </br>

    </br>

    <section>

    <body id="page-top">

 <!-- About Section -->

  <section class="col-md-12 col-sm-12 col-xs-12" id="about">

      <!-- About Section Heading -->

      <!-- About Section Content -->

    <br>

            <br>

            <br>

         <div class="col-md-12 col-sm-12 col-xs-12" style="justify-content: center; margin-left: 40%;">

          <form action="http://localhost:5000/preview" name="fs" id="fs" method="post" enctype=multipart/form-data>

                       <br/>

          <input type="file" name="datasetfile" id="file1" required />

                       <br/><br/>

        <input type="submit" style="margin-right:200px" class="btn btn-primary btn-large" value="Upload">

                      </form>

      </div>

  </section>

</body>

     </div>

        </section>

    <!-- ================ contact section end ================= -->

    </br>

    </br>

        <script>

            $('#datepicker').datepicker({

                iconsLibrary: 'fontawesome',

                icons: {

                 rightIcon: '<span class="fa fa-caret-down"></span>'

             }

            });

            $('#datepicker2').datepicker({

                iconsLibrary: 'fontawesome',

                icons: {

                 rightIcon: '<span class="fa fa-caret-down"></span>'

             }

            });

        </script>

    </body>

    </html>

**Preview.html**

<!doctype html>

<html class="no-js" lang="zxx">

<head>

    <meta charset="utf-8">

    <meta http-equiv="x-ua-compatible" content="ie=edge">

    <title>Training</title>

    <meta name="description" content="">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- <link rel="manifest" href="site.webmanifest"> -->

    <link rel="shortcut icon" type="image/x-icon" href="img/favicon.png">

    <!-- CSS here -->

      <link rel="stylesheet" href="../static/css/bootstrap.min.css">

    <link rel="stylesheet" href="../static/css/style.css">

</head>

<body>

    <!-- header-start -->

    <header>

        <div class="header-area ">

            <div id="sticky-header" class="main-header-area">

                <div class="container">

                    <div class="row align-items-center">

                        <div class="col-xl-3 col-lg-3">

                            <div class="logo-img">

                                <a href="index.html">

                                    <img src="img/logo.png" alt="">

                                </a>

                            </div>

                        </div>

                        <div class="col-xl-9 col-lg-9">

                            <div class="menu\_wrap d-none d-lg-block">

                                <div class="menu\_wrap\_inner d-flex align-items-center justify-content-end">

                                    <div class="main-menu">

                                        <nav>

                                            <ul id="navigation">

                                                <li><a href="{{url\_for('index')}}">Home</a></li>

                                                <li><a href="{{url\_for('login')}}">Login</a></li>

                                                <li><a href="{{url\_for('upload')}}">Upload</a></li>

                                            </ul>

                                        </nav>

                                    </div>

                                </div>

                            </div>

                        </div>

                        <div class="col-12">

                            <div class="mobile\_menu d-block d-lg-none"></div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </header>

    <!-- header-end -->

    <!-- bradcam\_area\_start  -->

    <div class="bradcam\_area breadcam\_bg\_1">

        <div class="container">

            <div class="row">

                <div class="col-xl-12">

                    <div class="bradcam\_text">

                        <h3>Train the model</h3>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <!-- bradcam\_area\_end  -->

    <!-- ================ contact section start ================= -->

    </br>

    </br>

    <section>

                <div class="row">

                    <div class="col-12">

                         <center><h2>Dataset preview</h2></center>

                    </div>

 <body id="page-top">

  <!-- Navigation -->

  <!-- Contact Section -->

  <section class="page-section" id="contact">

    <div class="container">

     <br>

     <br>

      <!-- Contact Section Heading -->

      <h2 class="text-center text-uppercase text-secondary mb-0"> </h2>

      <!-- Icon Divider -->

      <div class="divider-custom">

        <div class="divider-custom-line"></div>

        <div class="divider-custom-icon">

        </div>

        <div class="divider-custom-line"></div>

      </div>

      <!-- Contact Section Form -->

      <div class="row">

        <div class="col-lg-8 mx-auto">

    {{ df\_view.tail().to\_html(classes="table striped",na\_rep="-") | safe}}

        </div>

      </div>

  </section>

 <div class="form-group" style="padding:0px 250px 10px 40px;height:200px">

                    <input style="margin-left:500px" type="button" onclick="hideLoader()" class="btn btn-primary" value="Click to Train" />

                    <div id="loading" style="display:None;margin-top:552950px"></div>

                    </div>

  <!-- Copyright Section -->

  <section class="copyright py-4 text-center text-white">

    <div class="container">

    </div>

  </section>

  <!-- Scroll to Top Button (Only visible on small and extra-small screen sizes) -->

  <div class="scroll-to-top d-lg-none position-fixed ">

    <a class="js-scroll-trigger d-block text-center text-white rounded" href="#page-top">

      <i class="fa fa-chevron-up"></i>

    </a>

  </div>

  <!-- Bootstrap core JavaScript -->

  <script type='text/javascript' src='https://ajax.googleapis.com/ajax/libs/jquery/2.2.4/jquery.min.js'></script>

   <script type='text/javascript'>

function hideLoader() {

$('#loading').show(0).delay(1000).hide(0,function(){

     alert("Training finished!");

     window.location = "{{url\_for('prediction')}}";

   });

}

   </script>

</body>

    </div>

    </section>

    <!-- ================ contact section end ================= -->

    </br></br>

    </br></br>

    <footer class="footer">

        <div class="footer\_top">

            <div class="container">

                <div class="row">

                    <div class="col-xl-4 col-md-6 col-lg-4 ">

                        <div class="footer\_widget">

                        </div>

                    </div>

                    <div class="col-xl-4 col-md-6 col-lg-4">

                    </div>

                </div>

            </div>

        </div>

        <div class="copy-right\_text">

            <div class="container">

                <div class="row">

                    <div class="bordered\_1px "></div>

                    <div class="col-xl-12">

                    </div>

                </div>

            </div>

        </div>

    </footer>

        <script>

            $('#datepicker').datepicker({

                iconsLibrary: 'fontawesome',

                icons: {

                 rightIcon: '<span class="fa fa-caret-down"></span>'

             }

            });

            $('#datepicker2').datepicker({

                iconsLibrary: 'fontawesome',

                icons: {

                 rightIcon: '<span class="fa fa-caret-down"></span>'

             }

            });

        </script>

    </body>

</html>

**Prediction.html**

<!doctype html>

<html class="no-js" lang="zxx">

<head>

    <meta charset="utf-8">

    <meta http-equiv="x-ua-compatible" content="ie=edge">

    <title>Prediction</title>

    <meta name="description" content="">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- <link rel="manifest" href="site.webmanifest"> -->

    <link rel="shortcut icon" type="image/x-icon" href="img/favicon.png">

    <!-- Place favicon.ico in the root directory -->

    <!-- CSS here -->

    <link rel="stylesheet" href="../static/css/bootstrap.min.css">

    <link rel="stylesheet" href="../static/css/owl.carousel.min.css">

    <link rel="stylesheet" href="../static/css/style.css">

    <style>

            h1 {

                color:#f44336;

            }

            b{

            color: #2196F3;

            }

    </style>

</head>

<body>

    <!-- header-start -->

    <header>

        <div class="header-area ">

            <div id="sticky-header" class="main-header-area">

                <div class="container">

                    <div class="row align-items-center">

                        <div class="col-xl-3 col-lg-3">

                            <div class="logo-img">

                                <a href="index.html">

                                    <img src="img/logo.png" alt="">

                                </a>

                            </div>

                        </div>

                        <div class="col-xl-9 col-lg-9">

                            <div class="menu\_wrap d-none d-lg-block">

                                <div class="menu\_wrap\_inner d-flex align-items-center justify-content-end">

                                    <div class="main-menu">

                                        <nav>

                                            <ul id="navigation">

                                                  <li><a href="{{url\_for('index')}}">Home</a></li>

                                                <li><a href="{{url\_for('login')}}">Login</a></li>

                                                <li><a href="{{url\_for('upload')}}">Upload</a></li>

                                                        <li><a href="{{url\_for('prediction')}}">Prediction</a></li>

                                            </ul>

                                        </nav>

                                    </div>

                                </div>

                            </div>

                        </div>

                        <div class="col-12">

                            <div class="mobile\_menu d-block d-lg-none"></div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </header>

    <!-- header-end -->

    <!-- bradcam\_area\_start  -->

    <div class="bradcam\_area breadcam\_bg\_1">

        <div class="container">

            <div class="row">

                <div class="col-xl-12">

                    <div class="bradcam\_text">

                        <h3>Prediction </h3>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <!-- bradcam\_area\_end  -->

    <!-- ================ contact section start ================= -->

    <section>   </br>

    </br>

    </br>

                <div class="row">

                    <div class="col-12">

                        <center><b>Heart Disease prediction</b></center>

                    </div>

         <div class="col-md-12 col-sm-12 col-xs-12" style="margin-left:50px;">

                         <form action="{{ url\_for('predict')}}"method="post">

         <div class="span5">

                <div class="control-group">

                    <center><label class="control-label" for="inputMessage"><b></b></label></br></center>

                    </br>

                     <div class="controls">

               <center> <b> Age: </b> <input type="text" name="Age" placeholder="Age" required="required" width="300" style="width: 300px"/></center>

                   </div>

                   </br>

                     <div class="controls">

                   <center><b>sex:</b>    <select   type="text"  name="sex" required="required" placeholder="sex" width="300" style="width: 300px"></center>

                            <option value="0">Female</option>

                        <option value="1">Male</option>

                    </select>

                   </div>

                       </br>

                    <div class="controls">

                   <center><b> chest pain type:</b>    <select   type="text"  name="chest\_pain\_type" required="required" placeholder="chest\_pain\_type" width="300" style="width: 300px"></center>

                            <option value="0">Typical angina</option>

                           <option value="1">Atypical angina</option>

                            <option value="2"> Non-anginal pain</option>

                            <option value="3"> Asymptomatic</option>

                    </select> &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

                   </div>

                       </br>

 <div class="controls">

                 <center>  <b>  Resting blood pressure:</b><input type="text" name=" resting\_blood\_pressure" placeholder="Above 100" required="required"  width="300" style="width: 300px"/> &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</center>

                   </div>

                       </br>

 <div class="controls">

               <center>   <b> Serum cholestoral in mg/dL: </b><input type="text" name="serum\_cholestoral" placeholder="above 200"  width="300" style="width: 300px" required="required" />   &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; &nbsp;</center>

                   </div>

                       </br>

  <div class="controls">

                   <center><b>Fasting blood sugar :</b>    <select   type="text"  name="fasting\_blood\_sugar " required="required" placeholder="fasting blood sugar > 120 mg/dl" width="300" style="width: 300px"> &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</center>

                            <option value="0">False</option>

                        <option value="1">True</option>

                    </select> &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; </center>

                   </div>

                       </br>

  <div class="controls">

                   <center><b>Resting electrocardiographic :</b>    <select   type="text"  name="resting\_electrocardiographic " required="required" placeholder="resting electrocardiographic" width="300" style="width: 300px"></center>

                            <option value="0">Nothing to note</option>

                        <option value="1">ST-T Wave abnormality</option>

                      <option value="2">Possible or definite left ventricular hypertrophy</option>

                    </select>  &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</center>

                   </div>

                       </br>

<div class="controls">

               <center>   <b> Mximum heart rate: </b><input type="text" name="Mximum\_heart\_rate" placeholder="Mximum heart rate"  width="300" style="width: 300px" required="required" /> &nbsp;&nbsp;&nbsp; &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</center>

                   </div>

                       </br>

             <div class="controls">

                   <center><b>Exercise induced angina :</b>    <select   type="text"  name="exercise\_induced\_angina " required="required" placeholder="exercise\_induced\_angina > 120 mg/dl" width="300" style="width: 300px"></center>

                            <option value="0">No</option>

                        <option value="1">Yes</option>

                    </select>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</center>

                   </div>

                       </br>

                       <div class="controls">

               <center>   <b> oldpeak: </b><input type="text" name="oldpeak " placeholder="oldpeak "  width="300" style="width: 300px" required="required" />  </center>

                   </div>

                       </br>

                        <div class="controls">

                   <center><b>slope :</b>    <select   type="text"  name="slope " required="required" placeholder="slope" width="300" style="width: 300px"></center>

                            <option value="0">Upsloping</option>

                        <option value="1">Flatsloping</option>

                      <option value="2">Downslopins</option>

                    </select>  </center>

                   </div>

                       </br>

   <center>  <button type="submit" class="btn btn-primary">Predict </button></center>

    </form></br>

    </br>

         <center><h1> Prediction is : {{ prediction\_text }} </h1></center>

                </div>

                </div>

        </section>

    <!-- ================ contact section end ================= -->

    </br>   </br>   </br>   </br>

        <script>

            $('#datepicker').datepicker({

                iconsLibrary: 'fontawesome',

                icons: {

                 rightIcon: '<span class="fa fa-caret-down"></span>'

             }

            });

            $('#datepicker2').datepicker({

                iconsLibrary: 'fontawesome',

                icons: {

                 rightIcon: '<span class="fa fa-caret-down"></span>'

             }

            });

        </script>

    </body>

</html>

**9. TESTING**

**9.1. SOFTWARE TESTING TECHNIQUES**

Software Testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding, Testing presents an interesting anomaly for the software engineer.

**9.1.1. Testing Objectives**

1. Testing is a process of executing a program with the intent of finding an error.
2. A good test case is one that has a probability of finding an as yet
3. undiscovered error.
4. A successful test is one that uncovers an undiscovered error.
5. These above objectives imply a dramatic change in view port.

Testing cannot show the absence of defects, it can only show that software errors are present.

**9.1.2. Test Case Design**

Any engineering product can be tested in one of two ways:

**White Box Testing**

This testing is also called as glass box testing. In this testing, by knowing the specified function that a product has been designed to perform test can be conducted that demonstrates each function is fully operation at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

**Basis Path Testing**

* Flow graph notation
* Cyclomatic Complexity

Deriving test cases Control Structure Testing

* Condition testing
* Data flow testing
* Loop testing

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

The steps involved in black box test case design are:

* Graph based testing methods
* Equivalence partitioning
* Boundary value analysis
* Comparison testing
* Graph matrices

**9.2. SOFTWARE TESTING STRATEGIES**

A Strategy for software testing integrates software test cases into a series of well planned steps that result in the successful construction of software. Software testing is a broader topic for what is referred to as Verification and Validation. Verification refers to the set of activities that ensure that the software correctly implements a specific function. Validation refers he set of activities that ensure that the software that has been built is traceable to customer’s requirements.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* All links should take the user to the correct page.

**9.2.1. Unit Testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**9.2.2. Integration Testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**9.2.3. Functional Testing**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**9.2.4. Validation Testing**

At the end of integration testing software is completely assembled as a package. Validation testing is the next stage, which can be defined as successful when the software functions in the manner reasonably expected by the customer. Reasonable expectations are those defined in the software requirements specifications. Information contained in those sections form a basis for validation testing approach.

**9.2.5. System Testing**

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work to verify that all system elements have been properly integrated to perform allocated functions. System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**9.2.6. Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

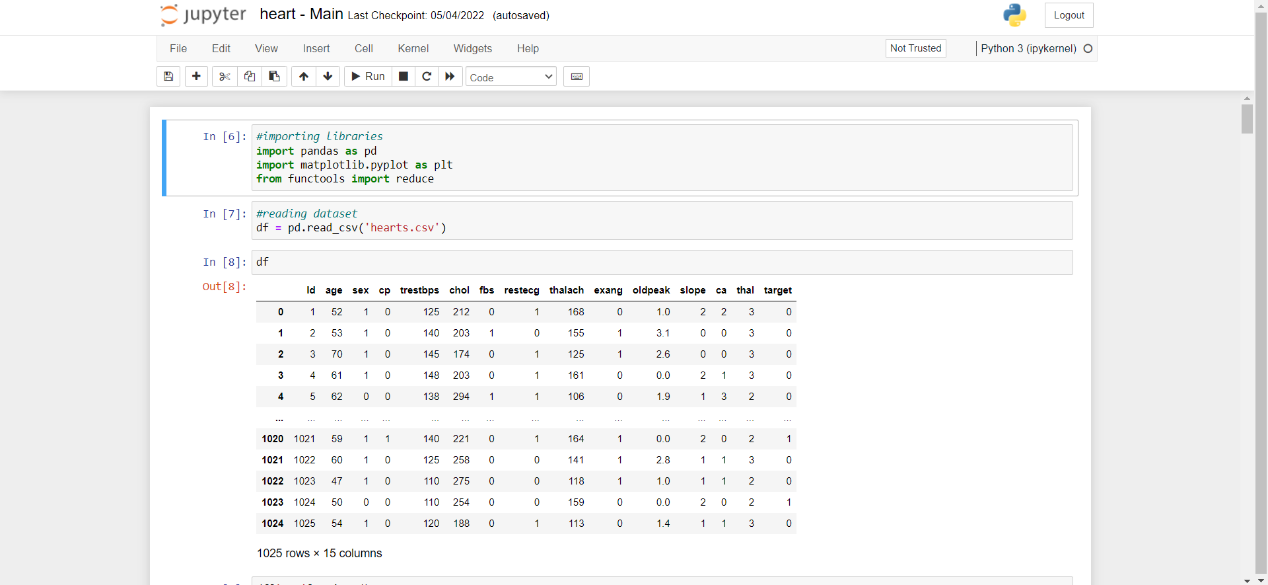
**9.2.7. Performance Testing**

This method is designed to test runtime performance of software within the context of an integrated system.

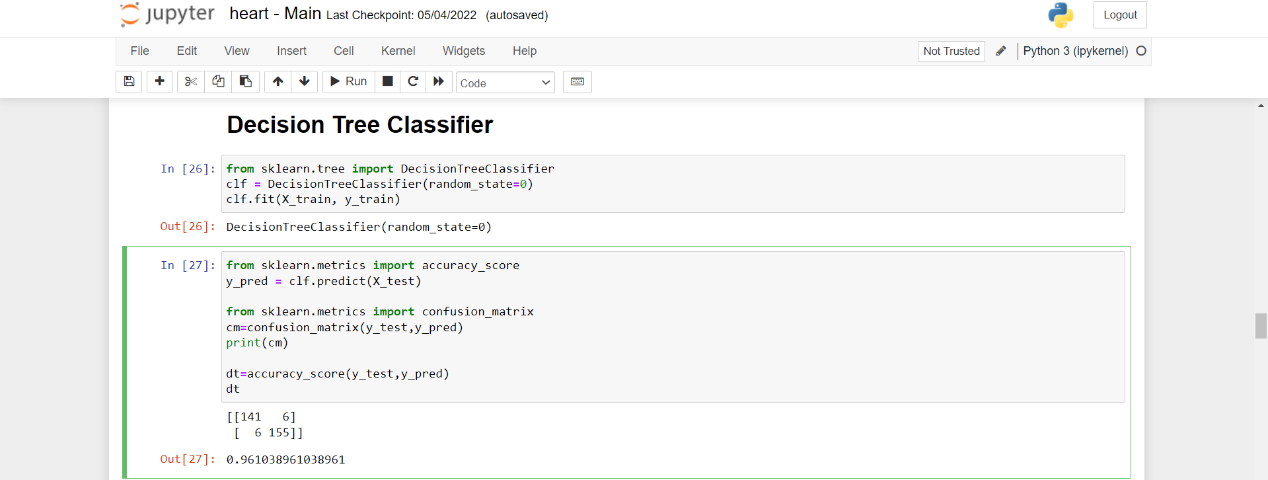
**9.3. TEST CASES**

Table 9.1: Test Case Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test**  **Sl.No.** | **Input** | **Expected Behavior** | **Observed**  **behavior** | **Status**  **P = Passed**  **F = Failed** |
| 1 | Login as admin with correct login details | User Home page for manager should be displayed | -do- | P |
| 2 | Login as user or admin with wrong login details | Error message should be displayed | -do- | P |
| 3 | HTML anchor tags | All links should take the user to the correct page. | -do- | P |
| 4 | Verify that the entries are of the correct format | Incorrect entry formats results in error and shows error message. | -do- | P |

**10. SCREENS**

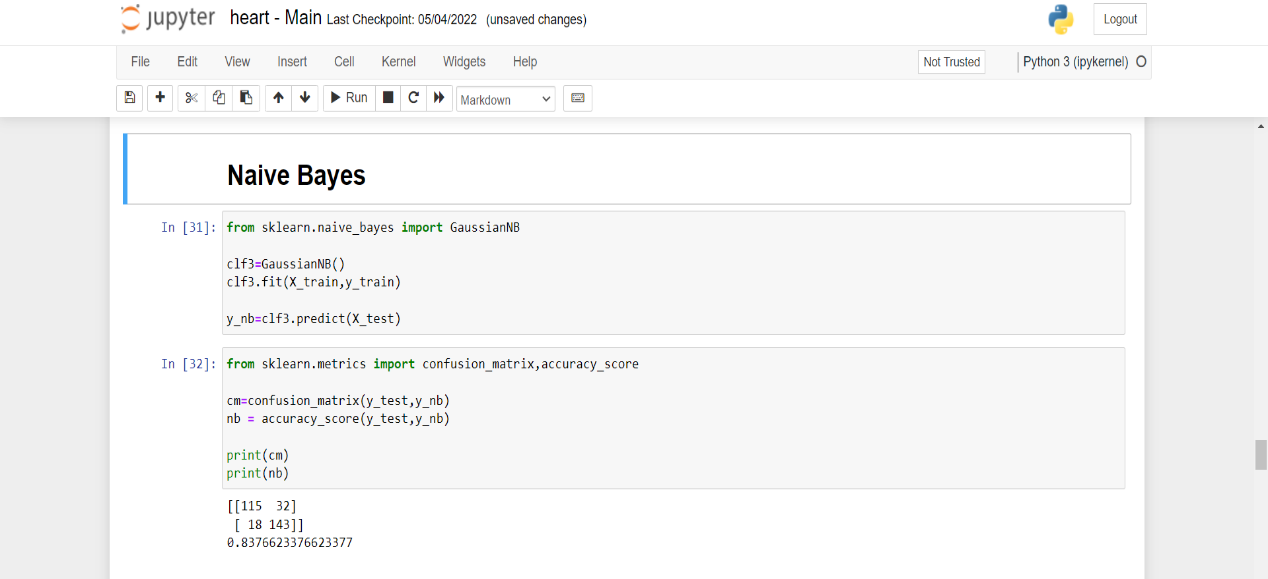
**Screen 10.1 : Reading Dataset**



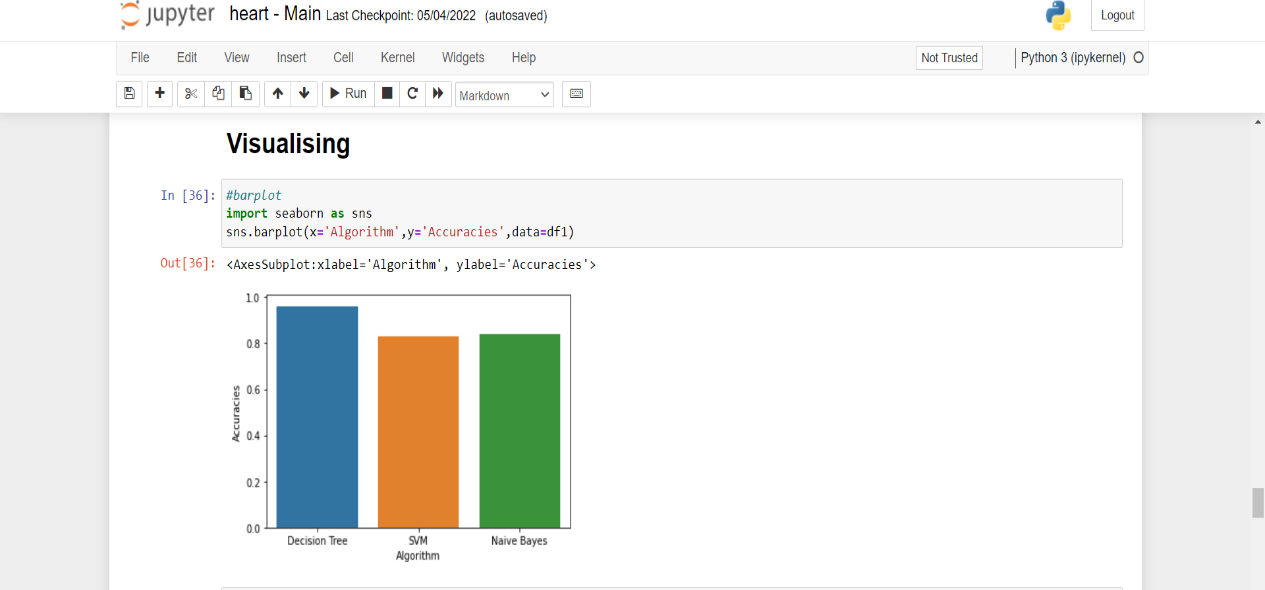
**Screen 10.2 : Implementing Decision Tree Classifier**



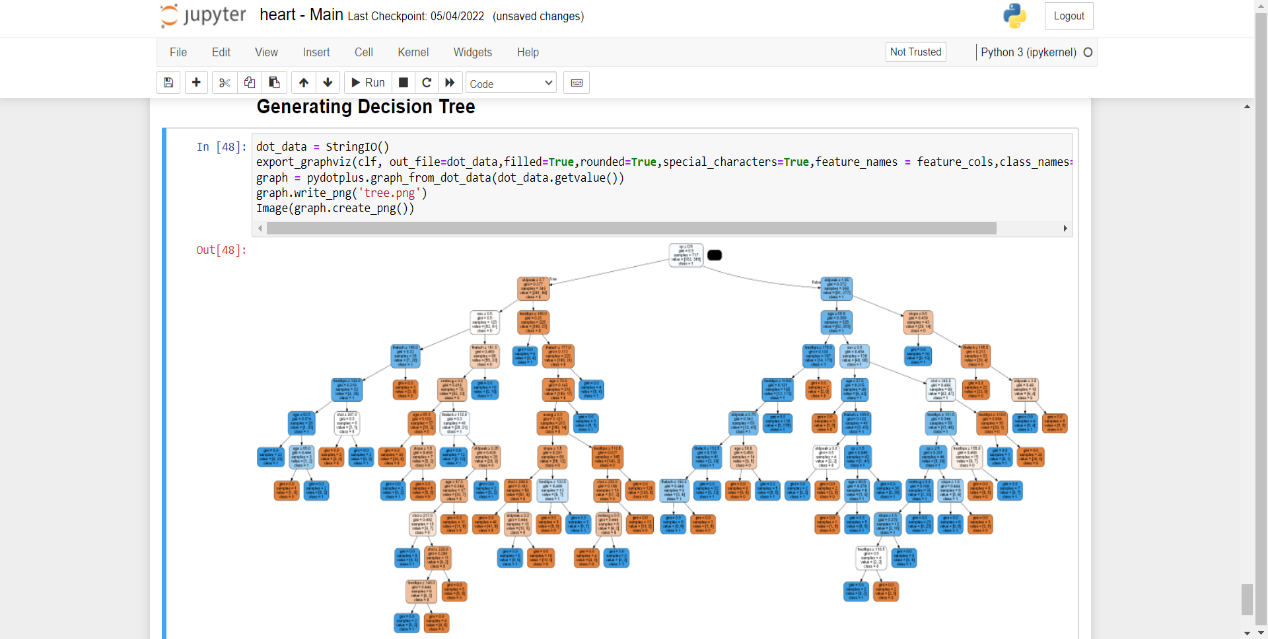
**Screen 10.3 : Implementing Support Vector Machine**



**Screen 10.4 : Implementing Naïve Bayes Classification**



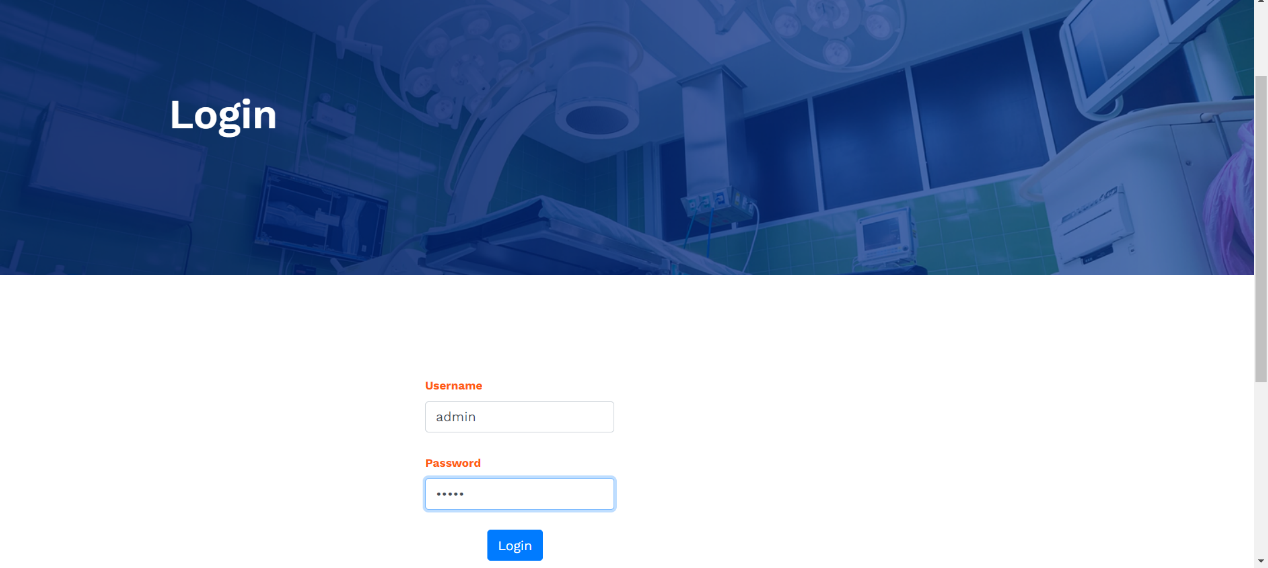
**Screen 10.5 : Comparing accuracies of different ML Algorithms using Barplot**

****

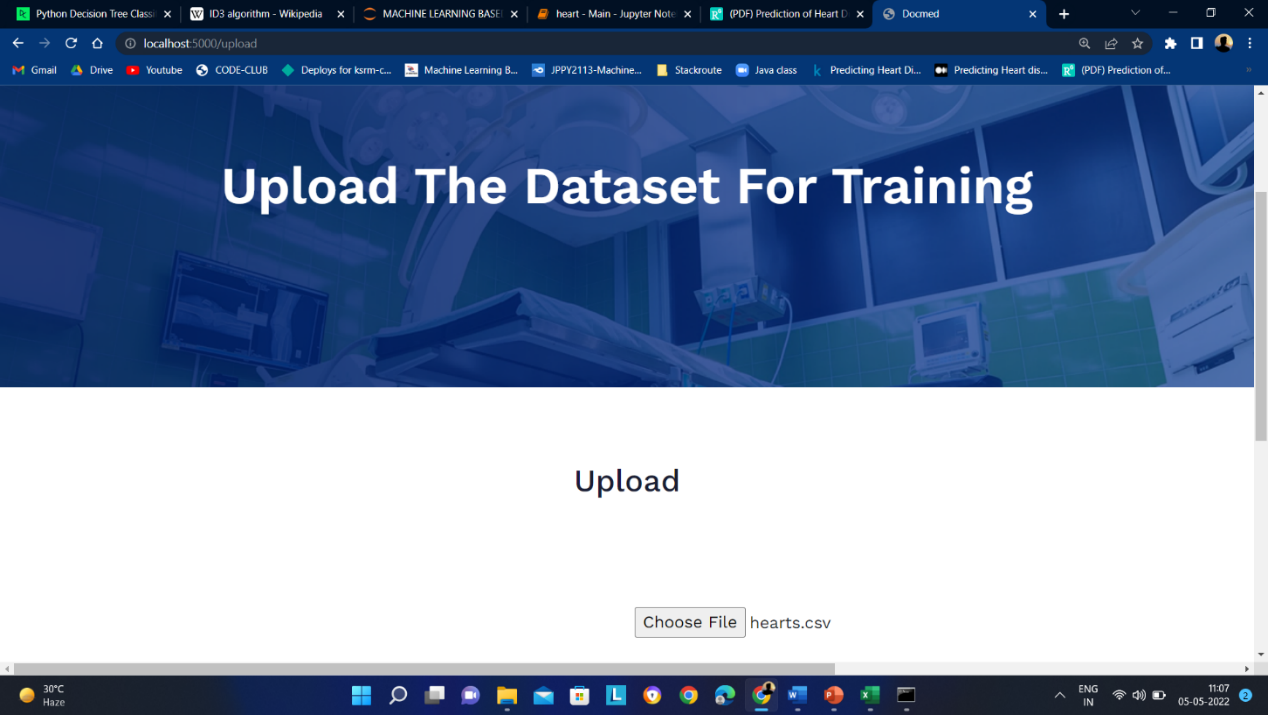
**Screen 10.6 : Generating Decision Tree using python**



**Screen 10.7 : Home Page**



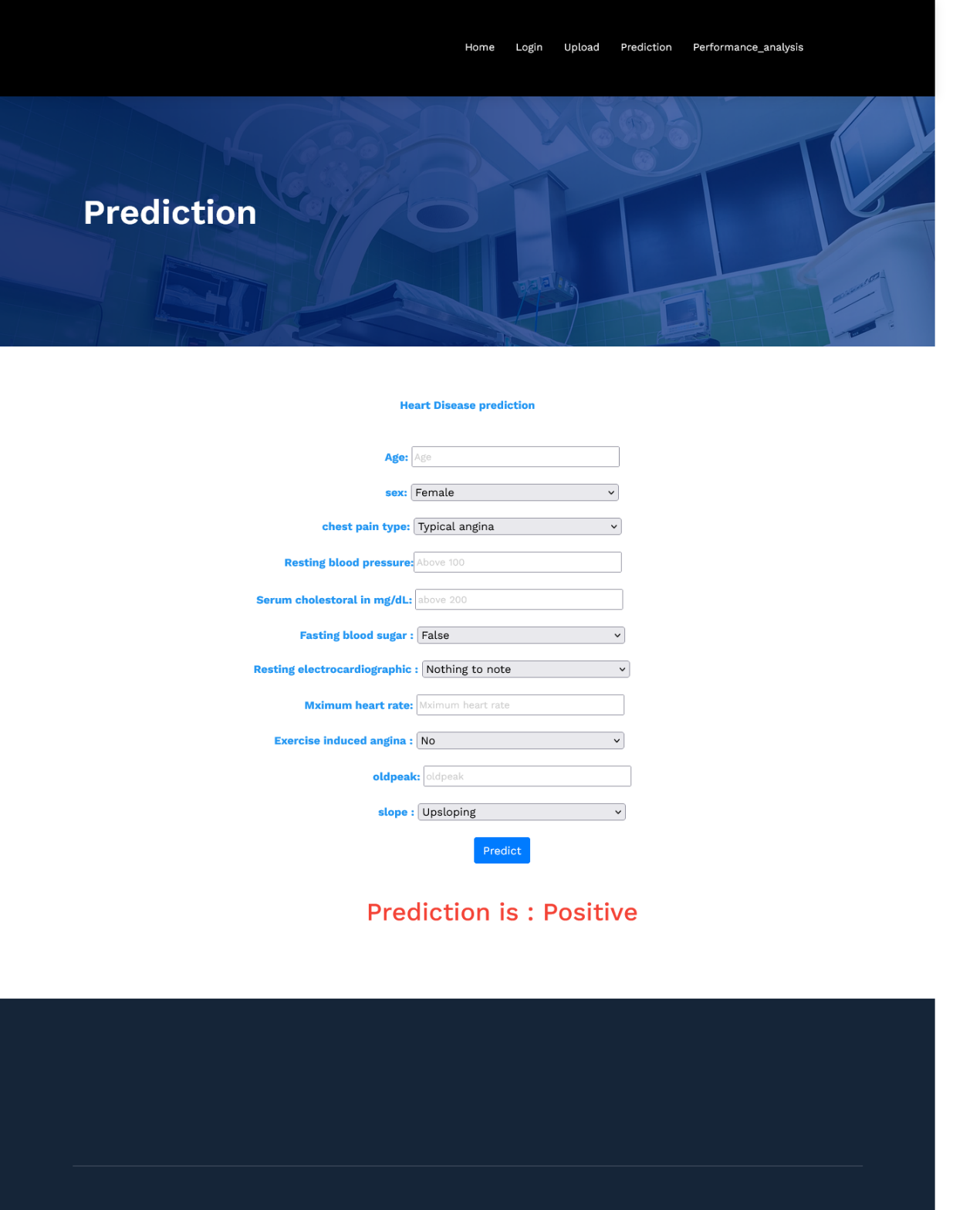
**Screen 10.8 : Login Page**



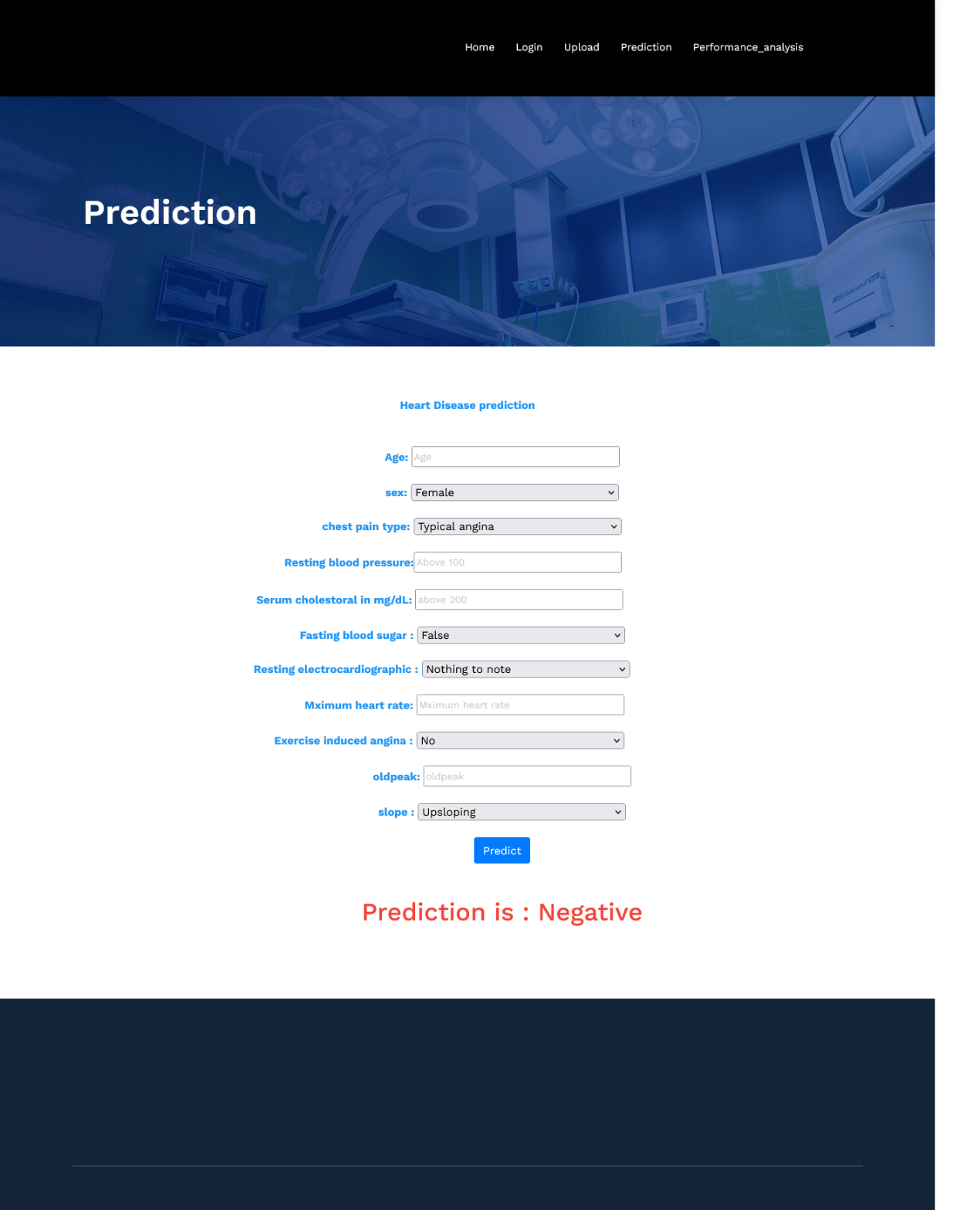
**Screen 10.9 : Select and Upload Dataset**



**Screen 10.10 : Train the Model**



**Screen 10.11 : Prediction Screen**



**Screen 10.12 : Prediction Screen**

**11. CONCLUSION**

Decision Tree is an efficient algorithm for classification problems. Predicting Heart disease is a classification problem with classes Positive and Negative. In this, what we found is during small datasets in some other cases most of time other algorithms direct us to a solution which is not accurate, but when we look at Decision Tree results we are getting more accurate results with 96% accuracy. This algorithm constructs decision tree based on the important attributes and classifies the new input. So accuracy of prediction at early stages is achieved effectively. Processing of healthcare data i.e., data related to heart will help in early detection of heart disease or abnormal condition of heart which results in saving of long term deaths. Heart disease prediction is a major challenge in the present modern life. With this application if the patient/user is away from reach of doctor, he/she can make use of the application in prediction of disease just by entering the report values. And can proceed further whether to consult a doctor or not.

**12. FUTURE SCOPE**

In future this application can extended by updating some features like, if the user is effected with heart disease all his family members will be notified with a message in early. And also the information should be passed to the nearest hospital. Another feature is there should be online doctor consultation with the nearest doctor available.

In this regard, it is important to note that, ML applications using various efficient algorithms are utilized not only in disease prediction and diagnosis but also in the field of radiology, bioinformatics and medical imaging diagnosis etc.

**REFERENCES**

[1] Kaan Uyar and Ahmet İlhan, "Diagnosis of heart disease using genetic algorithm based trained recurrent fuzzy neural networks" in B.V ICTASC, Elsevier.

[2] Theresa Princy and R. J. Thomas, "Human Heart Disease Prediction System using Data Mining Techniques", IEEE ICCPCT, 2016.

[3] M.M. Kirmani and S.I. Ansarullah, "Prediction of heart disease using decision tree a data mining technique", IJCSN Int. J. Comput. Sci. Netw., vol. 5, no. 6, pp. 885-892, 2016.

[4] M. Snehith Raja; M. Anurag; Ch. Prachetan Reddy; Nageswara Rao Sirisala, “Machine Learning Based Heart Disease Prediction System”, IEEE Conference, 2021

[5] Nichenametla, Rajesh & Maneesha, T. & Hafeez, Shaik & Krishna, Hari. (2018). Prediction of Heart Disease Using Machine Learning Algorithms. International Journal of Engineering and Technology(UAE).

[6] [Predicting Heart disease using Machine Learning | by Pulkit Khandelwal | Analytics Vidhya | Medium](https://medium.com/analytics-vidhya/predicting-heart-disease-using-machine-learning-ce10cfce41a6)

[7] N. Friedman, D. Geiger, and M. Goldszmidt, “Bayesian network classifiers,” Machine Learning,( 1997).

[8] M.A. Jabbar, B.L.Deekshatulu, and Priti Chandra, “Intelligent heart disease prediction system using random forest and evolutionary approach”, Journal of Network and Innovative Computing, Vol. 4, pp.174-184, 2016.

[9] JD Berry, DM Lloyd-Jones, DB Garside et al., "Framingham risk score and prediction of coronary heart disease death in young men", Am Heart J., vol. 154, no. 1, pp. 80-6, 2007.

[10] Ayon Dey, Jyoti Singh, N. Singh “Analysis of supervised machine learning algorithms for heart disease prediction” International journal of computer applications, published in 2006.