# An Enhanced Leaf Disease Detection System Using CNN-LeNet Approach

## A

## Project Report Submitted

**in partial fulfillment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE & ENGINEERING**

**By**

|  |  |
| --- | --- |
| **Sk. Sadhik** | **19761A0556** |
| **P. Firoz** | **19761A0546** |
| **G. Roji** | **20765A0503** |
|  |  |
|  |  |

**Under the esteemed guidance of Dr. J. Nageswara Rao**

### Associate Professor



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**LAKIREDDY BALI REDDY COLLEGE OF ENGINEERING (AUTONOMOUS)**

Approved by AICTE, New Delhi & Affiliated to JNTUK, Kakinada Accredited by NAAC with ‘A’ Grade& NBA and ISO 9001:2015 Certified Institute

LB Reddy Nagar, Mylavaram, NTR Dt, A.P

### 2019 – 2023

**LAKIREDDY BALI REDDY COLLEGE OF ENGINEERING (AUTONOMOUS)**

Approved by AICTE, New Delhi & Affiliated to JNTUK, Kakinada Accredited by NAAC with ‘A’ Grade& NBA and ISO 9001:2015 Certified Institute

LB Reddy Nagar, Mylavaram, NTR Dt, A.P

**Department of**

**COMPUTER SCIENCE & ENGINEERING**



**CERTIFICATE**

This is to certify that the project entitled **“An** **Enhanced Leaf Disease Detection System Using CNN-LeNet Approach”** is being submitted by:

|  |  |
| --- | --- |
| **Sk. Sadhik** | **19761A0556** |
| **P. Firoz** | **19761A0546** |
| **G. Roji** | **20765A0503** |
|  |  |

in partial fulfillment of the requirements for the award of degree of **B. Tech.** in **Computer Science & Engineering** from **Jawaharlal Nehru Technological University Kakinada** is a record of bonafide work carried out by them at **Lakireddy Bali Reddy College of Engineering (A).**

The results embodied in this Project Report have not been submitted to any other University or Institute for the award of any degree or diploma.

|  |  |  |
| --- | --- | --- |
| **PROJECT GUIDE**  **(Dr. J. Nageswara Rao)** | **EXTERNAL EXAMINER** | **HEAD OF THE DEPARTMENT**  **(Dr. D. Veeraiah)** |

# DECLARATION

We are here by declaring that the project entitled **“An Enhanced Leaf Disease Detection System Using CNN-LeNet Approach”** work done by us. We certify that the work contained in the report is original and has been done by me under the guidance of supervisor. The work has not been submitted to any other institute in preparing for any degree or diploma. We have followed the guidelines provided by the institute in preparing the report. We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the institute. Whenever we have used materials (data, theoretical analysis, figures, and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their details in the references. Further, we have taken permission from the copyright’s owner of the sources, whenever necessary**.**

## Name of the Student(s) Signature of the Student

|  |
| --- |
| **Sk. Sadhik(19761A0556)** |
| **P. Firoz(19761A0546)** |
| **G. Roji(20765A0503)** |
|  |

**ACKNOWLEDGEMENT**

We take great pleasure to express our deep sense of gratitude to our Project Guide, **Dr. J. Nageswara Rao,** Sr**.** Assistant Professor, Department of CSE for his valuable guidance during our project work.

We would like to thank **Dr. D. Veeraiah**, Professor & Head of the Department of Computer Science & Engineering for his encouragement.

We would like to express our heart-felt thanks to **Dr K. Appa Rao,** Principal, Lakireddy Balireddy College of Engineering for providing all the facilities for our project.

Our utmost thanks to all the faculty members and Non-Teaching Staff of the Department of Computer Science & Engineering for their support throughout our project work.

Our Family Members and Friends receive our deepest gratitude and love for their support throughout our academic year.

|  |  |
| --- | --- |
| **SK. Sadhik** | **19761A0556** |
| **P. Firoz** | **19761A0546** |
| **G. Roji** | **20765A0503** |
|  |  |

##### Plant disease detection is an important task in crop management and can be done with the help of Convolutional Neural Networks (CNNs). CNNs are a type of artificial neural network designed specifically for image recognition tasks such as detecting patterns in images. In this paper, we propose a CNN-based method for plant disease detection which uses an available dataset to extract features from an image of a plant and then classifies it into one or more disease classes. We also found that our model was able to accurately identify subtle differences between diseased plants compared to healthy ones - something traditional methods were not capable of doing on their own. The proposed method can be used for real-time use in crop management as it helps reduce crop loss and increase yields by detecting diseases early before they spread further.

##### LIST OF CONTENTS

**CONTENTS PAGE NO**

##### INTRODUCTION 1 - 2

* 1. Overview of the Project
  2. Feasibility Study
  3. Scope

##### LITERATURE SURVEY 3 - 9

* 1. Existing System & Disadvantages
  2. Proposed System & Advantages

##### SYSTEM ANALYSIS 10 - 13

* 1. Overview of System Analysis
  2. Software used in the project
  3. System Requirements

##### SYSTEM DESIGN 14 - 23

* 1. Overview of System Design
  2. UML diagrams

##### CODING & IMPLEMENTATION 24 – 33

##### SYSTEM TESTING 34 – 35

##### RESULTS 36 – 46

##### CONCLUSION 47

##### REFERENCES 48 - 50

Figure 1: Jupyter environment 11

Figure 2: System Architecture 14

Figure 3: Splitting the dataset into training and testing datasets 15

Figure 4: KNN example 17

Figure 5: Enhanced Decision Tree example 18

Figure 6: Support Vector-Hyperplane 19

Figure 7: Use Case Diagram 21

Figure 8: Class Diagram 22

Figure 9: Activity Diagram 23

Figure 10: Comparision of Accuracy rate of Various Machine Learning Algorithms 43

# INTRODUCTION

## OVERVIEW OF THE PROJECT

Heart disease is the major cause of death in modern times. Through the media, heart disease has been proven to be a major cause of death for both men and women. It is very difficult to identify heart disease because there are many risk factors such as high blood pressure (bp), diabetes, abnormal heartbeat, high cholesterol, and many other things. Predicting heart disease is considered one of the most important issues in the field of medical data diagnostics. Scientists have turned to modern methods, such as study machines, for predicting the disease. To differentiate or differentiate healthy people from people with heart disease, machine learning methods are very accurate and effective. One of the most common applications for Learning Machines is outcome prediction. In this paper, we have taken a database of Cleveland heart disease. This database is available online at the University of California Irvine (UCI) repository used by many researchers. Separation is an important and powerful machine learning method used to predict the purpose. This method is used in databases using K-Nearest Neighbor machine learning techniques called KNN, Enhanced Decision Tree, Support Vector Machine (SVM) Modification Models. These types are used to increase the accuracy levels of classification strategies.

## FEASIBILITY STUDY

All activities are available for use, provided with unlimited resources and time indefinite. However, software development is due to a lack of resources, as well as complex supply costs. It has been compulsory and sensible, to assess the feasibility of a project in a shortest time.

### Economic Feasibility

The economic feasibility will involve assessing the financial viability of developing the leaf disease detection system. The cost of acquiring the necessary technology, including hardware and software, will be considered, along with the cost of training and maintaining the system. The potential revenue from the system will also be evaluated to determine the return on investment.

### Technical Feasibility

The technical feasibility will involve determining the availability of the necessary technology, including hardware and software, to develop the leaf disease detection system. The system will require high-performance computing resources such as GPU and specialized software for machine learning, such as Python, TensorFlow, and Keras.

### Operational Feasibility

The operational feasibility will involve determining the practicality of implementing the leaf disease detection system. This will include evaluating the ease of use of the system by farmers and other end-users, and the availability of the necessary infrastructure for the system to function efficiently.

**1.3 SCOPE**

The scope of an enhanced leaf disease detection system using the CNN-LeNet approach includes the development of a machine learning model that can accurately and efficiently identify various types of leaf diseases in crops. The system will use images of leaves that have been affected by diseases as input data and use the CNN-LeNet architecture to classify the diseases.

The system will be designed to be user-friendly and accessible to farmers and other end-users, allowing them to quickly and easily identify diseases in their crops. The system will also provide information on the appropriate treatments for each disease, allowing for timely intervention and improved crop yields.

The scope of the system will also include the development of a user interface that allows end-users to interact with the system, view the results of the disease classification, and access relevant information on treatments and prevention methods.

In conclusion, the scope of an enhanced leaf disease detection system using the CNN-LeNet approach includes the development of a machine learning model, a user-friendly interface, and integration with existing agricultural technologies. The system has the potential to benefit the agriculture industry by improving disease detection and prevention, leading to improved crop yields and reduced losses.

# LITERATURE SURVEY

We read some source material before starting this work in an effort to properly complete it. Numerous cutting-edge techniques have been developed by researchers, and a summary of their work is presented in this section to help with plant disease detection, classification, and surveying.

A method for the detection of visible plant stem and leaf diseases has been put forth by Al Bashish et al [1]. The developed framework is built around image processing and goes through the following steps: first, the images are segmented using the K-Means technique, and then the segments are fed into a trained neural network. The end results show that the suggested method accurately and automatically detects leaf diseases. The developed classifier, which is based on statistical classification, worked well and was able to identify and classify the diseases.

An algorithm for locating the diseased area of plants using image processing has been proposed by Camargo et al [2]. The color transformation is done after the image is first captured. Enhancement of transformed images is accomplished using a Gaussian filter. To find the area of interest, segmentation is next used. Finding the ideal threshold allows the segments to be divided. The segmented regions are then classified and labeled as sick or healthy using the SVM classifier.

The classification of the diseased tomato leaves into different classes, including Tomato Late Blight, Septoria Spot, Bacterial Spot, Bacterial Canker, Tomato Leaf Curl, and Healthy, has been done by the authors in [3]. The implementation has been performed on a dataset of 383 digital camera-captured images. The dataset has been subjected to Otsu's method of image segmentation. The RGB color components were used to create color features, while the regionprops function was used to create shape features, GLCM was used to create texture features, and RGB color components were used to create color features. The result is a feature extraction module, which is made up of all the extracted features. The decision tree classifier has been trained using supervised learning techniques for classification. Despite its high accuracy, the decision tree has several limitations, including overfitting in the case of noisy data and a limited amount of user control.

## EXISTING SYSTEM

In present days, heart disease has become the most deathful disease. If the patients want to get treatment to their heart, they must first get check-up made for their heart. This will take a long time. To reduce this time, an intelligent system is developed to predict whether a person is having heart disease or not. Classification techniques are generally used in medical industry, because of the capability of these techniques in processing large amount of data produced from the healthcare industry. Forecasting cardiac disease using classification and machine learning has been a continuous effort from the past two to three decades. In the Exiting system Machine learning techniques such as K- Nearest Neighbors (KNN) and Support vector machine are used to predict the heart disease.

**K-Nearest Neighbors** algorithm is one of the simplest machine Learning algorithms based on the Supervised learning technique. KNN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a good suite category by using this algorithm. With the help of KNN, we can easily identify the category or class of a particular dataset. This algorithm has disadvantages which affects the prediction of leaf disease.

**Disadvantages**

* Does not work well with large dataset as calculating distances between each data instance would be very costly
* Does not work well with high dimensionality as this will complicate the distance calculating process to calculate distance for each dimension.
* Sensitive to noisy and missing data.

**Support Vector Machine** or SVM is one of the most closely monitored learning algorithms, which helps for solving both classification and Regression problems. Mainly, it is used for Machine Learning problems. The purpose of the SVM algorithm is to create a dynamic line or that can divide a n-dimensional space into sections so that whenever a new datapoint is added then we can easily place that new data point in the appropriate class for great reinforcement. This line which makes the best decision is called the hyperplane.

**Disadvantages**

* SVM algorithm is not suitable for large data sets.
* SVM does not perform very well when the data set has more noise i.e., target classes are overlapping.
* In cases where the number of features for each data point exceeds the number of training data samples, the SVM will underperform.

## PROPOSED SYSTEM

Heart disease is a major cause of death in modern times. The heart is an important part of the body. It sends blood to all parts of the body. Lifestyle changes, work-related stress, and poor eating habits contribute to many heart diseases. According to the (WHO) World Health Organization [1], cardiovascular diseases cause 17.7 million deaths each year and 31% worldwide. In India, cardiovascular disease has become a major cause of death. Therefore, it is also possible that an accurate diagnosis of cardiovascular disease is essential. Medical institutions around the world collect information on a variety of health-related topics. This data can be used using a variety of machine learning methods to get useful ideas. Machine Learning algorithms have been very useful in identifying the presence or absence of heart disease in recent days.

To test whether a person has heart disease or not, the app can be developed using Mechanical Learning Methods that can predict the risk of heart disease given basic characteristics such as age, gender, blood pressure etc. Computer literacy (ML) has proven to be effective in helping you make decisions and make predictions based on big data generated by the healthcare sector. In our proposed system, we have used an Enhanced Decision Tree classifier to create a system to predict whether a person has heart disease or not. In this, we have used some enhanced features to improve the accuracy of the model when compared with existing model.

The contribution of the proposed research is to design a machine-learning-based medical intelligent decision support system for the diagnosis of heart disease. An Enhanced Decision tree is a decision support tool that uses a tree-like graph or model of decisions and

their possible consequences including chance event outcomes and utility. It is one of the ways to display an algorithm. Decision trees are commonly used in operations research, specifically in decision analysis to help and identify a strategy that will most likely reach the goal. It is also a popular tool in machine learning. A Decision tree can easily be transformed to a set of rules by mapping from the root node to the leaf nodes one by one. Finally, by following these rules, appropriate conclusions can be reached.

**Advantages**

* It is very efficient, does not require too many computational resources, it’s highly interpretable, it doesn’t require input features to be scaled, it doesn’t require any tuning, it’s easy to regularize, and it outputs well-calibrated predicted probabilities
* Enhanced Decision Tree solves the problem of machine learning by transforming the data into tree representation.
* Each internal node of the tree representation denotes an attribute, and each leaf node denotes a class label.
* Enhanced Decision tree algorithm can be used to solve both regression and classification problems.

# SYSTEM ANALYSIS

## OVERVIEW OF SYSTEM ANALYSIS

### Machine Learning

Machine Learning (ML) is one of the applications of Artificial Intelligence (AI) that gives machines or systems the ability to learn and develop from past experience without previously clearly planned. It simply makes the computer work without explaining it properly. In today's world every successful program has an algorithm learning heart in its heart. Take for example the Google Search engine, LinkedIn, Facebook etc., all of these programs have automated learning methods enclosed in their programs in one way or another. They make good use of the data that are collected from various channels that help them get a bigger picture of what are they doing and what they should do. It starts from the pre-processing phase of the data followed by feature selection, classification, performance testing and finally results with improved accuracy.

## SOFTWARE USED IN THE PROJECT

### Python

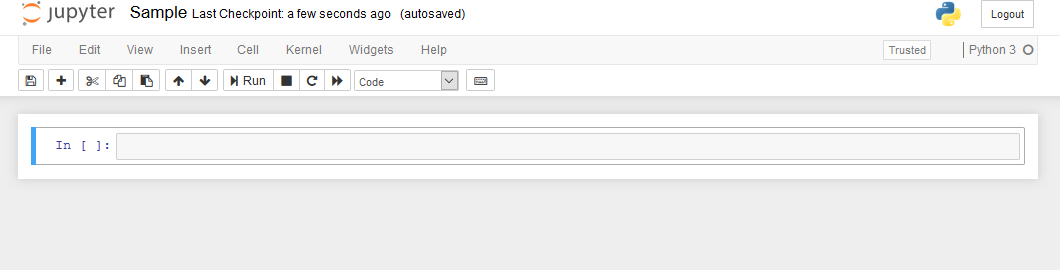
Python is a translated, highly functional, common programming language. It is the language of many programs. The most widely used programming language with the highest level of the general-purpose program. Apart from being an open-source programming language, python is a targeted, translated and active programming language. Python combines remarkable power with clear syntax. It has modules, classes, variations, very powerful data types, and powerful typing. There are network connections for most system phones and libraries, as well as various windows installation programs. Python is also used as language extensions for applications written in other languages ​​that require an easy-to-use script or interface. Python is widely regarded as the preferred language for teaching and learning ML (Machine Learning). Here are a few simple reasons:

* + - * Easy to work with. Compared with C, C++, and Java, the syntax is simpler, and in Python, it is also comprised of a lot of code and libraries for ease-of-use.
      * Although slower than other languages, the data management capacity is good.
      * Open Source: – Python, along with the R, it is gaining momentum and popularity in the field of the analysis, as both of these languages are open-source.
      * Ability to communicate with almost all third-party languages ​​and forums

### Jupyter Notebook

Jupyter Notebook is an incredibly powerful tool used for developing and remitting data science projects. This article shows you how to set up note books on local machine and using them for data science projects.





##### Figure 1: Jupyter environment

* + 1. **Packages**

**pandas:** pandas software library written in the language of the Python program for cheating and analyzing data. In particular, it provides data structure and function for managing numerical tables and time series. pandas are widely used for machine learning in the form of data frames. Pandas allows importing data for various file formats such as csv, excel etc.

**matplotlib.pyplot:** Provides a site-based API based on applications using standard GUI tools. pyplot shell-like interface in Matplotlib. Pyplot keeps status on all calls. It is useful for use in Jupyter or IPthon notebooks.

**sklearn:** sklearn (Scikit-learn) is a free online machine learning library for Python programs. It includes various classification, decoding and integration of algorithms. Provides a wide range of learning algorithms with a compatible environment in python.

### Dataset

### Dataset used: <https://www.kaggle.com/ronitf/heart-disease-uci>

### Analysis of the Kaggle Machine Learning Heart Disease dataset. The dataset consists of 14 main attributes and 303 instances used for performing the analysis. Various promising results are achieved and are validated using accuracy. Data set is used to predict various types of heart disease

##### *Table 1*: Dataset attributes with detailed information

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Description** | **Value Range** |
| Age | Patient’s age | 29 - 79 |
| Sex | Patient’s gender (female represented as 0 and male represented as 1) | 0,1 |
| Cp | The type of chest pain was categorised into 4 values: typical angina,  atypical angina, non-angina pain, asymptomatic | 1, 2, 3, 4 |
| Trestbps | Blood pressure level at resting mode (in mm / Hg at the time of admitting in the hospital) | 94 - 200 |
| Chol | Serum cholesterol in mg / dl | 126 - 564 |
| Fbs | Blood sugar levels on fasting > 120mg/dl. Represented as 1 in case of true and 0 in case of false | 0,1 |
| Restecg | Results of electro cardiogram while at rest are represented in 3 distinct values: Normal state is represented as value 0 and abnormality in ST-T wave as value 1, (which may include inversions of T-wave and/or depression or elevation of ST of > 0.05 mV) and any probability or certainty of LV hypertrophy by Estes’  criteria as value 2 | 0,1,2 |

|  |  |  |
| --- | --- | --- |
| Thalach | The accomplishment of the maximum rate of heart | 71 - 202 |
| Exang | Angina induced by exercise. (0 for depicting ‘no’ and 1 for  depicting ‘yes’) | 0,1 |
| Oldpeak | Exercise-induced ST depression in comparison with the state of rest | 1 – 3 |
| Slope | ST segment measured in terms of the slope during peak exercise  depicted in three values: unsloping, flat, downsloping | 1,2,3 |
| Ca | Fluoroscopy coloured major vessels numbers from 0 to 3 | 0 to 3 |
| Thal | Status of the heart illustrated through three distinctly numbered  values. Normal numbered as 3, fixed defect as 6 and reversible defect as 7. | 3,6,7 |
| Target | Heart disease diagnosis is represented in 2 values, 0 indicating the  absence and 1 representing the presence of heart disease. | 0 or 1 |

The attribute we need to find is the “target” attribute which depicts whether the person is having heart disease or not.

## SYSTEM REQUIREMENTS

### Software Requirements

Operating System : Windows7 or higher

Languages : Python

IDE : Jupyter Notebook

Dataset : Cleveland heart disease dataset

### Hardware Requirements

RAM : Minimum 4GB

Processor : Intel core i5

Secondary Storage : Minimum 500GB

# SYSTEM DESIGN

## OVERVIEW OF SYSTEM DESIGN

The data that we want to process may or may not be clean which means it may consist of noise or it may contain missing values that means null values. With this kind of data, we cannot obtain good results. To get good and accurate results, we have to clean it. We can fill those missing values and can remove noise by using techniques like filling the missing values with frequently occurring values or by taking the mean of the values etc. We may or may not require all the attributes in the dataset. So, to reduce complexity, we remove some attributes from the dataset that are not useful in our process. This is done by Feature Selection techniques like Information Gain, Chi-Square Test, Correlation coefficient etc. Figure 2 gives an idea about the flow of architecture of the process from a stage of dataset taken to final stage of results.

Diagram

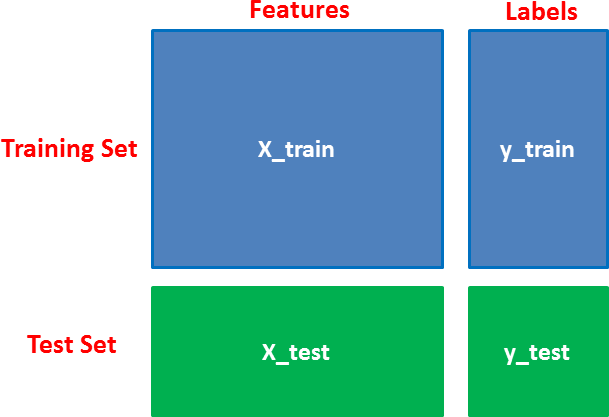
Description automatically generated

##### Figure 2: System Architecture.

Machine learning (ML) is the scientific study of algorithms and mathematical models used by computer programs to perform a specific task without using any other external instructions, depending upon the pattern and the position. It is the foundation of artificial intelligence. Machine learning algorithms create a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without any external planning to perform the task. Machine learning algorithms are used for a wide variety of applications, such as email filtering and computer viewing, in which it will be difficult and impossible to perform a standard performance for only one algorithm.

## Splitting the dataset

The data set is divided into two sets of which one is training dataset and another one is testing data set. The training dataset is used to train the model or algorithm and the test database is used to test the performance of the algorithm. Figure 3 shows how the attributed of the dataset are divided. In that, the training set contains x\_train and y\_train. X\_train is the input attributes or features to be given as input to the algorithm for training and y\_train contains the target variable which is already given. Similarly, the testing set will contain x\_test and y\_test. X\_test is the input attributes or features to be given as input to the algorithm for testing and y-test contains the target variable to be predicted.



##### Figure 3: Splitting the dataset into training and testing datasets

**CLASSIFICATION ALGORITHMS**

* + 1. **K-Nearest Neighbor (KNN)**

KNN is one of the most famous and easiest classification algorithms as of now in the industry because of its fast-learning technique and simplicity. It uses an easiest approach when dealing with a classification problem. When tested with the testing data sample, it looks thoroughly into the training data and finds the ‘K’ training samples that are closet to the testing sample that means it calculates the distance between the new sample and all the existing training samples and then it takes the K nearest samples. From these K nearest samples, it considers the class samples that are majority in number and assigns the new sample with the considered class.

**Working of KNN algorithm:**

The following steps will help us understand about the algorithm clearly.

**Step 1:** Firstly, divide or split the data into training and testing data.

**Step 2:** Next, we need to select “K” value, i.e., the number of nearest neighbors should be taken into consideration. For this purpose, follow Step 3.

**Step 3:** For every sample in the test data, the following step is executed.

Step 3.1: Calculate the Euclidean distance between the test sample and each row in the training data. The distance can be calculated using any of the distance functions namely Euclidean, Manhattan, Minkowski. The formulae are shown in below equations.

**Step 4:** Now, sort the distances in ascending order.

**Step 5:** Select the first “K” rows from the sorted distances.

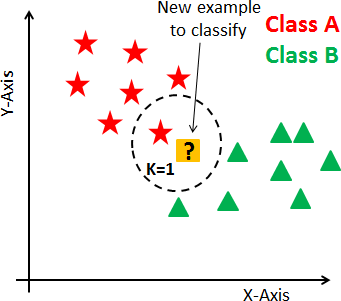
**Step 6:** Find the majority class from these “K” rows and assign the test sample with the majority class label. Thus, the resultant point will become the resultant class.

### Euclidean distance

### Manhattan distance

### Minkowski distance

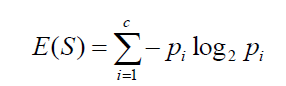
In the above equations, the variables x1, x2, ….xn represent the attributes of the training sample and y1, y2, …..yn represent the attributes of the testing sample where ‘n’ denotes number of attributes or features in the dataset. Figure 4 represents an example of KNN algorithm. The red colour stars belong to Class A and green colour triangles belong to Class B. In that ‘K’ value is taken as 1 (K = 1). So, the nearest neighbor is taken into consideration and the test sample is assigned to the corresponding class. Since the nearest neighbor belongs to Class A, the new sample will also be assigned to Class A.



##### Figure 4: KNN example

##### Enhanced Decision Tree

The Decision tree is a very sturdy and simpler algorithm for division and prediction. The decision tree is a tree-like structure, where each internal node defines a test in a qualification, each branch addresses a test outcome, and every terminal node holds a category label that works with category details and numbers. The decision tree is used to build structures such as trees. The decision tree is straightforward and broadly utilized for managing clinical databases. It is simple to use and to interpret the data in a tree-shaped structure. The algorithm partitions the data into two or more identical sets according to the most important indicators. For each characteristic, the entropy is determined, and the details are categorized, with indicators for higher or minimum entropy information:





##### The outcomes acquired are simpler to peruse and analyze. The Decision tree algorithm has higher exactness in contrast with different algorithms as it dissects the dataset in the tree shaped diagram. Notwithstanding, the data might be excessively grouped and just every trait is tested in turn for decision making.

##### 

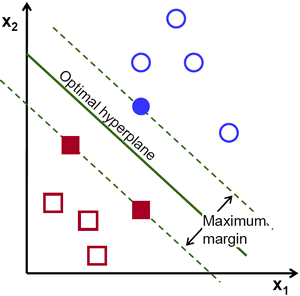
##### Figure 5: Decision Tree example

##### Support Vector Machine (SVM)

Support Vector Machine or SVM is one of the most closely monitored learning algorithms, which helps for solving both classification and Regression problems. Mainly, it is used for Machine Learning problems.

The purpose of the SVM algorithm is to create a dynamic line or that can divide a n-dimensional space into sections so that whenever a new datapoint is added then we can easily place that new data point in the appropriate class for great reinforcement. This line which makes the best decision is called the hyperplane.

SVM selects addon points that help to create the hyperplane. These extreme data points which on the planes that equally divides the hyperplane are called support vectors, that is why the algorithm is known as Support Vector Machine. Consider the diagram below in where there are two distinct classes divided by a boundary or hyperplane:



**Figure 6: Support Vector-Hyperplane**

**Hyperplane and Support Vectors**

**Hyperplane:** There may be as many lines as possible / restrictions for class distinctions in the wide space, but we need to find the best possible line that helps to separate data points. This best possible hyperplane is known as the SVM hyperplane.

The size of the hyperplane depends on the number of input features present in the database, which means that if there are 2 input features, then the hyperplane dividing a 2-dimensional space will be a straight line.

We always create hyperplane with high limits, which means a higher distance which can be between data points.

**Support Vectors:** Data points or vectors closest to the hyperplane and are called Support Vector. As these vectors support hyperplane, hence the name Support vector.

* 1. **UML DIAGRAMS**

In the field of software engineering, Unified Modeling Language (UML) is the preferred model for modeling modeling. UML is a standard purpose-oriented modeling language which in that includes a clear text used to produce a vague model of program, called the UML model. The model contains a "Semantic backplane" - texts such as written use cases that run model objects and drawings.

### Importance of UML in Modeling:

Model language is a language with its own words and pronouns that focus on the psychological and physical expression of the system. UML modeling language is thus a common language for software applications. UML is not a graphical programming language, but the models it generates can be directly linked to various of programming languages. This in further means that it is possible to create a map from a model in UML to the Java, C ++ or Visual Basic, or to even related data tables or a persistent database-focused store. This map allows advanced engineering: coding from UML model to programming language. Going back is also possible to re-create a model from usage back to UML. The UML is a programming language which is used for development of object-based software. To arrange a program code effectively, programmers often be creating "objects" which are structured within systems. The UML, which has been modeled on the Object Management Group (OMG), is designed for this purpose.

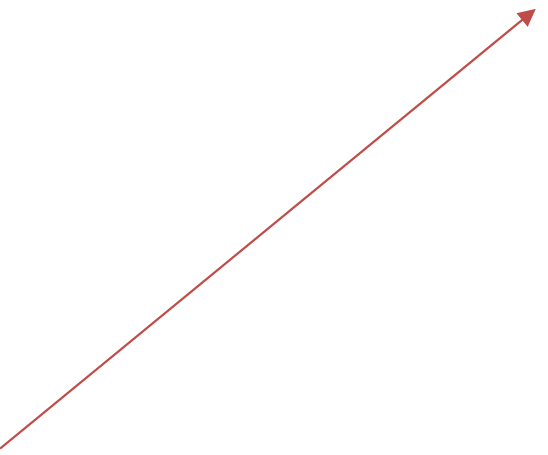
### A conceptual model of UML:

The three major elements of UML are

* + - The UML’s basic building blocks
    - The rules that dictate how those building blocks may be put together
    - Some common mechanism that applies throughout the UML

### Use case Diagram:

The use case diagram shows the set of various kinds of use cases with the characters and the relationships. We will use a case study diagram to illustrate the standalone visual of the system. Use case diagram is very crucial in planning and matching the behavioral behavior of a program.



Read input dataset

Train the algorithm with training data

Test the performance with testing data

User

Performance Evaluation

Display the result



##### Figure 7: Use Case Diagram

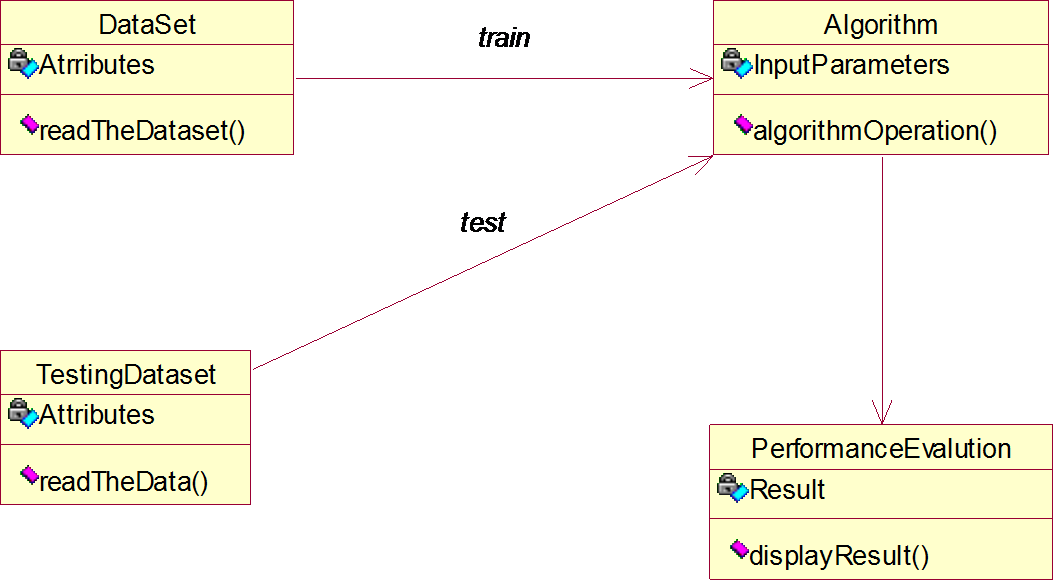
* + 1. **Class Diagram:**

Classes are the building blocks which are vital of any object aimed at an object. A paragraph is like a key of a set of objects that share similar attributes, function, relationships and semantics. Category uses one or more interfaces. Translated by drawings as rectangle.

##### IDENTIFYING RELATIONSHIPS:

In UML the means by which things can be connected wither mentally or physically are made as a relationship. In object-oriented modeling there are three types of relationships that are most important, viz

1. **Dependencies:** It is a relationship which can be explained that a change in a certain specification of one thing can automatically affect another thing that uses each. Graphically dependency is a dashed directed line.
2. **Generalization:** The relationship between a common object (called a parent) and a particular type of object (called a child). It means that the child's belongings can be used wherever the parent may appear. According to the drawing it is given as a solid line directed by the head of a large open arrow pointing to the parent.
3. **Association:** Relationships that specify that the objects of one entity are connected to the entities of another. Organizations which connect usually more than the two classes are called n-array organizations. According to the diagram it is translated as a line that connects the same or different classes.



##### Figure 8: Class Diagram

* + 1. **Activity Diagram:**

The aim of an activity diagram is to give an idea of ​​the flow and what happens within the application case or between several categories. An activity diagram simply describes the internal tasks performed and the changes caused by the completion of certain tasks. At the incomprehensible level it describes the sequence of tasks. This focuses on the events that occur in one thing as it responds to messages, an activity diagram can be used to perform the whole process. The sketch of the work usually contains:

##### Activity states and action states

To call the operation of an object that sends a signal for an object or to create or destroy an object, these useful figures are called provinces because they are the regions of the system, representing the action.

The Activity states can then be divided and their activity that is being represented by other activity diagrams can be failed or stopped and take some duration to complete.

##### Transitions

When an action or state function ceases the flow of control it immediately moves on to the next action or state of action. Explains this flow using changes to show the way from another action or status situation to the next action or work situation. You represent this as an easy-to-direct line.

##### Object Flow

Items may be involved in the control movement associated with the job drawing. The use of a dependent relationship is called the flow of an object because it indicates the participation of the object in the control movement.



User gives Dataset to Algorithm

User trains the Algorithm

Test with testing Dataset

Predict and Display output

##### Figure 9: Activity Diagram

**5. CODING AND IMPLEMENTATION**

## Importing Essential Libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

import os

print(os.listdir())

import warnings

warnings.filterwarnings('ignore')

**Importing Dataset**

dataset = pd.read\_csv("heart.csv")

**Printing out a few coloumns**

dataset.head(5)

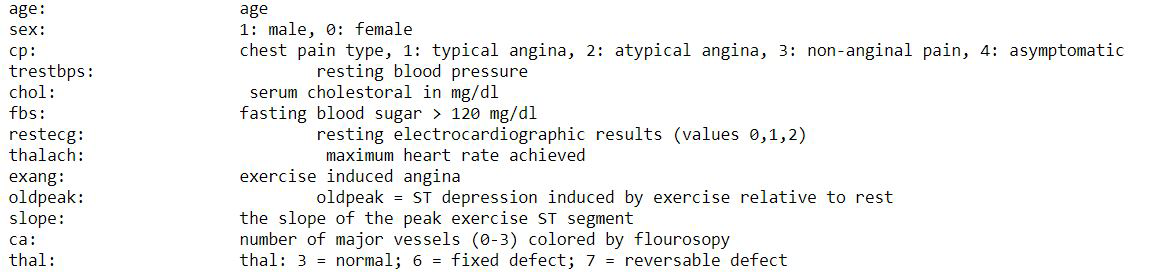
Table

Description automatically generated

#### Understanding Columns

info = ["age","1: male, 0: female","chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4:asymptomatic","resting blood pressure"," serum cholestoral in mg/dl","fasting blood sugar > 120 mg/dl","resting electrocardiographic results (values 0,1,2)"," maximum heart rate achieved","exercise induced angina","oldpeak = ST depression induced by exercise relative to rest","the slope of the peak exercise ST segment","number of major vessels (0-3) colored by flourosopy","thal: 3 = normal; 6 = fixed defect; 7 = reversable defect"]

for i in range(len(info)):

print(dataset.columns[i]+":\t\t\t"+info[i])

**Description**

dataset.describe()

Table

Description automatically generated

**Attributes information**

dataset.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 303 entries, 0 to 302

Data columns (total 14 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 age 303 non-null int64

1 sex 303 non-null int64

2 cp 303 non-null int64

3 trestbps 303 non-null int64

4 chol 303 non-null int64

5 fbs 303 non-null int64

6 restecg 303 non-null int64

7 thalach 303 non-null int64

8 exang 303 non-null int64

9 oldpeak 303 non-null float64

10 slope 303 non-null int64

11 ca 303 non-null int64

12 thal 303 non-null int64

13 target 303 non-null int64

dtypes: float64(1), int64(13)

memory usage: 33.3 KB

### Checking Correlation between Columns

### print(dataset.corr()["target"].abs().sort\_values(ascending=False))

|  |  |
| --- | --- |
| Target | 1.000000 |
| Exang | 0.436757 |
| Cp | 0.433798 |
| oldpeak | 0.430696 |
| thalach | 0.421741 |
| Ca | 0.391724 |
| slope | 0.345877 |
| thal | 0.344029 |
| Sex | 0.280937 |
| Age | 0.225439 |
| trestbps | 0.144931 |
| restecg | 0.137230 |
| chol | 0.085239 |
| Fbs | 0.028046 |

Name: target, dtype: float64

**Train-Test Split**

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

predictors = dataset.drop("target",axis=1)

target = dataset["target"]

X\_train,X\_test,Y\_train,Y\_test= train\_test\_split(predictors,target,test\_size=0.20,random\_state=0)

X\_train.shape

(242, 13)

X\_test.shape

(61, 13)

Y\_train.shape

(242,)

Y\_test.shape

(61,)

**Model Fitting**

from sklearn.metrics import accuracy\_score

**Support Vector Machine**

from sklearn import svm

sv = svm.SVC(kernel='linear')

sv.fit(X\_train, Y\_train)

Y\_pred\_svm = sv.predict(X\_test)

Y\_pred\_svm.shape

(61,)

score\_svm = round(accuracy\_score(Y\_pred\_svm,Y\_test)\*100,2)

print ("The accuracy score achieved using Linear SVM is: "+str(score\_svm)+" %")

The accuracy score achieved using Linear SVM is: **78.69 %**

**K-Nearest Neighbors (KNN)**

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n\_neighbors=7)

knn.fit(X\_train,Y\_train)

Y\_pred\_knn=knn.predict(X\_test)

Y\_pred\_knn.shape

(61,)

score\_knn = round(accuracy\_score(Y\_pred\_knn,Y\_test)\*100,2)

print ("The accuracy score achieved using KNN is: "+str(score\_knn)+" %")

The accuracy score achieved using KNN is: **67.21 %**

**Enhanced Decision Tree**

from sklearn.tree import DecisionTreeClassifier

max\_accuracy = 0

for x in range (200):

dt = DecisionTreeClassifier(random\_state=x)

dt.fit(X\_train,Y\_train)

Y\_pred\_dt = dt.predict(X\_test)

current\_accuracy = round(accuracy\_score(Y\_pred\_dt,Y\_test)\*100,2)

if(current\_accuracy>max\_accuracy):

max\_accuracy = current\_accuracy

best\_x = x

dt = DecisionTreeClassifier(random\_state=best\_x)

dt.fit(X\_train,Y\_train)

Y\_pred\_dt = dt.predict(X\_test)

print(Y\_pred\_dt.shape)

(61,)

score\_dt = round(accuracy\_score(Y\_pred\_dt,Y\_test)\*100,2)

print ("The accuracy score achieved using Decision Tree is: "+str(score\_dt)+" %")

The accuracy score achieved using Decision Tree is: **86.97 %**

# Output final score

scores = [score\_svm,score\_knn,score\_dt]

algorithms = ["Support Vector Machine","K-Nearest Neighbors","Enhanced Decision Tree"]

for i in range(len(algorithms)):

print("The accuracy score achieved using "+algorithms[i]+" is: "+str(scores[i])+" %")

The accuracy score achieved using Support Vector Machine is: 78.69 %

The accuracy score achieved using K-Nearest Neighbors is: 67.21 %

The accuracy score achieved using Enhanced Decision Tree is: 86.07 %

**Prediction on new data**

new\_data = pd.DataFrame({

'age':52,

'sex':1,

'cp':0,

'trestbps':125,

'chol':212,

'fbs':0,

'restecg':1,

'thalach':168,

'exang':0,

'oldpeak':1.0,

'slope':2,

'ca':2,

'thal':3,

}, index = [0])

Table

Description automatically generated

dt = DecisionTreeClassifier(random\_state=x)

dt.fit(X\_train,Y\_train)

Y\_pred\_rf = dt.predict(new\_data)

print(Y\_pred\_rf)

> [0]

p = dt.predict(new\_data)

if p[0]==0:

print("No Disease")

else:

print("Disease")

>No Disease

### Save Model Using joblib

### import joblib

### joblib.dump(dt,'model\_joblib\_heart')

### model = joblib.load('model\_joblib\_heart')

### model.predict(new\_data)

### dataset.tail()

### A screenshot of a computer Description automatically generated with low confidence

**Graphical User Interface (GUI)**

from tkinter import \*

import joblib

from tkinter import \*

import joblib

import numpy as np

from sklearn import \*

def show\_entry\_fields():

p1=int (e1.get ())

p2=int (e2.get ())

p3=int (e3.get ())

p4=int (e4.get ())

p5=int (e5.get ())

p6=int (e6.get ())

p7=int (e7.get ())

p8=int (e8.get ())

p9=int (e9.get ())

p10=float (e10.get ())

p11=int (e11.get ())

p12=int (e12.get ())

p13=int (e13.get ())

model = joblib.load('model\_joblib\_heart')

result=model. predict ([[p1, p2,p3,p4,p5,p6,p7,p8,p8,p10,p11,p12,p13]])

if result == 0:

Label (master, text="No Heart Disease"). grid(row=31)

else:

Label (master, text="Possibility of Heart Disease"). grid(row=31)

master = Tk ()

master.title("Heart Disease Prediction System")

label = Label (master, text = "Heart Disease Prediction System"

, bg = "black", fg = "white"). \

grid (row=0, columnspan=2)

Label (master, text="Enter Your Age"). grid(row=1)

Label (master, text="Male or Female [1/0]"). grid(row=2)

Label (master, text="Enter Value of CP"). grid(row=3)

Label (master, text="Enter Value of trestbps"). grid(row=4)

Label (master, text="Enter Value of chol"). grid(row=5)

Label (master, text="Enter Value of fbs"). grid(row=6)

Label (master, text="Enter Value of restecg"). grid(row=7)

Label (master, text="Enter Value of thalach"). grid(row=8)

Label (master, text="Enter Value of exang"). grid(row=9)

Label (master, text="Enter Value of oldpeak"). grid(row=10)

Label (master, text="Enter Value of slope"). grid(row=11)

Label (master, text="Enter Value of ca"). grid(row=12)

Label (master, text="Enter Value of thal"). grid(row=13)

e1 = Entry(master)

e2 = Entry(master)

e3 = Entry(master)

e4 = Entry(master)

e5 = Entry(master)

e6 = Entry(master)

e7 = Entry(master)

e8 = Entry(master)

e9 = Entry(master)

e10 = Entry(master)

e11 = Entry(master)

e12 = Entry(master)

e13 = Entry(master)

e1. grid (row=1, column=1)

e2. grid (row=2, column=1)

e3. grid (row=3, column=1)

e4. grid (row=4, column=1)

e5. grid (row=5, column=1)

e6. grid (row=6, column=1)

e7. grid (row=7, column=1)

e8. grid (row=8, column=1)

e9. grid (row=9, column=1)

e10.grid(row=10, column=1)

e11.grid(row=11, column=1)

e12.grid(row=12, column=1)

e13.grid(row=13, column=1)

Button (master, text='Predict', command=show\_entry\_fields). grid ()

mainloop()

**6. SYSTEM TESTING**

**6.1 OVERVIEW OF TESTING**

Software testing is a process that will be used to evaluate software quality. Software testing is a powerful technology test designed to provide participants with information about the quality of a tested product or service, depending on the context in which it is intended to operate. This information may include, but is not limited to, the application process or application for tracking errors. Quality is not absolute, it is the value of one person. It is for this reason that tests may not be fully detected by the accuracy of any software, and criticism of tests or comparisons that compare the status and behavior of the product is kept to a minimum. It is very important to ensure that software testing should be separated from a separate discipline, Software Quality Assurance (S. Q. A.), which covers all areas of business processes, not just tests.

## 6.2 TYPES OF TESTING METHODS

Software testing methods are basically classified into BLACK BOX TESTING and WHITE-BOX TESTING. These two methods can be used to explain the point that a test engineer takes when developing the test cases.

### 6.2.1 Black Box Testing

The black box test works with software as a black box without understanding internal behavior. It aims to test performance. Therefore, the tester enters the data and sees only the output of the test object. This standard of testing often requires that the full test cases provided to the inspector can simply confirm that of the given input, the amount of output (or character), is the same as the expected value stated in the test case. Methods of checking black boxes include: stock sorting, boundary analysis, all double checking, fuzz testing, model-based testing, matric tracking etc.

### 6.2.2 White Box Testing

The white box test, however, is when the tester achieves internal data structure, code and algorithms. White box check methods include creating tests to satisfy other coding methods. For example, a test designer can create a test to create that all statements in the system are made at least once. Some examples of white box tests are dating methods and error injection changes. The white box test includes all specific tests.

### Testing can be done on the following levels

**6.2.3 Unit Testing:**

Unit test tests for minimum component or module. Each unit (basic component) of the software has been tested to ensure that the detailed structure of the block is performed correctly. In an object-focused environment, this is usually done in the classroom, and testing of small units involves builders and destroyers.

### 6.2.4 Integration Testing

Integration testing reveals defects in interface and interactions between integrated components (modules). Largely continuous groups of tested software components related to building materials are compiled and tested till the software works as a system.

### 6.2.5 System Testing

System testing tests a fully integrated system to ensure it meets its requirements. System integration testing ensures that the system is integrated into any external or third-party programs defined in system requirements.

**7.RESULTS**

## Exploratory Data Analysis(EDA)

### Analysing Target Variable

y **=** dataset["target"]

​sns.countplot(y)

​target\_temp **=** dataset.target.value\_counts()

​print(target\_temp)

1 165

0 138

Name: target, dtype: int64



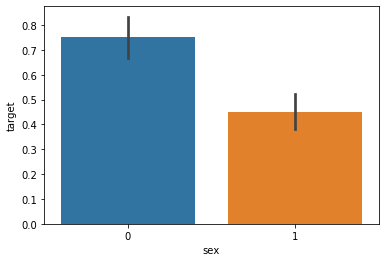
**Screenshot Of Result Showing the Number of Persons Having a Chance Of Getting**

**The Heart Disease**

##### We notice, that as expected, the 'sex' feature has 2 unique features

sns.barplot(dataset["sex"],y)

matplotlib.axes.\_subplots.AxesSubplot at 0x22f1416d1d0>



**Screenshot Showing Females Have More Chance to Get Heart Disease**

A picture containing background pattern

Description automatically generated

Chart, bar chart

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Chart, bar chart, box and whisker chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Chart

Description automatically generated

Text

Description automatically generated

**Chart

Description automatically generated**

**Chart

Description automatically generated**

Chart

Description automatically generated

Chart, histogram

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Chart, bar chart

Description automatically generated

**Figure 10: Comparison of Accuracy Rate of Various Machine Learning Algorithms**

**Output prediction on user input**

* To Verify the model, we are going to give user input to Predict the Disease.
* We have developed a Graphical User Interface (GUI) to read user input values
* After providing the user data it predicts whether the has a heart disease or not

**Output:**

* No Heart Disease
* Possibility of Heart Disease

**GUI PROGRAMING (Tkinter)**

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. All you need to do is perform the following steps

* Import the *Tkinter* module.
* Create the GUI application main window by using TK () method.
* Add one or more widgets to the GUI application. Here, we are using three widgets they are:
  + Displaying Text and Images with Label Widgets
  + Displaying Clickable Buttons with Button Widgets
  + Getting User Input with Entry Widgets
* Enter the main event loop to take action against each event triggered by the user.

Graphical user interface, table

Description automatically generated with medium confidence

**Output screen for person does not has a heart disease**

**Graphical user interface, table

Description automatically generated with medium confidence**

**Output screen for person who has a heart disease**

**8. CONCLUSION**

In this research, an intelligent machine-learning-based predictive system was proposed for the diagnosis of heart disease. We implemented an application using the Machine Learning Technique which can predict the vulnerability of a heart disease given basic attributes like age, gender, blood pressure, etc. Machine Learning algorithms such as the Enhanced decision tree are used to predict heart disease. Here, the Kaggle dataset is used to get more accurate results. The Enhanced Decision Tree classifier showed the best accuracy when compared with KNN and support vector machines. Due to the good performance of the Enhanced decision tree classifier with relief, it is a better predictive system in terms of accuracy. Furthermore, we have developed a Graphical User Interface (GUI) to take the input values from the users. After, taking the user input the model successfully predicts whether the person has heart disease or not and displays it on the screen. In the future, we will perform more experiments to increase the performance of these predictive classifiers for heart disease diagnosis by using other feature selection algorithms and optimizing techniques.

**9.REFERENCES**

1. Al Bashish, M. Braik, and S. Bani-Ahmed (2010), “A framework for detection and classification of plant leaf and stem diseases” proc. 2010 Int. Conf. Signal Image Process. ICSIP 2010, pp. 113-118
2. Camargo, A. and J. S. Smith. 2009. Image pattern classification for the identification of disease causing agents in plants". Com. Elect. Agr. 66: 121-125
3. H Sabrol and K Satish. “Tomato plant disease classification in digital images using classification tree”. In: Communication and Signal Processing (ICCSP), 2016 International Conference on. IEEE. 2016, pp. 1242–12
4. Umair Ayub, Syed Atif Moqurrab: Predicting Crop Diseases Using Data Mining Approaches: Classification. 2018 1st International Conference On Power, Energy And Smart Grid (Icpesg). Doi: 10.1109/Icpesg.2018.8384523.
5. Surampalli Ashok, Gemini Kishor: Tomato Leaf Disease Detection Using Deep Learning Techniques. Proceedings of the fifth International Conference on Communication and Electronics Systems(ICCES 2020).
6. SumitNema: Wheat Leaf Detection and Prevention Using Support Vector Machine. IEEE International Conference on Circuits and Systems in Digital Enterprise Technology 2018
7. Sandeep Kumar, KMVV Prasad: Leaf Disease Detection and Classification based on Machine Learning. International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE 2020)
8. Muhammad Shoaib: Deep learning-based segmentation and classification of leaf images for detection of tomato plant disease.
9. Abdulridha, J., Ampatzidis, Y., Kakarla, S. C., Roberts, P. (2020). Detection of target spot and bacterial spot diseases in tomato using UAV-based and benchtop-based hyperspectral imaging techniques. Precis. Agric. 21 (5), 955–978. DOI: 10.1007/s11119-019-09703-4
10. Agarwal, M., Singh, A., Arjaria, S., Sinha, A., Gupta, S. (2020). ToLeD: Tomato leafdisease detection using convolution neural network. Proc. Comput. Sci. 167 (2019), 293–301. doi: 10.1016/j.procs.2020.03.225
11. Choi, H., Jo, Y., Cho, W. K., Yu, J., Tran, P. T., Salaipeth, L., et al. (2020). Identification of viruses and viroids infecting tomato and pepper plants in Vietnam by metatranscriptomics. Int. J. Mol. Sci. 21 (20), 1–16. doi: 10.3390/ijms21207565
12. Md. Tariqul Islam: Plant Disease Detection using CNN Model and Image Processing. International Journal of Engineering Research &Technology (IJERT 2020).
13. M. Akila And P. Deepan : Detection And Classificationof Plant Leaf Diseases By Using Deep Learning Algorithm.(2018). International Journal Of Engineering Research & Technology (Ijert) Issn: 2278-0181 Published By, Www.Ijert.Org Iconnect - 2k18 Conference Proceedings
14. S. Phadikar And Et Al: Baye’s And Svm Classifier , Mean Fiitering Technique And Otsu’s Algorithm.
15. Umair Ayub, Syed Atif Moqurrab: Predicting Crop Diseases Using Data Mining Approaches: Classification. 2018 1st International Conference On Power, Energy And Smart Grid (Icpesg). Doi: 10.1109/Icpesg.2018.8384523.
16. Md. Selim Hossian, RokeyaMumtahanaMou: Recognition and Detection of Tea Leaf’s Diseases Using Support Vector Machine. 14th International Colloquium on Signal Processing & its Applications (CSPA 2018).
17. Jwo D. J., Chiu S. F., Gupta S., et al. Deep learning based automated detection of diseases from apple leaf images. Computers, Materials &Continua . 2022;71(1):1849–1866. DOI: 10.32604/cmc.2022.021875.