

# **DISEASE PREDICTION USING MACHINE LEARNING**

In Partial Fulfillment of the Requirements for the Degree

Of

## **BACHELOR OF TECHNOLOGY**

Submitted By:-

1. Md Sahil Zafar (19/IT/28)
2. Md Hasnain (19/IT/27)
3. Ramkrishna Patra (19/IT/42)

Supervised By:-

**Moumita Ghosh**

(Asst. Professor, Dept. of IT)

## **DEPARTMENT OF INFORMATION TECHNOLOGY**



## **HALDIA INSTITUTE OF TECHNOLOGY**

Haldia, Purba Medinipur, West Bengal – 721657, India

## ***PREFACE***

---

Engineering project plays a vital role in the progress of future engineers. Not only does it provide insights about the future concerned, it also bridges the gap between theory and practical knowledge. Project Report is an essential part of the **Project-I (PROJ-IT-781)** of the B.Tech course in Information Technology offered by Engineering or Technology Institutes affiliated to the Maulana Abul Kalam Azad University of Technology i.e. MAKAUT.

It is great opportunity for us to have the **Bachelor of Technology (B.Tech)** in **Department of Information Technology** at **Haldia Institute of Technology**, Haldia.

In the accomplishment of this degree, we are submitting our Project Report written on: **Disease Prediction Using Machine Learning**.

This project report has been prepared in fulfillment of our degree and has been carried out in seventh semester of our four year B.Tech course.

Subject to limitation of time, efforts and resources, every possible attempt has been made to study the problem deeply. Thus, we hope that this project will serve as a stepping stone for us in future and help us to carve a niche for ourselves in this field.

## **DECLARATION**

---

We hereby declare that the Project Report entitled on “**Disease Prediction Using Machine Learning**” submitted by us. This report submitted to the **Haldia Institute of Technology, Haldia** in partial fulfillment of the requirements for the degree of **B.Tech in Information Technology** under the guidance of Mrs. **Moumita Ghosh**.

We declare that the information and data given in the report is authentic to the best of our knowledge. We have adhered to all principles of academic honesty and integrity and have not mis-represented or fabricated or falsified any idea/data/fact source in our submission.

We also confirm that the report is only prepared for our academic requirements and not for any other purpose. We understood that any violation of the above will be cause for disciplinary action by the institute.

### **Name of the Students**

---

Md Hasnain(00219027)

---

Md Sahil Zafar(00219028)

---

Ramkrishna Patra(00219042)

**Date:** \_\_\_\_\_

## **ACKNOWLEDGEMENT**

---

We take this opportunity to express our sincere gratitude and respect to **Haldia Institute of Technology, Haldia** for providing us a platform to pursue our studies and carry out our final year project.

We would like to thank **Dr. Soumen Paul**, Professor and Head, Department of Information Technology, Haldia Institute of Technology, Haldia, who has been a constant support and encouragement throughout the course of this project.

We consider it a privilege and honor to express our sincere gratitude to our guide **Mrs. Moumita Ghosh**, Assistant Professor, Department of Information Technology, for the valuable guidance throughout the tenure of this review.

We also extend our thanks to all the faculty of **Information Technology** who directly or indirectly encouraged us.

Finally, we would like to thank our parents and friends for all their moral support they have given us during the completion of this work.

## *ABSTRACT*

---

It is a system which provides the user the information and tricks to take care of the health system of the user and it provides how to search out the disease using this prediction. Now a day's health industry plays major role in curing the diseases of the patients so this is often also some quite help for the health industry to inform the user and also it's useful for the user just in case he/she doesn't want to travel to the hospital or the other clinics, so just by entering the symptoms and every one other useful information the user can get to grasp the disease he/she is affected by and also the health industry may also get enjoy this method by just asking the symptoms from the stoner and entering within the system and in only many seconds they will tell the precise and over to some extent the accurate conditions. This Disease Prediction Using Machine Learning is totally through with the assistance of Machine Learning and Python programming language and also using the dataset that's available previously by the hospitals using that we are going to predict the diseases. Disease Prediction using Machine Learning is the system that is used to predict the diseases from the symptoms which are given by the patients or any user.

The system processes the symptoms provided by the user as input and gives the output as the probability of the disease. Naïve Bayes classifier is used in the prediction of the disease which is a supervised machine learning algorithm. The probability of the disease is calculated by the Naïve Bayes, Support Vector Machine and Random Forest algorithm. With an increase in biomedical and healthcare data, accurate analysis of medical data benefits early disease detection and patient care. By using Random Forest, Naïve Bayes we are predicting various diseases.

## ***TABLE OF CONTENTS***

---

<u>Topic</u>	<u>Page No.</u>
<b>1.Introduction</b>	<b>7</b>
<b>2.Survey of Technologies</b>	<b>9</b>
<b>3.Requirement and analysis</b>	<b>10</b>
<b>4. System Design</b>	<b>15</b>
<b>5. Algorithms</b>	<b>16</b>
<b>6. Dataset Preperation</b>	<b>33</b>
<b>7. Implementation</b>	<b>36</b>
<b>8. Conclusion</b>	<b>47</b>
<b>Future scope of project</b>	<b>48</b>
<b>References and Bibliography</b>	<b>49</b>

# Chapter 01: INTRODUCTION

There are many concerns about increasing reliance on technology. Nevertheless, as a society, we continue to push technology to new heights. From how we order food to how we provide healthcare, machine learning continue to help us surpass our wildest dreams. These are some advantages:

This is very important, especially if early detection and treatment can bring the best results. Using such an algorithm can literally save lives.

With the help of machine learning, the company is working to understand medical issues through crowdsourcin better. Having a larger and more diverse database can help the research be more accurate. In addition, these research methods become more accessible to people from marginalized communities who might otherwise not be able to participate. Finally, participating in research helps patients feel more capable while providing meaningful feedback.

## 1.1 OBJECTIVES:-

This article aims to contribute to the development of technologies related to **Machine Learning** applied to medicine by building a project where a neural network model can give a diagnosis from a chest x-ray image of a patient and explain its functioning as simply as possible. So, as we are going to deal mainly with lung-related issues, it's appropriate to know more about what they really are, in addition to their causes, effects, and history.

Respiratory illnesses are the ones that affect the respiratory system, which is responsible for the production of oxygen to feed the whole body. These illnesses are produced by infections, tobacco, smoke inhalation, and exposure to substances such as radon, asbestos... To this group of illnesses belongs illnesses such as asthma, pneumonia, tuberculosis, and Covid-19.

The first big epidemic belonging to this group of illnesses was the one produced by tuberculosis that affected the lungs. This epidemic was caused by the unacceptable labor conditions of the Industrial Revolution. This health problem was known lots of centuries before but was in that moment when it was first considered a huge health problem that provoked plenty of deaths and remarkable losses. Respiratory illnesses started being treated at the beginning of the XIX century with the invention of the stethoscope by the french doctor René Théophile Hyacinthe. Since that moment, the measures against this kind of illness have been divided into prevention (vaccines) and medical assistance to sick people.

## **1.2 PURPOSE AND SCOPE :**

### **1.2.1 Purpose**

With the continuous development and improvement of machine learning technology has also been extensively developed, which has promoted the development of computer vision, image processing, natural language processing, and other fields. This project aims to apply the ML Classification algorithm based on machine learning in the detection of diseases. This project introduces machine learning and related algorithms in detail and experiments like Naïve Bayes Algorithm based on machine learning. The experimental results show that Naïve Bayes Algorithm based on machine learning can well identify disease that are difficult to distinguish with the naked eye, with a recognition rate of up to 90%. Applying Naïve Bayes Algorithm based on machine learning in disease diagnosis has greatly improved the level of medical diagnosis.

### **1.2.2 Scope**

This Disease Prediction system can be used for urgent guidance on their illness according to the details and symptoms they will feed to the web-based application. Here, some intelligent data processing techniques are used to get the most accurate disease that would be related to the patient's details.



## CHAPTER 02: SURVEY OF TECHNOLOGIES

Now-a-days, people face various diseases due to the environmental condition and their living habits. So, the prediction of disease at earlier stage becomes important task. But the accurate prediction on the basis of symptoms becomes too difficult for doctor. [1] The correct prediction of disease is the most challenging task. To overcome this problem data mining plays an important role to predict the disease. Medical science has large amount of data growth per year. Due to increase amount of data growth in medical and healthcare field the accurate analysis on medical data which has been benefits from early patient care. With the help of disease data, data mining finds hidden pattern information in the huge amount of medical data. In this paper, they have proposed a general disease prediction based on symptoms of the patient. For the disease prediction, they used K-Nearest Neighbour (KNN) and Convolutional neural network (CNN) machine learning algorithm for accurate prediction of disease. For disease prediction required disease symptoms dataset. Ensemble classification technique is used in this model before prediction. In this general disease prediction, the living habits of person and check-up information consider for the accurate prediction. Machine Learning Algorithms Used: KNN (K-Nearest Neighbour), CNN (Convolutional neural network) Accuracy: CNN – 84.50%, KNN – 81.12%.

Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.[2] Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, agriculture, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks. A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers, but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. Some implementations of machine learning use data and neural networks in a way that mimics the working of a biological brain. In its application across business problems, machine learning is also referred to as predictive analytics.

## **CHAPTER 3: REQUIREMENT ANALYSIS**

### **3.1 Problem Definition :**

The primary goal is to develop a prediction system which will allow the users to check whether they have diabetes or heart disease sitting at home.[2] The user need not visit the doctor unless he has diabetes or heart disease, for further treatment. The prediction system requires a large dataset and efficient machine learning algorithms to predict the presence of the disease. Pre-processing the dataset to train the machine learning models, removing redundant, null, or invalid data for optimal performance of the prediction engine. Doctors rely on common knowledge for treatment. When common knowledge is lacking, studies are summarized after some number of cases have been studied. But this process takes time, whereas if machine learning is used, the patterns can be identified earlier. For using machine learning, a huge amount of data is required. There is very limited amount of data available depending on the disease. Also, the number of samples having no diseases is very high compared to the number of samples having the disease. This project is about performing two case studies to compare the performance of various machine learning algorithms to help identify such patterns in (i) and to create a platform for easier data sharing and collaboration. The primary aim of this project is to analyze the “Pima Indian Diabetes Dataset” and “Cleveland Heart Disease Dataset” and use Support Vector Machine, Naïve Bayes, and K-Nearest Neighbors for prediction. The secondary aim is to develop a web application that allows users to predict heart disease and diabetes utilizing the prediction engine.

### **3.2 Requirement specification:-**

#### **Data Collection:**

Data collection is defined as the procedure of collecting, measuring and analysing accurate insights for research using standard validated techniques. A researcher can evaluate their hypothesis based on collected data. In most cases, data collection is the primary and most important step for research, irrespective of the field of research. The approach of data collection is different for different fields of study, depending on the required information. The most critical objective of data collection is ensuring that information-rich and reliable data is collected for statistical analysis so that data-driven decisions can be made for research.

**Data Visualisation:**

Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

**Data Labelling:**

Supervised machine learning, which we'll talk about below, entails training a predictive model on historical data with predefined target answers. An algorithm must be shown which target answers or attributes to look for. Mapping these target attributes in a dataset is called labelling. Data labelling takes much time and effort as datasets sufficient for machine learning may require thousands of records to be labelled. For instance, if your image recognition algorithm must classify types of bicycles, these types should be clearly defined and labelled in a dataset.

**Data Selection :**

Data selection is defined as the process of determining the appropriate data type and source, as well as suitable instruments to collect data. Data selection precedes the actual practice of data collection. This definition distinguishes data selection from selective data reporting (selectively excluding data that is not supportive of a research hypothesis) and interactive/active data selection (using collected data for monitoring activities/events, or conducting secondary data analyses). The process of selecting suitable data for a research project can impact data integrity. After having collected all information, a data analyst chooses a subgroup of data to solve the defined problem. For instance, if you save your customers' geographical location, you don't need to add their cell phones and bank card numbers to a dataset. But purchase history would be necessary. The selected data includes attributes that need to be considered when building a predictive model.

**Data Pre-processing :**

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. The purpose of pre-processing is to convert raw data into a form that fits machine learning. Structured and clean data allows a data scientist to get more precise results from an applied machine learning model. The technique includes data formatting, cleaning, and sampling. Data formatting. The importance of data formatting grows when data is acquired from various sources by different people. The first task for a data scientist is to standardize record formats. A specialist checks whether variables representing each attribute are

recorded in the same way. Titles of products and services, prices, date formats, and addresses are examples of variables. The principle of data consistency also applies to attributes represented by numeric ranges. Data cleaning. This set of procedures allows for removing noise and fixing inconsistencies in data. A data scientist can fill in missing data using imputation techniques, e.g. substituting missing values with mean attributes. A specialist also detects outliers — observations that deviate significantly from the rest of distribution. If an outlier indicates erroneous data, a data scientist deletes or corrects them if possible. This stage also includes removing incomplete and useless data objects. Data anonymization. Sometimes a data scientist must anonymize or exclude attributes representing sensitive information (i.e. when working with healthcare and banking data). Data sampling. Big datasets require more time and computational power for analysis. If a dataset is too large, applying data sampling is the way to go. A data scientist uses this technique to select a smaller but representative data sample to build and run models much faster, and at the same time to produce accurate outcomes.

### **Data Transformation:**

Data transformation is the process of converting data from one format or structure into another format or structure. Data transformation is critical to activities such as data integration and data management. Data transformation can include a range of activities: you might convert data types, cleanse data by removing nulls or duplicate data, enrich the data, or perform aggregations, depending on the needs of your project. Scaling. Data may have numeric attributes (features) that span different ranges, for example, millimetres, meters, and kilometres. Scaling is about converting these attributes so that they will have the same scale, such as between 0 and 1, or 1 and 10 for the smallest and biggest value for an attribute. Decomposition. Sometimes finding patterns in data with features representing complex concepts is more difficult. Decomposition technique can be applied in this case. During decomposition, a specialist converts higher level features into lower level ones. In other words, new features based on the existing ones are being added. Decomposition is mostly used in time series analysis. For example, to estimate a demand for air conditioners per month, a market research analyst converts data representing demand per quarters. Aggregation. Unlike decomposition, aggregation aims at combining several features into a feature that represents them all. For example, you have collected basic information about your customers and particularly their age. To develop a demographic segmentation strategy, you need to distribute them into age categories, such as 16-20, 21-30, 31-40, etc. You use aggregation to create large-scale features based on smallscale ones. This technique allows you to reduce the size of a dataset without the loss of information.

### **Data Splitting:**

A dataset used for machine learning should be partitioned into three subsets — training, test, and validation sets. Training set. A data scientist uses a training set to train a model and define its

optimal parameters — parameters it must learn from data. Test set. A test set is needed for an evaluation of the trained model and its capability for generalization. The latter means a model's ability to identify patterns in new unseen data after having been trained over a training data. It is crucial to use different subsets for training and testing to avoid model overfitting, which is the incapacity for generalization we mentioned above. Validation set. The purpose of a validation set is to tweak a model's hyperparameters — higher-level structural settings that cannot be directly learned from data. These settings can express, for instance, how complex a model is and how fast it finds patterns in data. The proportion of a training and a test set is usually 80 to 20 percent, respectively. A training set is then split again, and its 20 percent will be used to form a validation set. At the same time, machine learning practitioner Jason Brownlee suggests using 66 percent of data for training and 33 percent for testing. A size of each subset depends on the total dataset size. The more training data a data scientist uses, the better the potential model will perform. Consequently, more results of model testing data lead to better model performance and generalization capability.

### **Modelling:**

After pre-processing the collected data and split it into three subsets, we can proceed with a model training.[3] This process entails “feeding” the algorithm with training data. An algorithm will process data and output a model that is able to find a target value (attribute) in new data — an answer you want to get with predictive analysis. The purpose of model training is to develop a model. Two model training styles are most common — supervised and unsupervised learning. The choice of each style depends on whether you must forecast specific attributes or group data objects by similarities. Model evaluation and testing: The goal of this step is to develop the simplest model able to formulate a target value fast and well enough. A data scientist can achieve this goal through model tuning. That is the optimization of model parameters to achieve an algorithm's best performance. One of the more efficient methods for model evaluation and tuning is cross-validation. Cross-validation: Cross-validation is the most used tuning method. It entails splitting a training dataset into ten equal parts (folds). A given model is trained on only nine folds and then tested on the tenth one (the one previously left out). Training continues until every fold is left aside and used for testing. As a result of model performance measure, a specialist calculates a cross-validated score for each set of hyperparameters. A data scientist trains models with different sets of hyperparameters to define which model has the highest prediction accuracy. The cross-validated score indicates average model performance across ten hold-out folds.

### **Model Deployment:**

Deployment is the method by which you integrate a machine learning model into an existing production environment to make practical business decisions based on data. It is one of the last stages in the machine learning life cycle and can be one of the most cumbersome. Often, an organization's IT systems are incompatible with traditional model-building languages, forcing data scientists and programmers to spend valuable time and brainpower rewriting them.

## **3.3 Software and Hardware Requirements :**

### **3.3.1 Hardware requirements:-**

- OS: Windows 10
- RAM: Minimum of 4GB

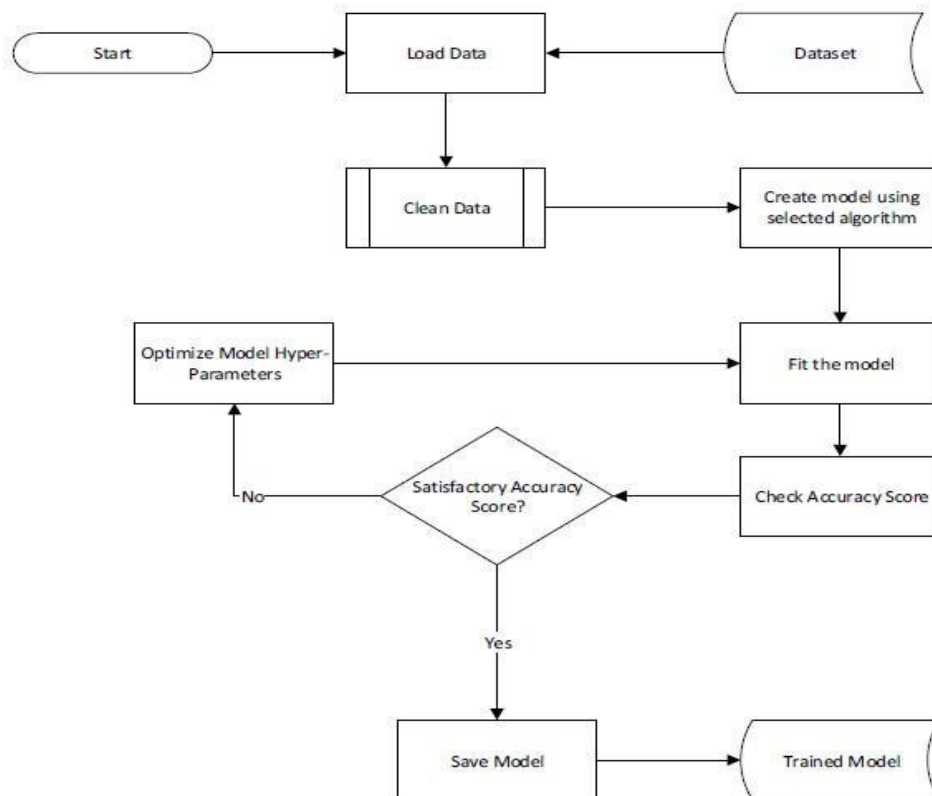
### **3.3.2 Software requirements:-**

- Basic Text-Editor: Visual Studio Code.
- Jupyter Notebook: Development of Python Scripts.

## CHAPTER 04 : SYSTEM DESIGN

### 4.1 Physical Design:

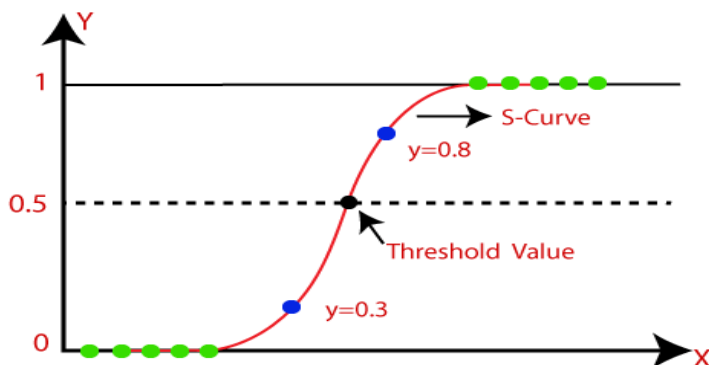
#### 4.1.1 Design Methodology



## CHAPTER 05 : ALGORITHMS

### 5.1 Logistic Regression:

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1.**
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems.**
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:





### 5.1.1 Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}; 0 \text{ for } y=0, \text{ and infinity for } y=1$$

- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$\log \left[ \frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

### 5.1.2 Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

- **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

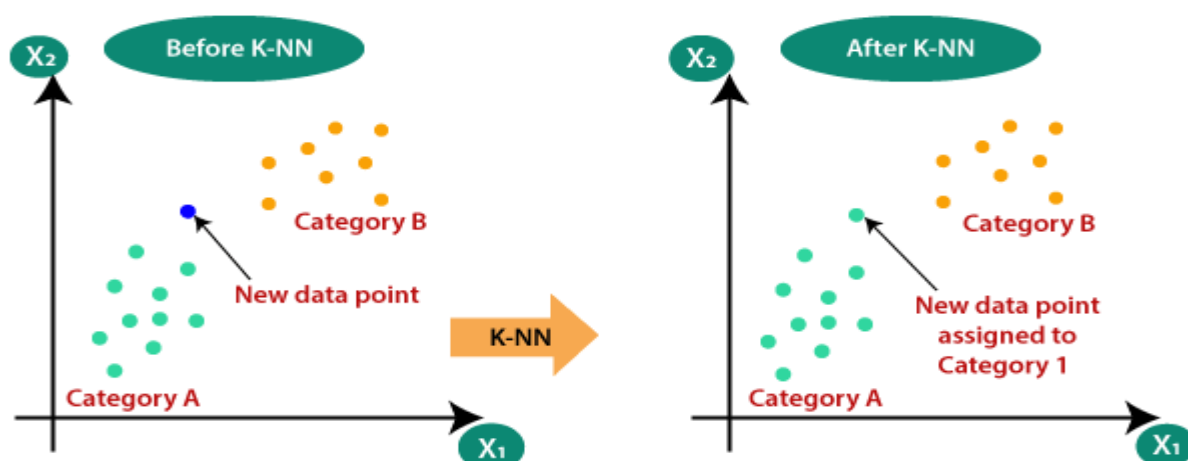
## 5.2 K Nearest Neighbors:

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN 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 well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
- Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.



### 5.2.1 Why do we need a K-NN Algorithm?

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point  $x_1$ , so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:

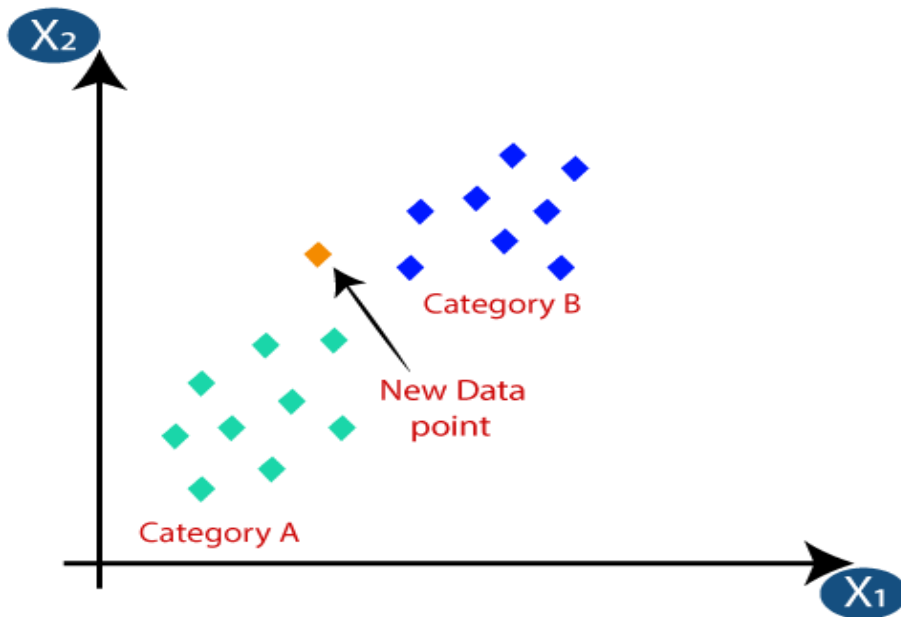


### 5.2.2 How does K-NN work?

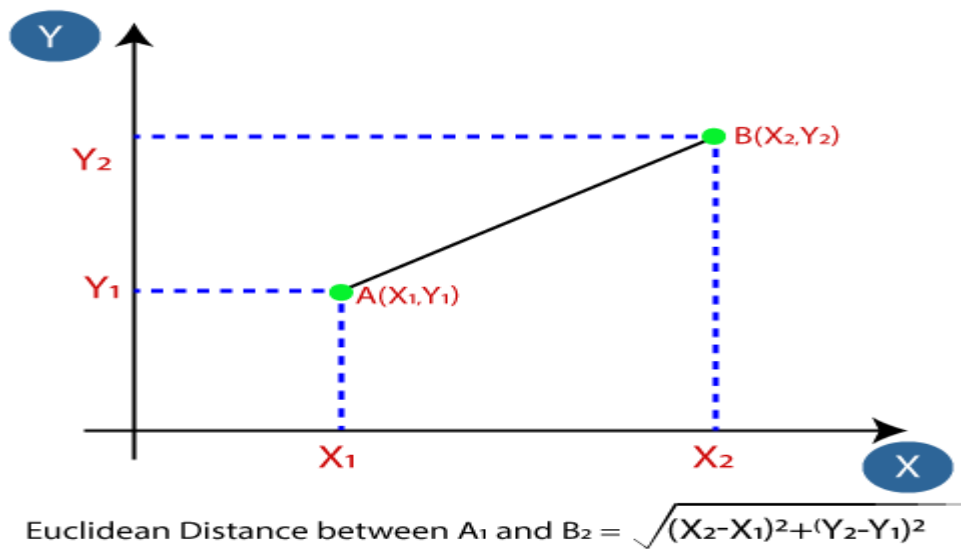
The K-NN working can be explained on the basis of the below algorithm:

- Step-1: Select the number  $K$  of the neighbors
- Step-2: Calculate the Euclidean distance of  $K$  number of neighbors
- Step-3: Take the  $K$  nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these  $k$  neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step-6: Our model is ready.

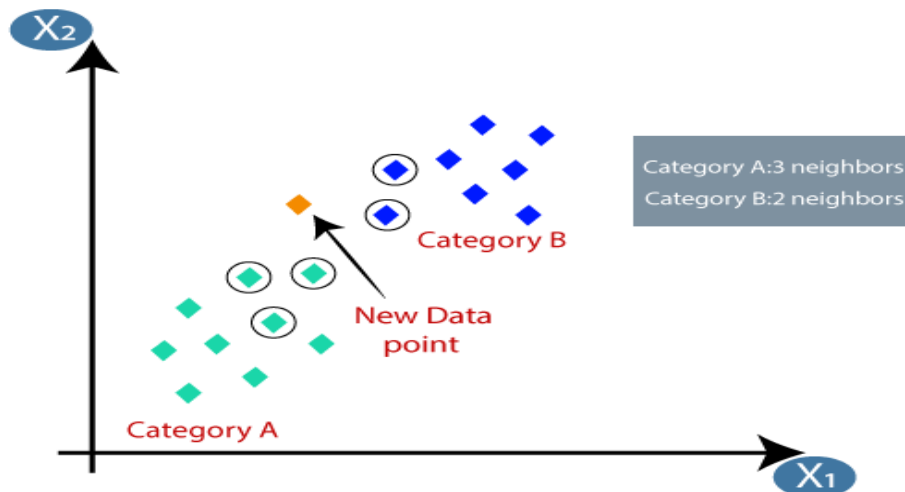
Suppose we have a new data point and we need to put it in the required category. Consider the below image:



- Firstly, we will choose the number of neighbors, so we will choose the  $k=5$ .
- Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:



### 5.2.3 How to select the value of K in the K-NN Algorithm?

Below are some points to remember while selecting the value of K in the K-NN algorithm:

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
- A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties.

### 5.2.4 Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

### 5.2.5 Disadvantages of KNN Algorithm:

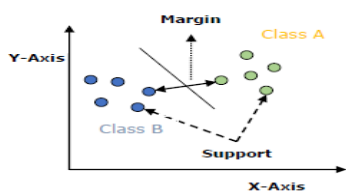
- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

## 5.3 Support Vector Classifier :

Support vector machines (SVMs) are powerful yet flexible supervised machine learning algorithms which are used both for classification and regression. But generally, they are used in classification problems. In 1960s, SVMs were first introduced but later they got refined in 1990. SVMs have their unique way of implementation as compared to other machine learning algorithms. Lately, they are extremely popular because of their ability to handle multiple continuous and categorical variables.

### 5.3.1 Working of SVM

An SVM model is basically a representation of different classes in a hyperplane in multidimensional space. The hyperplane will be generated in an iterative manner by SVM so that the error can be minimized. The goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH).



The followings are important concepts in SVM –

- ❖ **Support Vectors** – Datapoints that are closest to the hyperplane is called support vectors. Separating line will be defined with the help of these data points.
- ❖ **Hyperplane** – As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.
- ❖ **Margin** – It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin.

The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH) and it can be done in the following two steps –

- First, SVM will generate hyperplanes iteratively that segregates the classes in best way.
- Then, it will choose the hyperplane that separates the classes correctly.
- 
- **SVM Kernels**

In practice, SVM algorithm is implemented with kernel that transforms an input data space into the required form. [5] SVM uses a technique called the kernel trick in which kernel takes a low dimensional input space and transforms it into a higher dimensional space. In simple words, kernel converts non-separable problems into separable problems by adding more dimensions to it. It makes SVM more powerful, flexible and accurate. The following are some of the types of kernels used by SVM.

**Linear Kernel** : It can be used as a dot product between any two observations. The formula of linear kernel is as below –

$$K(x, x_i) = \sum (x * x_i)$$

From the above formula, we can see that the product between two vectors say  $x$  &  $x_i$  is the sum of the multiplication of each pair of input values.

**Polynomial Kernel** : It is more generalized form of linear kernel and distinguish curved or nonlinear input space. Following is the formula for polynomial kernel –

$$k(X, X_i) = 1 + \sum (X * X_i)^d$$

Here  $d$  is the degree of polynomial, which we need to specify manually in the learning algorithm.

**Radial Basis Function (RBF) Kernel** : RBF kernel, mostly used in SVM classification, maps input space in indefinite dimensional space. Following formula explains it mathematically –

$$K(x, x_i) = \exp(-\gamma * \sum (x - x_i)^2)$$

Here,  $\gamma$  ranges from 0 to 1. We need to manually specify it in the learning algorithm. A good default value of  $\gamma$  is 0.1. As we implemented SVM for linearly separable data, we can implement it in Python for the data that is not linearly separable. It can be done by using kernels.

## 5.4 Naïve Bayes

- It is mainly used in *text classification* that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.**
- Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles.**

### 5.4.1 Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- **Naïve:** It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes:** It is called Bayes because it depends on the principle of [Bayes' Theorem](#).

### 5.4.2 Bayes' Theorem:

- Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

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

Where,

**P(A|B) is Posterior probability:** Probability of hypothesis A on the observed event B.

**P(B|A) is Likelihood probability:** Probability of the evidence given that the probability of a hypothesis is true.



**P(A) is Prior Probability:** Probability of hypothesis before observing the evidence.

**P(B) is Marginal Probability:** Probability of Evidence.

#### 5.4.3 Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

#### 5.4.4 Disadvantages of Naïve Bayes Classifier:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

#### 5.4.5 Applications of Naïve Bayes Classifier:

- It is used for **Credit Scoring**.
- It is used in **medical data classification**.
- It can be used in **real-time predictions** because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as **Spam filtering** and **Sentiment analysis**.

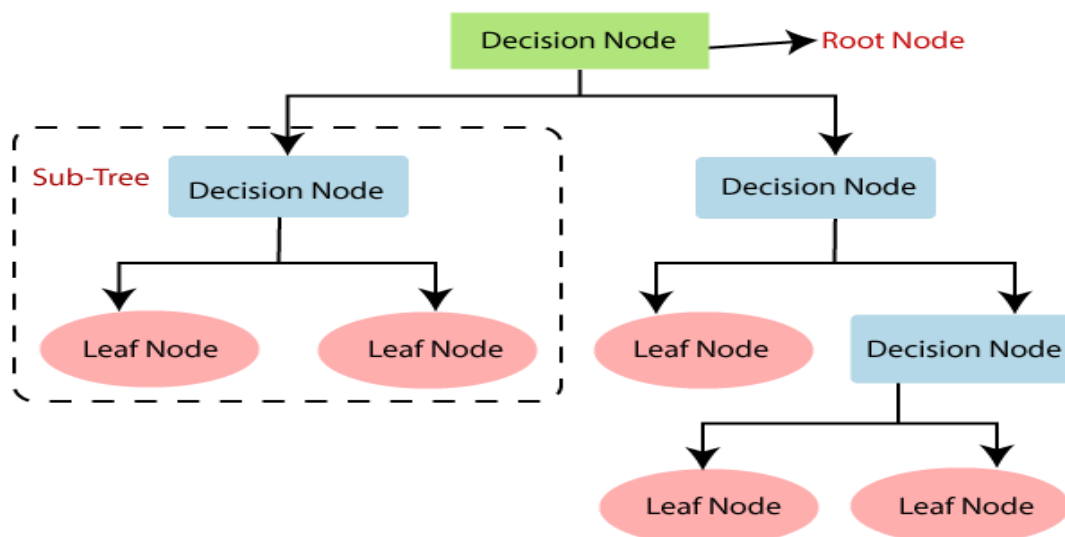
#### 5.4.6 Types of Naïve Bayes Model:

There are three types of Naive Bayes Model, which are given below:

- **Gaussian:** The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.
- **Multinomial:** The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics

## 5.5 Decision Tree

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- *It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.*
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:



### 5.5.1 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.

### 5.5.2 How does the Decision Tree algorithm Work?

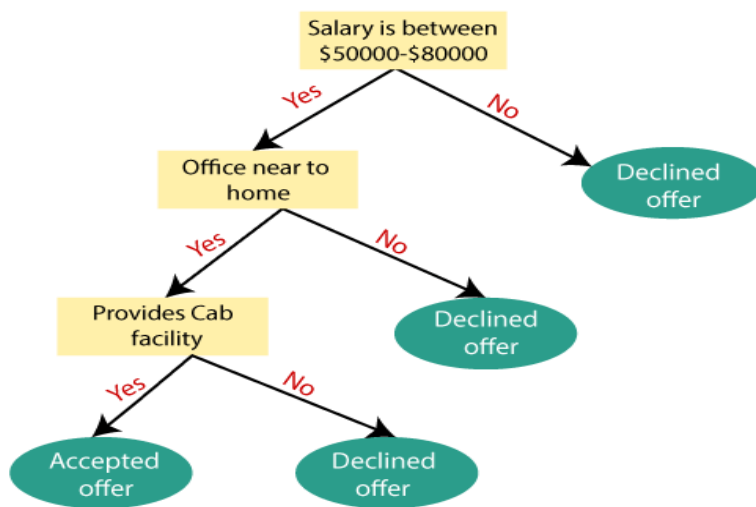
In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

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

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

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

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



### 5.5.3 Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM**. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- **Information Gain**
- **Gini Index**

#### 1. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

1.  $\text{Information Gain} = \text{Entropy}(S) - [(\text{Weighted Avg}) * \text{Entropy}(\text{each feature})]$

**Entropy:** Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

## 2. Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

$$\text{Gini Index} = 1 - \sum_j P_j^2$$

## Pruning: Getting an Optimal Decision tree

*Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.*

A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree **pruning** technology used:

- **Cost Complexity Pruning**
- **Reduced Error Pruning.**

#### 5.5.4 Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

#### 5.5.5 Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the **Random Forest algorithm**.
- For more class labels, the computational complexity of the decision tree may increase.

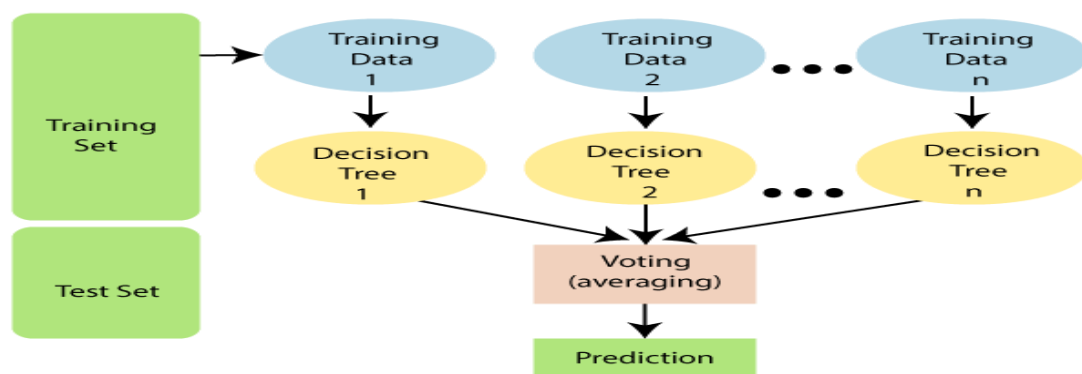
## 5.6 Random Forest Classifier :

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:



### Working Of Random Forest :

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

### **6.3.2 Applications :**

There are mainly four sectors where Random forest mostly used:

- ❖ **Banking:** Banking sector mostly uses this algorithm for the identification of loan risk.
- ❖ **Medicine:** With the help of this algorithm, disease trends and risks of the disease can be identified.
- ❖ **Land Use:** We can identify the areas of similar land use by this algorithm.
- ❖ **Marketing:** Marketing trends can be identified using this algorithm.



## **CHAPTER 06 : DATASET PREPERATION**

### **6.1 Data Gathering :**

Data preparation is the primary step for any machine learning problem. We will be using a dataset from Kaggle for this problem.

### **6.2 Data Cleaning :**

Cleaning is the most important step in a machine learning project. The quality of our data determines the quality of our machine learning model. So it is always necessary to clean the data before feeding it to the model for training. In our dataset all the columns are numerical, the target column i.e. prognosis is a string type and is encoded to numerical form using a label encoder.

### **6.3 Inference:**

After training the three models we will be predicting the disease for the input symptoms by combining the predictions of all three models. This makes our overall prediction more robust and accurate.

### **6.4 Dataset:**

The primary goal is to develop a prediction engine which will allow the users to check whether they have heart disease or not sitting at home. The user need not visit the doctor unless he has heart disease, for further treatment. The prediction engine requires a large dataset and efficient machine learning algorithms to predict the presence of the disease. Pre-processing the dataset to train the machine learning models, removing redundant, null or invalid data for optimal performance of the prediction engine.

The secondary aim is to develop a web application that allows users to predict heart disease and diabetes utilizing the prediction engine

## 6.5 Dataset Formulation

### 6.5.1 About Dataset

#### Context

This data set dates from 1988 and consists of four databases: Cleveland, Hungary, Switzerland, and Long Beach V. It contains 76 attributes, including the predicted attribute, but all published experiments refer to using a subset of 14 of them. The "target" field refers to the presence of heart disease in the patient. It is integer valued 0 = no disease and 1 = disease.

#### Content

Attribute Information:

1. age
2. sex
3. chest pain type (4 values)
4. resting blood pressure
5. serum cholestoral in mg/dl
6. fasting blood sugar > 120 mg/dl
7. resting electrocardiographic results (values 0,1,2)
8. maximum heart rate achieved
9. exercise induced angina
10. oldpeak = ST depression induced by exercise relative to rest
11. the slope of the peak exercise ST segment
12. number of major vessels (0-3) colored by flourosopy
13. thal: 0 = normal; 1 = fixed defect; 2 = reversable defect

The names and social security numbers of the patients were recently removed from the database, replaced with dummy values.

#### Dataset Link:

We have collected the dataset needed for our project from kraggle

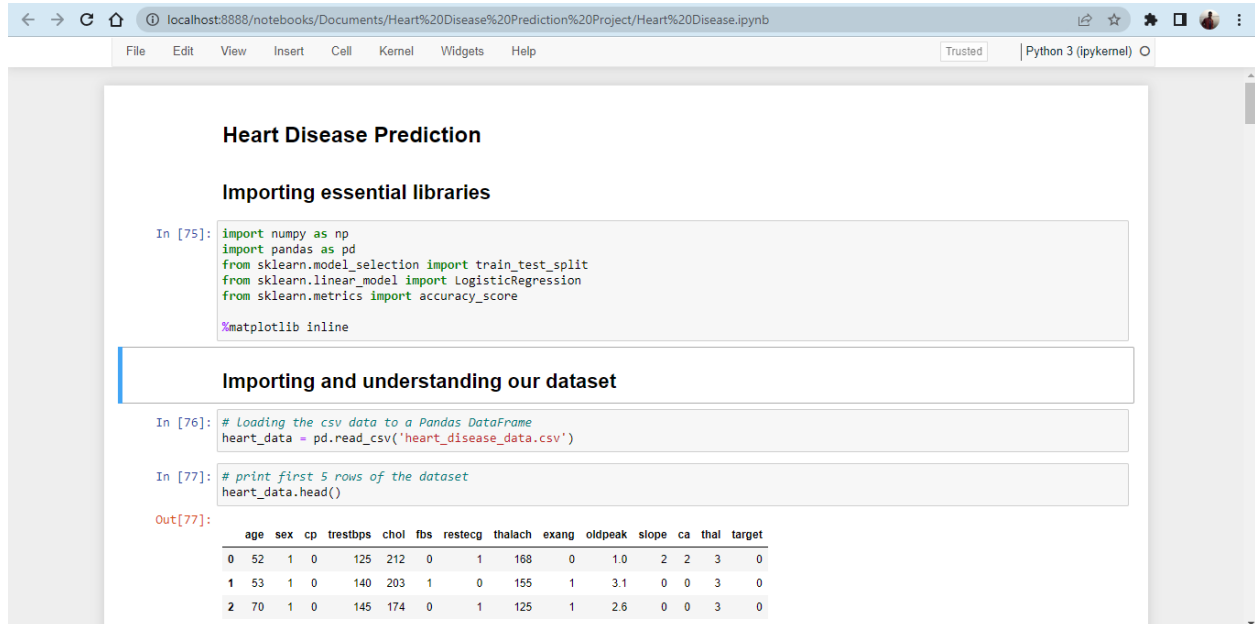
<https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset?resource=download>

The screenshot shows the Microsoft Excel interface with a dataset named 'heart\_disease\_data'. The dataset is displayed in a grid with 22 rows and 20 columns. The first column is labeled 'age', the second is 'sex', and the last is 'target'. The data represents heart disease risk factors and outcomes.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target						
2	52	1	0	125	212	0	1	168	0	1	2	2	3	0						
3	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0						
4	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0						
5	61	1	0	148	203	0	1	161	0	0	2	1	3	0						
6	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0						
7	58	0	0	100	248	0	0	122	0	1	1	0	2	1						
8	58	1	0	114	318	0	2	140	0	4.4	0	3	1	0						
9	55	1	0	160	289	0	0	145	1	0.8	1	1	3	0						
10	46	1	0	120	249	0	0	144	0	0.8	2	0	3	0						
11	54	1	0	122	286	0	0	116	1	3.2	1	2	2	0						
12	71	0	0	112	149	0	1	125	0	1.6	1	0	2	1						
13	43	0	0	132	341	1	0	136	1	3	1	0	3	0						
14	34	0	1	118	210	0	1	192	0	0.7	2	0	2	1						
15	51	1	0	140	298	0	1	122	1	4.2	1	3	3	0						
16	52	1	0	128	204	1	1	156	1	1	1	0	0	0						
17	34	0	1	118	210	0	1	192	0	0.7	2	0	2	1						
18	51	0	2	140	308	0	0	142	0	1.5	2	1	2	1						
19	54	1	0	124	266	0	0	109	1	2.2	1	1	3	0						
20	50	0	1	120	244	0	1	162	0	1.1	2	0	2	1						
21	58	1	2	140	211	1	0	165	0	0	2	0	2	1						
22	60	1	2	140	185	0	0	155	0	3	1	0	2	0						

# CHAPTER 07 : IMPLEMENTATION

## 7.1 Importing Libraries



**Heart Disease Prediction**

**Importing essential libraries**

```
In [75]: import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

%matplotlib inline
```

**Importing and understanding our dataset**

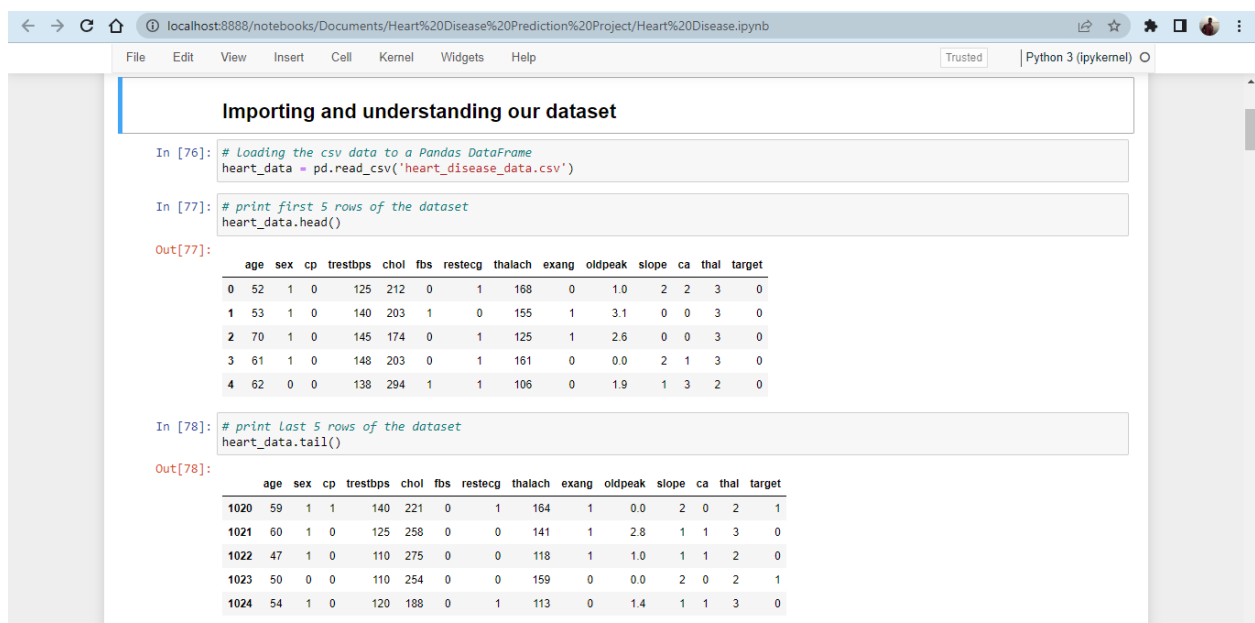
```
In [76]: # Loading the csv data to a Pandas DataFrame
heart_data = pd.read_csv('heart_disease_data.csv')
```

```
In [77]: # print first 5 rows of the dataset
heart_data.head()
```

Out[77]:

	age	sex	cp	trestbps	chol	fb	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	64	1	0	140	203	0	1	161	0	0.0	2	1	3	0

## 7.2 Importing Our Dataset



**Importing and understanding our dataset**

```
In [76]: # Loading the csv data to a Pandas DataFrame
heart_data = pd.read_csv('heart_disease_data.csv')
```

```
In [77]: # print first 5 rows of the dataset
heart_data.head()
```

Out[77]:

	age	sex	cp	trestbps	chol	fb	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

```
In [78]: # print Last 5 rows of the dataset
heart_data.tail()
```

Out[78]:

	age	sex	cp	trestbps	chol	fb	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

## 7.3 Exploring The Dataset

```
← → ↺ 🏠 localhost:8888/notebooks/Documents/Heart%20Disease%20Prediction%20Project/Heart%20Disease.ipynb
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Exploring The Dataset

In [7]: # number of rows and columns in the dataset
heart_data.shape

Out[7]: (1025, 14)

In [8]: # getting some info about the data
heart_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  ---
0    age         1025 non-null   int64
1    sex         1025 non-null   int64
2    cp          1025 non-null   int64
3    trestbps    1025 non-null   int64
4    chol        1025 non-null   int64
5    fbs         1025 non-null   int64
6    restecg     1025 non-null   int64
7    thalach     1025 non-null   int64
8    exang       1025 non-null   int64
9    oldpeak     1025 non-null   float64
10   slope       1025 non-null   int64
11   ca          1025 non-null   int64
12   thal        1025 non-null   int64
13   target      1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB

In [10]: heart_data.describe()
```

```
← → ↺ 🏠 localhost:8888/notebooks/Documents/Heart%20Disease%20Prediction%20Project/Heart%20Disease.ipynb
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [10]: heart_data.describe()

Out[10]:
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.00
mean	54.434146	0.695610	0.942439	131.611707	246.000000	0.149268	0.529756	149.114146	0.336585	1.071512	1.385366	0.75
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.005724	0.472772	1.175053	0.617755	1.03
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.00
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.000000	0.000000	0.000000	1.000000	0.00
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.000000	0.000000	0.800000	1.000000	0.00
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.000000	1.000000	1.800000	2.000000	1.00
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.00

```

In [81]: # checking for missing values
heart_data.isnull().sum()

Out[81]:
age      0
sex      0
cp       0
trestbps 0
chol     0
fbs      0
restecg  0
thalach  0
exang    0
oldpeak  0
slope    0
ca       0
thal     0
```

## 7.4 Checking For Missing Value

```
localhost:8888/notebooks/Documents/Heart%20Disease%20Prediction%20Project/Heart%20Disease.ipynb
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (pykernel)

In [81]: # checking for missing values
heart_data.isnull().sum()

Out[81]: age      0
sex        0
cp         0
trestbps   0
chol       0
fbs        0
restecg    0
thalach    0
exang      0
oldpeak    0
slope      0
ca         0
thal       0
target     0
dtype: int64

In [82]: # statistical measures about the data
heart_data.describe()

Out[82]:
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.00	
mean	54.434146	0.695610	0.942439	131.611707	246.000000	0.149268	0.529756	149.114146	0.336585	1.071512	1.385366	0.75
std	9.072290	0.460373	1.029641	17.516718	51.592511	0.356527	0.527878	23.005724	0.472772	1.175053	0.617755	1.03
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.00
25%	48.000000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	132.000000	0.000000	0.000000	1.000000	0.00
50%	56.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	152.000000	0.000000	0.800000	1.000000	0.00
75%	61.000000	1.000000	2.000000	140.000000	275.000000	0.000000	1.000000	166.000000	1.000000	1.800000	2.000000	1.00

localhost:8888/notebooks/Documents/Heart%20Disease%20Prediction%20Project/Heart%20Disease.ipynb

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
In [83]: # checking the distribution of Target Variable
heart_data['target'].value_counts()
```

```
Out[83]: 1    526
         0    499
         Name: target, dtype: int64
```

```
In [84]: X = heart_data.drop(columns='target', axis=1)
         Y = heart_data['target']
```

```
In [85]: print(X)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
0	52	1	0	125	212	0	1	168	0	1.0	
1	53	1	0	140	203	1	0	155	1	3.1	
2	70	1	0	145	174	0	1	125	1	2.6	
3	61	1	0	148	203	0	1	161	0	0.0	
4	62	0	0	138	294	1	1	106	0	1.9	
...	...	...	...	...	...	...	...	...	...	...	
1020	59	1	1	140	221	0	1	164	1	0.0	
1021	60	1	0	125	258	0	0	141	1	2.8	
1022	47	1	0	110	275	0	0	118	1	1.0	
1023	50	0	0	110	254	0	0	159	0	0.0	
1024	54	1	0	120	188	0	1	113	0	1.4	

	slope	ca	thal
0	2	2	3
1	0	0	3
2	0	0	3
3	2	1	3
4	1	3	2
...	...	...	...
1020	2	0	2
1021	1	1	3
1022	1	1	3

```

localhost:8888/notebooks/Documents/Heart%20Disease%20Prediction%20Project/Heart%20Disease.ipynb
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

1020 59 1 1 140 221 0 1 164 1 0.0
1021 60 1 0 125 258 0 0 141 1 2.8
1022 47 1 0 110 275 0 0 118 1 1.0
1023 50 0 0 110 254 0 0 159 0 0.0
1024 54 1 0 120 188 0 1 113 0 1.4

      slope ca thal
0         2 2 3
1         0 0 3
2         0 0 3
3         2 1 3
4         1 3 2
...      ... ..
1020      2 0 2
1021      1 1 3
1022      1 1 2
1023      2 0 2
1024      1 1 3

[1025 rows x 13 columns]

In [86]: print(Y)
0      0
1      0
2      0
3      0
4      0
..
1020    1
1021    0
1022    0
1023    1
1024    0
Name: target, Length: 1025, dtype: int64

```

## 7.5 Train Test Split

```

1      0
2      0
3      0
4      0
..
1020    1
1021    0
1022    0
1023    1
1024    0
Name: target, Length: 1025, dtype: int64

Train Test split

In [87]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_state=2)

In [88]: print(X.shape, X_train.shape, X_test.shape)
(1025, 13) (820, 13) (205, 13)

```

## 7.6 Logistic Regression

### Logistic Regression

```
In [89]: model = LogisticRegression()

In [90]: # training the LogisticRegression model with Training data
model.fit(X_train, Y_train)
Y_pred_lr = model.predict(X_test)

C:\Users\user\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear\_model.html#logistic-regression
n_iter_1 = _check_optimize_result(

In [91]: Y_pred_lr.shape
Out[91]: (285,)

In [92]: score_lr = round(accuracy_score(Y_pred_lr, Y_test)*100,2)
print("The accuracy score achieved using Logistic Regression is: "+str(score_lr)+" %")

The accuracy score achieved using Logistic Regression is: 80.49 %
```





## 7.7 Naïve Bayes

### Naive Bayes

```
In [93]: from sklearn.naive_bayes import GaussianNB
         nb = GaussianNB()
         nb.fit(X_train, Y_train)
         Y_pred_nb = nb.predict(X_test)

In [94]: Y_pred_nb.shape
Out[94]: (205,)

In [95]: score_nb = round(accuracy_score(Y_pred_nb, Y_test)*100,2)
         print("The accuracy score achieved using Naive Bayes is: "+str(score_nb)+" %")
The accuracy score achieved using Naive Bayes is: 78.05 %

In [96]: from sklearn.neighbors import KNeighborsClassifier
         knn = KNeighborsClassifier(n_neighbors=7)
         knn.fit(X_train, Y_train)
         Y_pred_knn=knn.predict(X_test)
```



## 7.8 KNN

### K Nearest Neighbors

```
In [97]: from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(X_train, Y_train)
Y_pred_knn=knn.predict(X_test)

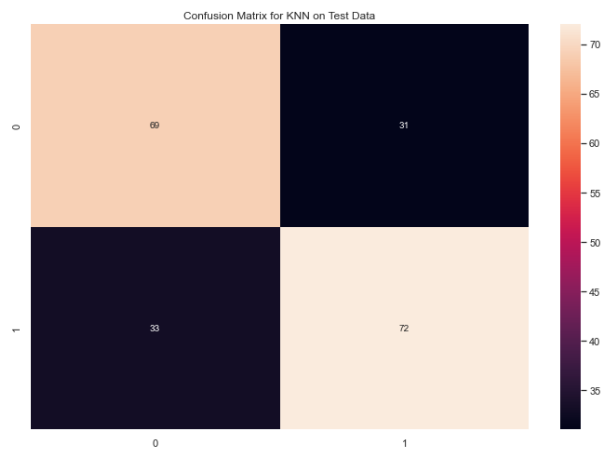
In [98]: Y_pred_knn.shape
Out[98]: (205,)

In [99]: 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: 68.78 %
```

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (ipykernel)

```
In [45]: cf_matrix = confusion_matrix(Y_test, Y_pred_knn)
plt.figure(figsize=(12,8))
sns.heatmap(cf_matrix, annot=True)
plt.title("Confusion Matrix for KNN on Test Data")
plt.show()
```



## 7.9 SVM

### SVM

```
In [100]: from sklearn import svm  
sv = svm.SVC(kernel='linear')  
sv.fit(X_train, Y_train)  
Y_pred_svm = sv.predict(X_test)
```

```
In [101]: Y_pred_svm.shape
```

```
Out[101]: (205,)
```

```
In [102]: 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: 82.44 %
```

File Edit View Insert Cell Kernel Widgets Help

Not Trusted

Python 3 (ipykernel) O

```
In [46]: cf_matrix = confusion_matrix(Y_test, Y_pred_svm)  
plt.figure(figsize=(12,8))  
sns.heatmap(cf_matrix, annot=True)  
plt.title("Confusion Matrix for SVM on Test Data")  
plt.show()
```



## 7.10 Decision Tree

```
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel) O
```

**Decision Tree**

```
In [103]: 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

#print(max_accuracy)
#print(best_x)

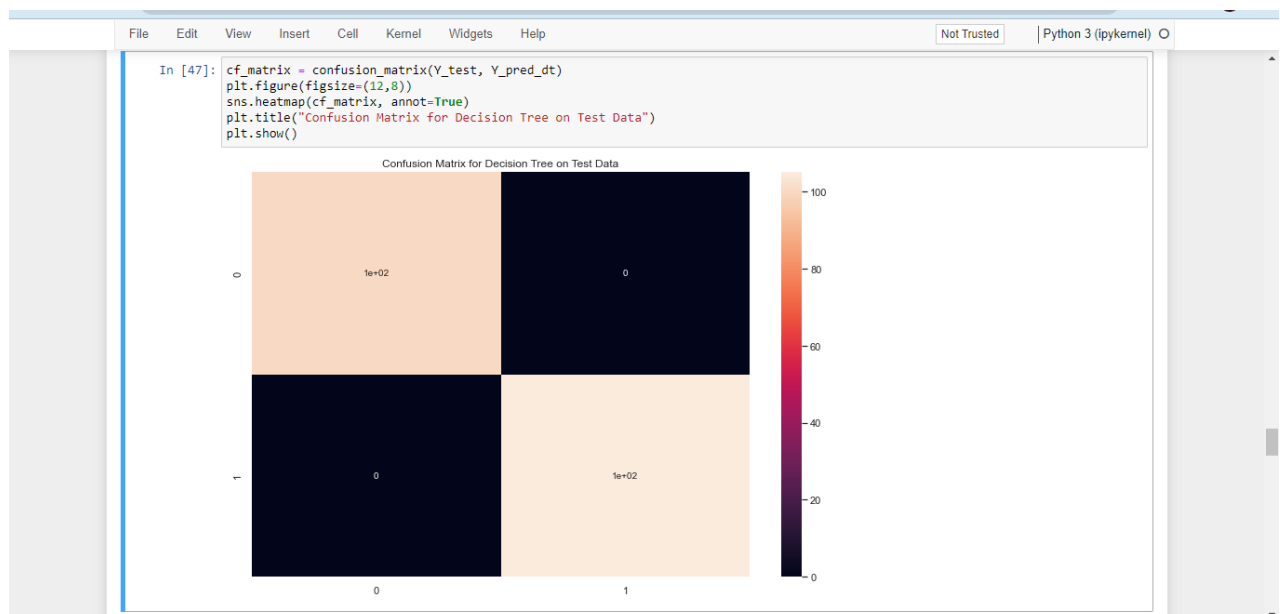
dt = DecisionTreeClassifier(random_state=best_x)
dt.fit(X_train,Y_train)
Y_pred_dt = dt.predict(X_test)

In [104]: print(Y_pred_dt.shape)
(205,)
```

```
In [105]: 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: 100.0 %
```



## 7.11 Accuracy Scores

### Accuracy Scores of all the algorithms

```
In [106]: scores = [score_lr,score_nb,score_svm,score_knn,score_dt]
          algorithms = ["Logistic Regression","Naive Bayes","Support Vector Machine","K-Nearest Neighbors","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 Logistic Regression is: 80.49 %
The accuracy score achieved using Naive Bayes is: 78.05 %
The accuracy score achieved using Support Vector Machine is: 82.44 %
The accuracy score achieved using K-Nearest Neighbors is: 68.78 %
The accuracy score achieved using Decision Tree is: 100.0 %
```



## 7.12 Saving the model

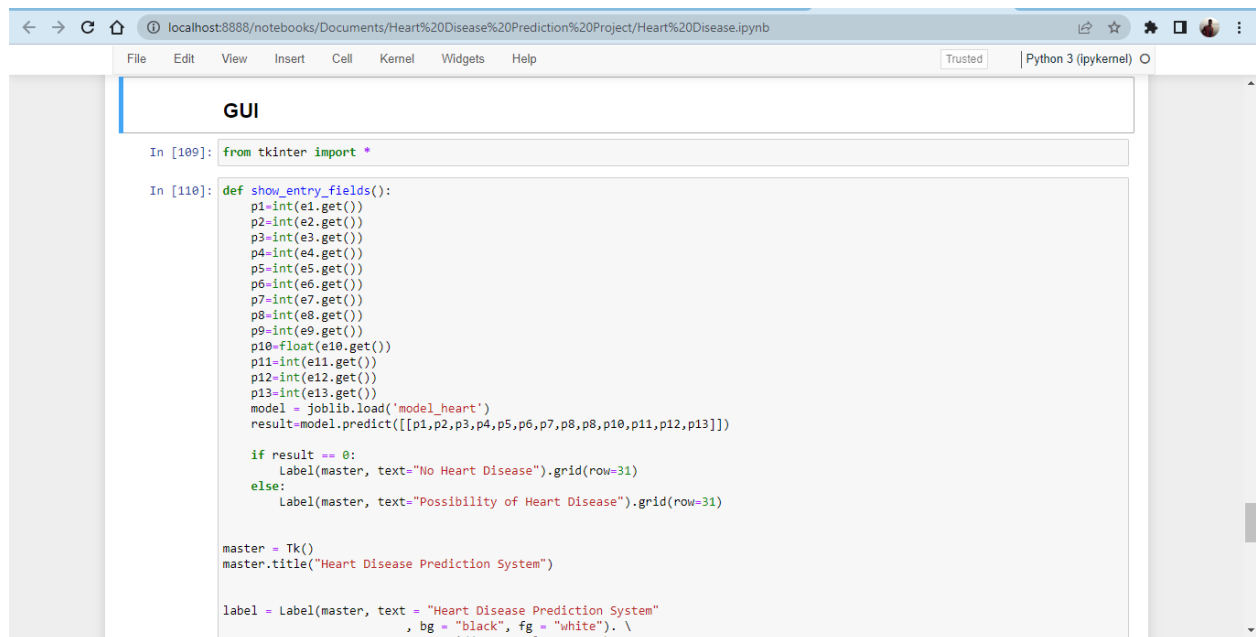
### Saving The Model

```
In [107]: import joblib

In [108]: joblib.dump(dt,'model_heart')

Out[108]: ['model_heart']
```

## 7.13 GUI



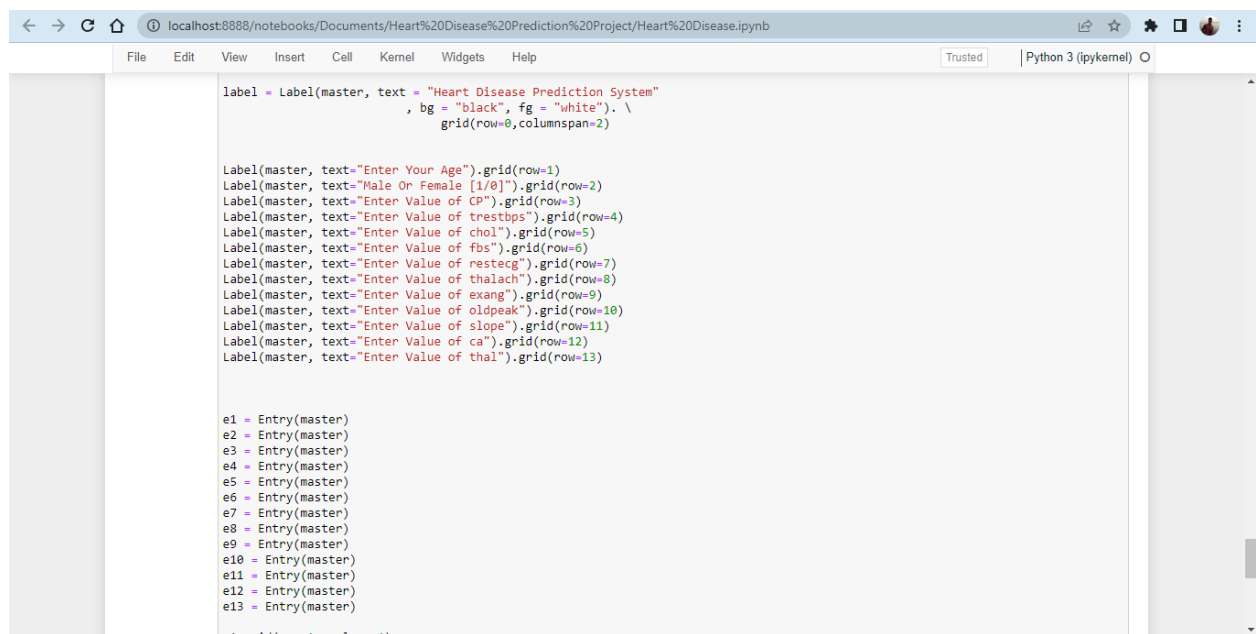
```
In [109]: from tkinter import *

In [110]: 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_heart')
    result=model.predict([[p1,p2,p3,p4,p5,p6,p7,p8,p9,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"). \
```



```
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)
```

```
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel) O

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()

C:\Users\user\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
C:\Users\user\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
C:\Users\user\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
C:\Users\user\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

# User Input:

Heart Disease Prediction System

Heart Disease Prediction System

Enter Your Age	19
Male Or Female [1/0]	1
Enter Value of CP	1
Enter Value of trestbps	150
Enter Value of chol	234
Enter Value of fbs	0
Enter Value of restecg	1
Enter Value of thalach	178
Enter Value of exang	0
Enter Value of oldpeak	3.2
Enter Value of slope	2
Enter Value of ca	0
Enter Value of thal	3

Predict

Possibility of Heart Disease

Heart Disease Prediction System

Heart Disease Prediction System

Enter Your Age	61
Male Or Female [1/0]	1
Enter Value of CP	0
Enter Value of trestbps	148
Enter Value of chol	203
Enter Value of fbs	0
Enter Value of restecg	1
Enter Value of thalach	161
Enter Value of exang	0
Enter Value of oldpeak	0.0
Enter Value of slope	2
Enter Value of ca	1
Enter Value of thal	3

Predict

Pos: No Heart Disease ease



Heart Disease Prediction System

Heart Disease Prediction System

Enter Your Age	19
Male Or Female [1/0]	1
Enter Value of CP	1
Enter Value of trestbps	125
Enter Value of chol	201
Enter Value of fbs	0
Enter Value of restecg	1
Enter Value of thalach	158
Enter Value of exang	0
Enter Value of oldpeak	1.0
Enter Value of slope	2
Enter Value of ca	2
Enter Value of thal	4
Predict	

Possibility of Heart Disease

Heart Disease Prediction System

Heart Disease Prediction System

Enter Your Age	57
Male Or Female [1/0]	0
Enter Value of CP	0
Enter Value of trestbps	189
Enter Value of chol	203
Enter Value of fbs	0
Enter Value of restecg	1
Enter Value of thalach	161
Enter Value of exang	0
Enter Value of oldpeak	0.0
Enter Value of slope	2
Enter Value of ca	1
Enter Value of thal	3
Predict	

Pos: No Heart Disease ease

## CHAPTER 06: CONCLUSION

This project aims to predict the disease on the basis of the symptoms. The project is designed in such a way that the system takes symptoms from the user as input and produces output i.e. predict diseases. In conclusion, for disease risk modeling, the accuracy of risk prediction depends on the diversity feature of the hospital data. The Prediction engine provides an optimal performance compared to other state of art approaches. The Prediction Engine makes use of three algorithms to predict the presence of a disease namely: Support Vector Machine (SVM), Random Forest and Naïve Bayes. The reason to choose these three algorithms are:

- They are effective, if the training data is large.
- A single dataset can be provided as an input to all these 3 algorithms with minimal or no modification.
- A common scalar can be used to normalize the input provided to these 3 algorithms.

## **Future Scope of the Project**

- To enhance the functionality of the prediction engine providing the details of 5 nearest hospitals or medical facilities to the user input location.
- Provide a user account which allows the user to keep track of their medical test data and get suggestions or support to meet the right specialists or the tests to be taken
- Provide admin controls to upload, delete the dataset which will be used to train the model.
- Automate the process of training the model and extracting pickle files of the trained models which will be consumed by the API's to predict the disease.
- Mail the detailed report of the prediction engine results along with the information of 5 nearest medical facilities details having location and contact information

## REFERENCES & BIBLIOGRAPHY

---

- [1] “Dhiraj Dahiwade, Gajanan Patle, Ektaa Meshram. Designing disease prediction model using machine learning approach. 3<sup>rd</sup> International Conference on Computing Methodologies and communication. 2019 August 29.” <https://ieeexplore.ieee.org/abstract/document/8819782>
- [2] “Shahadat Uddin, Arif Khan, Md Ekramul Hossain, Md Ali Moni. Comparing different supervised machine learning algorithms for disease prediction. BMC Medical Informatics & Decision Making. 2019 December 21.” <https://link.springer.com/article/10.1186/s12911-019-1004-8>
- [3] “Senthilkumar Mohan, Chandrasegar Thirumalai, Gautam Srivastava. Effective Heart Disease Prediction Using Hybrid Machine Learning Techniques. IEEE Access ( Volume: 7). 2019 June 19.” <https://ieeexplore.ieee.org/abstract/document/8740989>
- [4] “K Pingale, Sushant Surwase, Vaibhav Kulkarni, Saurabh Sarage, Abhijeet Karve. Disease prediction using machine learning. International Research Journal of Engineering and Technology 2019 October 9.” [https://d1wqtxts1xzle7.cloudfront.net/61714216/IRJET-V6I1212220200108-1822-1wxmovz-libre.pdf?1578481824=&response-content-disposition=inline%3B+filename%3DIRJET\\_Disease\\_Prediction\\_using\\_Machine\\_L.pdf&Expires=1670838529&Signature=Xxwbv4FSjK9hmr9poNG-x0c0p24](https://d1wqtxts1xzle7.cloudfront.net/61714216/IRJET-V6I1212220200108-1822-1wxmovz-libre.pdf?1578481824=&response-content-disposition=inline%3B+filename%3DIRJET_Disease_Prediction_using_Machine_L.pdf&Expires=1670838529&Signature=Xxwbv4FSjK9hmr9poNG-x0c0p24)
- [5] “Mehrbakhsh Nilashi, Othman bin Ibrahim, Hossein Ahmadi, Leila Shahmoradian. An analytical method for diseases prediction using machine learning techniques. Computers & Chemical Engineering Volume 106. 2017 November 2.” <https://www.sciencedirect.com/science/article/abs/pii/S0098135417302570>