Symptom Based Disease Prediction using Multiple Models

Anusha Mettu

# INTRODUCTION

The primary objective of this project is to develop a user-friendly model that can accurately predict a disease based on the symptoms provided by the user. To achieve this, the model will utilize advanced machine learning algorithms that will be run simultaneously for the same set of symptoms. This will improve the accuracy of disease detection, as symptoms can vary from person to person even for the same illness. Moreover, the project will include the development of a customized user interface that will allow the user to input their symptoms and select their preferred algorithm for disease detection. Ultimately, the project aims to provide a reliable and accessible tool that can serve as an initial step in the diagnosis of diseases based on the symptoms presented.

# PROBLEM STATEMENT

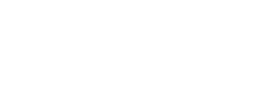
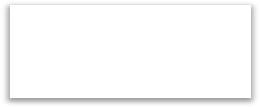
The healthcare industry has grown into a multibillion-dollar enterprise. The healthcare business generates vast amounts of health-care data on a regular basis, which can be utilized to extract information for forecasting sickness that may occur to a patient in the future utilizing treatment history and health data. This hidden information in healthcare data will be used afterwards to make affective health decisions for patients. This sector also needs to be improved by utilizing informative data in healthcare. So here is one of the methods for analyzing and predicting the disease from their symptoms using machine learning.

# OBJECTIVE

The primary objective of this project is to develop a user-friendly model that can accurately predict a disease based on the symptoms provided by the user. To achieve this, the model will utilize advanced machine learning algorithms that will be run simultaneously for the same set of symptoms. This will improve the accuracy of disease detection, as symptoms can vary from person to person even for the same illness. Moreover, the project will include the development of a customized user interface that will allow the user to input their symptoms and select their preferred algorithm for disease detection. Aims to provide a reliable and accessible tool that can serve as an initial step in the diagnosis of diseases based on the symptoms presented.

# FRAMEWORK

In the proposed system we are trying to predict multiple diseases with the symptoms as input from the user. However not all users have same symptoms from a disease so here we are using multiple algorithms to predict the disease.



**Data**

**Gathering**

**Data**

**Cleaning**

**Algorithm**

**models**

**Indepth analysis**

**of algorithms**

**Predictions**

**GUI Building**

# IMPLEMENTATION:

The given dataset comprises of binary data, represented by 0s and 1s. Therefore, it is imperative to eliminate any missing values from the dataset to ensure its accuracy. Following this, the data labels are transformed into numerical format, and the dataset is segregated into two parts, namely training and testing.

Specifically, 80% of the dataset is allocated for training, and the remaining 20% is reserved for testing purposes.

To construct our model, we have employed the K-fold cross-validation technique along with popular machine learning algorithm such as Decision Tree.

To construct our model, we have employed the K-fold cross-validation technique along with popular machine learning algorithms such as Decision Tree, Random Forest, and Naive Bayes classifier.

To facilitate the process of disease prediction, we have developed a Python GUI that accepts users' symptoms as input, and then machine learning algorithms are employed to predict the possible disease based on the given symptoms.

# DATA SET COLLECTION:

We have divided our data into two sets. One is a training data set and the other one is a testing data set.

In our data set we have 133 rows. In this we will have 132 rows that collect data regarding the symptoms of disease and the last column explains the type of disease.



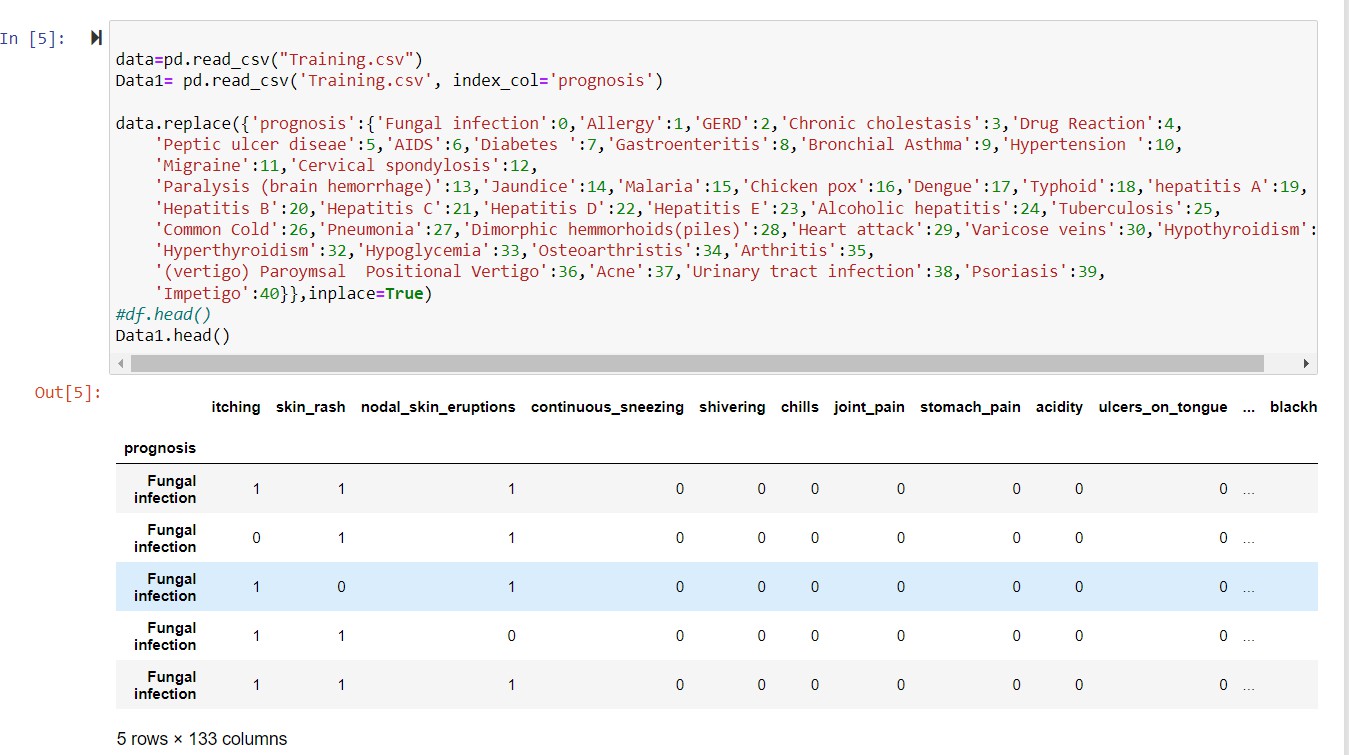
So here we have 132 attributes, and each attribute is a symptom of a particular disease. And a particular set of symptoms leads to a specific disease.

Similarly, we have 41 number of rows that explain the different possible types of disease that may occur due to a particular set of symptoms.



### Read data set:

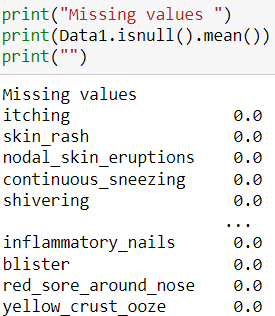
Displaying the symptoms and the diseases caused by the set of symptoms.



# DATA CLEANING

### Missing data:

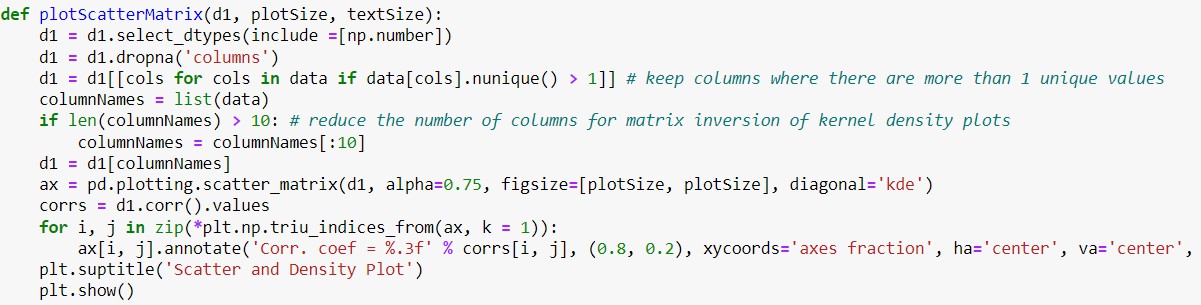
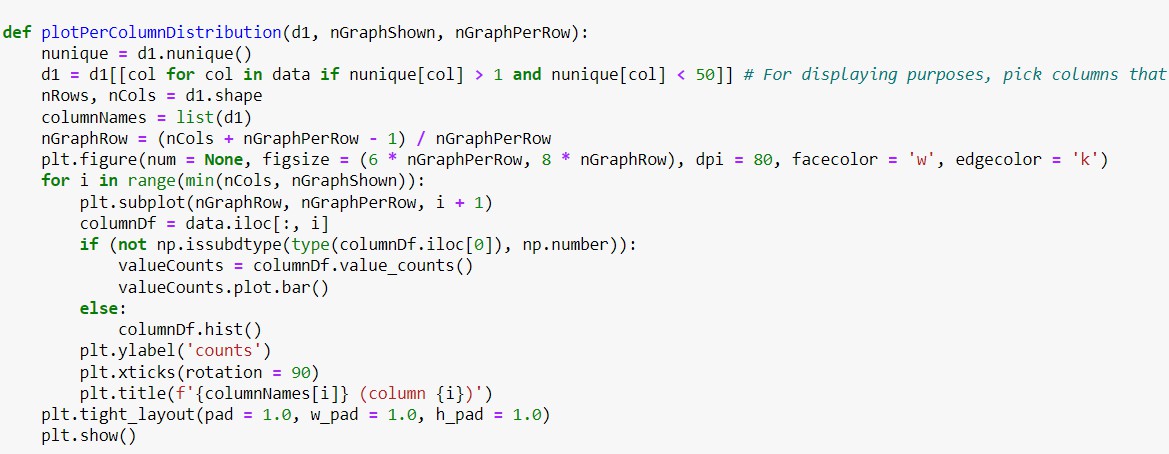
In our data set we do not have any missing data. Our data set is clean.

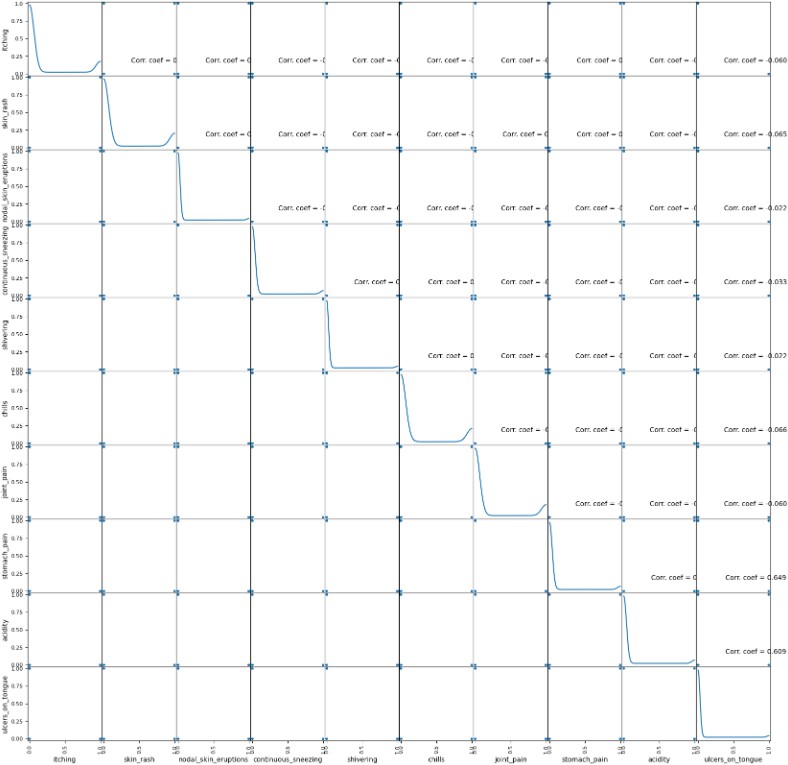


# DATA VISUALIZATION:

## Scatter plot:

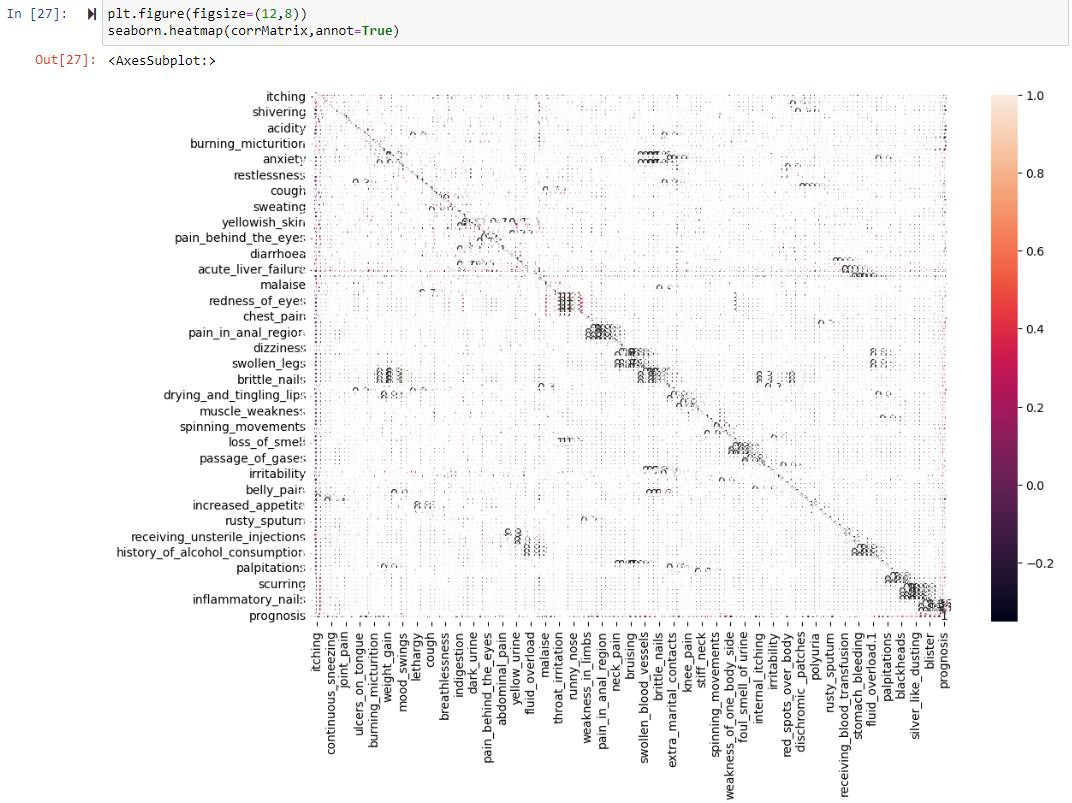
We have used scatter to visualize our data. This plot helps us to find the relation between each attribute.





## Correlation matrix:

The correlation matrix helps us to find out correlation between each variable. Using this plot, we can know if a particular variable has an effect in our model or not.

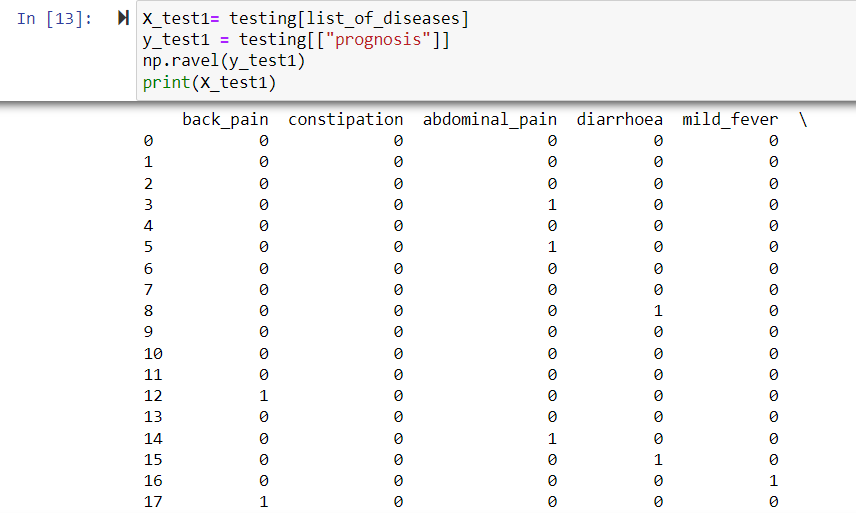


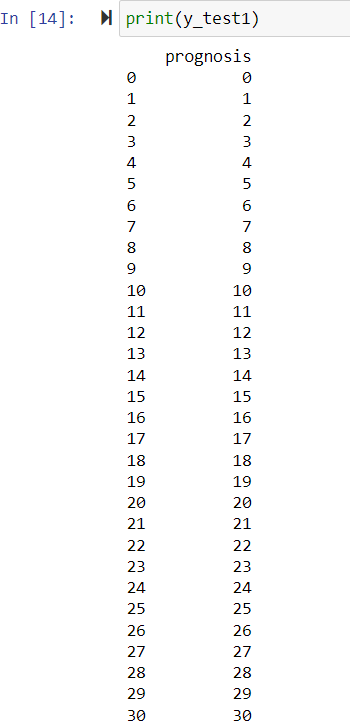
Using this table, we can figure out that every variable has its own significance. Hence all the variables should be used during the testing process.

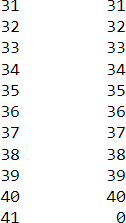
# DATA PROCESSING

**TESTING:**

We do X-test and Y test to our model. X- test is used to test the independent variables. Y-Test is used to test the dependent variables.







# ALGORITHMS

In our model we have used three different algorithms. They are:

* 1. Decision tree algorithm
  2. Random forest algorithm
  3. Naïve bayes algorithm

## Decision Tree Algorithm:

Decision Trees are a versatile supervised learning method that can effectively tackle classification and regression problems. While they are suitable for both, Decision Trees are most commonly used for classification tasks. These tree-structured classifiers have internal nodes that denote the dataset's features, branches representing decision rules, and leaf nodes that represent the final outcomes.

* + A Decision Tree comprises of two fundamental nodes: the **Decision Node** and the

### Leaf Node.

* + Decision Nodes are responsible for making decisions and can have multiple branches, whereas Leaf Nodes serve as the output of those decisions and do not contain any further branches. These decisions, or tests, are conducted based on the dataset's features.
  + Decision Trees provide a graphical representation of all possible solutions to a problem or decision based on the given conditions. The term "Decision Tree" is derived from its tree-like structure that starts with a root node and expands into branches.
  + To construct a Decision Tree, we utilize the Classification and Regression Tree **(CART)** algorithm. A Decision Tree poses a question, and based on the response (Yes/No), it further partitions the tree into subtrees.
* **Root Node:** The root node initiates the decision tree and represents the entire dataset, which is then divided into homogeneous subsets.
* **Leaf Node:** it represents the final output of the decision tree, and no further segregation is possible after getting a leaf node.
* **Splitting:** Splitting involves dividing the decision or root node into sub-nodes based on specific conditions.
* **Branch/Sub Tree:** A tree formed by splitting the tree.
* **Pruning:** Pruning involves removing unnecessary branches from the decision tree.
* **Parent/Child node:** The tree's root node is the parent node, and any other nodes that derive from it are considered child nodes.

**SUB**



### Working of the algorithm:

To predict the class of a given dataset using a decision tree, the algorithm begins at the root node and compares the root attribute's value to the record attribute's value. Based on this comparison, the algorithm follows the corresponding branch to the next node.

The algorithm compares the attribute value to the sub-nodes for the next node and repeats the process until it reaches the tree's leaf node. The algorithm's complete process is illustrated in the following algorithm:

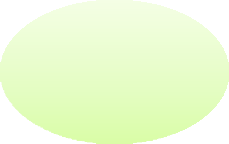
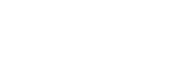
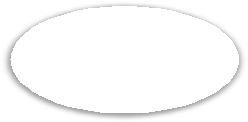
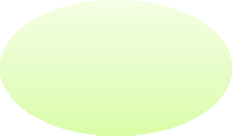
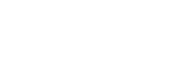
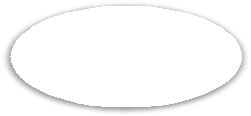
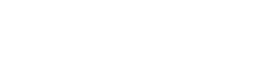
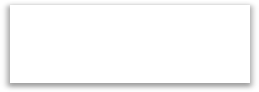
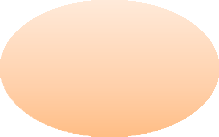
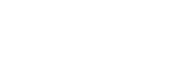
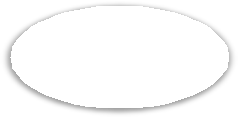
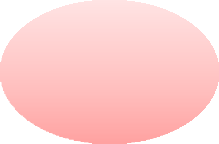
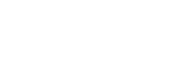
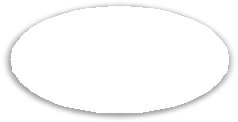
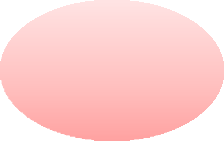
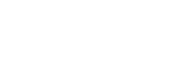
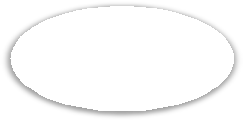
**Step 1:** Start with the root node S, which represents the complete dataset.

**Step 2:** Determine the best attribute in the dataset using an Attribute Selection Measure (ASM).

**Step 3:** Create subsets of S based on the possible values for the best attribute.

**Step 4:** Generate a decision tree node that includes the best attribute.

**Step 5:** Recursively repeat steps 2-4 for each subset of the dataset until a leaf node is reached, and no further classification is possible.



Leaf

Node

Leaf

node

Leaf

Node

**TREE**

Leaf Node

Leaf Node

Decision Node

Decision Node

Decision Node

**ROOT NODE**

Decision Node

Using the Decision tree algorithm for our model will give 92.857% accuracy.

Text  Description automatically generated

## Random Forest Algorithm:

A random forest is a machine learning technique that utilizes ensemble learning to solve classification and regression problems. This algorithm consists of many decision trees that are trained through bagging or bootstrap aggregating. By taking the average or mean of the output from various trees, the random forest predicts the outcome based on the predictions of the decision trees. Increasing the number of trees increases the precision of the outcome. This technique overcomes the limitations of a decision tree algorithm, reduces overfitting of datasets, increases precision, and generates predictions without requiring many configurations in packages.

## Why Random Forest?

* + The Random Forest algorithm requires less time for training when compared to other algorithms.
  + Even for large datasets, the Random Forest algorithm is efficient in predicting output with high accuracy.
  + Additionally, the Random Forest algorithm can maintain accuracy when dealing with a significant proportion of missing data.

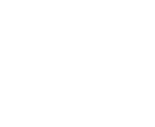
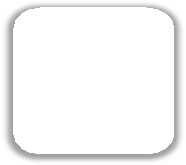
## Working:

The Random Forest algorithm operates in two phases. In the first phase, N decision trees are combined to create the random forest, while in the second phase, predictions are made for each tree created in the first phase. The working process of this algorithm can be explained through the following steps and diagram

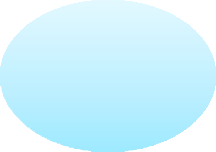
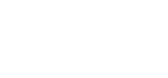
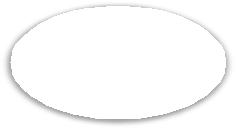
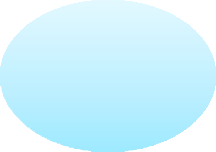
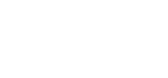
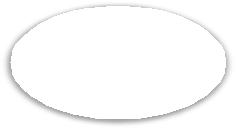
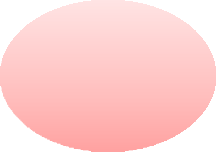
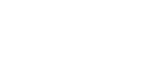
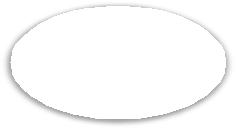
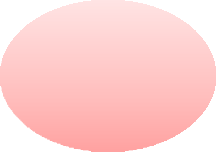
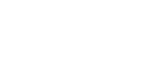
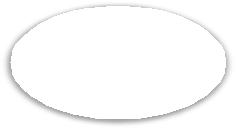
Step 1: involves selecting K random data points from the training set. step 2: decision trees are built using subsets of the selected data points. step 3: The number of decision trees to be built (N) is chosen.

Step 4: Steps 1 and 2 are repeated.

step 5: Finally, predictions are made for new data points by finding the predictions of each decision tree, and the new data points are assigned to the category that has received most of the votes.



**Test set**



**Training**

**date 2**

**Training**

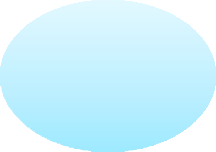
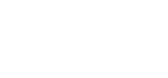
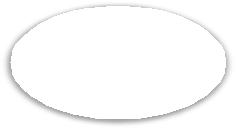
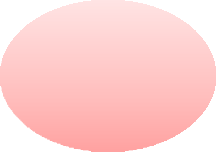
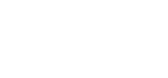
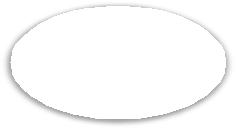
**data n**

**Decision**

**Tree 2**

**Decision**

**Tree n**



**Training**

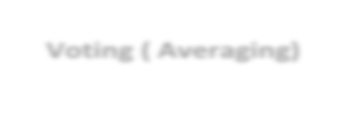
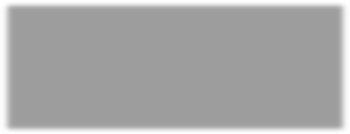
**Set**

**Training**

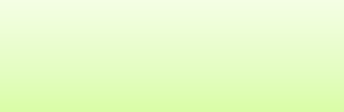
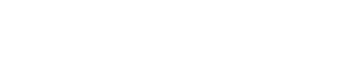
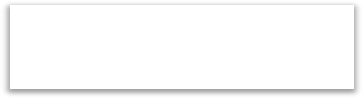
**Date 1**

**Decision**

**Tree 1**



**Voting ( Averaging)**



**Prediction**

Using the Random Forest algorithm for our model will give 92.857% accuracy.

Text  Description automatically generated with medium confidence

## Naïve Bayes Algorithm:

The algorithm is a supervised learning technique that utilizes Bayes' theorem for solving classification problems. This algorithm is particularly useful for text classification problems that involve high-dimensional training datasets. The Naïve Bayes classifier is a simple yet highly effective classification algorithm that facilitates the speedy construction of machine learning models capable of quick predictions. As a probabilistic classifier, it relies on the object's probability to make predictions. Several applications of the Naïve Bayes algorithm exist, including spam filtration, sentimental analysis, and article classification.

### Bayes' Theorem:

Bayes' theorem, sometimes referred to as Bayes' Rule or Bayes' law, is a mathematical principle utilized to calculate the probability of a hypothesis given prior knowledge. It involves conditional probability and can be expressed mathematically using the following formula:

### P(A|B) =[ P(B|A) \* P(A) ] / P(B)

**Were,**

**Posterior probability (P(A|B))** is the probability of hypothesis A given the observed event

B. On the other hand

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

**Prior Probability (P(A))** is the probability of the hypothesis before considering any evidence.

Finally, **Marginal Probability (P(B))** refers to the probability of the evidence irrespective of any hypothesis.

### Working of the algorithm with an example:

Let's utilize Bayes' theorem for our coin example, where we consider two coins and calculate the probability of getting two heads or at least one tail directly from the sample space.

To analyze this, we define event A as the second coin being head, and event B as the first coin being tails, in reverse order to find out the second event.

We focus on event A and express it as the probability of A given B using Bayes' theorem. Probability = P(A|B)

= [ P(B|A) \* P(A) ] / P(B)

= [ P(First coin being tail given the second coin is the head) \* P(Second coin being head) ] / P(First coin being tail)

= [ (1/2) \* (1/2) ] / (1/2)

= 1/2 = 0.5

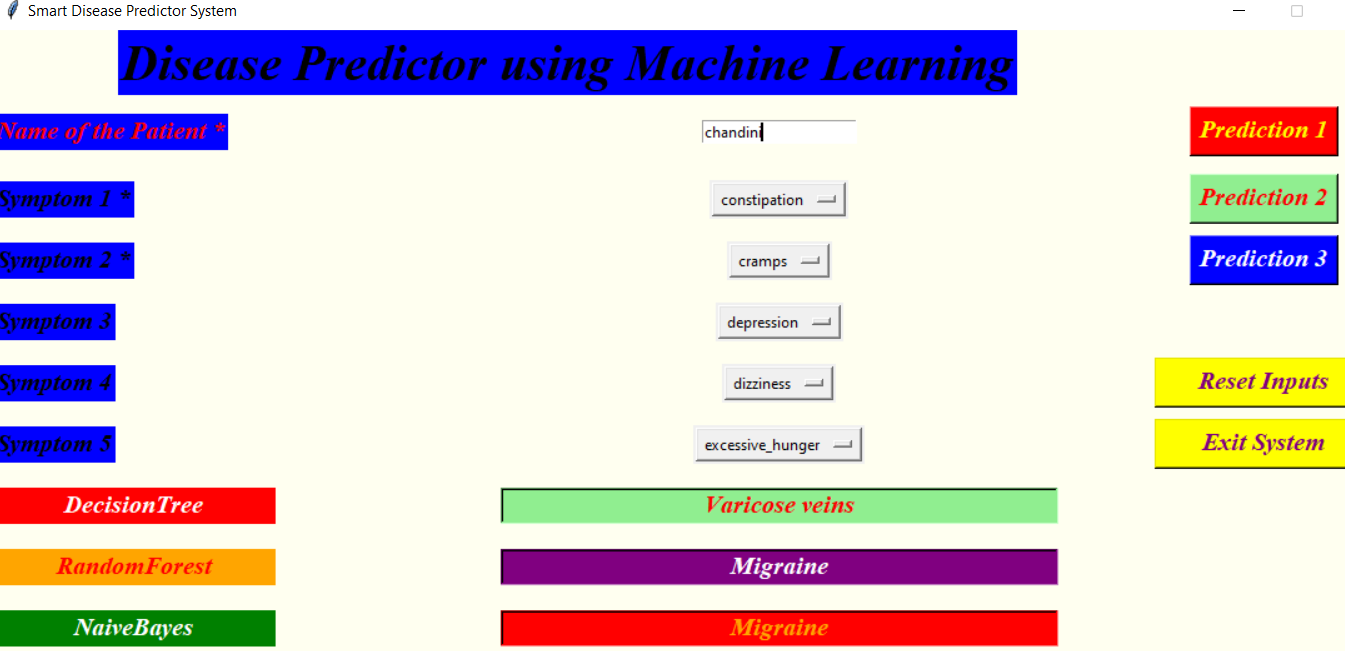
Bayes' theorem enables the computation of the probability of an event given previous knowledge of conditions that may be associated with the event. To effectively use the Naive Bayes algorithm as a machine learning tool, it is essential to comprehend its place in the hierarchy of other techniques.

Using the Naïve Bayes algorithm for our model will give 92.857% accuracy.

Text  Description automatically generated

# OUTPUT:

We need to enter the patient’s name and their symptoms. Three symptoms of the patient is must and the other two symptoms are optional. Based on the given symptoms we need to click on prediction 1, prediction 2 and prediction 3 buttons. Prediction 1 gives the output using decision tree algorithm. Prediction 2 gives the output using Random Forest algorithm. Prediction 3 gives the output using NaivesBayes algorithm.



# CHALLENGES:

1. The dataset used for this project was extensive, which made it time-consuming to remove irrelevant data during the cleaning process.
2. Developing a customized graphical user interface specific to this project presented challenges as the entire interface was built through hard coding.
3. Additionally, selecting the most suitable classifier from the many machine learning classifiers available was a challenging task.
4. Moreover, enhancing the accuracy of the selected classifier is a time-consuming process. Despite these challenges, the project aims to deliver reliable disease diagnosis by utilizing advanced machine learning techniques.

# FUTURE WORK:

Present model we are using only 3 algorithms for the prediction of disease, but in the future, we plan to give the user an option to select the drop down from the list of n- number of classifiers to choose for prediction. Improve the accuracy of the current model to the maximum. We also plan to add more diseases and symptoms to the model.

# CONCLUSION:

We were able to achieve 93% accuracy on average. With a system like this people who do not have access to diagnosis center can find the disease and take the precautionary steps as first step pf treatment. We have included a method for storing the user input which can be used in the future to build a better version. We were also able to see how different supervised machine learning algorithms work and were analyzed as well. It provided insight into how technologies such as machine learning can help the healthcare industry.