MINI PROJECT - II

(2021-22)

BRAINY CHATBOT

FINAL REPORT

Institute of Engineering & Technology

Team Members:

Jaideep Lalchandani
(181500290)
Mansi Goyal
(181500370)
Prashant Tomar
(181500492)
Radhika Singh
(181500529)
Siddhant Gupta
(181500708)

Supervised By:

Mr. Sharad Gupta
Technical Trainer
Department of Computer Engineering & Applications

ACKNOWLEDGEMENT

We take this opportunity to thank all those who have helped us in completing the project successfully.

We would like to express our gratitude to **Mr. Sharad Gupta**, who as our guide/mentor provided us with every possible support and guidance throughout the development of project. This project would never have been completed without his encouragement and support.

Our heartiest thanks to Dr. (Prof). **Anand Singh Jalal**, Head of Dept., Department of CEA for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal.

.We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

DECLARATION

We hereby declare that the work which is being presented in the Mini Project **"Brainy ChatBOT"** in partial fulfilment of the requirements for mini project viva voice, is an authentic record of my own work carried under the supervision of "GLA UNIVERSITY MATHURA".

Signature of Candidate:
Name of Candidate: Jaideep Lalchandani, Mansi Goyal, Prashant Tomar, Radhika Singh, Siddhant Gupta
Roll. No.: 181500290, 181500370, 181500492, 181500529, 181500708
Course: B.Tech. (Computer Science & Engineering)
Year: 3rd
Semester: VI

ABSTRACT

The project named "**Brainy ChatBOT**", is a machine learning based model created in Python using Machine learning Algorithms. The purpose of the project is to provide a model that can predict the diseases of a patient on the basis of his/her symptoms and provide the concerned doctor's link to contact. In this project patient can open the model and just input yes for regarding symptoms that he/she have and then the model will predict diseases. We have used csv data files for this project so when new diseases will come, we can easily enhance our data and doctors also can add their data in the doctor's dataset. This chat-bot model can reduce the time which taken when we don't know about our diseases that we have and the time to search the doctor and as we know medical field is in growing phase so today by this model patient can identify some of their disease and directly contact to the doctors.

Contents

Acknowledgement 2

Declaration 3

Abstract 4

1. Introduction **6**

1.1 Motivation……………………………………………….… 6

1.2 Overview…………………………………………………. 6

1.3 Objective……………………………………………. … 6

2. Software Requirement Analysis **8**

2.1 Problem Statement…………………………………………. 8

2.2 Modules………………………………………………….. 8

2.3 Specific Requirement……………………………………... 10

2.4 Tools and Technology used…………………………………... 11

3. Software Design **27**

3.1 Data Flow Diagram…………………………………………….. 27

3.2 UML Diagram………………………………………………… 28

3.3 Database Design………………………………………………. 30

4. Software Testing **33**

5. Implementation and User Interface **35**

6. Appendices **38**

Future Scope 44

Bibliography/References 45

Chapter 1. Introduction

1.1 Motivation

We all know that medical field is growing there are lots of software available to predict medicines according to disease but not that type of software which can analyze our disease according to our given symptoms and provide the concerned doctor's link to contact that can reduce the time to find a better doctor.

1.2 Overview

In this project, the model will start showing the symptoms and the patient have to choose the right one that he/she have and according to those symptoms model will show the disease name that he may have and the link to contact the regarding doctor.

1.3 Objective

In "**Brainy ChatBOT**" project the objective is to reduce the time which taken by a patient when he or she doesn't know about their disease and if they know about it other scenario is that they don't know which doctor they should consult about it.

So by this project it will be very easy to analyze our disease at any place just at fingertips. There is not such type of software available that can predict the disease and the doctor also.

Chapter 2. Software Requirement Analysis

2.1 Problem Statement

There is not such type of software available that can predict the disease and the doctor also. So, the problem statement is to create a model that predict the disease according to the symptoms present in the patient. Also, the symptoms will be asked one by one to the patient.

2.2 Modules

The project is based on several modules:

2.2.1 Product perspective

Following services will be provided.

1. Information regarding disease of the patient and the link of the doctor to contact will be displayed.
2. Presence of particular symptoms will be asked so that a patient can select right one analyze disease.

2.2.2 Product Functions

A summary of the major functions that the model will perform:

1. Provide facility to patient to do their check-up and that saves the time.
2. Doctors and the Medical staff can register on this model by adding there data in the dataset.
3. Patient has to register only single time then he can access it by username and password.

2.2.2.1 Patient

- ✓ Can detect their disease.
- ✓ If disease detected that he or she can contact to doctor which is suggested.

2.2.2.2 Doctors

- ✓ Can update their data in the dataset
- ✓ Check-up the patient which is sent to him or her by the model

2.2.3 User Characteristics

a. Educational level: Users should be comfortable with the English language.

b. **Experience:** Users should have prior information regarding the names of general diseases and medical facilities.

c. **Skills:** Users should have basic knowledge and should be comfortable using general purpose applications on computers.

2.2.4 General Constraints

- Since the CSV Files is used as data source for this project so there can be some noise in data.

- Programming is done by Python3.

2.3 Specific Requirements

This specific requirements describe the specific constraints impost on the requirements :

- Processor-i3(Minimum)
- Hard Disk-64GB
- Memory- 4 GB RAM(Minimum)

2.3.2 Hardware Interfaces

- Screen resolution of at least 800X600 is required for proper and complete viewing of screens. Higher resolution will be accepted.

2.3.3 Software Interfaces

- Any Windows/Linux/Mac based operating system.
- MS Excel to open the CSV files.
- Anaconda (Spyder) / Jupyter Notebook for developing code.

2.4 Technologies and Tools used

2.4.1 Introduction of Python in Machine Learning

- To reiterate, Machine Learning is simply recognizing patterns in your data to be able to make improvements and intelligent decisions on its own.
- Python is the most suitable programming language for this because it is easy to understand and you can read it for yourself.
- Its readability, non-complexity, and ability for fast prototyping make it a popular language among developers and programmers around the world.
- Many of these inbuilt libraries are for Machine Learning and Artificial Intelligence, and can easily be applied out of the box.
- Some of the libraries used:

- **scikit-learn** for data mining, analysis, and Machine Learning;
- **Tensorflow**, a high-level neural network library;
- **pylearn2** which is also ideal for data mining and Machine Learning, but more flexible than scikit-learn.

2.4.2 Tools Used

1. **Python**

Python is an interpreter, object-oriented programming language that has gained popularity because of its clear syntax and readability. Python can be used on a server to create web applications. First and foremost reason why Python is much popular because it is highly productive as compared to other programming languages like C++ and Java. Python was created by Guido van Rossum, a former resident of the Netherlands, whose favourite comedy group at the time was Monty Python's Flying Circus.

2. Anaconda Navigator:

Anaconda Navigator is a desktop GUI that comes with Anaconda Individual Edition. It makes it easy to launch applications and manage packages and environments without using command-line commands.

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed.

3. Spyder

Spyder is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python. Pythostack,including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open source software. It is released under the MIT license.

2.4.3 Libraries Used

1. NumPy

NumPy is a python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. It was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python. NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. It is written partially in Python, but most of the parts that require fast computation are written in C or C++.

2. Pandas

Pandas is an open source library that is used to analyze data in Python. Pandas allows importing data from various file formats such as comma-separated values, JSON, SQL, Microsoft Excel and creates an object with rows and columns called a data frame Pandas allows various data manipulation operations such as merging , reshaping[, selecting, as well as data cleaning, and data wrangling features.

3. Matplotlib

Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter. It can be used in Python and IPython shells, Jupyter notebook and web application servers also.

4. **Scikit Learn**:

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib

2.4.4 Decision Tree

2.4.4.1 Introduction

Classification is a two-step process, learning step and prediction step, in machine learning. In the learning step, the model is developed based on given training data. In the prediction step, the model is used to predict the response for given data. Decision Tree is one of the easiest and popular classification algorithms to understand and interpret.

2.4.4.2 Types of Decision Trees

Types of decision trees are based on the type of target variable we have. It can be of two types:

1. **Categorical Variable Decision Tree:** Decision Tree which has a categorical target variable then it called a **Categorical variable decision tree.**

2. **Continuous Variable Decision Tree:** Decision Tree has a continuous target variable then it is called **Continuous Variable Decision Tree.**

Example:- Let's say we have a problem to predict whether a customer will pay his renewal premium with an insurance company (yes/ no). Here we know that the income of customers is a significant variable but the insurance company does not have income details for all customers. Now, as we know this is an important variable, then we can build a decision tree to predict customer income based on occupation, product, and various other variables. In this case, we are predicting values for the continuous variables.

2.4.4.3 Important Terminology related to Decision Trees

1. **Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.

2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.

3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called the decision node.

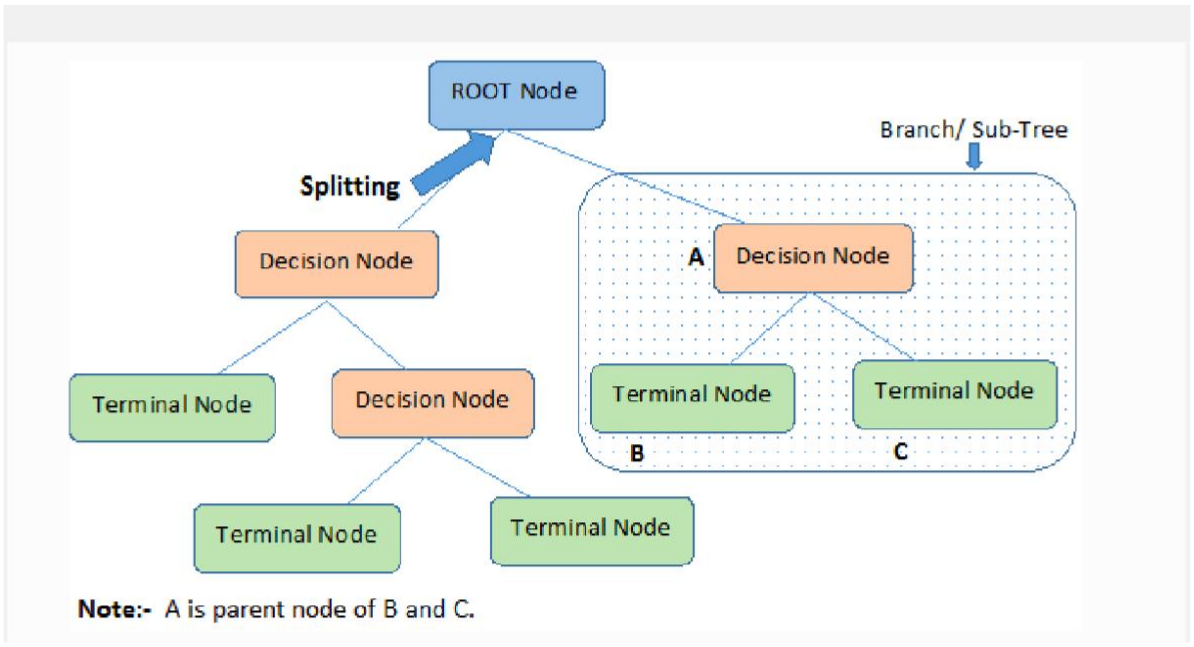
4. **Leaf / Terminal Node:** Nodes do not split is called Leaf or Terminal node.

5. **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of

splitting.

6. **Branch / Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.

7. **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.



Decision trees classify the examples by sorting them down the tree from the root to some leaf/terminal node, with the leaf/terminal node providing the classification of the example.

Each node in the tree acts as a test case for some attribute, and each edge descending from the node corresponds to the possible answers to the test case. This process is recursive in nature and is repeated for every sub tree rooted at the new node.

Assumptions while creating Decision Tree

Below are some of the assumptions we make while using Decision tree:

- In the beginning, the whole training set is considered as the **root.**
- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- Records are **distributed recursively** on the basis of attribute values.

- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.

Decision Trees follow **Sum of Product (SOP) r**epresentation. The Sum of product (SOP) is also known as **Disjunctive Normal Form**. For a class, every branch from the root of the tree to a leaf node having the same class is conjunction (product) of values, different branches ending in that class form a disjunction (sum).

The primary challenge in the decision tree implementation is to identify which attributes do we need to consider as the root node and each level. Handling this is to know as the attributes selection. We have different attributes selection measures to identify the attribute which can be considered as the root note at each level.

How do Decision Trees work?

The decision of making strategic splits heavily affects a tree’s accuracy. The decision criteria are different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable. The decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

The algorithm selection is also based on the type of target variables. Let us look at some algorithms used in Decision Trees:

ID3 → (extension of D3)

C4.5 → (successor of ID3)

CART → (Classification And Regression Tree)

The ID3 algorithm builds decision trees using a top-down greedy search approach through the space of possible branches with no backtracking. A greedy algorithm, as the name suggests, always makes the choice that seems to be the best at that moment.

Steps in ID3 algorithm:

1. It begins with the original set S as the root node.

2. On each iteration of the algorithm, it iterates through the very unused attribute of the set S and calculates **Entropy(H)** and **Information gain(IG)** of this attribute.

3. It then selects the attribute which has the smallest Entropy or Largest Information gain.

4. The set S is then split by the selected attribute to produce a subset of the data.

5. The algorithm continues to recur on each subset, considering only attributes never selected before.

Attribute Selection Measures

The dataset consists of **N** attributes then deciding which attribute to place at the root or at different levels of the tree as internal nodes is a complicated step. By just randomly selecting any node to be the root can't solve the issue. If we follow a random approach, it may give us bad results with low accuracy.
For solving this attribute selection problem, researchers worked and devised some solutions.

They suggested using some *criteria* like:

Entropy,

Information gain,

Gini index,

Gain Ratio,

Reduction in Variance

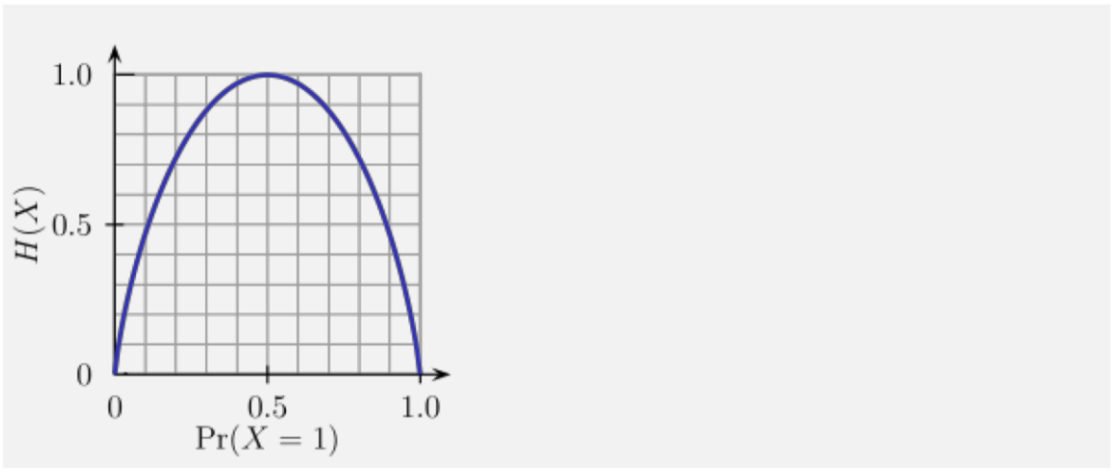
Chi-Square

These criterions will calculate values for every attribute. The values are sorted, and attributes are placed in the tree by following the order i.e., the attribute with a high value(in case of information gain) is placed at the root.
While using Information Gain as a criterion, we assume attributes to be categorical, and for the Gini index, attributes are assumed to be continuous.

Entropy

Entropy is a measure of the randomness in the information being processed.

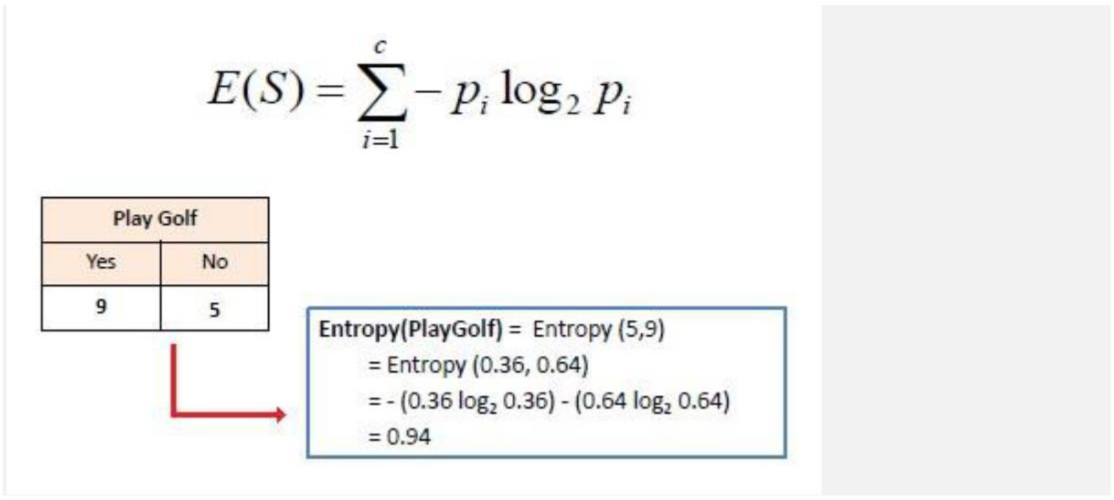
The higher the entropy, the harder it is to draw any conclusions from that information. Flipping a coin is an example of an action that provides information that is random.



From the above graph, it is quite evident that the entropy H(X) is zero when the probability is either 0 or 1. The Entropy is maximum when the probability is 0.5 because it projects perfect randomness in the data and there is no chance if perfectly determining the outcome.

ID3 follows the rule — A branch with an entropy of zero is a leaf node and a branch with entropy more than zero needs further splitting.

Mathematically Entropy for 1 attribute is represented as:



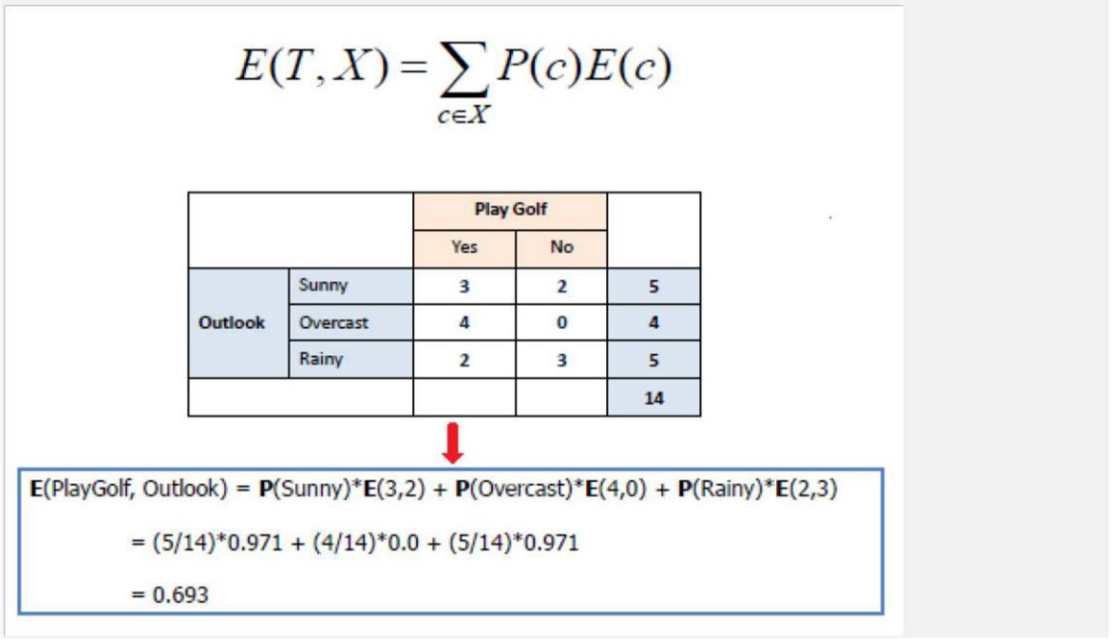
$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94

Where S → **Current state, and Pi → Probability of an event *i* of state** S **or Percentage of class *i* in a node of state S.**

Mathematically Entropy for multiple attributes is represented as:



$$E(T, X) = \sum_{c \in X} P(c)E(c)$$

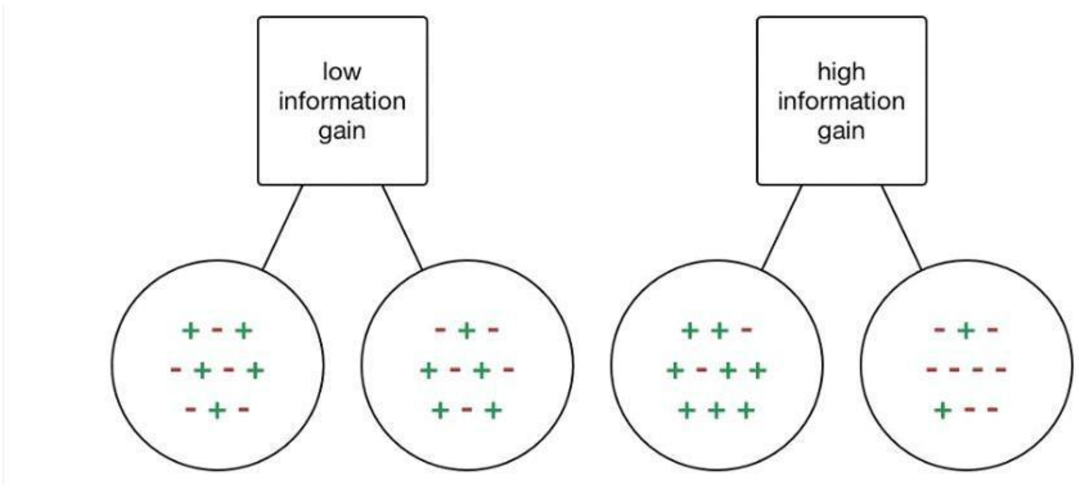
		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)***E**(3,2) + **P**(Overcast)***E**(4,0) + **P**(Rainy)***E**(2,3)
= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971
= 0.693

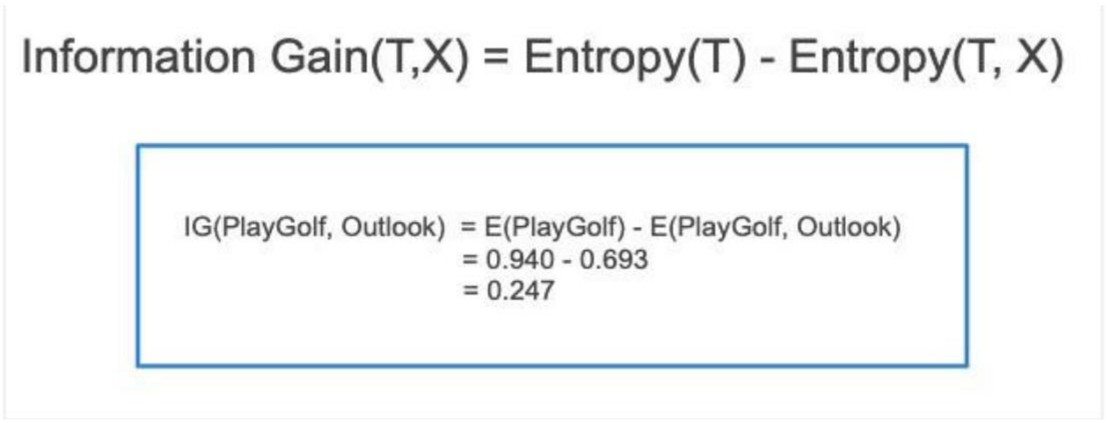
where **T→ Current state and X → Selected attribute**

Information Gain

Information gain or **IG** is a statistical property that measures how well a given attribute separates the training examples according to their target classification. Constructing a decision tree is all about finding an attribute that returns the highest information gain and the smallest entropy.



Information gain is a decrease in entropy. It computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values. ID3 (Iterative Dichotomiser) decision tree algorithm uses information gain. Mathematically, IG is represented as:



Information Gain(T,X) = Entropy(T) - Entropy(T, X)

```
IG(PlayGolf, Outlook)  = E(PlayGolf) - E(PlayGolf, Outlook)
                       = 0.940 - 0.693
                       = 0.247
```

In a much simpler way we can conclude that,

$$Information\ Gain = Entropy(before) - \sum_{j=1}^{K} Entropy(j,\ after)$$

Where "before" is the dataset before the split, K is the number of subsets generated by the split, and (j, after) is subset j after the split.

Gini Index

You can understand the Gini index as a cost function used to evaluate splits in the dataset. It is calculated by subtracting the sum of the squared probabilities of each class from one. It favors larger partitions and easy to implement whereas information gain favors smaller partitions

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

Gini Index works with the categorical target variable "Success" or "Failure". It performs only Binary splits.

Higher the value of Gini index higher the homogeneity.

Steps to Calculate Gini index for a split

1. Calculate Gini for sub-nodes, using the above formula for success(p) and failure(q) (p^2+q^2).

2. Calculate the Gini index for split using the weighted Gini score of each node of that split.

CART (Classification and Regression Tree) uses the Gini index method to create split points.

Gain ratio

Information gain is biased towards choosing attributes with a large number of values as root nodes. It means it prefers the attribute with a large number of distinct values.

C4.5, an improvement of ID3, uses Gain ratio which is a modification of Information gain that reduces its bias and is usually the best option. Gain ratio overcomes the problem with information gain by taking into account the number of branches that would result before making the split. It corrects information gain by taking the intrinsic information of a split into account.

$$Gain\ Ratio = \frac{Information\ Gain}{SplitInfo} = \frac{Entropy\ (before) - \sum_{j=1}^{K} Entropy(j, after)}{\sum_{j=1}^{K} w_j \log_2 w_j}$$

Where “before” is the dataset before the split, K is the number of subsets generated by the split, and (j, after) is subset j after the split.

Reduction in Variance

Reduction in variance is an algorithm used for continuous target variables (regression problems). This algorithm uses the standard formula of variance to choose the best split. The

split with lower variance is selected as the criteria to split the population:

$$\text{Variance} = \frac{\Sigma(X - \overline{X})^2}{n}$$

Above X-bar is the mean of the values, X is actual and n is the number of values.

Steps to calculate Variance:

1. Calculate variance for each node.
2. Calculate variance for each split as the weighted average of each node variance.

Chapter 3. Software Design

3.1 Data Flow Diagram

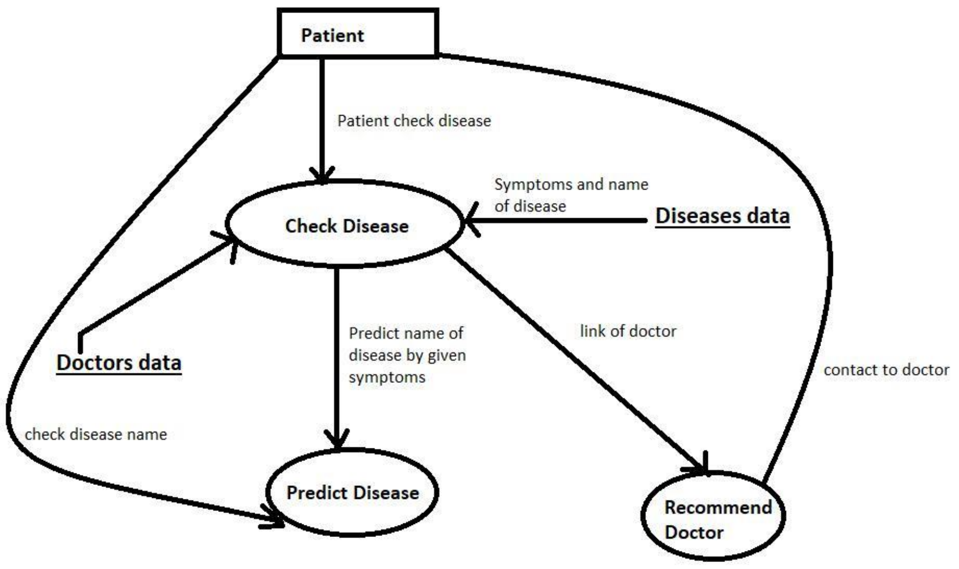
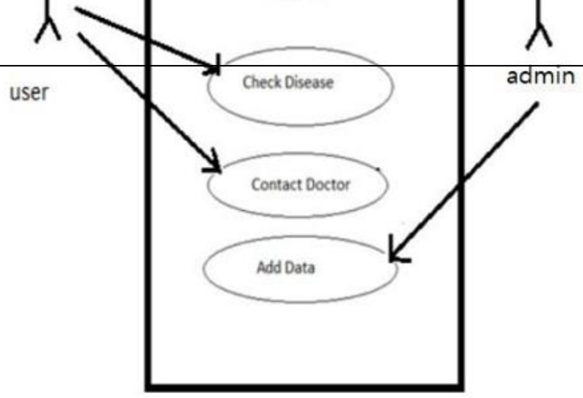


Fig-1


user
Check Disease
Contact Doctor
Add Data
admin

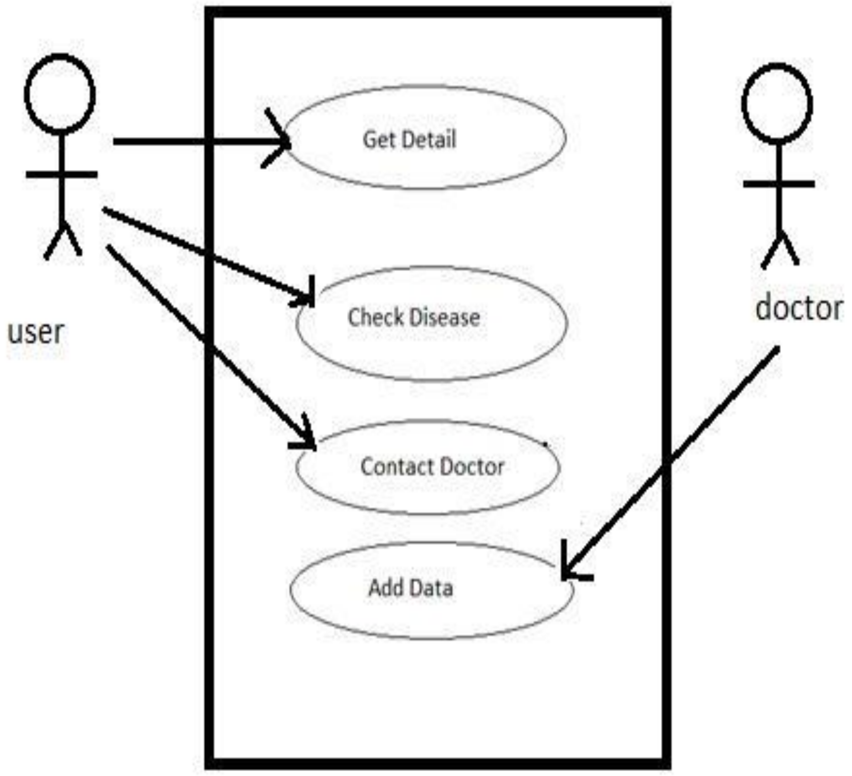

user
Get Detail
Check Disease
Contact Doctor
Add Data
doctor

Fig-2

3.2.2 Sequence Diagram

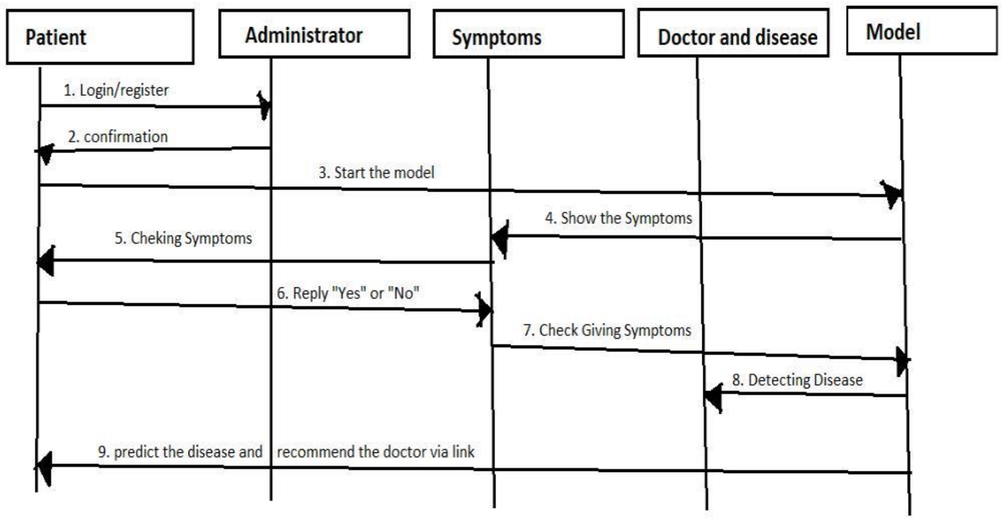


Fig-3

3.3 Database Design

3.3.1 E-R Diagram

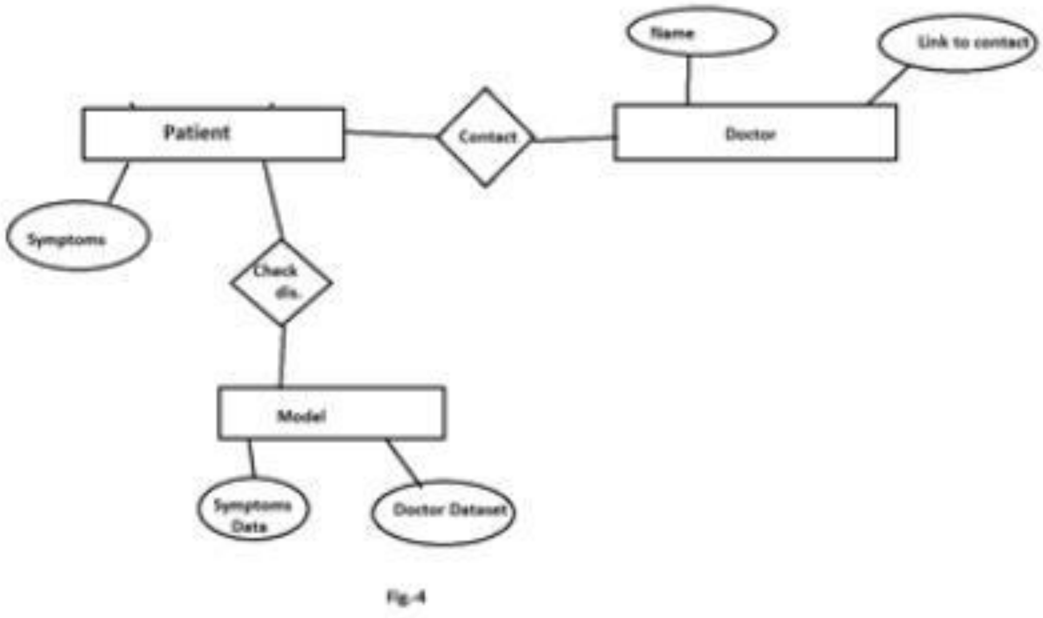
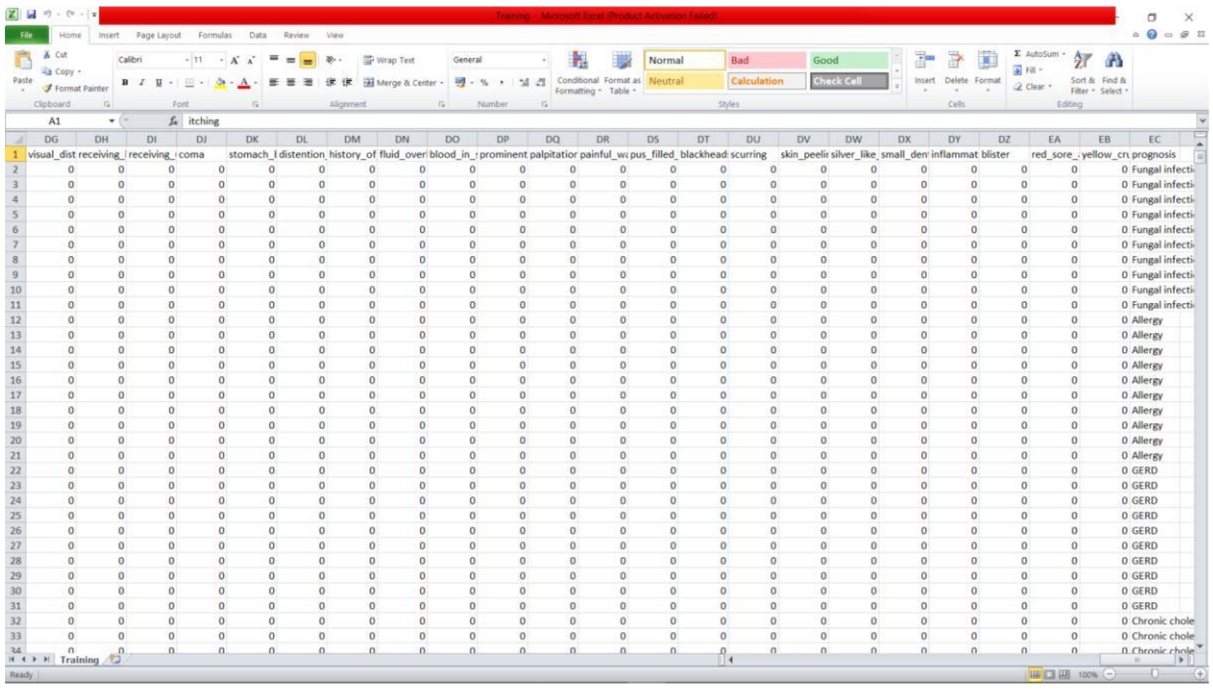


Fig.-4

3.3.2 Tables



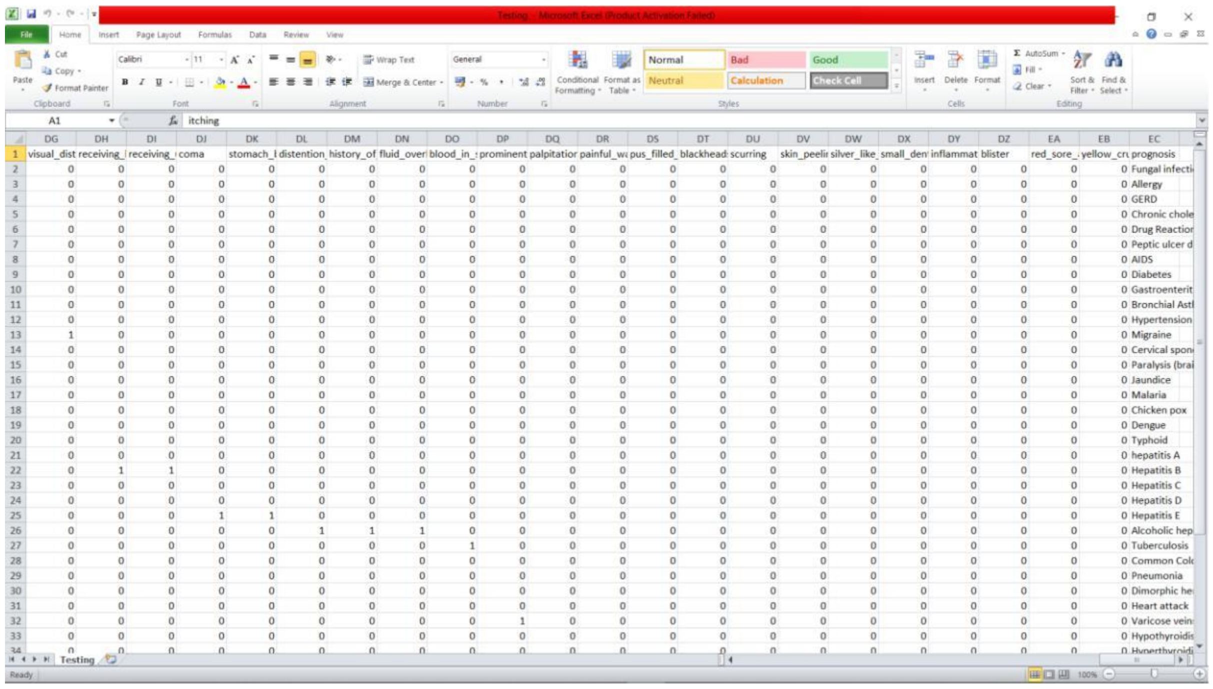
Training - Microsoft Excel (Product Activation Failed)

A1 itching

	DG	DH	DI	DJ	DK	DL	DM	DN	DO	DP	DQ	DR	DS	DT	DU	DV	DW	DX	DY	DZ	EA	EB	EC
1	visual_dist	receiving_	receiving_	coma	stomach_	distention	history_of	fluid_over	blood_in_	prominent	palpitatior	painful_wa	pus_filled_	blackhead	scurring	skin_peeli	silver_like_	small_den	inflammat	blister	red_sore_	yellow_cru	prognosis
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Chronic chole
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Chronic chole
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Chronic chole

Training

Table No.-1



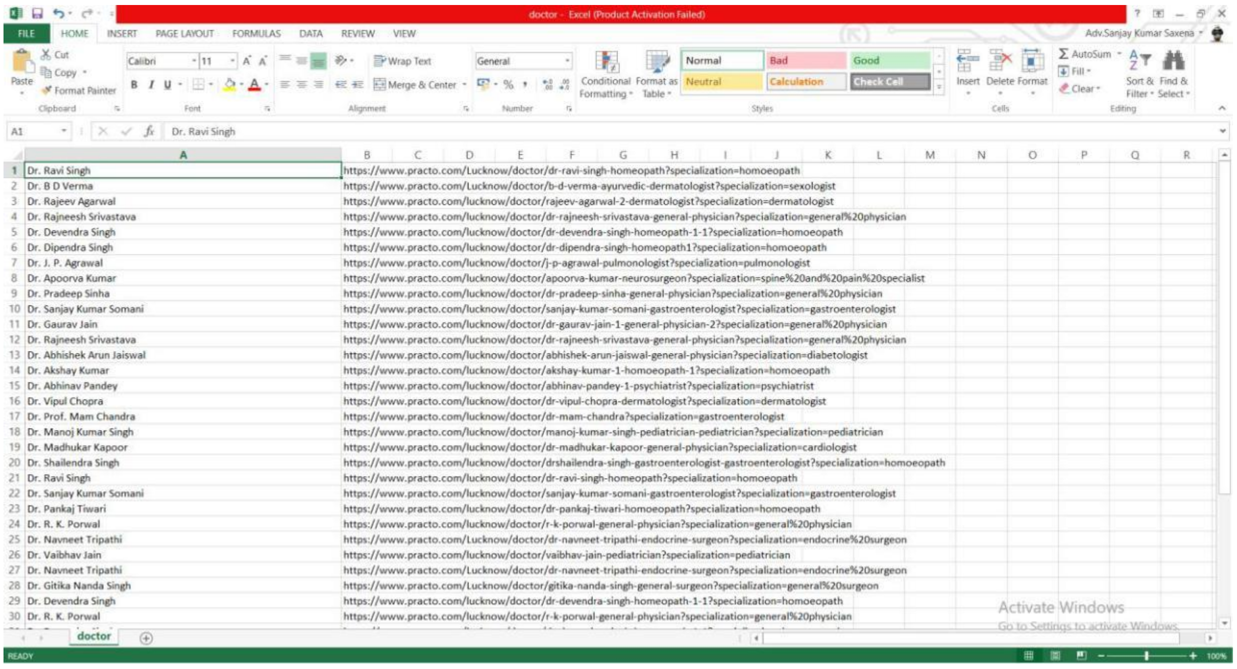
Testing - Microsoft Excel (Product Activation Failed)

A1 itching

	DG	DH	DI	DJ	DK	DL	DM	DN	DO	DP	DQ	DR	DS	DT	DU	DV	DW	DX	DY	DZ	EA	EB	EC
1	visual_dist	receiving_	receiving_	coma	stomach_	distention	history_of	fluid_over	blood_in_	prominent	palpitatior	painful_wa	pus_filled_	blackhead	scurring	skin_peeli	silver_like_	small_den	inflammat	blister	red_sore_	yellow_cru	prognosis
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Fungal infecti
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Allergy
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	GERD
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Chronic chole
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Drug Reactior
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Peptic ulcer d
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	AIDS
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Diabetes
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Gastroenterit
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Bronchial Astl
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hypertension
13	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Migraine
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Cervical spon
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Paralysis (brai
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Jaundice
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Malaria
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Chicken pox
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Dengue
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Typhoid
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	hepatitis A
22	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hepatitis B
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hepatitis C
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hepatitis D
25	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hepatitis E
26	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Alcoholic hep
27	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	Tuberculosis
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Common Colc
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Pneumonia
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Dimorphic he
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Heart attack
32	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	Varicose vein
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hypothyroidis
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hyperthyroidi

Testing

Table No.-2



	A	B
1	Dr. Ravi Singh	https://www.practo.com/Lucknow/doctor/dr-ravi-singh-homeopath?specialization=homoeopath
2	Dr. B D Verma	https://www.practo.com/Lucknow/doctor/b-d-verma-ayurvedic-dermatologist?specialization=sexologist
3	Dr. Rajeev Agarwal	https://www.practo.com/lucknow/doctor/rajeev-agarwal-2-dermatologist?specialization=dermatologist
4	Dr. Rajneesh Srivastava	https://www.practo.com/lucknow/doctor/dr-rajneesh-srivastava-general-physician?specialization=general%20physician
5	Dr. Devendra Singh	https://www.practo.com/lucknow/doctor/dr-devendra-singh-homeopath-1-1?specialization=homoeopath
6	Dr. Dipendra Singh	https://www.practo.com/lucknow/doctor/dr-dipendra-singh-homeopath1?specialization=homoeopath
7	Dr. J. P. Agrawal	https://www.practo.com/lucknow/doctor/j-p-agrawal-pulmonologist?specialization=pulmonologist
8	Dr. Apoorva Kumar	https://www.practo.com/lucknow/doctor/apoorva-kumar-neurosurgeon?specialization=spine%20and%20pain%20specialist
9	Dr. Pradeep Sinha	https://www.practo.com/lucknow/doctor/dr-pradeep-sinha-general-physician?specialization=general%20physician
10	Dr. Sanjay Kumar Somani	https://www.practo.com/lucknow/doctor/sanjay-kumar-somani-gastroenterologist?specialization=gastroenterologist
11	Dr. Gaurav Jain	https://www.practo.com/lucknow/doctor/dr-gaurav-jain-1-general-physician-2?specialization=general%20physician
12	Dr. Rajneesh Srivastava	https://www.practo.com/lucknow/doctor/dr-rajneesh-srivastava-general-physician?specialization=general%20physician
13	Dr. Abhishek Arun Jaiswal	https://www.practo.com/lucknow/doctor/abhishek-arun-jaiswal-general-physician?specialization=diabetologist
14	Dr. Akshay Kumar	https://www.practo.com/lucknow/doctor/akshay-kumar-1-homoeopath-1?specialization=homoeopath
15	Dr. Abhinav Pandey	https://www.practo.com/lucknow/doctor/abhinav-pandey-1-psychiatrist?specialization=psychiatrist
16	Dr. Vipul Chopra	https://www.practo.com/lucknow/doctor/dr-vipul-chopra-dermatologist?specialization=dermatologist
17	Dr. Prof. Mam Chandra	https://www.practo.com/lucknow/doctor/dr-mam-chandra?specialization=gastroenterologist
18	Dr. Manoj Kumar Singh	https://www.practo.com/lucknow/doctor/manoj-kumar-singh-pediatrician-pediatrician?specialization=pediatrician
19	Dr. Madhukar Kapoor	https://www.practo.com/lucknow/doctor/dr-madhukar-kapoor-general-physician?specialization=cardiologist
20	Dr. Shailendra Singh	https://www.practo.com/lucknow/doctor/drshailendra-singh-gastroenterologist-gastroenterologist?specialization=homoeopath
21	Dr. Ravi Singh	https://www.practo.com/lucknow/doctor/dr-ravi-singh-homeopath?specialization=homoeopath
22	Dr. Sanjay Kumar Somani	https://www.practo.com/lucknow/doctor/sanjay-kumar-somani-gastroenterologist?specialization=gastroenterologist
23	Dr. Pankaj Tiwari	https://www.practo.com/lucknow/doctor/dr-pankaj-tiwari-homoeopath?specialization=homoeopath
24	Dr. R. K. Porwal	https://www.practo.com/lucknow/doctor/r-k-porwal-general-physician?specialization=general%20physician
25	Dr. Navneet Tripathi	https://www.practo.com/Lucknow/doctor/dr-navneet-tripathi-endocrine-surgeon?specialization=endocrine%20surgeon
26	Dr. Vaibhav Jain	https://www.practo.com/lucknow/doctor/vaibhav-jain-pediatrician?specialization=pediatrician
27	Dr. Navneet Tripathi	https://www.practo.com/Lucknow/doctor/dr-navneet-tripathi-endocrine-surgeon?specialization=endocrine%20surgeon
28	Dr. Gitika Nanda Singh	https://www.practo.com/Lucknow/doctor/gitika-nanda-singh-general-surgeon?specialization=general%20surgeon
29	Dr. Devendra Singh	https://www.practo.com/lucknow/doctor/dr-devendra-singh-homeopath-1-1?specialization=homoeopath
30	Dr. R. K. Porwal	https://www.practo.com/lucknow/doctor/r-k-porwal-general-physician?specialization=general%20physician

Table No.-3

Chapter 4. Software Testing

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under testing. Software testing is a process of executing a program or application with intent of finding the software bugs. It can also be stated as the process of validating and verifying that a software program or application.

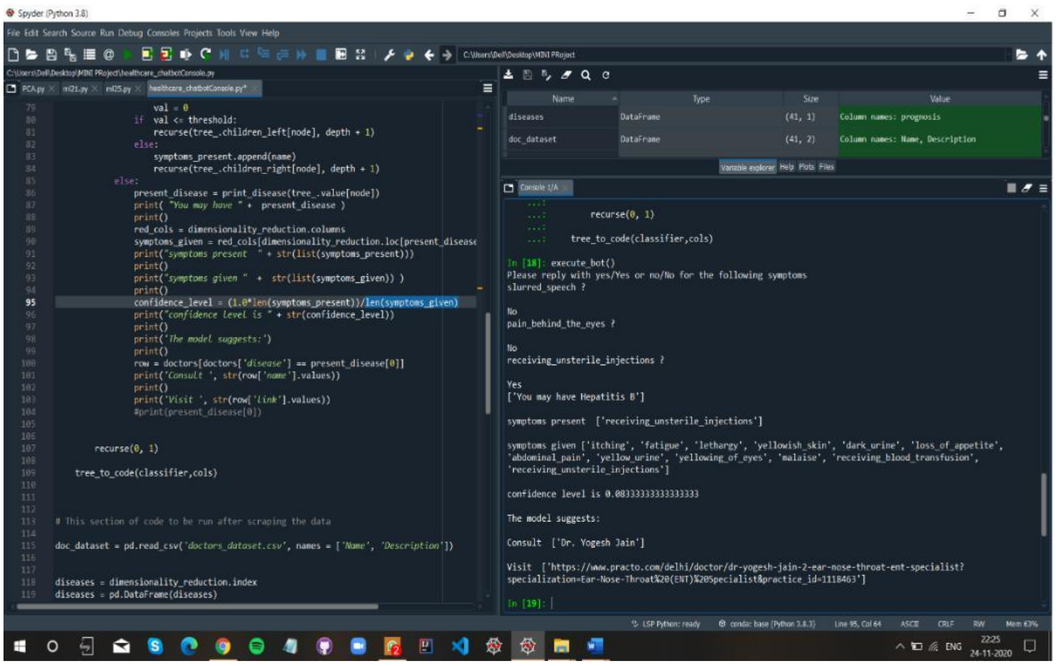
4.1 Testing of Model –

In testing we have used one function name confidence level to check how much accurate we are predicting the disease according to the symptoms given.

$$= \frac{(1.0 * \quad (\quad \_ \quad))}{(\quad \_ \quad)}$$

Here, Len is indicating as lenngth and the Symptoms_present indicating how much no of symptoms are present and Symptoms given indicating what symptom we have given to the Model.

4.1.1 Testing

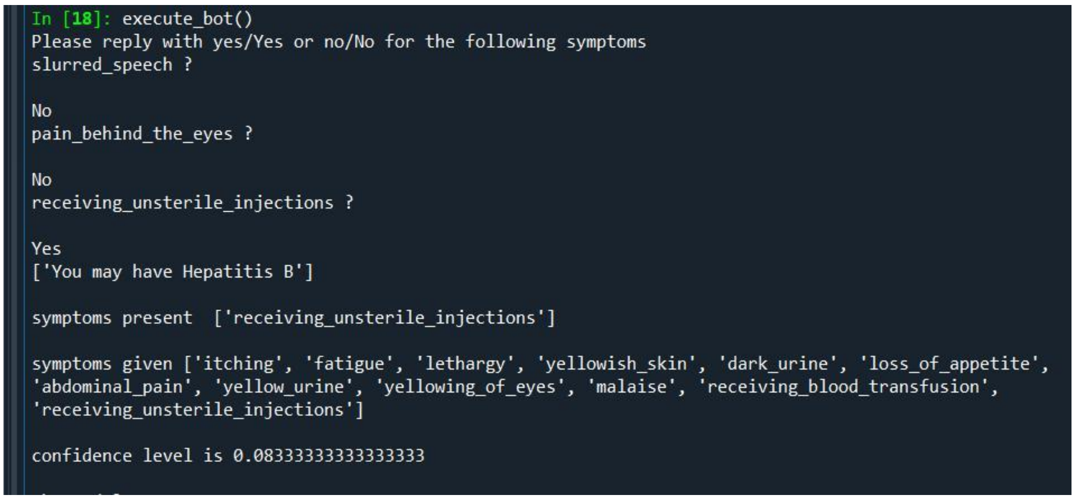


```
                    val = 0
                if  val <= threshold:
                    recurse(tree_.children_left[node], depth + 1)
                else:
                    symptoms_present.append(name)
                    recurse(tree_.children_right[node], depth + 1)
            else:
                present_disease = print_disease(tree_.value[node])
                print( "You may have " +  present_disease )
                print()
                red_cols = dimensionality_reduction.columns
                symptoms_given = red_cols[dimensionality_reduction.loc[present_disease
                print("symptoms present  " + str(list(symptoms_present)))
                print()
                print("symptoms given "  +  str(list(symptoms_given)) )
                print()
                confidence_level = (1.0*len(symptoms_present))/len(symptoms_given)
                print("confidence level is " + str(confidence_level))
                print()
                print('The model suggests:')
                print()
                row = doctors[doctors['disease'] == present_disease[0]]
                print('Consult ', str(row['name'].values))
                print()
                print('Visit ', str(row['link'].values))
                #print(present_disease[0])


        recurse(0, 1)

    tree_to_code(classifier,cols)



# This section of code to be run after scraping the data

doc_dataset = pd.read_csv('doctors_dataset.csv', names = ['Name', 'Description'])


diseases = dimensionality_reduction.index
diseases = pd.DataFrame(diseases)
```

Name	Type	Size	Value
diseases	DataFrame	(41, 1)	Column names: prognosis
doc_dataset	DataFrame	(41, 2)	Column names: Name, Description

```
   ...:
   ...:         recurse(0, 1)
   ...:
   ...:     tree_to_code(classifier,cols)

In [18]: execute_bot()
Please reply with yes/Yes or no/No for the following symptoms
slurred_speech ?

No
pain_behind_the_eyes ?

No
receiving_unsterile_injections ?

Yes
['You may have Hepatitis B']

symptoms present  ['receiving_unsterile_injections']

symptoms given ['itching', 'fatigue', 'lethargy', 'yellowish_skin', 'dark_urine', 'loss_of_appetite',
'abdominal_pain', 'yellow_urine', 'yellowing_of_eyes', 'malaise', 'receiving_blood_transfusion',
'receiving_unsterile_injections']

confidence level is 0.08333333333333333

The model suggests:

Consult  ['Dr. Yogesh Jain']

Visit  ['https://www.practo.com/delhi/doctor/dr-yogesh-jain-2-ear-nose-throat-ent-specialist?
specialization=Ear-Nose-Throat%20(ENT)%20Specialist&practice_id=1118463']

In [19]:
```

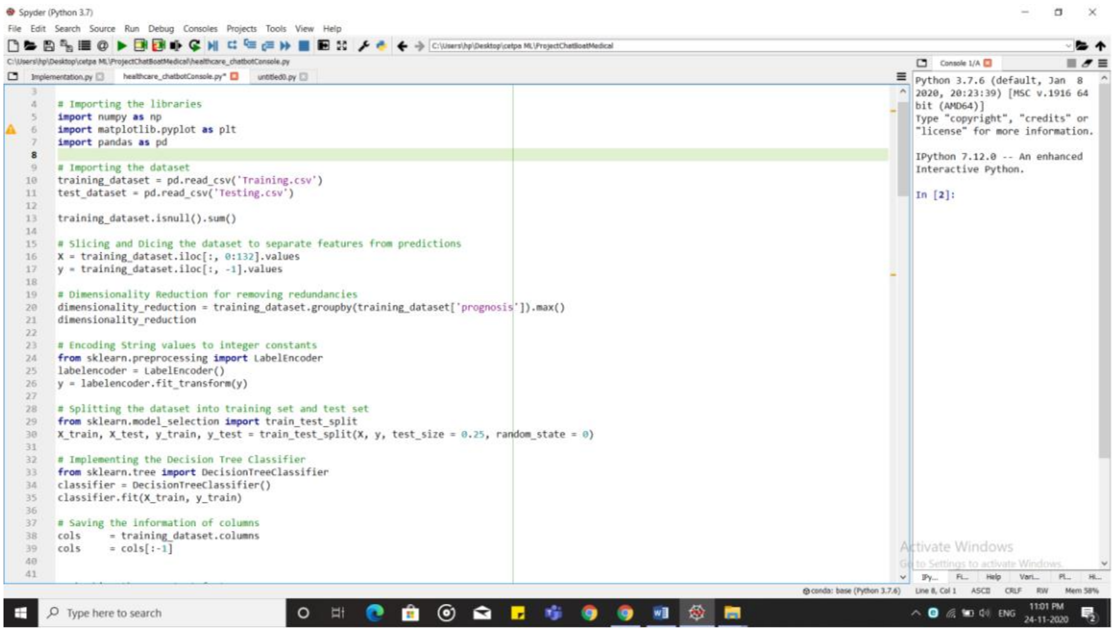
Fig-1



```
In [18]: execute_bot()
Please reply with yes/Yes or no/No for the following symptoms
slurred_speech ?

No
pain_behind_the_eyes ?

No
receiving_unsterile_injections ?

Yes
['You may have Hepatitis B']

symptoms present  ['receiving_unsterile_injections']

symptoms given ['itching', 'fatigue', 'lethargy', 'yellowish_skin', 'dark_urine', 'loss_of_appetite',
'abdominal_pain', 'yellow_urine', 'yellowing_of_eyes', 'malaise', 'receiving_blood_transfusion',
'receiving_unsterile_injections']

confidence level is 0.08333333333333333
```

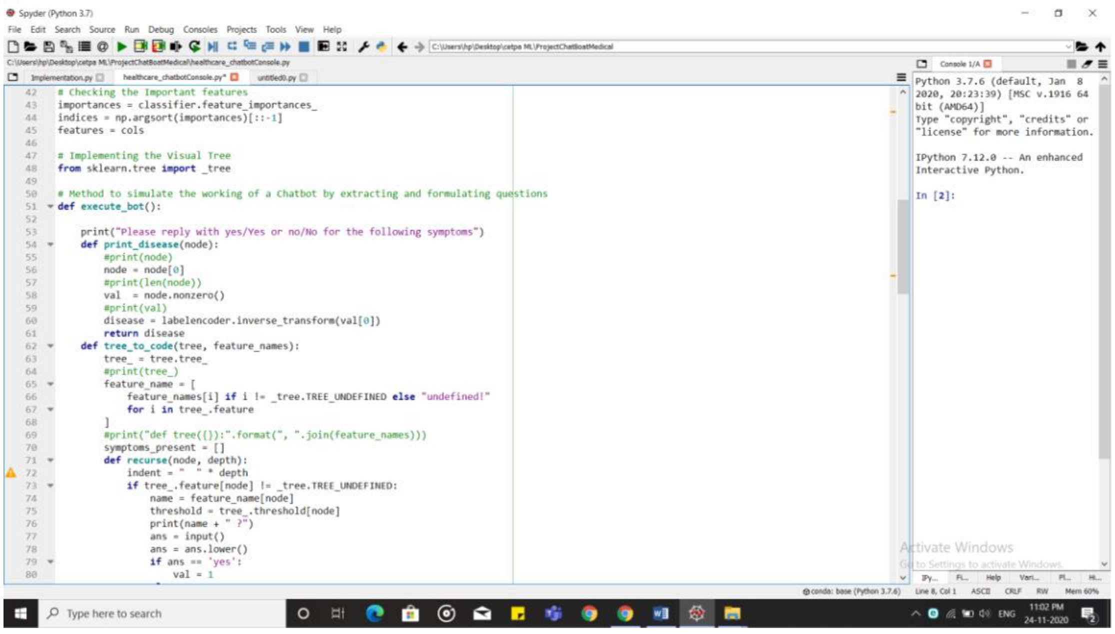
Fig -2

Chapter 5. Implementation and User Interface



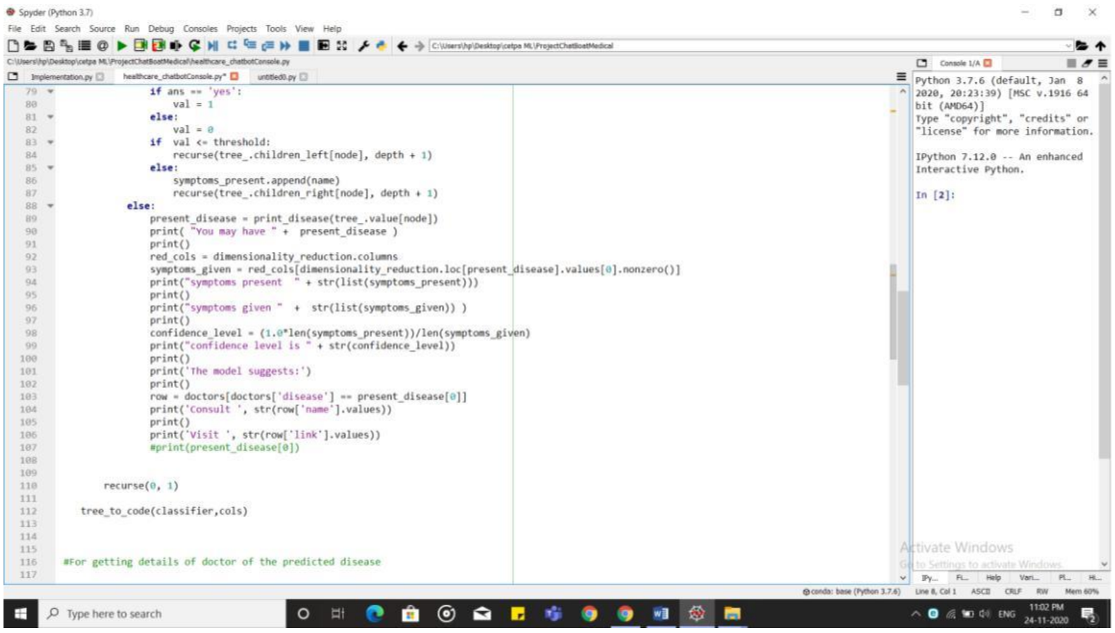
```python
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
training_dataset = pd.read_csv('Training.csv')
test_dataset = pd.read_csv('Testing.csv')

training_dataset.isnull().sum()

# Slicing and Dicing the dataset to separate features from predictions
X = training_dataset.iloc[:, 0:132].values
y = training_dataset.iloc[:, -1].values

# Dimensionality Reduction for removing redundancies
dimensionality_reduction = training_dataset.groupby(training_dataset['prognosis']).max()
dimensionality_reduction

# Encoding String values to integer constants
from sklearn.preprocessing import LabelEncoder
labelencoder = LabelEncoder()
y = labelencoder.fit_transform(y)

# Splitting the dataset into training set and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Implementing the Decision Tree Classifier
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(X_train, y_train)

# Saving the information of columns
cols     = training_dataset.columns
cols     = cols[:-1]
```

```
Python 3.7.6 (default, Jan  8
2020, 20:23:39) [MSC v.1916 64
bit (AMD64)]
Type "copyright", "credits" or
"license" for more information.

IPython 7.12.0 -- An enhanced
Interactive Python.

In [2]:
```



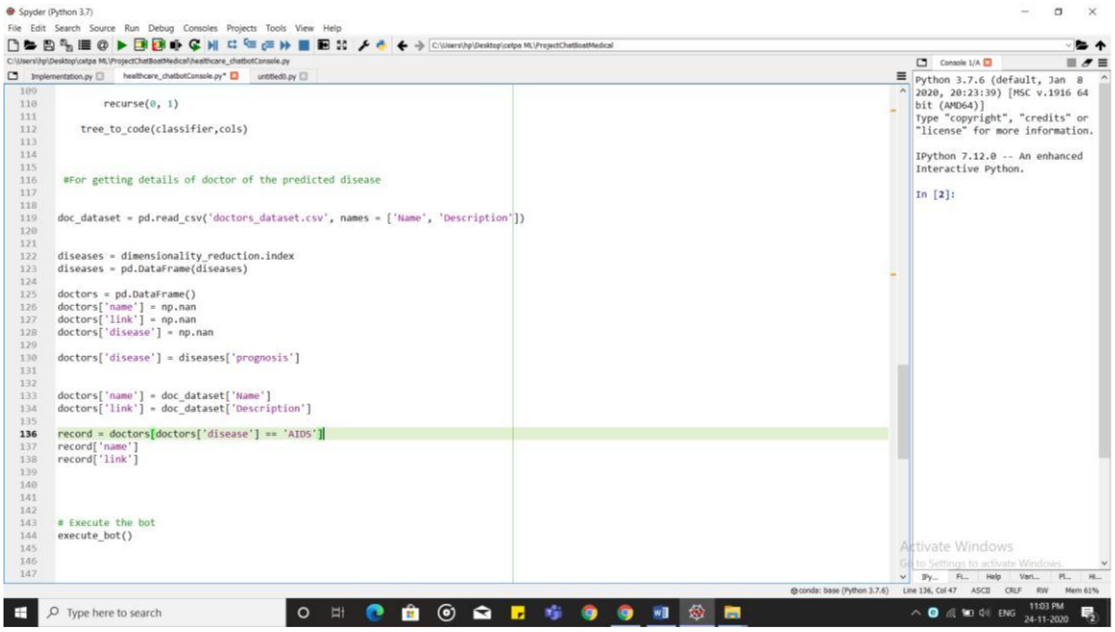
```python
# Checking the Important features
importances = classifier.feature_importances_
indices = np.argsort(importances)[::-1]
features = cols

# Implementing the Visual Tree
from sklearn.tree import _tree

# Method to simulate the working of a Chatbot by extracting and formulating questions
def execute_bot():

    print("Please reply with yes/Yes or no/No for the following symptoms")
    def print_disease(node):
        #print(node)
        node = node[0]
        #print(len(node))
        val  = node.nonzero()
        #print(val)
        disease = labelencoder.inverse_transform(val[0])
        return disease
    def tree_to_code(tree, feature_names):
        tree_ = tree.tree_
        #print(tree_)
        feature_name = [
            feature_names[i] if i != _tree.TREE_UNDEFINED else "undefined!"
            for i in tree_.feature
        ]
        #print("def tree({}):".format(", ".join(feature_names)))
        symptoms_present = []
        def recurse(node, depth):
            indent = "  " * depth
            if tree_.feature[node] != _tree.TREE_UNDEFINED:
                name = feature_name[node]
                threshold = tree_.threshold[node]
                print(name + " ?")
                ans = input()
                ans = ans.lower()
                if ans == 'yes':
                    val = 1
```

```
Python 3.7.6 (default, Jan  8
2020, 20:23:39) [MSC v.1916 64
bit (AMD64)]
Type "copyright", "credits" or
"license" for more information.

IPython 7.12.0 -- An enhanced
Interactive Python.

In [2]:
```



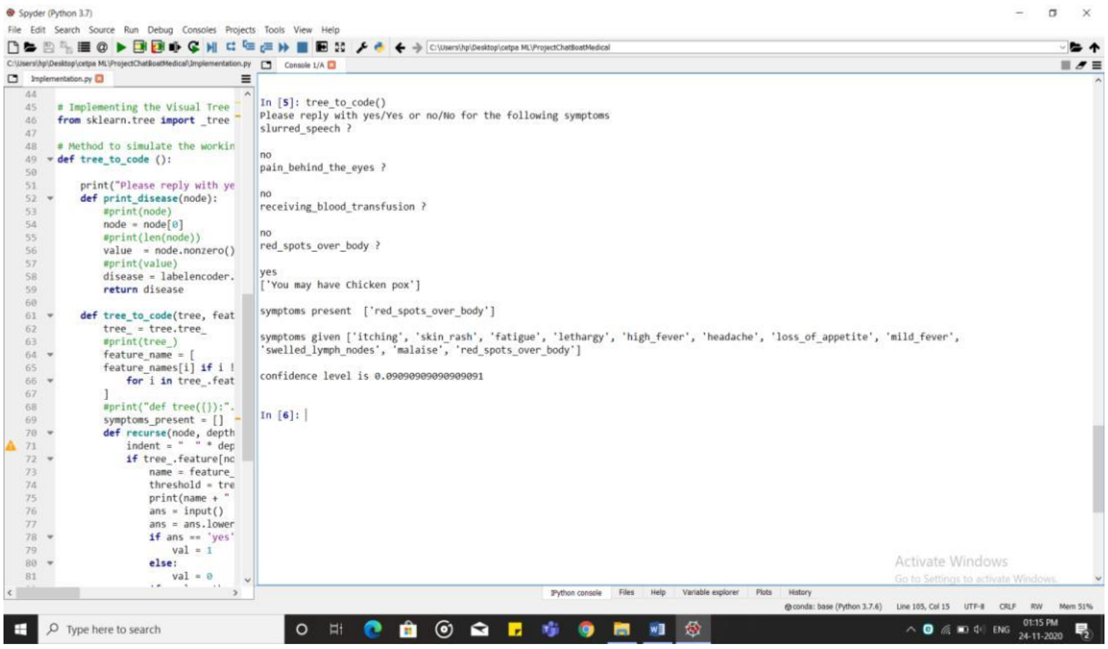
```
                if ans == 'yes':
                    val = 1
                else:
                    val = 0
                if  val <= threshold:
                    recurse(tree_.children_left[node], depth + 1)
                else:
                    symptoms_present.append(name)
                    recurse(tree_.children_right[node], depth + 1)
            else:
                present_disease = print_disease(tree_.value[node])
                print( "You may have " +  present_disease )
                print()
                red_cols = dimensionality_reduction.columns
                symptoms_given = red_cols[dimensionality_reduction.loc[present_disease].values[0].nonzero()]
                print("symptoms present  " + str(list(symptoms_present)))
                print()
                print("symptoms given "  +  str(list(symptoms_given)) )
                print()
                confidence_level = (1.0*len(symptoms_present))/len(symptoms_given)
                print("confidence level is " + str(confidence_level))
                print()
                print('The model suggests:')
                print()
                row = doctors[doctors['disease'] == present_disease[0]]
                print('Consult ', str(row['name'].values))
                print()
                print('Visit ', str(row['link'].values))
                #print(present_disease[0])


        recurse(0, 1)

    tree_to_code(classifier,cols)



 #For getting details of doctor of the predicted disease

```

```
Python 3.7.6 (default, Jan  8
2020, 20:23:39) [MSC v.1916 64
bit (AMD64)]
Type "copyright", "credits" or
"license" for more information.

IPython 7.12.0 -- An enhanced
Interactive Python.

In [2]:
```



```

        recurse(0, 1)

    tree_to_code(classifier,cols)



 #For getting details of doctor of the predicted disease


doc_dataset = pd.read_csv('doctors_dataset.csv', names = ['Name', 'Description'])


diseases = dimensionality_reduction.index
diseases = pd.DataFrame(diseases)

doctors = pd.DataFrame()
doctors['name'] = np.nan
doctors['link'] = np.nan
doctors['disease'] = np.nan

doctors['disease'] = diseases['prognosis']


doctors['name'] = doc_dataset['Name']
doctors['link'] = doc_dataset['Description']

record = doctors[doctors['disease'] == 'AIDS']
record['name']
record['link']




# Execute the bot
execute_bot()
```

```
Python 3.7.6 (default, Jan  8
2020, 20:23:39) [MSC v.1916 64
bit (AMD64)]
Type "copyright", "credits" or
"license" for more information.

IPython 7.12.0 -- An enhanced
Interactive Python.

In [2]:
```



```
# Implementing the Visual Tree
from sklearn.tree import _tree

# Method to simulate the workin
def tree_to_code ():

    print("Please reply with ye
    def print_disease(node):
        #print(node)
        node = node[0]
        #print(len(node))
        value  = node.nonzero()
        #print(value)
        disease = labelencoder.
        return disease

    def tree_to_code(tree, feat
        tree_ = tree.tree_
        #print(tree_)
        feature_name = [
        feature_names[i] if i !
            for i in tree_.feat
        ]
        #print("def tree({}):".
        symptoms_present = []
        def recurse(node, depth
            indent = "  " * dep
            if tree_.feature[nc
                name = feature_
                threshold = tre
                print(name + "
                ans = input()
                ans = ans.lower
                if ans == 'yes'
                    val = 1
                else:
                    val = 0
```

```
In [5]: tree_to_code()
Please reply with yes/Yes or no/No for the following symptoms
slurred_speech ?

no
pain_behind_the_eyes ?

no
receiving_blood_transfusion ?

no
red_spots_over_body ?

yes
['You may have Chicken pox']

symptoms present  ['red_spots_over_body']

symptoms given ['itching', 'skin_rash', 'fatigue', 'lethargy', 'high_fever', 'headache', 'loss_of_appetite', 'mild_fever',
'swelled_lymph_nodes', 'malaise', 'red_spots_over_body']

confidence level is 0.09090909090909091


In [6]: 
```

Chapter 6 <u>**Appendices**</u>

6.1 Coding of project:

```
# Importing the libraries
import numpy as np
import    matplotlib.pyplot    as
plt import pandas as pd

# Importing the dataset
train_dataset = pd.read_csv('Training.csv')
test_dataset = pd.read_csv('Testing.csv')

train_dataset.isnull().sum()

# Slicing   and   Dicing   the   dataset   to   separate   features   from
predictions X = train_dataset.iloc[:, 0:132].values
y = train_dataset.iloc[:, -1].values

# Dimensionality Reduction for removing redundancies
minimised_dataset                                                     =
train_dataset.groupby(train_dataset['prognosis']).max() minimised_dataset

# Encoding  String  values  to  integer  constants
from          sklearn.preprocessing          import
LabelEncoder labelencoder = LabelEncoder()
```

```
y = labelencoder.fit_transform(y)

# Splitting the dataset into training set and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25,
random_state = 0)

# Implementing the Decision Tree Classifier
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(X_train, y_train)

# Saving columns
column    = train_dataset.columns
column    = column[:-1]



# Checking the Important features importances
= classifier.feature_importances_ indices =
np.argsort(importances)[::-1] features =
column

# Implementing the Visual Tree
from sklearn.tree import _tree

# Method to simulate the working of a Chatbot by extracting and formulating
questions
```

```
def execute_bot():

    print("Please reply with yes/Yes or no/No for the following symptoms")
    def print_disease(node):
        #print(node)
        node = node[0]
        #print(len(node))
        val = node.nonzero()
        #print(val)
        disease = labelencoder.inverse_transform(val[0])
        return disease
    def tree_to_code(tree, feature_names):
        tree_ = tree.tree_
        #print(tree_)
        feature_name = [
            feature_names[i] if i != _tree.TREE_UNDEFINED else
            "undefined!" for i in tree_.feature
        ]
        symptoms_present = []
        def recurse(node, depth):
            indent = " " * depth
            if tree_.feature[node] != _tree.TREE_UNDEFINED:
                name = feature_name[node]
                threshold = tree_.threshold[node]
                print(name + " ?")
                ans = input()
                ans = ans.lower()
                if ans == 'yes':
                    val = 1
                else:
                    val = 0
                if val <= threshold:
                    recurse(tree_.children_left[node], depth + 1)
                else:
                    symptoms_present.append(name)
                    recurse(tree_.children_right[node], depth + 1)
            else:
                present_disease = print_disease(tree_.value[node])
                print( "You may have " + present_disease )
                print()
                red_cols = minimised_dataset.columns
                symptoms_given =
red_cols[minimised_dataset.loc[present_disease].values[0].nonzero()]
                print("symptoms present " + str(list(symptoms_present)))
                print()
                print("symptoms given " + str(list(symptoms_given)) )
                print()
                confidence_level =
(1.0*len(symptoms_present))/len(symptoms_given)
                print("confidence level is " + str(confidence_level))
                print()
                print('The model suggests:')
                print()
                row = doctors[doctors['disease'] == present_disease[0]]
```

```
                print('Consult ', str(row['name'].values))
                print()
                print('Visit ', str(row['link'].values))
                #print(present_disease[0])



            recurse(0, 1)

        tree_to_code(classifier,column)




#For getting details of doctor of the predicted disease

doc_dataset = pd.read_csv('doctor.csv', names = ['Name', 'Description'])



diseases = minimised_dataset.index
diseases = pd.DataFrame(diseases)

doctors = pd.DataFrame()
doctors['name'] = np.nan
doctors['link'] = np.nan
doctors['disease'] = np.nan

doctors['disease'] = diseases['prognosis']
```

```
doctors['name'] = doc_dataset['Name']
doctors['link'] = doc_dataset['Description']

record = doctors[doctors['disease'] == 'AIDS']
record['name']
record['link']




# Execute the bot
execute_bot()
```

Future Scope

There are some of the tasks that can be done in future work related to this project study-

- GUI can be developed for more effective use.
- The design of an intelligent system can be prepared that can show the information of doctors, based on the location.

Bibliography/References

[1] https://www.analyticsvidhya.com/blog/

[2] https://medium.com/data-science-simplified/

[3] https://data-flair.training/blogs/

[4] https://www.wikipedia.org/

[5] https://www.python.org/

[6] https://pandas.pydata.org/

[7] https://www.geeksforgeeks.org/

[8] https://www.anaconda.com/