MINI PROJECT - I

(2020-21)

HEALTHBOT

MID TERM REPORT

Institute of Engineering & Technology

Team Members

Achal Paliwal

(181500032)

Akshita Saxena

(181500066)

Nandani Bansal

(181500413)

Prashant Agrawal

(181500486)

Shashank Gupta

(181500653)

Supervised By

Mr. Mohd. Amir Khan

Technical Trainer

Department of Computer Engineering & Applications

Contents

Abstract

The project named "**Health Care Chat-bot**", is a Window-based application 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.

1. Introduction

We all know that medical field is growing there are lots of soft wares 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.

In this project, when a patient start the project by clicking start button model will start showing the symptoms 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.

2. Objective

In "**Health care Chat-bot**" 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.

3. Modules

The project is based on several modules:

3.1 Product perspective

1. User Interface: The application will have a user-friendly and menu-based interface. Following frames will be provided.
2. A login frame for entering the username, the password will be provided. Access to main screen of the model.
3. There is a frame for displaying information regarding disease of the patient and the link of the doctor to contact.
4. There is a frame for displaying Symptoms so that a patient can select right one analyze disease.
5. There is a button to start the model when the patient will click on it the model will starting asking symptoms to the patient.
6. There is a frame for displaying the disease of the patient.

3.2 Product Functions

The Model will allow access only to authorized users or the user who have registered themselves already in it. A summary of the major functions that the model will perform:

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

3.2.1 Patient

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

3.2.2 Doctors

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

3.3 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.

3.4 General Constraints

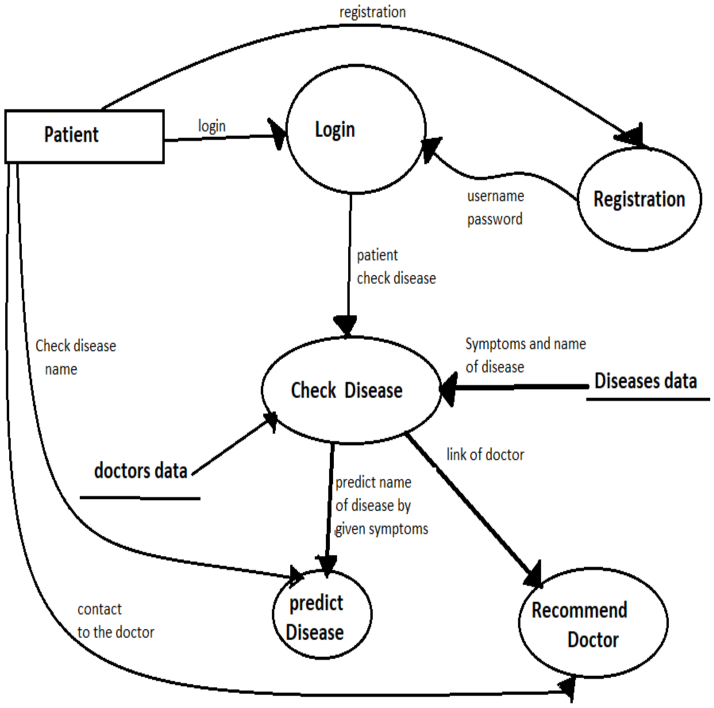
- Since the CSV Files is used as data source for this project so there can be some noise in data.
- Due to the less features of Tkinter framework the GUI for this project is moderate.
- An extra login feature is added to authorize the person who want's is using it
- Programming is done by Python3.

4. Specific Requirements

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

- **Hardware Requirement**
  - Processor - Intel i5
  - Operating System – Windows /8/10, Linux, Mac OS
  - RAM – 4GB(minimum)
  - Hard disk – 64 GB
  - Hardware Devices – Computer System
- **Tools Required**
  - Anaconda Navigator
  - Spyder
  - MS Excel

5. Implementation



PART 1. Get the Data

PART 2. Discovery & Visualization to gain insights

PART 3. Data Preprocessing

PART 4. Select and train a machine learning model for prediction of disease

PART 5. Testing

PART 6. Creating GUI

6. Progress

Part 1 is completed

PART 1: Get the Data

- Collect training data
- Collect testing data

Part 2 is completed

PART 2: Discovery & Visualization to gain insights

- Discover correlations
- Experiment with attribute combinations

Part 3 is completed

PART 3: Data Preprocessing

- Handling missing values
- Handling categorical values

Part 4 is completed

PART 4: Select and train a Machine Learning model for prediction of disease

- Experiment with various algorithms

Part 5 is pending

PART 5: Testing

- Testing model to measure performance

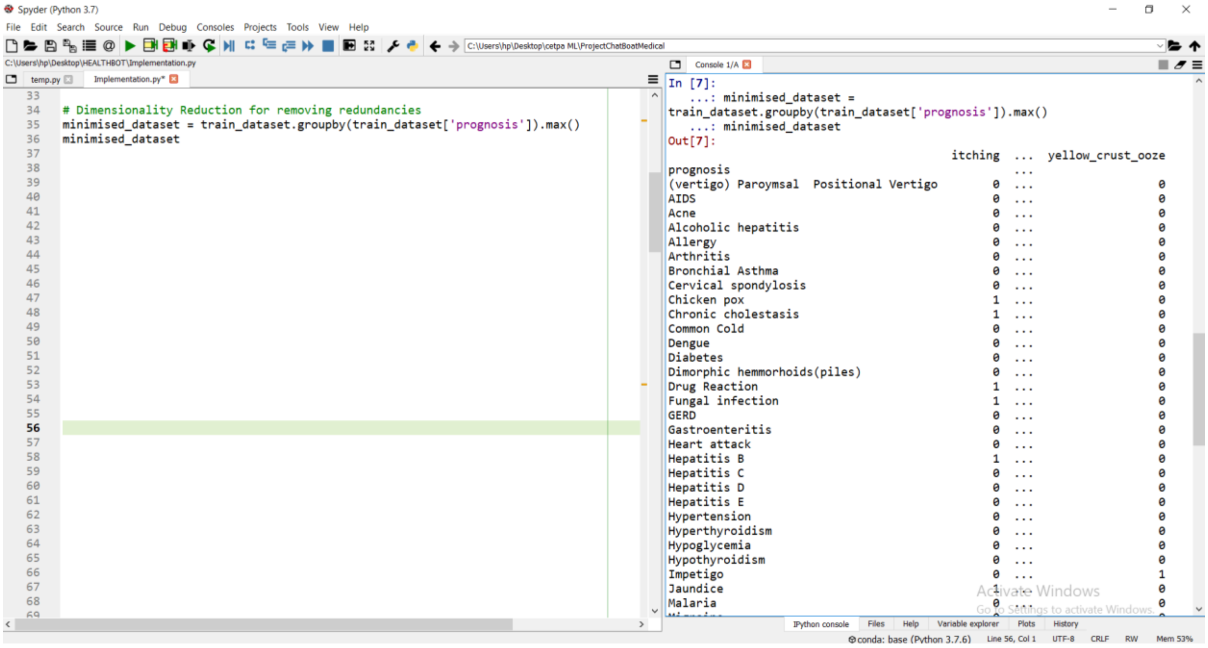
Part 6 is pending

PART 6: Creating GUI

- **Patient Registration Screen**
  - Registration Name
  - Password

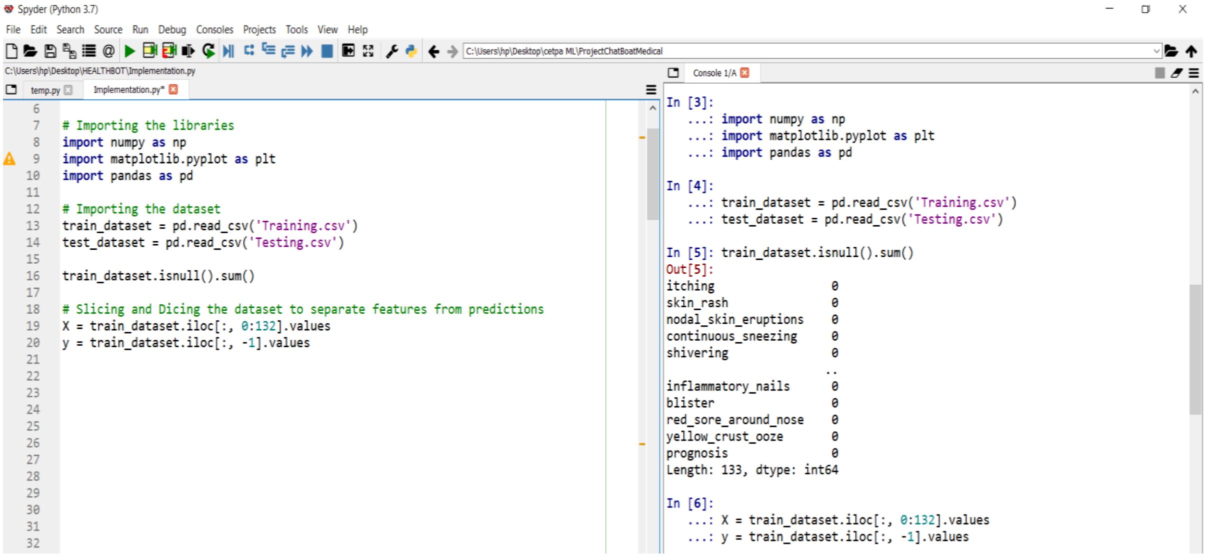
- **Patient Login Screen**
  - Login Name
  - Password
- **Main Frame**
  - Start Button
  - Yes/No Button
  - Symptoms Showing Frame
  - Result Frame
- **Symptoms Asking Frame**
  - Symptoms Name
- **Disease and Doctor prediction frame**
  - Disease Name
  - Doctor Link

7. Screenshots


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hp\Desktop\cetpa ML\ProjectChatBoatMedical
C:\Users\hp\Desktop\HEALTHBOT\Implementation.py
temp.py
Implementation.py*
Console 1/A
IPython console
Files
Help
Variable explorer
Plots
History
conda: base (Python 3.7.6)
Line 56, Col 1
UTF-8
CRLF
RW
Mem 53%

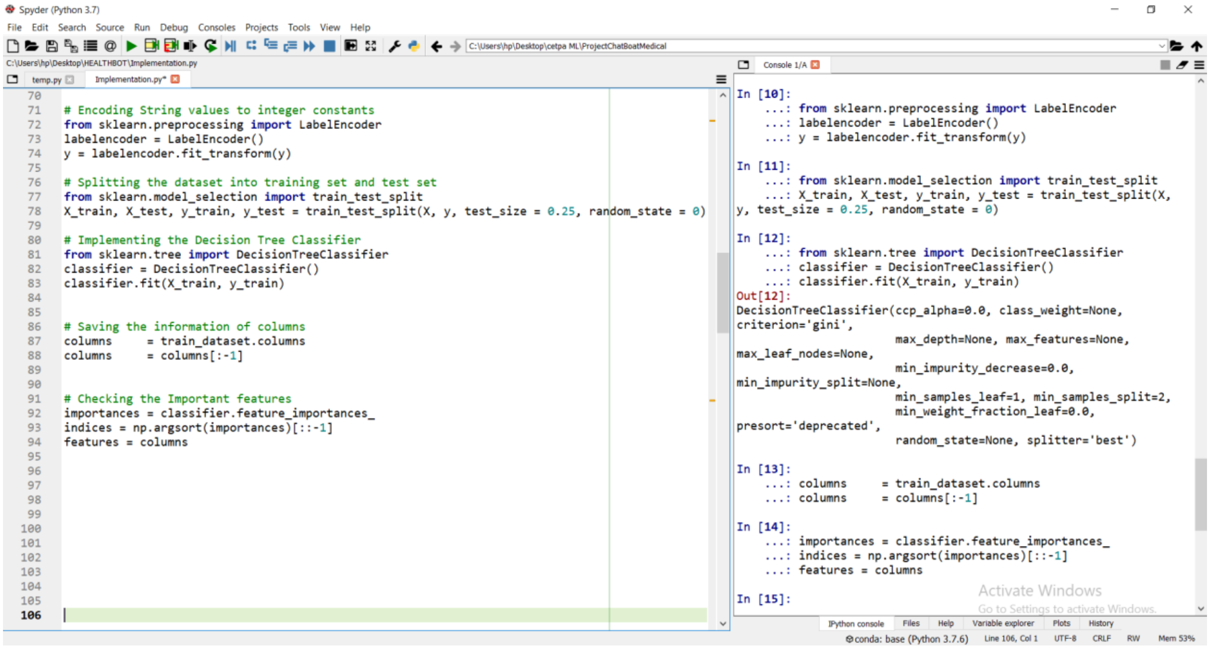
```
# Dimensionality Reduction for removing redundancies
minimised_dataset = train_dataset.groupby(train_dataset['prognosis']).max()
minimised_dataset
```

```
In [7]:
   ...: minimised_dataset =
train_dataset.groupby(train_dataset['prognosis']).max()
   ...: minimised_dataset
Out[7]:
                                     itching  ...  yellow_crust_ooze
prognosis                                     ...
(vertigo) Paroymsal  Positional Vertigo    0  ...                  0
AIDS                                       0  ...                  0
Acne                                       0  ...                  0
Alcoholic hepatitis                        0  ...                  0
Allergy                                    0  ...                  0
Arthritis                                  0  ...                  0
Bronchial Asthma                           0  ...                  0
Cervical spondylosis                       0  ...                  0
Chicken pox                                1  ...                  0
Chronic cholestasis                        1  ...                  0
Common Cold                                0  ...                  0
Dengue                                     0  ...                  0
Diabetes                                   0  ...                  0
Dimorphic hemmorhoids(piles)               0  ...                  0
Drug Reaction                              1  ...                  0
Fungal infection                           1  ...                  0
GERD                                       0  ...                  0
Gastroenteritis                            0  ...                  0
Heart attack                               0  ...                  0
Hepatitis B                                1  ...                  0
Hepatitis C                                0  ...                  0
Hepatitis D                                0  ...                  0
Hepatitis E                                0  ...                  0
Hypertension                               0  ...                  0
Hyperthyroidism                            0  ...                  0
Hypoglycemia                               0  ...                  0
Hypothyroidism                             0  ...                  0
Impetigo                                   0  ...                  1
Jaundice                                   1  ...                  0
Malaria                                    0  ...                  0
```


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hp\Desktop\cetpa ML\ProjectChatBoatMedical
C:\Users\hp\Desktop\HEALTHBOT\Implementation.py
temp.py
Implementation.py*
Console 1/A

```

# 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
```

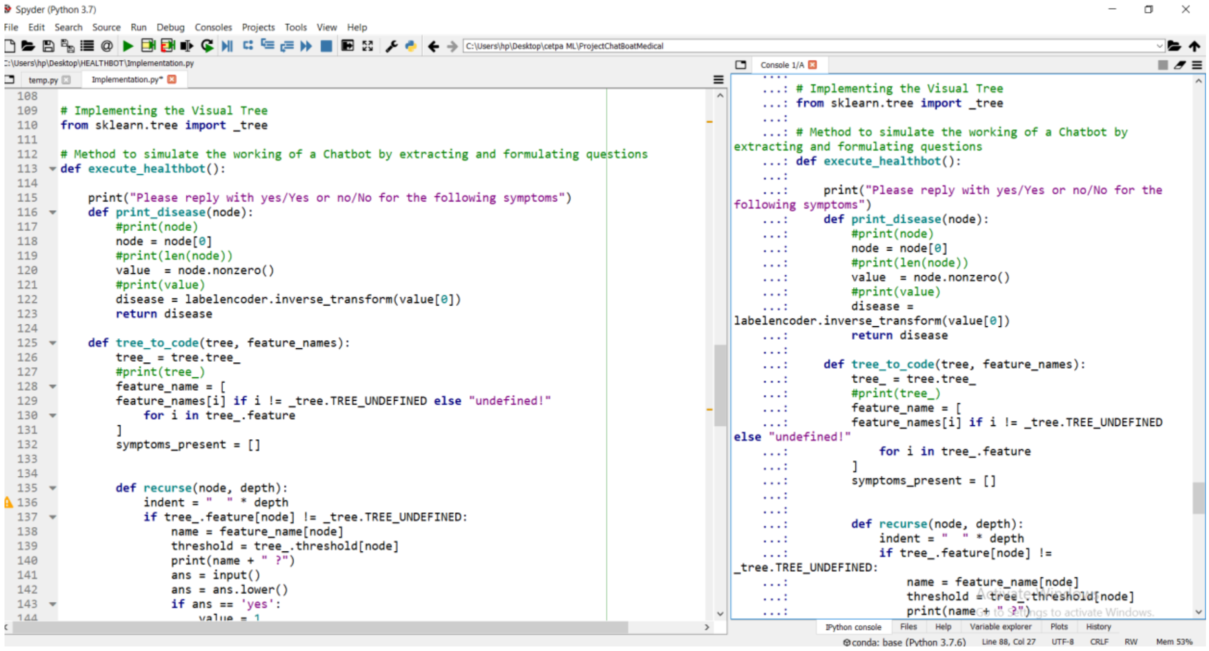
```
In [3]:
   ...: import numpy as np
   ...: import matplotlib.pyplot as plt
   ...: import pandas as pd

In [4]:
   ...: train_dataset = pd.read_csv('Training.csv')
   ...: test_dataset = pd.read_csv('Testing.csv')

In [5]: train_dataset.isnull().sum()
Out[5]:
itching                 0
skin_rash               0
nodal_skin_eruptions    0
continuous_sneezing     0
shivering               0
                       ..
inflammatory_nails      0
blister                 0
red_sore_around_nose    0
yellow_crust_ooze       0
prognosis               0
Length: 133, dtype: int64

In [6]:
   ...: X = train_dataset.iloc[:, 0:132].values
   ...: y = train_dataset.iloc[:, -1].values
```


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hp\Desktop\cetpa ML\ProjectChatBoatMedical
C:\Users\hp\Desktop\HEALTHBOT\Implementation.py
temp.py
Implementation.py*
Console 1/A
IPython console Files Help Variable explorer Plots History
conda: base (Python 3.7.6) Line 106, Col 1 UTF-8 CRLF RW Mem 53%

```python

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


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

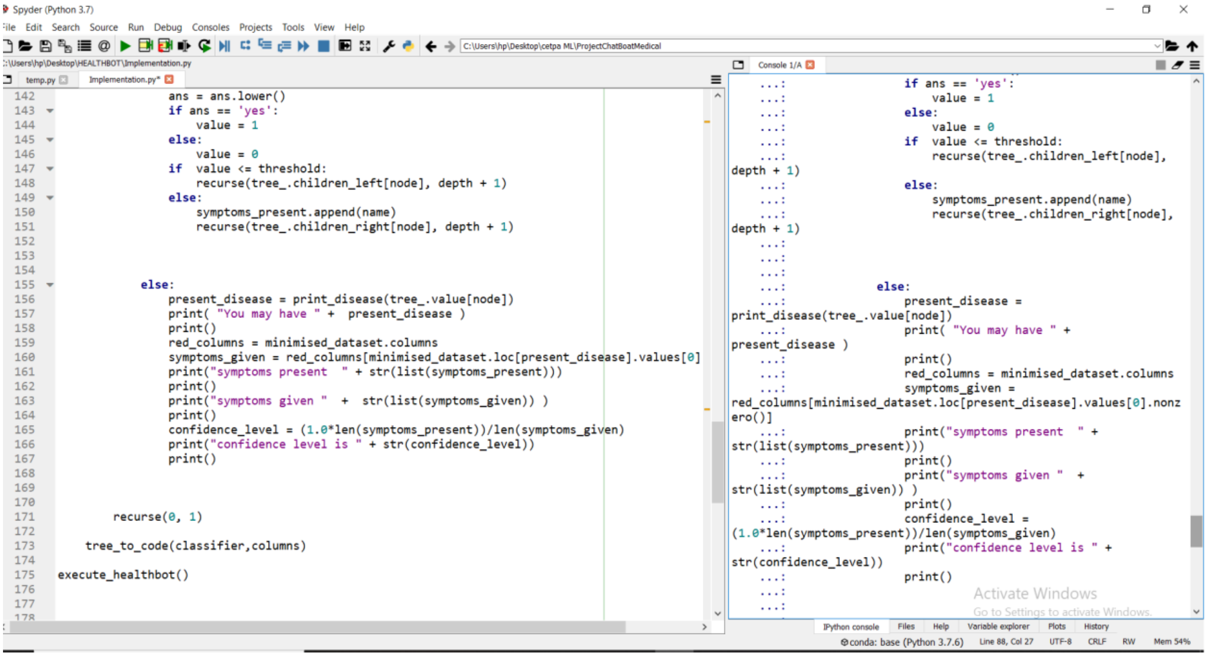
```
In [10]:
    ...: from sklearn.preprocessing import LabelEncoder
    ...: labelencoder = LabelEncoder()
    ...: y = labelencoder.fit_transform(y)

In [11]:
    ...: 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)

In [12]:
    ...: from sklearn.tree import DecisionTreeClassifier
    ...: classifier = DecisionTreeClassifier()
    ...: classifier.fit(X_train, y_train)
Out[12]:
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
                       max_depth=None, max_features=None,
max_leaf_nodes=None,
                       min_impurity_decrease=0.0,
min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0,
presort='deprecated',
                       random_state=None, splitter='best')

In [13]:
    ...: columns      = train_dataset.columns
    ...: columns      = columns[:-1]

In [14]:
    ...: importances = classifier.feature_importances_
    ...: indices = np.argsort(importances)[::-1]
    ...: features = columns

In [15]:
```

Activate Windows
Go to Settings to activate Windows.


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hp\Desktop\cetpa ML\ProjectChatBoatMedical
:\Users\hp\Desktop\HEALTHBOT\Implementation.py
temp.py
Implementation.py*
Console 1/A
IPython console Files Help Variable explorer Plots History
conda: base (Python 3.7.6) Line 88, Col 27 UTF-8 CRLF RW Mem 53%

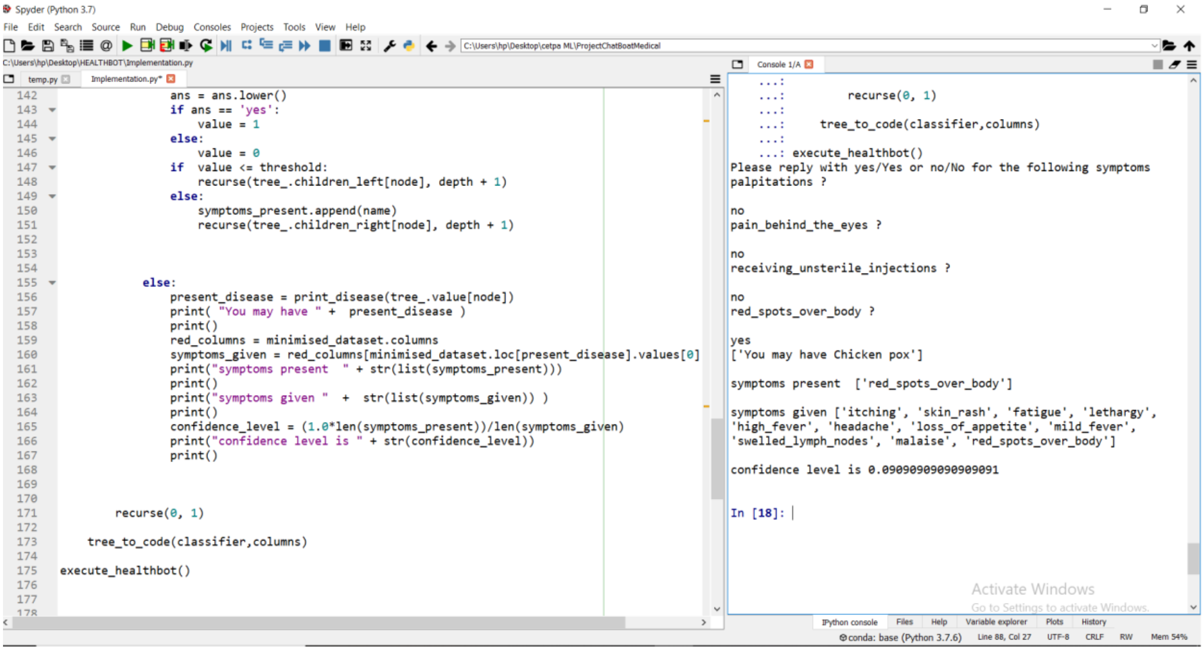
```python

# Implementing the Visual Tree
from sklearn.tree import _tree

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

    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))
        value  = node.nonzero()
        #print(value)
        disease = labelencoder.inverse_transform(value[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':
                    value = 1
```

```
    ...:
    ...: # Implementing the Visual Tree
    ...: from sklearn.tree import _tree
    ...:
    ...: # Method to simulate the working of a Chatbot by
extracting and formulating questions
    ...: def execute_healthbot():
    ...:
    ...:     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))
    ...:         value  = node.nonzero()
    ...:         #print(value)
    ...:         disease =
labelencoder.inverse_transform(value[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 + " ?")
```


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hp\Desktop\cetpa ML\ProjectChatBoatMedical
C:\Users\hp\Desktop\HEALTHBOT\Implementation.py
temp.py
Implementation.py*
Console 1/A
IPython console
Files
Help
Variable explorer
Plots
History
conda: base (Python 3.7.6)
Line 88, Col 27
UTF-8
CRLF
RW
Mem 54%

```
                ans = ans.lower()
                if ans == 'yes':
                    value = 1
                else:
                    value = 0
                if  value <= 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_columns = minimised_dataset.columns
                symptoms_given = red_columns[minimised_dataset.loc[present_disease].values[0]
                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()



        recurse(0, 1)

    tree_to_code(classifier,columns)

execute_healthbot()
```

```
    ...:                     if ans == 'yes':
    ...:                         value = 1
    ...:                     else:
    ...:                         value = 0
    ...:                     if  value <= 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_columns = minimised_dataset.columns
    ...:                     symptoms_given =
red_columns[minimised_dataset.loc[present_disease].values[0].nonz
ero()]
    ...:                     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()
    ...:
    ...:
```


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hp\Desktop\cetpa ML\ProjectChatBoatMedical
C:\Users\hp\Desktop\HEALTHBOT\Implementation.py
temp.py
Implementation.py*
Console 1/A
IPython console
Files
Help
Variable explorer
Plots
History
conda: base (Python 3.7.6)
Line 88, Col 27
UTF-8
CRLF
RW
Mem 54%

```
                ans = ans.lower()
                if ans == 'yes':
                    value = 1
                else:
                    value = 0
                if  value <= 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_columns = minimised_dataset.columns
                symptoms_given = red_columns[minimised_dataset.loc[present_disease].values[0]
                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()



        recurse(0, 1)

    tree_to_code(classifier,columns)

execute_healthbot()
```

```
    ...:
    ...:         recurse(0, 1)
    ...:
    ...:     tree_to_code(classifier,columns)
    ...:
    ...: execute_healthbot()
Please reply with yes/Yes or no/No for the following symptoms
palpitations ?

no
pain_behind_the_eyes ?

no
receiving_unsterile_injections ?

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 [18]:
```

8. References

The following references were used in this project:

1. https://www.python.org
2. https://www.geeksforgeeks.org
3. https://www.anaconda.org
4. https://www.wikipedia.org
5. https://www.numpy.org
6. https://www.analyticsvidhya.com/blog/2020/06/4-ways-split-decision-tree/