

HEALTHCARE CHATBOT FOR DISEASE
PREDICTION AND RECOMMENDATION USING
MACHINE LEARNING IN PYTHON

BACHELOR OF ENGINEERING IN
COMPUTER SCIENCE & ENGINEERING



Submitted To:

Er. Shreya Bansal

Submitted By :

Prakeerti Misra

18BCS2956

Rohil Nagar

18BCS2925

Afaq Sheikh

18BCS2882

Prateek Sandhey

18BCS2854

CONTENTS:

1. Abstract
2. Introduction about the project
3. Feasibility Study
4. Methodology
5. Hardware Software Requirements
6. Work till date
 - Phase 1: Datasets
 - Phase 2: New login page
 - Phase 3: Questions Diagnosis GUI using Tkinter
7. Bibliography

INTRODUCTION

A chatbot is a software application used to conduct an on-line chat conversation via text or text-to-speech, in lieu of providing direct contact with a live human agent.^[1] Designed to convincingly simulate the way a human would behave as a conversational partner, chatbot systems typically require continuous tuning and testing, and many in production remain unable to adequately converse or pass the industry standard Turing test. The term "ChatterBot" was originally coined by Michael Mauldin (creator of the first Verbot) in 1994 to describe these conversational programs. Chatbots are used in dialog systems for various purposes including customer service, request routing, or for information gathering. While some chatbot applications use extensive word-classification processes, natural language processors, and sophisticated AI, others simply scan for general keywords and generate responses using common phrases obtained from an associated library or database. Most chatbots are accessed on-line via website popups or through virtual assistants. They can be classified into usage categories that include: commerce (e-commerce via chat), education, entertainment, finance, health, news, and productivity.

Healthcare Chatbot:-

A Healthcare chatbot is a fully automated piece of software that has a conversation with your prospects to capture and qualify leads in your digital marketing campaigns.

Feasibility Study

Chatbots have been storming the industry since last decade. Many big companies have made their own healthcare chatbots, these include: IBM Watson, Mfine, Practo etc. Benefits of employing bots in healthcare

- Provide answers to frequently asked questions quickly and efficiently
- Schedule appointments and consultations
- Track patients' care to reduce readmissions
- Send alerts and notifications for prescription refills and care guidelines
- Update record systems with patients' medical history
- Allow the exchange of data from currently disparate health systems
- Automate data entry to significantly reduce error and avoid double entry

Methodology

The project consists of a basic chatbot which reads through the dataset, looks for trends in it and provides the conclusions on the basis of the trends. The project also recommends the type of doctor the user should see on the basis of the solutions. The project is divided into 3 major parts.

1. Getting the data: The first and foremost step towards our bot is to collect data. The main work will be done on this data to provide optimal solutions. We will scrap a health dataset and divide it into test and training sets. Further we will look for doctors data set for the recommendation part.
2. Making a GUI (graphical user interface): It is really important for the developer to make its software user friendly i.e. easy to understand by the user and easy manageable by them. This part will comprise of a login page and a page that will comprise of questions that the chatbot will ask the user. It is these questions, on behalf of which our algorithm will come into action and give suitable results regarding which disease is a person prone to.
3. Healthcare chatbot console: Last we have the console where all exact programming of the chatbot will occur. This is the place where algorithms come into action, previous packages that were created are used to give it a final look. Applying the correct algorithms: To get optimal results we have to study the data, its type, clean the data, find of trends using classifications and regressions. Later we have to decide and apply the best possible algorithm out of naïve byes, decision tree etc. The final prompt, final coding takes place in this phase to give our project a final look.

Software And Hardware Requirements

There are many platforms available for creating a chatbot. You can use that. Or if you are for sure to create a chatbot from scratch then I would recommend python as the language of choice. More detailed thought process would be required to think about hardware. All will depend upon the chatbot usecase. Number of users who will use, core purpose of the chatbot etc.

The basic hardware requirements are: (a basic PC OR laptop with these specification)

- 8GB RAM
- 256GB SSD
- Windows 10
- I5 7th Gen atleast

The basic software requirements are:

- MS EXCEL
- Anaconda Navigator to perform coding in Python (we generally use spyder)

SPYDER: SCIENTIFIC PYTHON DEVELOPMENT ENVIRONMENT

Team Work Till Date

The project has successfully implemented its first and 50% of second phase. The first phase required us to collect the data. The data on which we need to apply algorithms and get the results for our project. The second phase required us to apply the correct algorithms out of the immense number of algorithms available for us. Let's dig into the work we got so far.

PHASE 1: Getting the data

First, we brought up our test data set which includes 132 columns and 43 rows. The 132 columns contain the list of symptoms that our chatbot will be looking through and the 43 rows contains the 0-1 probability of a person having that symptom. 0 stand for NO and 1 stands for YES. The last column of our dataset contains what disease a person had when he went through selected symptoms.

a) Test Set:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	itching	skin_rash	nodal_skin	continuous	shivering	chills	joint_pain	stomach_acidity	ulcers_on	muscle_wi	vomiting	burning_m	spotting_	fatigue	weight_ga	anxiety	cold_hand	mood_swi	weight_lo	restlessne	lethargy	patches_irr	
2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	
5	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
6	1	1	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	
7	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	
10	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
16	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	
17	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
18	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	
19	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	
20	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
21	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
22	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	
25	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
26	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
27	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	
28	0	Testing	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	

We can clearly see in the last column the names of the disease a person was diagnosed with earlier.

[illegible]

c) Doctors Dataset:

This is the dataset we got from GITHUB which includes the names of doctors with their emails and contact information. Once our algorithms are applied to give the best results our application will take you to a page where you can find links and contact information of doctors regarding your related disease.

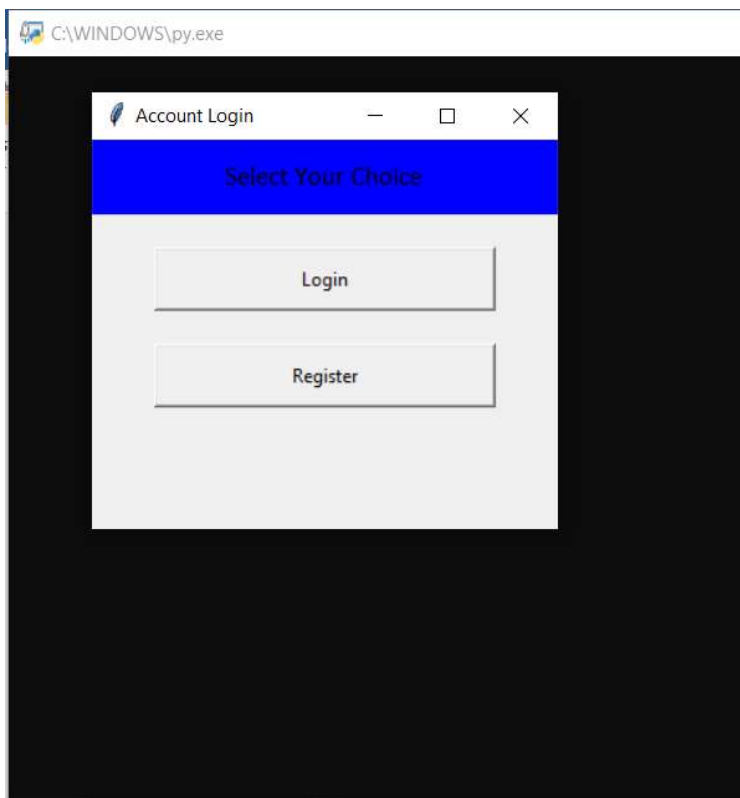
[illegible]

Phase 2: New login page

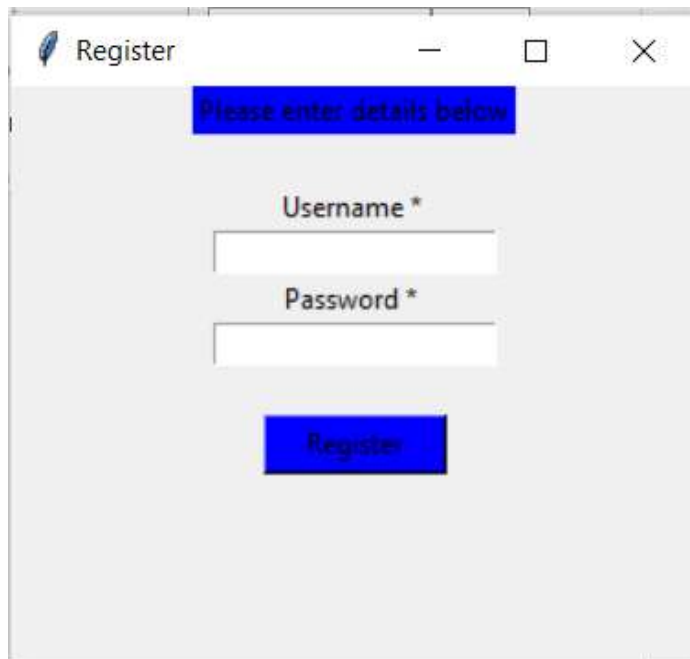
In this phase our team has successfully implemented the code to create a new login page. This is a part of graphical user interface to make our application more user friendly.

Here is our login page:

STEP1:

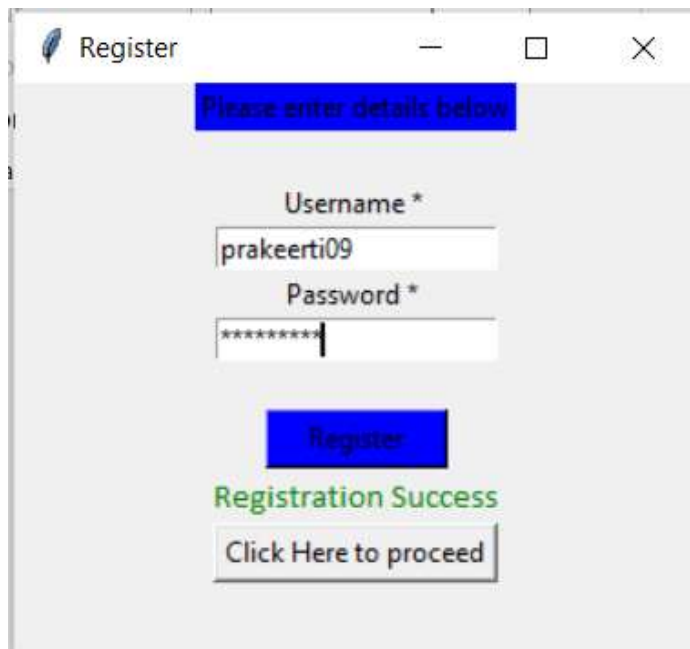


STEP 2:



A screenshot of a web browser window titled "Register". The window has a light gray background. At the top, there is a blue banner with the text "Please enter details below". Below the banner, there are two input fields: "Username *" and "Password *". Both fields are empty. Below the input fields, there is a blue button labeled "Register".

STEP 3:



A screenshot of the same "Register" web browser window. The "Username *" field now contains the text "prakeerti09". The "Password *" field contains a series of asterisks "*****". Below the input fields, there is a blue button labeled "Register". Below the button, there is a green text message "Registration Success" and a button labeled "Click Here to proceed".

Phase 3: Questions Diagnosis GUI using Tkinter

Question	
Digonosis	
No	Yes
Clear	Start

This is a package we created names questions() which we will be calling in the main chatbot console to get full chatbot experience. We used tkinter to give it a display, a user specific display. In this simply a question will be put up in the questions box and the patient just has to press Yes or No according to their symptoms. These results will be saved and processed by our algorithms in the next and the last phase of the project. In the last phase we will be introducing our datasets to train our machine to give the most optimal results.

Phase 4: Chatbot console

This is the final phase of our chatbot. In this phase we introduce the test and training datasets and apply algorithms to it to get the output result. The chatbot runs perfectly in the python console.

This phase has stages of its own

I. Introducing the libraries and the datasets

```
1 ##### A Healthcare Domain Chatbot to simulate the predictions of a General Physician ####
2 ##### A pragmatic Approach for Diagnosis #####
3
4 # Importing the libraries
5 import numpy as np
6 import matplotlib.pyplot as plt
7 import pandas as pd
8
9 # Importing the dataset
10 training_dataset = pd.read_csv('Training.csv')
11 test_dataset = pd.read_csv('Testing.csv')
12
```

II. Cleaning the data and preparing it for analysis

```
13 # Slicing and Dicing the dataset to separate features from predictions
14 X = training_dataset.iloc[:, 0:132].values
15 y = training_dataset.iloc[:, -1].values
16
17 # Dimensionality Reduction for removing redundancies
18 dimensionality_reduction = training_dataset.groupby(training_dataset['prognosis']).max()
19
20 # Encoding String values to integer constants
21 from sklearn.preprocessing import LabelEncoder
22 labelencoder = LabelEncoder()
23 y = labelencoder.fit_transform(y)
24
25 # Splitting the dataset into training set and test set
26 from sklearn.model_selection import train_test_split
27 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
28
```

III. Applying the decision tree algorithm

Decision tree algorithm: Decision Tree algorithm belongs to the family of supervised learning algorithms. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision **rules** inferred from prior data(training data). In Decision Trees, for predicting a class label for a record we start from the **root** of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

```

28
29 # Implementing the Decision Tree Classifier
30 from sklearn.tree import DecisionTreeClassifier
31 classifier = DecisionTreeClassifier()
32 classifier.fit(X_train, y_train)
33
34 # Saving the information of columns
35 cols = training_dataset.columns
36 cols = cols[:-1]
37
38
39 # Checking the Important features
40 importances = classifier.feature_importances_
41 indices = np.argsort(importances)[::-1]
42 features = cols
43
44 # Implementing the Visual Tree
45 from sklearn.tree import _tree
46

```

IV. Method to simulate the working of a Chatbot by extracting and formulating questions

```

47 # Method to simulate the working of a Chatbot by extracting and formulating questions
48 def execute_bot():
49
50     print("Please reply with yes/Yes or no/No for the following symptoms")
51     def print_disease(node):
52         #print(node)
53         node = node[0]
54         #print(len(node))
55         val = node.nonzero()
56         #print(val)
57         disease = labelencoder.inverse_transform(val[0])
58         return disease
59     def tree_to_code(tree, feature_names):
60         tree_ = tree.tree_
61         #print(tree_)
62         feature_name = [
63             feature_names[i] if i != _tree.TREE_UNDEFINED else "undefined!"
64             for i in tree_.feature
65         ]
66         #print("def tree({}):".format(", ".join(feature_names)))
67         symptoms_present = []
68         def recurse(node, depth):
69             indent = " " * depth
70             if tree_.feature[node] != _tree.TREE_UNDEFINED:
71                 name = feature_name[node]
72                 threshold = tree_.threshold[node]
73                 print(name + " ?")
74                 ans = input()
75                 ans = ans.lower()
76                 if ans == 'yes':
77                     val = 1
78                 else:
79                     val = 0
80                 if val <= threshold:
81                     recurse(tree_.children_left[node], depth + 1)
82                 else:
83                     symptoms_present.append(name)
84                     recurse(tree_.children_right[node], depth + 1)
85             else:
86                 present_disease = print_disease(tree_.value[node])
87                 print("You may have " + present_disease)

```



```

84         recurse(tree_.children_right[node], depth + 1)
85     else:
86         present_disease = print_disease(tree_.value[node])
87         print( "You may have " + present_disease )
88         print()
89         red_cols = dimensionality_reduction.columns
90         symptoms_given = red_cols[dimensionality_reduction.loc[present_disease].value]
91         print("symptoms present " + str(list(symptoms_present)))
92         print()
93         print("symptoms given " + str(list(symptoms_given)) )
94         print()
95         confidence_level = (1.0*len(symptoms_present))/len(symptoms_given)
96         print("confidence level is " + str(confidence_level))
97         print()
98         print('The model suggests:')
99         print()
100        row = doctors[doctors['disease'] == present_disease[0]]
101        print('Consult ', str(row['name'].values))
102        print()
103        print('Visit ', str(row['link'].values))
104        #print(present_disease[0])
105
106
107        recurse(0, 1)
108
109        tree_to_code(classifier,cols)
110
111

```

V. Introducing the doctors dataset

```

113 # This section of code to be run after scraping the data
114
115 doc_dataset = pd.read_csv('doctors_dataset.csv', names = ['Name', 'Description'])
116
117
118 diseases = dimensionality_reduction.index
119 diseases = pd.DataFrame(diseases)
120
121 doctors = pd.DataFrame()
122 doctors['name'] = np.nan
123 doctors['link'] = np.nan
124 doctors['disease'] = np.nan
125
126 doctors['disease'] = diseases['prognosis']
127
128
129 doctors['name'] = doc_dataset['Name']
130 doctors['link'] = doc_dataset['Description']
131
132 record = doctors[doctors['disease'] == 'AIDS']
133 record['name']
134 record['link']
135

```

VI. Execute the bot and see results:

```

138
139 # Execute the bot and see it in Action
140 execute_bot()
141

```

OUTPUT:

Random case 1:

```
Console 1/A x
In [1]: runcell(0, 'C:/Users/prake/OneDrive/Desktop/sem 5 project/healthcare_chatbotConsole.py')
Please reply with yes/Yes or no/No for the following symptoms
slurred_speech ?

yes
['You may have Hypoglycemia']

symptoms present ['slurred_speech']

symptoms given ['vomiting', 'fatigue', 'anxiety', 'sweating', 'headache', 'nausea',
'blurred_and_distorted_vision', 'excessive_hunger', 'drying_and_tingling_lips', 'slurred_speech',
'irritability', 'palpitations']

confidence level is 0.08333333333333333

The model suggests:

Consult ['Dr. Jyoti Arora Monga']

Visit ['https://www.practo.com/delhi/doctor/dr-jyoti-arora-ayurveda?
specialization=Ayurveda&practice_id=693424']

In [2]:

Removing all variables...
```

Random case 2:

```
In [2]: runcell(0, 'C:/Users/prake/OneDrive/Desktop/sem 5 project/healthcare_chatbotConsole.py')
Please reply with yes/Yes or no/No for the following symptoms
slurred_speech ?

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

The model suggests:

Consult ['Dr. Inderjeet Singh']

Visit ['https://www.practo.com/delhi/doctor/inderjeet-singh-ayurveda-sexologist?
specialization=Homoeopath&practice_id=1219975']
```


Progress rate: Our team members have worked hard and have successfully implemented the whole project.

References And Bibliography

Firstly I would like to thank our mentor Miss Shreya Bansal for helping us through out the project and guiding us in our queries and mistakes.

Next we would extend our tanks to:

- Github
- Kaggle
- Wikipedia
- Courseera
- Udemy
- <https://www.softwebsolutions.com/healthcare-bot-development.html>
- https://scikit-learn.org/stable/modules/naive_bayes.html
- <https://chatbotslife.com/guide-to-making-a-brilliant-chatbot-8a554882205e>
- <https://hbr.org/2015/11/a-refresher-on-regression-analysis>
- <https://machinelearningmastery.com/types-of-classification-in-machinelearning/#:~:text=In%20machine%20learning%2C%20classification%20refers,one%20of%20the%20known%20characters.>

