Team: -Goal Diggers

Lifesy - Health Care

Diagnosis based on symptoms

Problem Statement

There are a few issues that the medical services area faces each day. Nonetheless, innovation can help you stay on top of things and assist at putting you among the top medical services suppliers. Here are some of the primary healthcare problems that this project will solve for a better and healthier world.

- Healthcare web applications
- Social Networking App for Healthcare Professionals
- Cloud + Data analytics Capabilities
- Prediction of diseases with symptoms
- Recommending doctors of nearby area
- Location of pharmacies and hospitals
- Blogs and newspaper of health by single subscription

Project Description

Health information data needs are likewise changing the data looking for conduct and can be seen all throughout the planet. Difficulties looked by numerous individuals are looking on the web for wellbeing data in regards to illnesses, analysis and various medicines. On the off chance that a proposal framework can be made for specialists and medication while utilizing survey mining will save a great deal of time. In this kind of framework, the patient deals with issues in understanding the heterogeneous clinical vocabulary as the clients are laymen. Patient is confused in light of the fact that a lot of clinical data on various mediums are available. The thought behind the recommender system is to adjust to adapt to the uncommon requirements of the wellbeing area related with patients.

Introduction

With the ascent in number of patients and sickness consistently clinical framework is overburdened and with time has gotten overrated in numerous nations. A large portion of the illness includes a conference with specialists to get treated. With adequate information expectation of infection by a calculation can be extremely simple what's more, modest. Expectation of illness by taking a gander at the side effects is an indispensable piece of treatment. In our task we have attempted precisely foresee a sickness by taking a gander at the indications of the patient. We have utilized 4 distinct calculations for this reason and acquired an exactness of 92-95%. Such a framework can have an exceptionally huge potential in patients treatment of things to come. We have additionally planned an intelligent interface to work with connection with the framework. We have likewise endeavored to show also, imagined the consequence of our examination and this undertaking.

Web Development

Front- End: -

1. Created the Frontend - Page’s with the help of fundamental technologies.

2. Like HTML, CSS, JavaScript, bootstrap, JQuery – libraries and some other resources.

3. Like Icon, Fonts, API, Images etc.

4. For the device-compatibility/Responsiveness – used bootstrap & Media Query.

5. created the “COVID tracker and VACCINATION center near me” Using API’s.

6. Created the basic ‘CHATBOT’ UI using JQuery library.

7. Also Customize some web parts like scrolling bar, cursor etc.

8. We have also added the embed “Feedback Form” using Google forms service.

Back-End: -

1. Creating Login sessions for users e.g. doctor and patient.

2. Save the data entered by the user to the database of MYSQL.

3. Create an authentication process with generating ID's.

4. Connection of API's to display the data.

5. Online booking Appointment system for patients developed in PHP.

6. Doctors can easily update their profiles linked to the MYSQL database.

7. Doctors can mention their availability and patients will book the appointment based on availability of doctors.

8. Designing of Dashboards for Patients and Doctors separately.

9. All the required data will be fetched with PHP and display the information in the dashboard.

10. Users can search nearby hospitals.

11. Users can also enter their symptoms and disease prediction GUI built by our data analytics team which is displayed on the website.

Database Collection

Dataset for this project was collected from a study of university of Columbia performed at New York Presbyterian Hospital during 2004. Link of dataset is given below.

http://people.dbmi.columbia.edu/~friedma/Projects/DiseaseSymptomKB/index.html

Library Used

In this project standard libraries for database analysis and model creation are used. The following are the libraries used in this project.

1. tkinter: -

It’s a standard GUI library of python. Python when combined with tkinter provides fast and easy way to create GUI. It provides a powerful object-oriented tool for creating GUI.

It provides various widgets to create GUI some of the prominent ones being:

· Button
· Canvas
· Label
· Entry
· Check Button
· List box
· Message
· Text
· Message box

Some of these were used in this project to create our GUI namely message box, button, label, Option Menu, text and title. Using tkinter we were able to create an interactive GUI for our model.

2. Numpy: -

Numpy is the core library of scientific computing in python. It provides powerful tools to deal with various multi-dimensional arrays in python. It is a general purpose array processing package.

Numpy’s main purpose is to deal with multidimensional homogeneous arrays. It has tools ranging from array creation to its handling. It makes it easier to create a n dimensional array just by using np.zeros() or handle its contents using various other methods such as replace, arrange, random, save, load it also helps I array processing using methods like sum, mean, std, max, min, all, etc.

Array created with numpy also behave differently than arrays created normally when they are operated upon using operators such as +,-,*,/.

All the above qualities and services offered by numpy array makes it highly suitable for our purpose of handling data. Data manipulation occurring in arrays while performing various operations need to give the desired results while predicting outputs require such high operational capabilities.

3. pandas: -

It is the most popular python library used for data analysis. It provides highly optimized performance with back-end source code purely written in C or python.

Data in python can be analyzed with 2 ways

- · Series
- · Data frames

Series is a one dimensional array defined in pandas used to store any data type.

Data frames are two-dimensional data structure used in python to store data consisting of rows and columns.

Pandas data frame is used extensively in this project to use datasets required for training and testing the algorithms. Data frames make it easier to work with attributes and results. Several of
its inbuilt functions such as replace were used in our project for data manipulation and preprocessing.

4. sklearn: -

Sklearn is an open source python library which implements a huge range of machine learning, pre-processing, cross-validation and visualization algorithms. It features various simple and efficient tools for data mining and data processing. It features various classification, regression and clustering algorithm such as support vector machine, random forest classifier, decision tree, Gaussian naïve-Bayes, KNN to name a few.

In this project we have used sklearn to get advantage of inbuilt classification algorithms like decision tree, random forest classifier, KNN and naïve Bayes. We have also used inbuilt cross validation and visualization features such as classification report, confusion matrix and accuracy score.

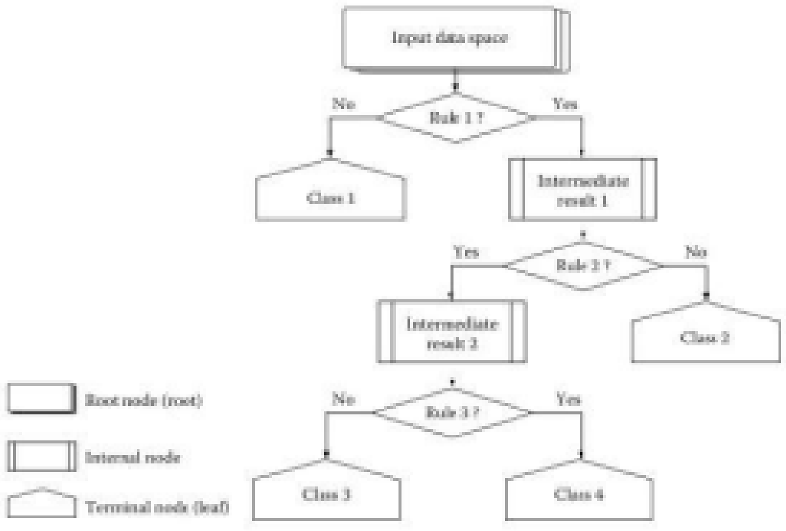
Models

There are four different kinds of models present in our project to predict the disease these are

- · Decision tree
- · Random forest tree
- · Gaussian Naïve Bayes

Decision trees are classified as a very effective and versatile classification technique. It is used in pattern recognition and classification for image. It is used for classification in very complex problems due to its high adaptability. It is also capable of engaging problems of higher dimensionality. It mainly consists of three parts: root, nodes and leaf.

Roots consists of attribute which has most effect on the outcome, leaf tests for value of certain attribute and leaf gives out the output of tree.



Decision tree is the first prediction method we have used in our project. It gives us an accuracy of ~95%.

Random Forest Algorithm is a supervised learning algorithm used for both classification and regression. This algorithm works on 4 basic steps –

1. It chooses random data samples from the dataset.
2. It constructs decision trees for every sample dataset chosen.
3. At this step every predicted result will be compiled and voted on.
4. At last most voted prediction will be selected and be presented as a result of classification.

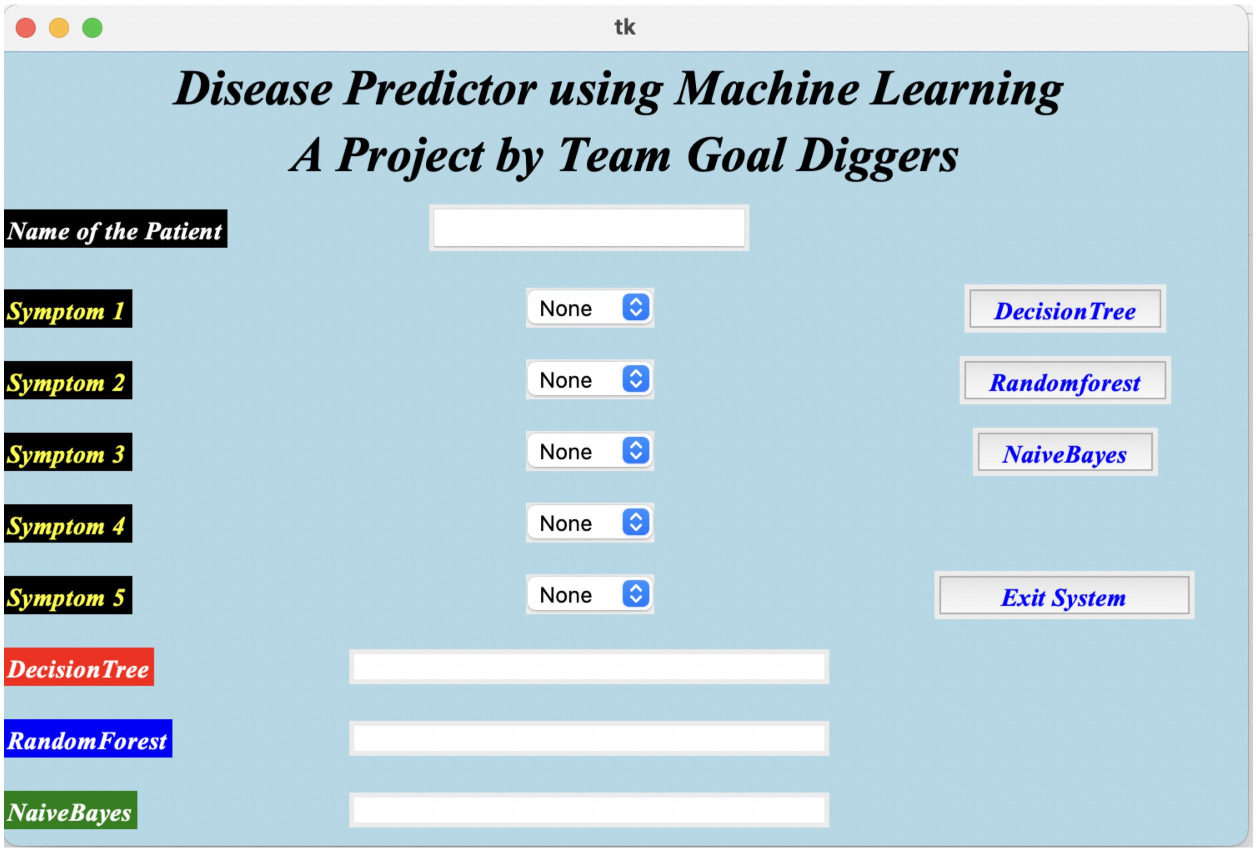
In this project we have used random forest classifier with 100 random samples and the result given is ~95% accuracy.

The Naïve Bayes algorithm is a family of algorithms based on the naïve bayes theorem. They share a common principle that is every pair of prediction is independent of each other. It also makes an assumption that features make an independent and equal contribution to the prediction.

In our project we have used the naïve Bayes algorithm to gain a ~95% accurate prediction.

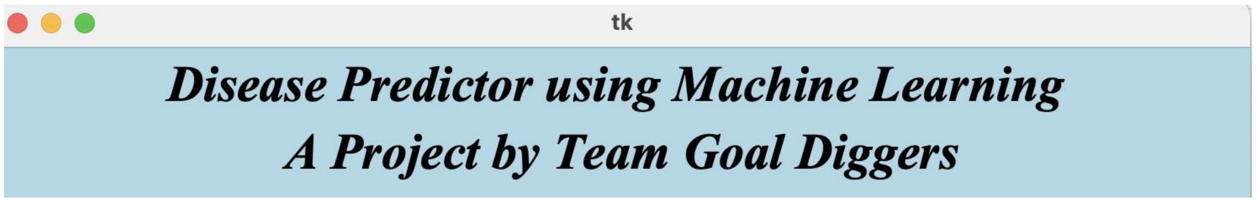
GUI

GUI made for this project is a simple tkinter GUI consisting of labels, message box, button, text, title and option menu

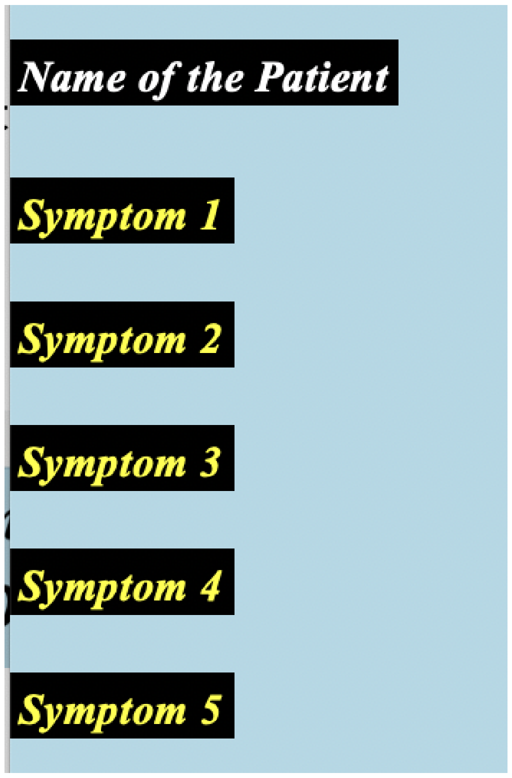


Root.title() is used to set the the title as Smart Disease Predictor System

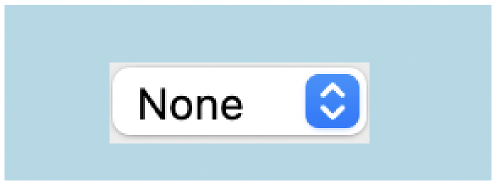
Label is used to add heading and contributors section.


tk
Disease Predictor using Machine Learning
A Project by Team Goal Diggers

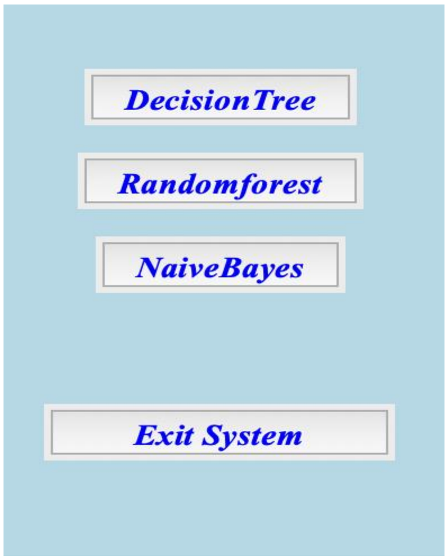
Labels are further used for different sections



Option Menu is used to create drop down menu

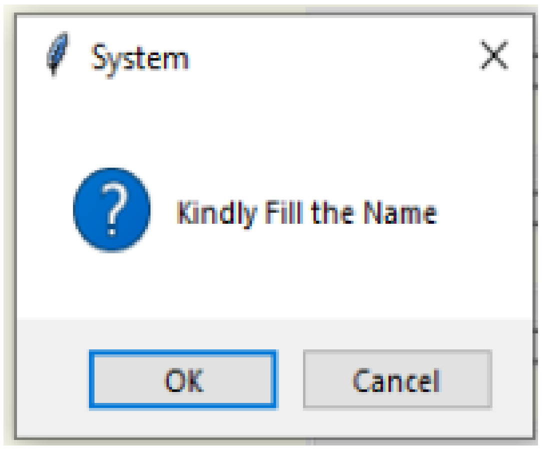


Buttons are used to give functionalities and predict the outcome of models also two utility buttons namely exit and rest are also created.



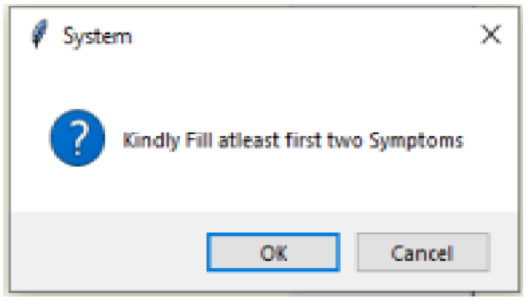
Text is used to show output of the prediction using blank space.
Message box are used at three different places, one- to

restrain the to enter name



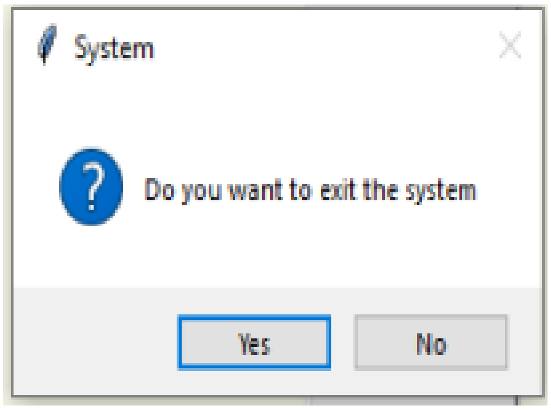
Kindly Fill the Name

Two- to ask for at least two symptoms,



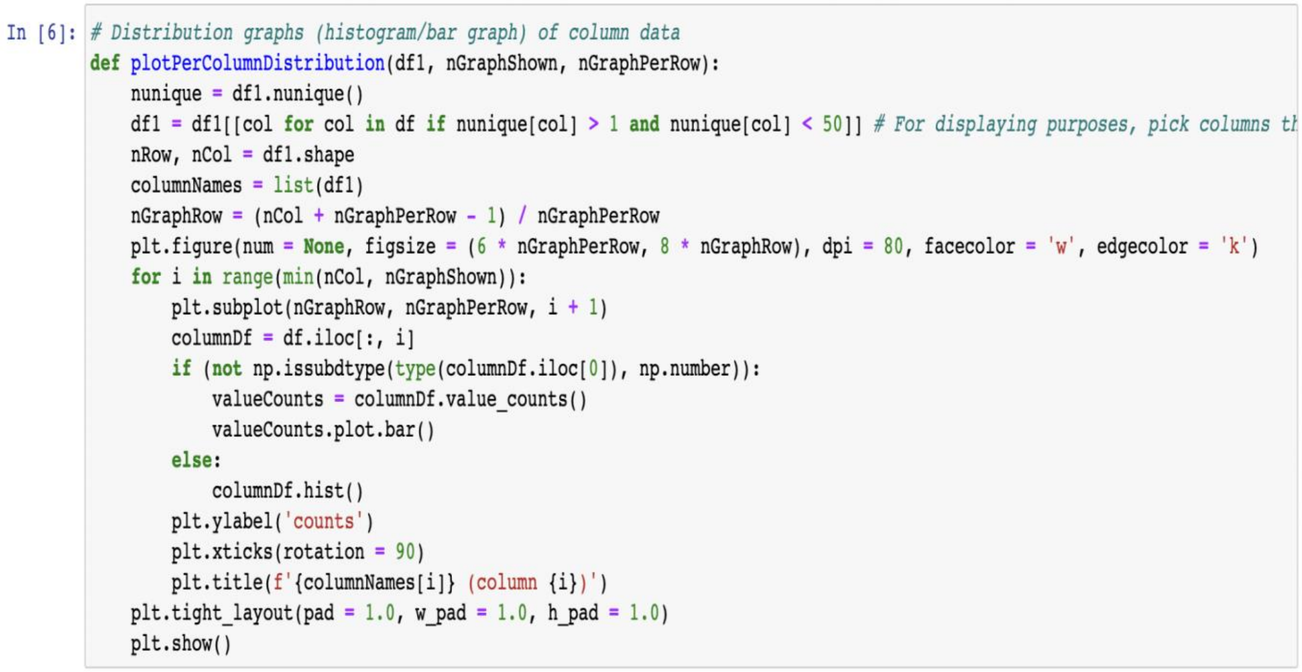
Kindly Fill atleast first two Symptoms

Three- to confirm to exit system.

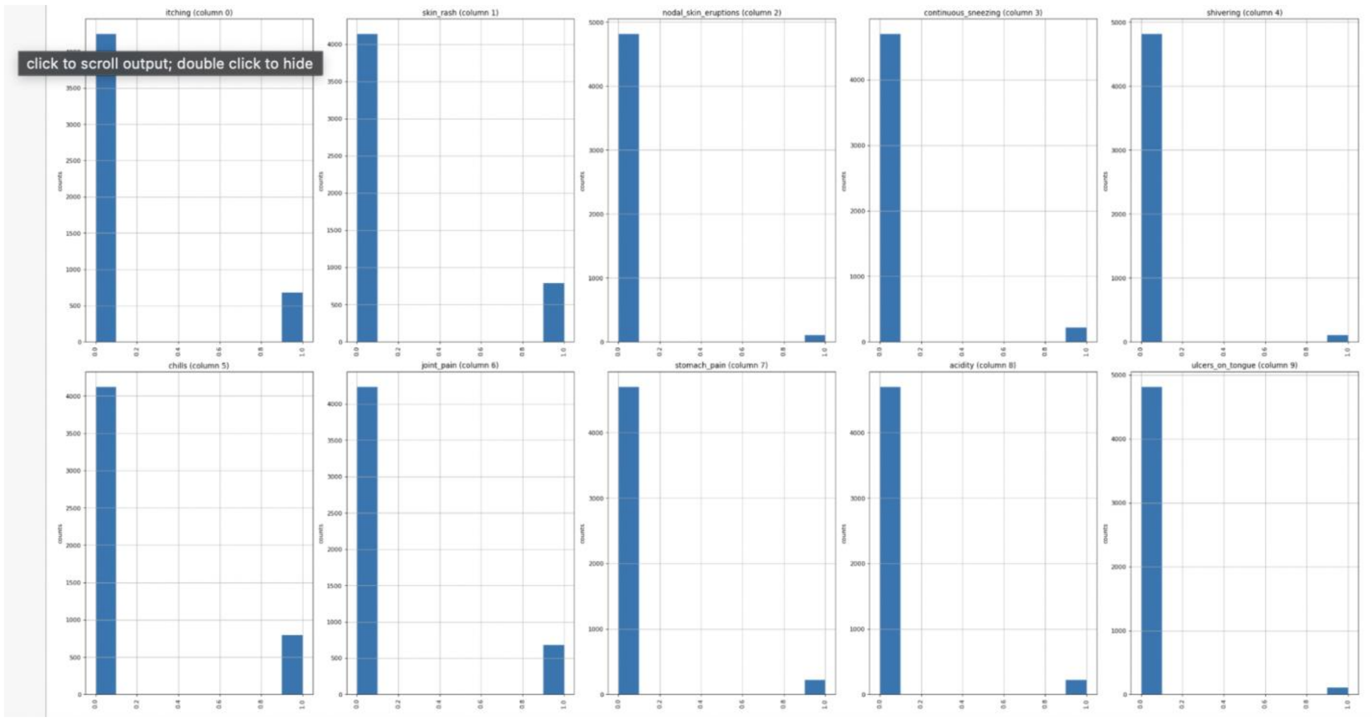


Do you want to exit the system

Modules



```
In [6]: # Distribution graphs (histogram/bar graph) of column data
def plotPerColumnDistribution(df1, nGraphShown, nGraphPerRow):
    nunique = df1.nunique()
    df1 = df1[[col for col in df if nunique[col] > 1 and nunique[col] < 50]] # For displaying purposes, pick columns th
    nRow, nCol = df1.shape
    columnNames = list(df1)
    nGraphRow = (nCol + nGraphPerRow - 1) / nGraphPerRow
    plt.figure(num = None, figsize = (6 * nGraphPerRow, 8 * nGraphRow), dpi = 80, facecolor = 'w', edgecolor = 'k')
    for i in range(min(nCol, nGraphShown)):
        plt.subplot(nGraphRow, nGraphPerRow, i + 1)
        columnDf = df.iloc[:, i]
        if (not np.issubdtype(type(columnDf.iloc[0]), np.number)):
            valueCounts = columnDf.value_counts()
            valueCounts.plot.bar()
        else:
            columnDf.hist()
        plt.ylabel('counts')
        plt.xticks(rotation = 90)
        plt.title(f'{columnNames[i]} (column {i})')
    plt.tight_layout(pad = 1.0, w_pad = 1.0, h_pad = 1.0)
    plt.show()
```

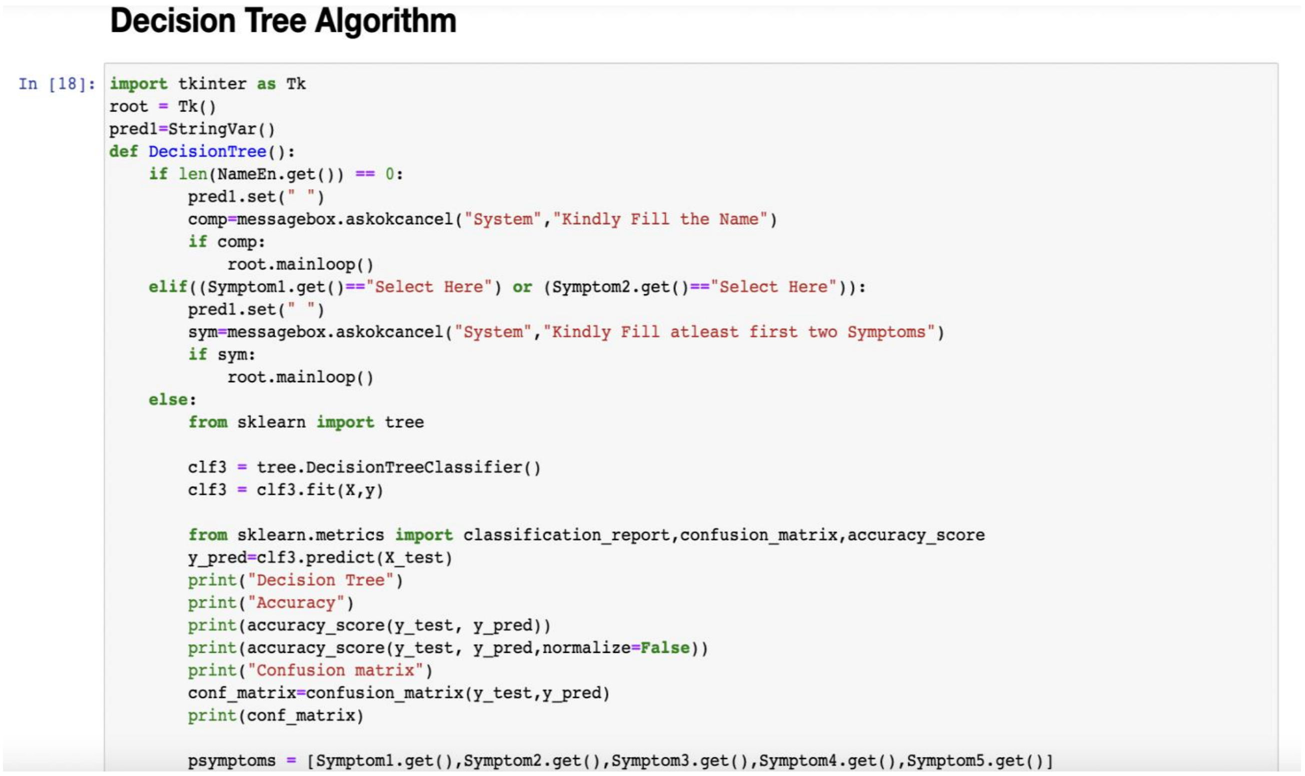


Functions like plotpercolumbdistribution() plotScatterMatrix() is used to visualize the data.

```
#list1 = DF['prognosis'].unique()
def scatterplt(disea):
    x = ((DF.loc[disea]).sum())#total sum of symptom reported for given disease
    x.drop(x[x==0].index,inplace=True)#droping symptoms with values 0
    print(x.values)
    y = x.keys()#storing nameof symptoms in y
    print(len(x))
    print(len(y))
    plt.title(disea)
    plt.scatter(y,x.values)
    plt.show()

def scatterinp(sym1,sym2,sym3,sym4,sym5):
    x = [sym1,sym2,sym3,sym4,sym5]#storing input symptoms in y
    y = [0,0,0,0,0]#creating and giving values to the input symptoms
    if(sym1!='Select Here'):
        y[0]=1
    if(sym2!='Select Here'):
        y[1]=1
    if(sym3!='Select Here'):
        y[2]=1
    if(sym4!='Select Here'):
        y[3]=1
    if(sym5!='Select Here'):
        y[4]=1
    print(x)
    print(y)
    plt.scatter(x,y)
    plt.show()
```

Functions like scatter plt and scattering are used to compare input to training data.



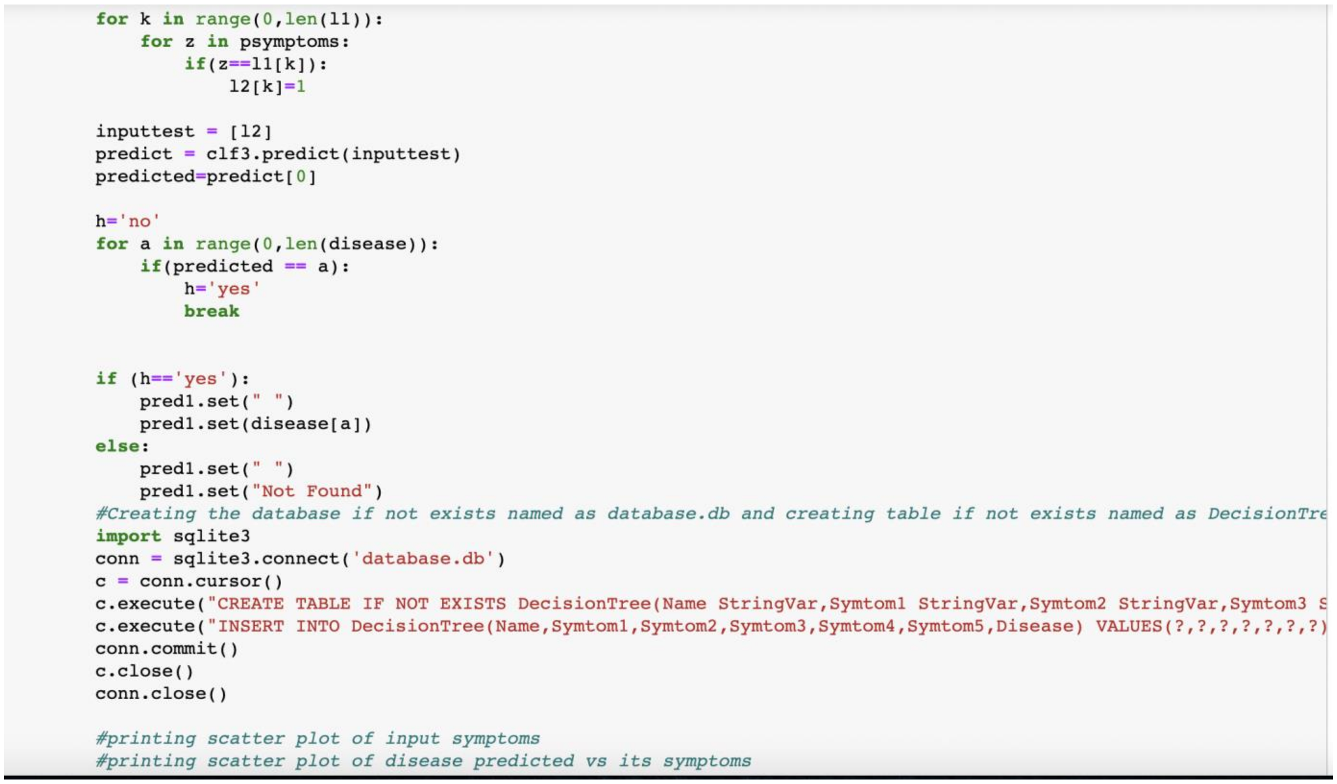
Decision Tree Algorithm

```
In [18]: import tkinter as Tk
         root = Tk()
         pred1=StringVar()
         def DecisionTree():
             if len(NameEn.get()) == 0:
                 pred1.set(" ")
                 comp=messagebox.askokcancel("System","Kindly Fill the Name")
                 if comp:
                     root.mainloop()
             elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
                 pred1.set(" ")
                 sym=messagebox.askokcancel("System","Kindly Fill atleast first two Symptoms")
                 if sym:
                     root.mainloop()
             else:
                 from sklearn import tree

                 clf3 = tree.DecisionTreeClassifier()
                 clf3 = clf3.fit(X,y)

                 from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
                 y_pred=clf3.predict(X_test)
                 print("Decision Tree")
                 print("Accuracy")
                 print(accuracy_score(y_test, y_pred))
                 print(accuracy_score(y_test, y_pred,normalize=False))
                 print("Confusion matrix")
                 conf_matrix=confusion_matrix(y_test,y_pred)
                 print(conf_matrix)

                 psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]
```



```
    for k in range(0,len(l1)):
        for z in psymptoms:
            if(z==l1[k]):
                l2[k]=1

    inputtest = [l2]
    predict = clf3.predict(inputtest)
    predicted=predict[0]

    h='no'
    for a in range(0,len(disease)):
        if(predicted == a):
            h='yes'
            break


    if (h=='yes'):
        pred1.set(" ")
        pred1.set(disease[a])
    else:
        pred1.set(" ")
        pred1.set("Not Found")
    #Creating the database if not exists named as database.db and creating table if not exists named as DecisionTre
    import sqlite3
    conn = sqlite3.connect('database.db')
    c = conn.cursor()
    c.execute("CREATE TABLE IF NOT EXISTS DecisionTree(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 S
    c.execute("INSERT INTO DecisionTree(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)
    conn.commit()
    c.close()
    conn.close()

    #printing scatter plot of input symptoms
    #printing scatter plot of disease predicted vs its symptoms
```

- Algorithm of decision tree and database storage.
- Algorithm of random forest classifier.
- Algorithm of naïve Bayes classifier

All these classifiers are connected to the database and GUI to function seamlessly.

- Code of GUI to set initial values of labels.
- Code of message box.
- Code of option menu.
- Code of buttons.
- Code of result display.

Conclusions

We set out to create a system which can predict disease on the basis of symptoms given to it. Such a system can decrease the rush at OPDs of hospitals and reduce the workload on medical staff. We were successful in creating such a system and use 4 different algorithms to do so. On an average we achieved accuracy of ~94%. Such a system can be largely reliable to do the job. Creating this system we also added a way to store the data entered by the user in the database which can be used in future to help in creating better version of such system. Our system also has an easy to use interface. It also has various visual representations of data collected and results achieved.

References

http://people.dbmi.columbia.edu/~friedma/Projects/DiseaseSymptomKB/index.html

Connect Us: -

Facebook | Instagram | LinkedIn

Goal Diggers