Project Report

on

Disease Prediction Using Symptoms Data Analysis

Submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

Session 2022-23
in

ABES ENGINEERING COLLEGE GHAZIABAD, UTTARPRADESH

By

ANANYA SHUKLA
1900320100023

Under the guidance of

Industrial Guide - Dr Anil Kumar Dubey
Current Mentor – Ms. Shikha Jain

ABES ENGINEERING COLLEGE, GHAZIABAD

AFFILIATED TO
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, U.P., LUCKNOW
(Formerly UPTU)

STUDENT'S DECLARATION

We hereby declare that the work being presented in this report entitled "DISEASE PRIDICTION USING SYMPTOMS" is an authentic record of my own work carried out under the supervision of Dr. Anil Kumar Dubey". The matter embodied in this report has not been submitted by me for the award of any other degree.

Dated: **Signature of student**

Department: CSE

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Dr Anil Kumar Dubey
Associate Professor
CSE Department

CERTIFICATE

This is to certify that Project Report entitled "**DISEASE PRIDICTION USING SYMPTOMS**" which is submitted by Ananya Shukla in partial fulfillment of the requirement for the award of Data Science Project in Department of Computer Science Engineering of Dr. A.P.J. Abdul Kalam Technical University, formerly Uttar Pradesh Technical University is a record of the candidate own work carried out by her under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

Supervisor

Date

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the Data Science Project undertaken during B. Tech. Third Year. I owe special debt of gratitude to Dr. Anil Kumar Dubey, Ms. Shikha Jain, Department of Computer Science & Engineering, ABESEC Ghaziabad for their constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

I also take the opportunity to acknowledge the contribution of Professor (Dr.) Divya Mishra, Head, Department of Computer Science & Engineering, ABESEC Ghaziabad for her full support and assistance during the development of the project.

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

Signature:

Name :

Roll No.:

Date :

ABSTRACT

Correct and on-time evaluation of any fitness-related trouble is important for the prevention and remedy of the infection. The traditional way of analysis may not be enough in the case of a severe ailment. Growing a clinical diagnosis gadget based on device studying (ML) algorithms for the prediction of any disorder can help in an extra accurate analysis than the conventional technique. Based on the symptoms gives the output as the disease that the individual might be suffering from. The project is based on that how to predict a disease using symptoms. The dataset has been taken from Kaggle. Firstly, analyzed the data set by simply opening the "CSV" file. Then I pre-processed the whole data set to remove the insignificant columns and empty rows. Also, to get to know all the unique values of each attribute. Further, I performed Visualization and Some Machine Learning Algorithms on the dataset. Then Exploratory analysis of each to get an idea about how the values are entered in each attribute with reference to no. of entries. All these activities are done using Python Modules on our dataset: Numpy, Pandas, Matplotlib, Seaborn, sklearn.

This dataset consists of 4 CSV files: dataset.csv, symptom_Description.csv, symptom_precaution.csv, and Symptom-severity.csv all these are used for different algorithms of Machine Learning.

TABLE OF CONTENTS

LIST OF TABLES

There are basically Four Tables

1) **Dataset**
   Contain following Attributes:
   - Disease
   - Symptoms 1 – 17

2) **Symptoms_Description**
   Contain following Attributes:
   - Disease
   - Description

3) **Symptoms_precautions**
   Contain following Attributes:
   - Disease
   - Precaution 1 – 4

4) **Symptoms-severity**
   Contain following Attributes:
   - Symptom
   - Weight

CHAPTER 1

INTRODUCTION

1.1. Problem Introduction

Medicine and healthcare are essential additives of the financial system and human existence. There is a remarkable quantity of alternate withinside the international we're residing in now and the arena that existed some weeks back. Everything has grown to become ugly and divergent. In this situation, in which the whole thing has grew to become digital, the docs and nurses are placing up most efforts to store humans' lives even though they need to chance their very own. There also are a few far off villages that lack clinical facilities. Virtual docs are board-licensed docs who select to exercise online thru video and call appointments, instead of in-individual appointments however this isn't viable withinside the case of an emergency. Machines are usually idea to be advanced to humans because, without human error, they can whole jobs greater quick and consistently. A disorder predictor may be known as a digital doctor, that may expect the disorder of any affected person with nonhuman error.

The number one aim became to increase several fashions to outline which one in every of them offers the maximum correct predictions. While the extent and complexity of system studying tasks vary, the primary shape stays the same. Several rule-primarily based totally strategies had been drawn from system studying to keep in mind the improvement and deployment of the predictive version. Several fashions had been initiated via way of means of the use of numerous systems studying (ML) algorithms that accrued uncooked statistics after which bifurcated it in line with gender, age group, and signs. The dataset became then processed in numerous ML fashions. According to ML fashions, the accuracy varied.

1.1.1. Motivation

With the help of the results from our dataset, it will help people and doctors to predict the disease one has in the most accurate manner by just seeing symptoms. It is a very important task in anyone's day-to-day life to be happy and healthy.

1.1.2. Project Objective

The support of an objective is to clean data and provide the maximum solutions of the queries generated from this dataset using the above Modules, to visualize them for better understanding and apply Machine learning Algorithms to the datasets.

1.1.3. Scope of the Project

The scope of the project is very vast, as it can be used in many fields of science & technology, mostly used by doctors to predict the disease of their patients, one can use it for its own purpose too, Helpful for our Indian Army too who live in different climate conditions.

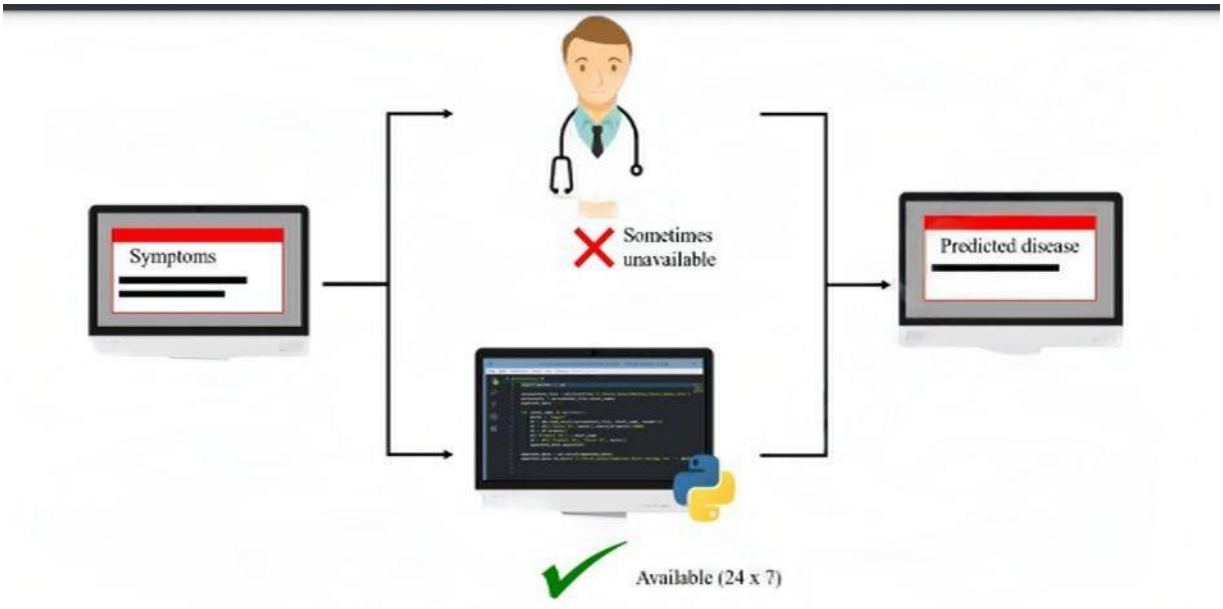


Fig. 1 Proposed system for disease prediction. The doctor may not be available always when needed. But, in the modern time scenario, according to necessity one can always use this prediction system anytime. The symptoms of an individual along with the age and gender can be given to the ML model to further process. After preliminary processing of the data, the ML model uses the current input, trains and tests the algorithm resulting in the predicted disease.

CHAPTER 2

LITERATURE SURVEY

Numerous research works have been carried out for the prediction of diseases based on the symptoms shown by an individual using machine learning algorithms.

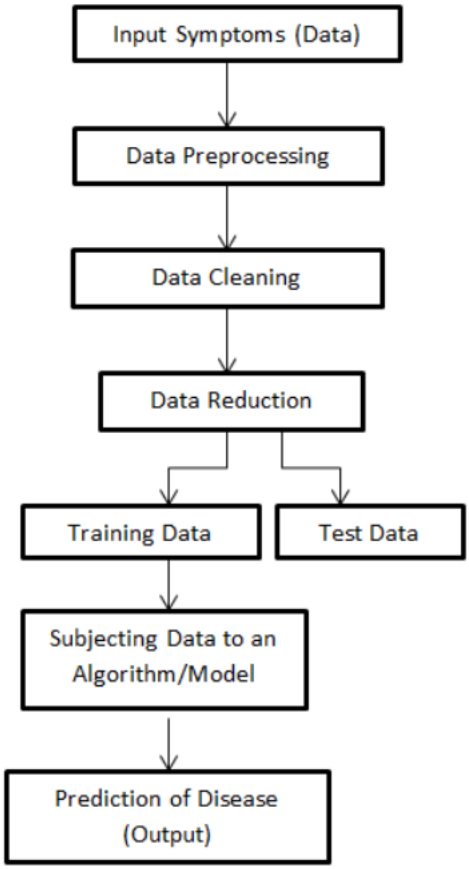
Monto et al. [3] designed a statistical model to predict whether a patient had influenza or not. They included 3744 unvaccinated adults and adolescent patients of influenza who had a fever and at least 2 other symptoms of influenza. Out of 3744, 2470 were confirmed to have influenza by the laboratory. Based on this data, their model gave an accuracy of 79 %. Sreevalli et al. [4] used the random forest machine-learning algorithm to predict the disease based on the symptoms. The system resulted in low time consumption and minimal cost for the prediction of diseases. The algorithm resulted in an accuracy of 84.2 %. Langbehn et al. created a number of tools. [5] to detect Alzheimer's disease. Data for 29 adults were used for the training purpose of the ML algorithm. They had developed classification models to detect reliable absolute changes in the scores with the help of SmoteBOOST and wRACOG algorithms.

Karayilan et al. [6] proposed a heart disease prediction system that uses the artificial neural network backpropagation algorithm. 13 clinical features were used as input for the neural network and then the neural network was trained with the backpropagation algorithm to predict the absence or presence of heart disease with an accuracy of 95 %. Various machine learning algorithms were streamlined for the effective prediction of chronic disease. The information gathered for the training was insufficient. To overcome this, a latent factor model was used. A new convolutional neural network-based multimodal disease risk prediction (CNN-MDRP) was structured. The algorithm reached an accuracy of around 94.8 %. Chae et al. [7] used 4 different deep learning models namely deep neural networks (DNN), long short term memory (LSTM), ordinary least squares (OLS), and an autoregressive integrated moving average (ARIMA) for monitoring 80 infectious diseases in 6 groups. Of all the models used, DNN and LSTM models had a better performance. The DNN model performed better in terms of average performance and the LSTM model gave close predictions when occurrences were large. Haq et al. [6] used a database that contained information about patients having any heart disease.

CHAPTER 3

METHODOLOGY

The dataset we looked at included 17 symptoms, which when combined or staged, result in 41 illnesses. In light of the 4920 documents from various patient samples, the goal is to develop a forecast algorithm that takes into account the side effects of various clients and predicts the illness to which the person would be exposed.



CHAPTER 4

IMPLEMENTATION

4.1. Software and Hardware Requirements

Python modules required like Numpy, Pandas, Matpltlib, sklearn, seaborn etc.

4.2. Basic Implementation

```
#import library
import pandas as pd
```

```
# Load Data
data = pd.read_csv('dataset.csv')
```

```
#The info() method prints information about the DataFrame. The information contains the number of columns, column labels, column
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4920 entries, 0 to 4919
Data columns (total 18 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Disease     4920 non-null   object
 1   Symptom_1   4920 non-null   object
 2   Symptom_2   4920 non-null   object
 3   Symptom_3   4920 non-null   object
 4   Symptom_4   4572 non-null   object
 5   Symptom_5   3714 non-null   object
 6   Symptom_6   2934 non-null   object
 7   Symptom_7   2268 non-null   object
 8   Symptom_8   1944 non-null   object
 9   Symptom_9   1692 non-null   object
 10  Symptom_10  1512 non-null   object
 11  Symptom_11  1194 non-null   object
 12  Symptom_12  744 non-null    object
 13  Symptom_13  504 non-null    object
 14  Symptom_14  306 non-null    object
 15  Symptom_15  240 non-null    object
 16  Symptom_16  192 non-null    object
 17  Symptom_17  72 non-null     object
dtypes: object(18)
memory usage: 692.0+ KB
```

```
#Using describe() function to get the basic idea of all the Int variable of data set.
data.describe()
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	S
count	4920	4920	4920	4920	4572	3714	2934	2268	1944	1692	
unique	41	34	48	54	50	38	32	26	21	22	
top	Dimorphic hemmorhoids(piles)	vomiting	vomiting	fatigue	high_fever	headache	nausea	abdominal_pain	abdominal_pain	yellowing_of_eyes	yellow
freq	120	822	870	726	378	348	390	264	276	228	

```
## Head shows first 10 rows
data.head(10)
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10	S
0	Fungal infection	itching	skin_rash	nodal_skin_eruptions	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	
1	Fungal infection	skin_rash	nodal_skin_eruptions	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	Fungal infection	itching	nodal_skin_eruptions	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	Fungal infection	itching	skin_rash	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	Fungal infection	itching	skin_rash	nodal_skin_eruptions	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
5	Fungal infection	skin_rash	nodal_skin_eruptions	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
6	Fungal infection	itching	nodal_skin_eruptions	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
7	Fungal infection	itching	skin_rash	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
8	Fungal infection	itching	skin_rash	nodal_skin_eruptions	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

```
# tail shows last 5 rows
data.tail()
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Sym
4915	(vertigo) Paroymsal Positional Vertigo	vomiting	headache	nausea	spinning_movements	loss_of_balance	unsteadiness	NaN	NaN	
4916	Acne	skin_rash	pus_filled_pimples	blackheads	scurring	NaN	NaN	NaN	NaN	
4917	Urinary tract infection	burning_micturition	bladder_discomfort	foul_smell_of urine	continuous_feel_of_urine	NaN	NaN	NaN	NaN	
4918	Psoriasis	skin_rash	joint_pain	skin_peeling	silver_like_dusting	small_dents_in_nails	inflammatory_nails	NaN	NaN	
4919	Impetigo	skin_rash	high_fever	blister	red_sore_around_nose	yellow_crust_ooze	NaN	NaN	NaN	

```
# columns gives column names of features
data.columns
```

```
Index(['Disease', 'Symptom_1', 'Symptom_2', 'Symptom_3', 'Symptom_4',
       'Symptom_5', 'Symptom_6', 'Symptom_7', 'Symptom_8', 'Symptom_9',
       'Symptom_10', 'Symptom_11', 'Symptom_12', 'Symptom_13', 'Symptom_14',
       'Symptom_15', 'Symptom_16', 'Symptom_17'],
      dtype='object')
```

```
# shape gives number of rows and columns in a tuple
data.shape
```

```
(4920, 18)
```

```
#The dtypes property is used to find the dtypes in the DataFrame. This returns a Series with the data type of each column.
data.dtypes
```

```
Disease       object
Symptom_1     object
Symptom_2     object
Symptom_3     object
Symptom_4     object
Symptom_5     object
Symptom_6     object
Symptom_7     object
Symptom_8     object
Symptom_9     object
Symptom_10    object
Symptom_11    object
Symptom_12    object
Symptom_13    object
Symptom_14    object
Symptom_15    object
Symptom_16    object
Symptom_17    object
dtype: object
```

4.3. Visualization

Whisker Plot or Box Plot

• A box plot, also known as a whisker plot, shows the minimum, first quartile, median, third quartile, and maximum values for a collection of data. In a box plot, we draw a box from the first to the third quartile.

• A vertical line runs through the box at the median. Each quartile's whiskers lead towards the minimum or maximum.

• A box plot is a visual depiction of numerical data groups depicted through their quartiles.

• There are five components to a box plot.

• Minimum

• First Quartile: 25% • Median (Second Quartile): 50%

• 75 percent in the third quartile

• Maximum

```
#import packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

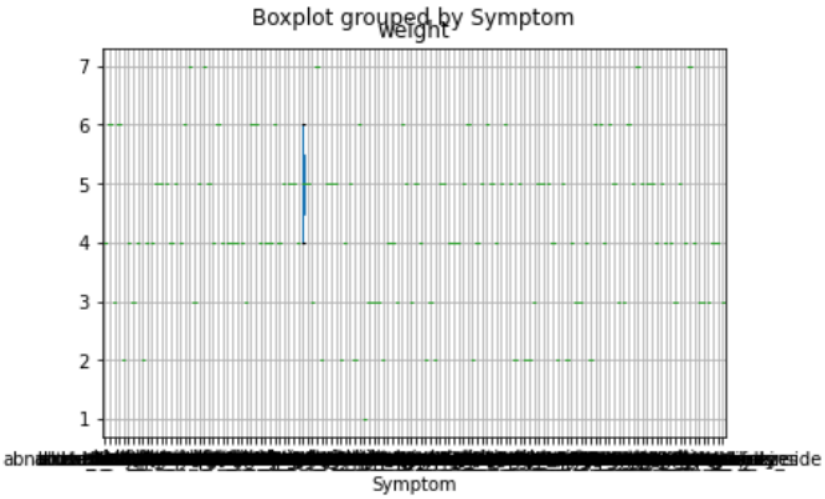
```
# Load Data
data = pd.read_csv('dataset.csv')
data.head(1)
```

```
d=pd.read_csv('Symptom-severity.csv')
d.head(1)
```

	Symptom	weight
0	itching	1

```
d.boxplot(column='weight',by='Symptom')
```

```
<AxesSubplot:title={'center':'weight'}, xlabel='Symptom'>
```



Scatter Plot

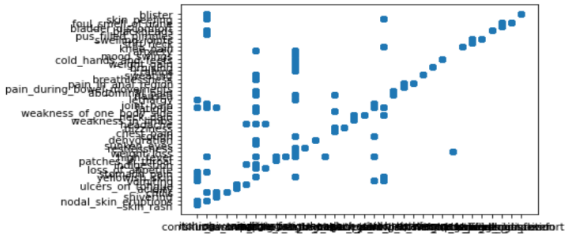
Some other commonly used plot kind is the simple scatter plot, a close cousin of the line plot. Instead of points being joined by line segments, where the points are represented individually with a dot, circle, or another shape

```
# Load Data
data = pd.read_csv('dataset.csv')
data.head(1)
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10	Symptor
0	Fungal infection	itching	skin_rash	nodal_skin_eruptions	dischromic _patches	NaN	NaN	NaN	NaN	NaN	NaN	

```
plt.scatter(data['Symptom_1'],data['Symptom_2'])
```

```
<matplotlib.collections.PathCollection at 0x288f1c2ef10>
```



Histogram

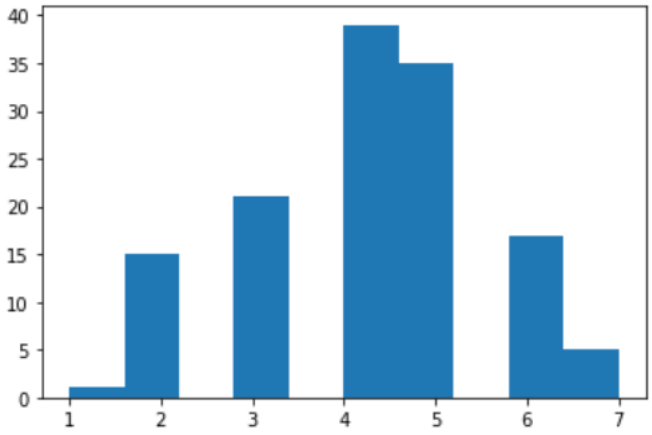
• A histogram shows the frequency on the vertical axis and the horizontal axis is another size

• usually it has containers, wherein every bin has a minimum and maximum fee. each bin additionally has a frequency between x and limitless

• Bin refers back to the variety of values that are divided into a chain of periods

```
plt.hist(d['weight'])
```

```
(array([ 1., 15.,  0., 21.,  0., 39., 35.,  0., 17.,  5.]),
 array([1. , 1.6, 2.2, 2.8, 3.4, 4. , 4.6, 5.2, 5.8, 6.4, 7. ]),
 <BarContainer object of 10 artists>)
```



4.4. SPLIT DATA FOR TEST AND TRAIN BY CLEANING

Keeping apart records into training and checking out units is a critical part of evaluating data mining fashions. generally, whilst you separate a data set into a training set and try outset, most of the records are used for schooling, and a smaller part of the data is used for checking out. by way of using comparable facts for schooling and testing, you could minimize the effects of statistics discrepancies and better apprehend the characteristics of the model. After a model has been processed via the use of the training set, we take a look at the version by making predictions towards the take a look at the set. due to the fact, that the information in the checking outset already contains

recognized values for the attribute that you need to predict, it is simple to decide whether or not the version's guesses are accurate.

Train Dataset: to fit the machine learning model.

Test Dataset: to evaluate the fit machine learning model.

```
: ## import data

df = pd.read_csv('dataset.csv')
df = shuffle(df, random_state = 42)
df.head()
```

:

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8
373	Acne	skin_rash	blackheads	scurring	NaN	NaN	NaN	NaN	NaN
4916	Acne	skin_rash	pus_filled_pimples	blackheads	scurring	NaN	NaN	NaN	NaN
1550	Hyperthyroidism	fatigue	mood_swings	weight_loss	restlessness	sweating	diarrhoea	fast_heart_rate	excessive_hunger
3081	AIDS	muscle_wasting	patches_in_throat	high_fever	extra_marital_contacts	NaN	NaN	NaN	NaN
3857	Chronic cholestasis	itching	vomiting	yellowish_skin	nausea	loss_of_appetite	abdominal_pain	yellowing_of_eyes	NaN

```
: ## charactieristics of data

df.describe()
```

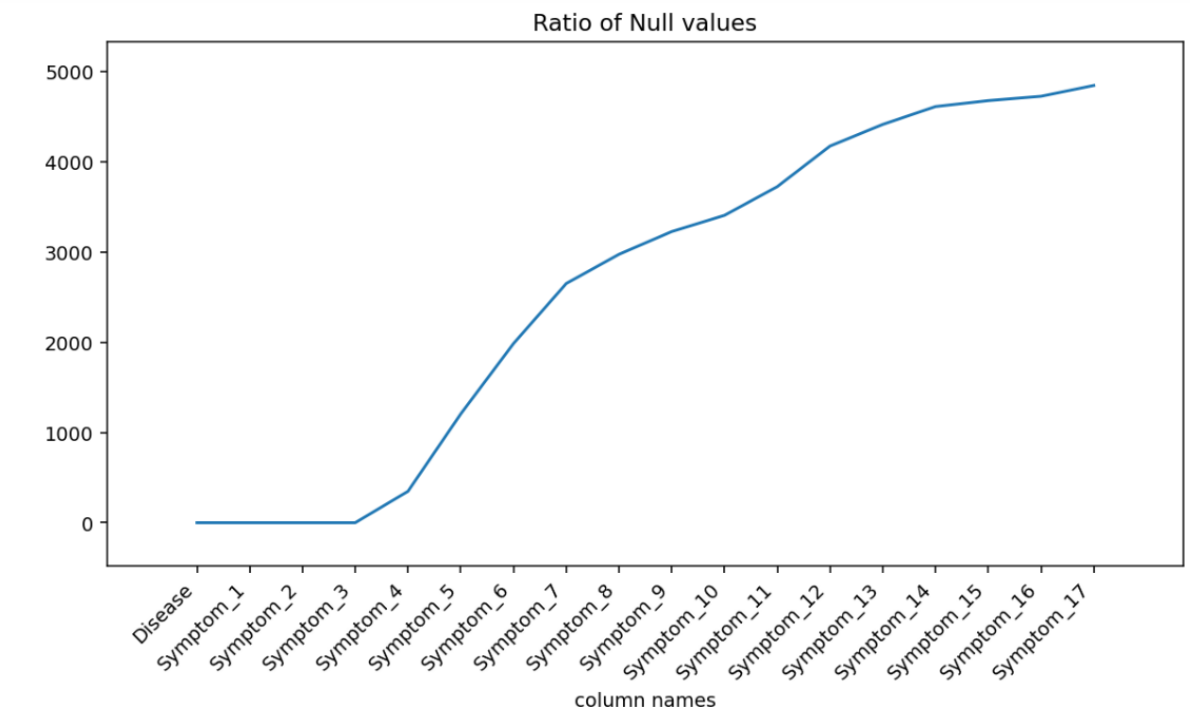
:

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10	S
count	4920	4920	4920	4920	4572	3714	2934	2268	1944	1692	1512	
unique	41	34	48	54	50	38	32	26	21	22	21	
top	Dimorphic hemmorhoids(piles)	vomiting	vomiting	fatigue	high fever	headache	nausea	abdominal pain	abdominal pain	yellowing of eyes	yellowing of eyes	
freq	120	822	870	726	378	348	390	264	276	228	198	

```
## check null values

null_checker = df.apply(lambda x: sum(x.isnull())).to_frame(name='count')
print(null_checker)
```

```
            count
Disease         0
Symptom_1       0
Symptom_2       0
Symptom_3       0
Symptom_4     348
Symptom_5    1206
Symptom_6    1986
Symptom_7    2652
Symptom_8    2976
Symptom_9    3228
Symptom_10   3408
Symptom_11   3726
Symptom_12   4176
Symptom_13   4416
Symptom_14   4614
Symptom_15   4680
Symptom_16   4728
Symptom_17   4848
```

```
## plot of null value

plt.figure(figsize=(10, 5), dpi=140)
plt.plot(null_checker.index, null_checker['count'])
plt.xticks(null_checker.index, null_checker.index, rotation = 45, horizontalalignment = 'right')
plt.title('Ratio of Null values')
plt.xlabel('column names')
plt.margins(0.1)
plt.show()
```




```
cols = df.columns

data = df[cols].values.flatten()

reshaped = pd.Series(data)
reshaped = reshaped.str.strip()
reshaped = reshaped.values.reshape(df.shape)

df = pd.DataFrame(reshaped, columns = df.columns)
df.head()
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10	Symptom_
0	Acne	skin rash	blackheads	scurring	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
1	Acne	skin rash	pus filled pimples	blackheads	scurring	NaN	NaN	NaN	NaN	NaN	NaN	N
2	Hyperthyroidism	fatigue	mood swings	weight loss	restlessness	sweating	diarrhoea	fast heart rate	excessive hunger	muscle weakness	irritability	abnorr menstruat
3	AIDS	muscle wasting	patches in throat	high fever	extra marital contacts	NaN	NaN	NaN	NaN	NaN	NaN	N
4	Chronic cholestasis	itching	vomiting	yellowish skin	nausea	loss of appetite	abdominal pain	yellowing of eyes	NaN	NaN	NaN	N

```
## lets fill nan values

df = df.fillna(0)
df.head()
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10	Symptom_
0	Acne	skin rash	blackheads	scurring	0	0	0	0	0	0	0	
1	Acne	skin rash	pus filled pimples	blackheads	scurring	0	0	0	0	0	0	
2	Hyperthyroidism	fatigue	mood swings	weight loss	restlessness	sweating	diarrhoea	fast heart rate	excessive hunger	muscle weakness	irritability	abnorr menstruat
3	AIDS	muscle wasting	patches in throat	high fever	extra marital contacts	0	0	0	0	0	0	
4	Chronic cholestasis	itching	vomiting	yellowish skin	nausea	loss of appetite	abdominal pain	yellowing of eyes	0	0	0	

```
## lets explore symptom severity

df_severity = pd.read_csv('Symptom-severity.csv')
df_severity['Symptom'] = df_severity['Symptom'].str.replace('_',' ')
df_severity.head(10)
```

	Symptom	weight
0	itching	1
1	skin rash	3
2	nodal skin eruptions	4
3	continuous sneezing	4
4	shivering	5
5	chills	3
6	joint pain	3
7	stomach pain	5
8	acidity	3
9	ulcers on tongue	4

```
## overall list
df_severity['Symptom'].unique()
```

```
array(['itching', 'skin rash', 'nodal skin eruptions',
       'continuous sneezing', 'shivering', 'chills', 'joint pain',
       'stomach pain', 'acidity', 'ulcers on tongue', 'muscle wasting',
       'vomiting', 'burning micturition', 'spotting urination', 'fatigue',
       'weight gain', 'anxiety', 'cold hands and feets', 'mood swings',
       'weight loss', 'restlessness', 'lethargy', 'patches in throat',
       'irregular sugar level', 'cough', 'high fever', 'sunken eyes',
       'breathlessness', 'sweating', 'dehydration', 'indigestion',
       'headache', 'yellowish skin', 'dark urine', 'nausea',
       'loss of appetite', 'pain behind the eyes', 'back pain',
       'constipation', 'abdominal pain', 'diarrhoea', 'mild fever',
       'yellow urine', 'yellowing of eyes', 'acute liver failure',
       'fluid overload', 'swelling of stomach', 'swelled lymph nodes',
       'malaise', 'blurred and distorted vision', 'phlegm',
       'throat irritation', 'redness of eyes', 'sinus pressure',
       'runny nose', 'congestion', 'chest pain', 'weakness in limbs',
       'fast heart rate', 'pain during bowel movements',
       'pain in anal region', 'bloody stool', 'irritation in anus',
       'neck pain', 'dizziness', 'cramps', 'bruising', 'obesity',
       'swollen legs', 'swollen blood vessels', 'puffy face and eyes',
       'enlarged thyroid', 'brittle nails', 'swollen extremeties',
       'excessive hunger', 'extra marital contacts',
       'drying and tingling lips', 'slurred speech', 'knee pain',
       'hip joint pain', 'muscle weakness', 'stiff neck',
       'swelling joints', 'movement stiffness', 'spinning movements',
       'loss of balance', 'unsteadiness', 'weakness of one body side',
       'loss of smell', 'bladder discomfort', 'foul smell ofurine',
       'continuous feel of urine', 'passage of gases', 'internal itching',
       'toxic look (typhos)', 'depression', 'irritability', 'muscle pain',
       'altered sensorium', 'red spots over body', 'belly pain',
       'abnormal menstruation', 'dischromic patches',
       'watering from eyes', 'increased appetite', 'polyuria',
       'family history', 'mucoid sputum', 'rusty sputum',
       'lack of concentration', 'visual disturbances',
```

```
'receiving blood transfusion', 'receiving unsterile injections',
'coma', 'stomach bleeding', 'distention of abdomen',
'history of alcohol consumption', 'blood in sputum',
'prominent veins on calf', 'palpitations', 'painful walking',
'pus filled pimples', 'blackheads', 'scurring', 'skin peeling',
'silver like dusting', 'small dents in nails',
'inflammatory nails', 'blister', 'red sore around nose',
'yellow crust ooze', 'prognosis'], dtype=object)
```

```
## lets encode sysptoms in the data

vals = df.values
symptoms = df_severity['Symptom'].unique()

for i in range(len(symptoms)):
    vals[vals == symptoms[i]] = df_severity[df_severity['Symptom'] == symptoms[i]]['weight'].values[0]
```

```
df_processed = pd.DataFrame(vals, columns=cols)
df_processed.head()
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10	Symptom_
0	Acne	3	2	2	0	0	0	0	0	0	0	
1	Acne	3	2	2	2	0	0	0	0	0	0	
2	Hyperthyroidism	4	3	3	5	3	6	5	4	2	2	
3	AIDS	3	6	7	5	0	0	0	0	0	0	
4	Chronic cholestasis	1	5	3	5	4	4	4	0	0	0	

```
## assign symptoms with no rank to zero

df_processed = df_processed.replace('dischromic  patches', 0)
df_processed = df_processed.replace('spotting  urination', 0)
df_processed = df_processed.replace('foul smell of urine', 0)
```

```
#Split Data
data = df_processed.iloc[:,1:].values
labels = df['Disease'].values
```

```
## split train and test data
# help(train_test_split)
X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, random_state=42)
```

```
print(X_train.shape, y_train.shape, X_test.shape, y_test.shape)
```

```
(3936, 17) (3936,) (984, 17) (984,)
```

```
## shape of train data

print(X_train[0])
print(X_train[1])
```

```
[6 4 0 6 0 0 0 0 0 0 0 0 0 0 0 0 0]
[3 5 3 5 4 4 3 2 3 0 0 0 0 0 0 0 0]
```

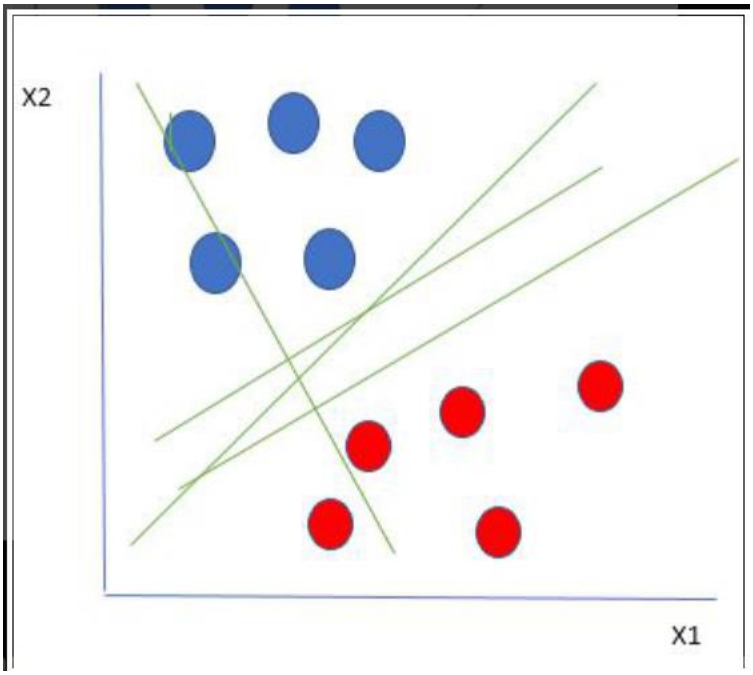
4.5 MACHINE LEARNING ALGORITMS

Compute the F1 score, also known as balanced F-score or F-measure.

The F1 score is interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score is F1 = 2 * (precision * recall) / (precision + recall)

SVM:

Support Vector Machine (SVM) is a supervised machine learning algorithm that is used for both classification and regression. Though we are saying regression issues as properly it's exceptionally desirable for classification. The goal of the SVM algorithm is to discover a hyperplane in an N-dimensional space that exceptionally classifies the information points.



```
x_train, x_test, y_train, y_test = train_test_split(data, labels, train_size = 0.8,random_state=42)
print(x_train.shape, x_test.shape, y_train.shape, y_test.shape)
```

```
(3936, 17) (984, 17) (3936,) (984,)
```

```
SVM_unhyperd= SVC()
SVM_unhyperd.fit(x_train, y_train)
```

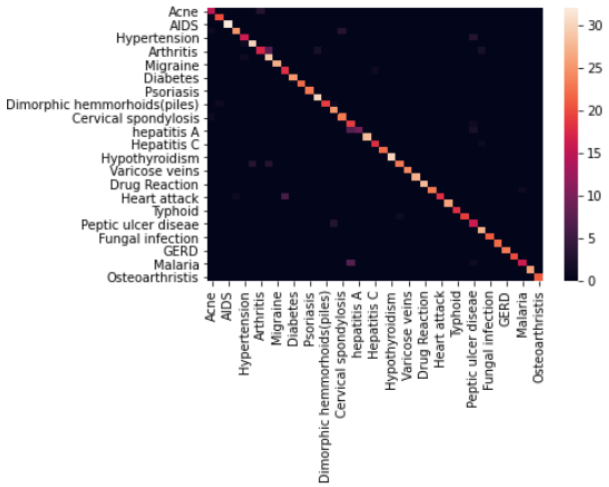
```
SVC()
```

```
preds = SVM_unhyperd.predict(x_test)
conf_mat = confusion_matrix(y_test, preds)
df_cm = pd.DataFrame(conf_mat, index=df['Disease'].unique(), columns=df['Disease'].unique())
print('F1-score% =', f1_score(y_test, preds, average='macro')*100, '|', 'Accuracy% =', accuracy_score(y_test, preds)*100,'|',
      'Precision% =', precision_score(y_test, preds,average='macro')*100)
```

```
F1-score% = 93.10485856410196 | Accuracy% = 93.4959349593496 | Precision% = 94.2446688327875
```

```
#Plot the confusion matrix for 25 diseases
sns.heatmap(df_cm)
```

```
<AxesSubplot:>
```




```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
SVM_unhyperd_train =cross_val_score(SVM_unhyperd, x_train, y_train, cv=kfold, scoring='accuracy')
pd.DataFrame(SVM_unhyperd_train,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (SVM_unhyperd_train.mean()*100.0, SVM_unhyperd_train.std()*100.0))
```

```
Mean Accuracy: 92.988%, Standard Deviation: (0.85%)
```

```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
SVM_unhyperd_test =cross_val_score(SVM_unhyperd, x_test, y_test, cv=kfold, scoring='accuracy')
pd.DataFrame(SVM_unhyperd_test,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (SVM_unhyperd_test.mean()*100.0, SVM_unhyperd_test.std()*100.0))
```

```
Mean Accuracy: 82.623%, Standard Deviation: (2.54%)
```

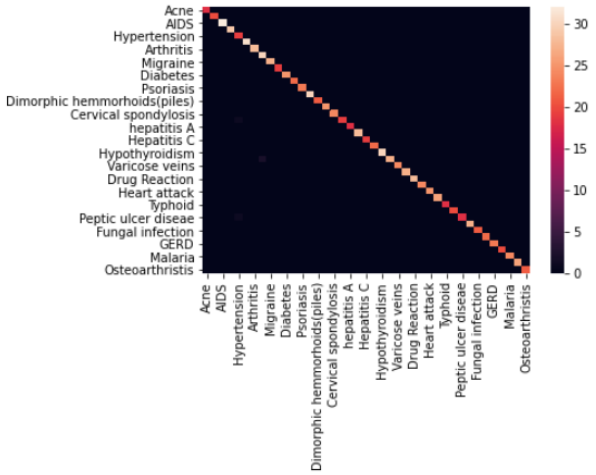
```
print(classification_report(y_test, preds))
```

```
                                   precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       0.88      0.83      0.86        18
                                AIDS       0.95      1.00      0.98        20
                                Acne       1.00      1.00      1.00        32
                 Alcoholic hepatitis       0.96      0.86      0.91        29
                             Allergy       0.89      0.84      0.86        19
                           Arthritis       0.91      0.97      0.94        31
                  Bronchial Asthma       0.85      0.61      0.71        28
                Cervical spondylosis       0.74      0.97      0.84        30
                         Chicken pox       1.00      1.00      1.00        27
                 Chronic cholestasis       0.75      0.95      0.84        19
                         Common Cold       1.00      1.00      1.00        25
                              Dengue       1.00      1.00      1.00        22
                            Diabetes       1.00      1.00      1.00        23
          Dimorphic hemmorhoids(piles)       0.94      1.00      0.97        30
                        Drug Reaction       1.00      0.90      0.95        21
                    Fungal infection       0.86      1.00      0.93        25
                                GERD       0.88      0.96      0.92        24
                     Gastroenteritis       0.58      0.95      0.72        20
                        Heart attack       1.00      0.50      0.67        18
                         Hepatitis B       1.00      1.00      1.00        28
                         Hepatitis C       0.95      0.95      0.95        19
                         Hepatitis D       1.00      1.00      1.00        22
                         Hepatitis E       1.00      1.00      1.00        30
                        Hypertension       0.96      0.79      0.87        29
                    Hyperthyroidism       1.00      1.00      1.00        24
                       Hypoglycemia       1.00      1.00      1.00        27
                     Hypothyroidism       1.00      1.00      1.00        28
                            Impetigo       1.00      0.96      0.98        24
                            Jaundice       1.00      0.72      0.84        25
                             Malaria       1.00      1.00      1.00        26
                            Migraine       1.00      1.00      1.00        18
                       Osteoarthristis       1.00      0.95      0.97        20
          Paralysis (brain hemorrhage)       0.70      0.84      0.76        19

        Peptic ulcer diseae       0.90      1.00      0.95        27
                  Pneumonia       1.00      1.00      1.00        21
                  Psoriasis       1.00      1.00      1.00        22
               Tuberculosis       1.00      1.00      1.00        23
                    Typhoid       1.00      1.00      1.00        20
    Urinary tract infection       0.94      0.67      0.78        24
             Varicose veins       1.00      1.00      1.00        26
                hepatitis A       1.00      1.00      1.00        21

                   accuracy                           0.93       984
                  macro avg       0.94      0.93      0.93       984
               weighted avg       0.94      0.93      0.93       984
```

```
#Hyperparameter tuning with GridSearchCV
SVM_hyperd = SVC(C=0.02, gamma=0.3, kernel='poly')
SVM_hyperd.fit(x_train, y_train)
preds = SVM_hyperd.predict(x_test)
conf_mat = confusion_matrix(y_test, preds)
df_cm = pd.DataFrame(conf_mat, index=df['Disease'].unique(), columns=df['Disease'].unique())
print('F1-score% =', f1_score(y_test, preds, average='macro')*100, '|', 'Accuracy% =', accuracy_score(y_test, preds)*100)
sns.heatmap(df_cm)
```

```
F1-score% = 99.58380389536958 | Accuracy% = 99.59349593495935
```

```
<AxesSubplot:>
```




```
#Using 10-Fold Cross Validation to estimate the performance of machine learning models
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
SVM_hyperd_train =cross_val_score(SVM_hyperd, x_train, y_train, cv=kfold, scoring='accuracy')
pd.DataFrame(SVM_hyperd_train,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (SVM_hyperd_train.mean()*100.0, SVM_hyperd_train.std()*100.0))
```

```
Mean Accuracy: 99.492%, Standard Deviation: (0.38%)
```

```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
SVM_hyperd_test =cross_val_score(SVM_hyperd, x_test, y_test, cv=kfold, scoring='accuracy')
pd.DataFrame(SVM_hyperd_test,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (SVM_hyperd_test.mean()*100.0, SVM_hyperd_test.std()*100.0))
```

```
Mean Accuracy: 96.639%, Standard Deviation: (1.94%)
```

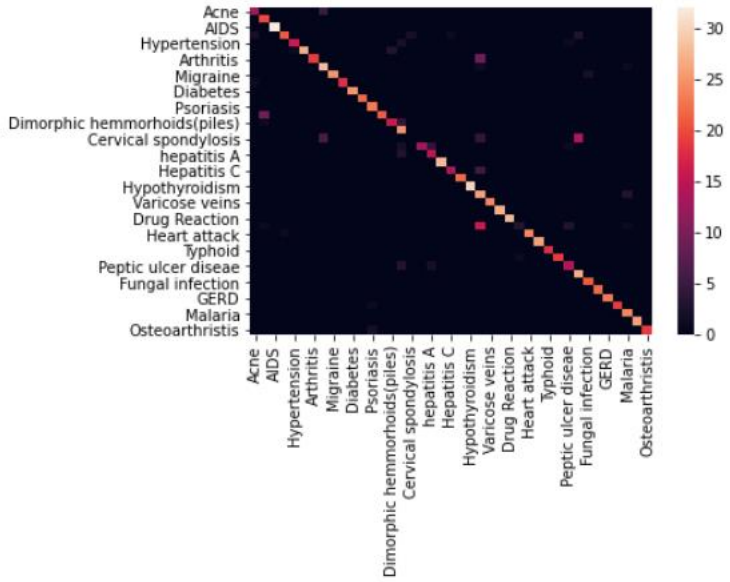
Naive Bayes Model:

Naive Bayes classifiers are a set of classification algorithms based on Bayes' Theorem. It isn't a single algorithm but a family of algorithms where they all share a common principle, i.e., each pair of capabilities being categorized is unbiased of each other.

```
#NAIVE BAYES
from sklearn.naive_bayes import GaussianNB
gaussian = GaussianNB()
gaussian.fit(x_train, y_train)
preds=gaussian.predict(x_test)
conf_mat = confusion_matrix(y_test, preds)
df_cm = pd.DataFrame(conf_mat, index=df['Disease'].unique(), columns=df['Disease'].unique())
print('F1-score% =', f1_score(y_test, preds, average='macro')*100, '|', 'Accuracy% =', accuracy_score(y_test, preds)*100)
sns.heatmap(df_cm)
```

```
F1-score% = 86.58357842139314 | Accuracy% = 87.70325203252033
```

```
<AxesSubplot:>
```




```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
gaussian_train =cross_val_score(gaussian, x_train, y_train, cv=kfold, scoring='accuracy')
pd.DataFrame(gaussian_train,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (gaussian_train.mean()*100.0, gaussian_train.std()*100.0))
```

```
Mean Accuracy: 86.840%, Standard Deviation: (1.65%)
```

```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
gaussian_test =cross_val_score(gaussian, x_test, y_test, cv=kfold, scoring='accuracy')
pd.DataFrame(gaussian_test,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (gaussian_test.mean()*100.0, gaussian_test.std()*100.0))
```

```
Mean Accuracy: 85.666%, Standard Deviation: (1.77%)
```

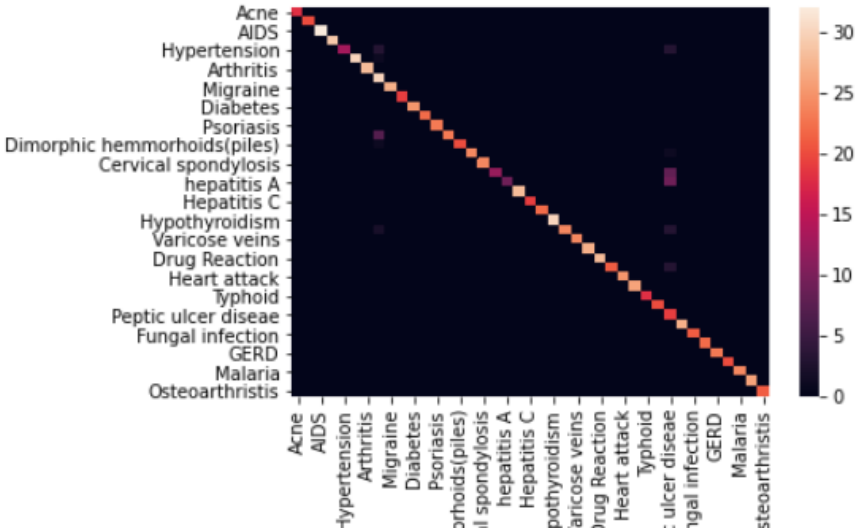
Decision Tree:

Decision tree is the most effective and famous tool for prediction and classification. A Decision tree is a flowchart like tree structure, where every internal node denotes a check on an attribute, every branch represents an outcome of the test, and every leaf node (terminal node) holds a class label.

```
#Decision Tree
tree =DecisionTreeClassifier(criterion='gini',random_state=42,max_depth=13)
tree.fit(x_train, y_train)
preds=tree.predict(x_test)
conf_mat = confusion_matrix(y_test, preds)
df_cm = pd.DataFrame(conf_mat, index=df['Disease'].unique(), columns=df['Disease'].unique())
print('F1-score% =', f1_score(y_test, preds, average='macro')*100, '|', 'Accuracy% =', accuracy_score(y_test, preds)*100)
sns.heatmap(df_cm)
```

```
F1-score% = 95.7812664665753 | Accuracy% = 95.83333333333334
```

```
<AxesSubplot:>
```




```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
DS_train =cross_val_score(tree, x_train, y_train, cv=kfold, scoring='accuracy')
pd.DataFrame(DS_train,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (DS_train.mean()*100.0, DS_train.std()*100.0))
```

```
Mean Accuracy: 95.706%, Standard Deviation: (1.99%)
```

```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
DS_test =cross_val_score(tree, x_test, y_test, cv=kfold, scoring='accuracy')
pd.DataFrame(DS_test,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (DS_test.mean()*100.0, DS_test.std()*100.0))
```

```
Mean Accuracy: 94.198%, Standard Deviation: (3.57%)
```

Random Forest:

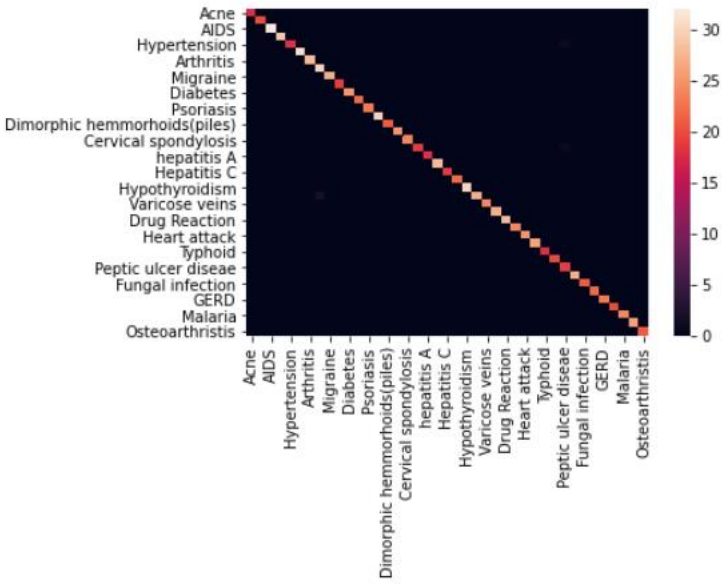
Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset

```
#Random Forest
rfc=RandomForestClassifier(random_state=42)
```

```
rnd_forest = RandomForestClassifier(random_state=42, max_features='sqrt', n_estimators= 500, max_depth=13)
rnd_forest.fit(x_train,y_train)
preds=rnd_forest.predict(x_test)
conf_mat = confusion_matrix(y_test, preds)
df_cm = pd.DataFrame(conf_mat, index=df['Disease'].unique(), columns=df['Disease'].unique())
print('F1-score% =', f1_score(y_test, preds, average='macro')*100, '|', 'Accuracy% =', accuracy_score(y_test, preds)*100)
sns.heatmap(df_cm)
```

```
F1-score% = 99.58380389536958 | Accuracy% = 99.59349593495935
```

```
<AxesSubplot:>
```




```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
rnd_forest_train =cross_val_score(rnd_forest, x_train, y_train, cv=kfold, scoring='accuracy')
pd.DataFrame(rnd_forest_train,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (rnd_forest_train.mean()*100.0, rnd_forest_train.std()*100.0))
```

```
Mean Accuracy: 99.187%, Standard Deviation: (0.44%)
```

```
kfold = KFold(n_splits=10,shuffle=True,random_state=42)
rnd_forest_test =cross_val_score(rnd_forest, x_test, y_test, cv=kfold, scoring='accuracy')
pd.DataFrame(rnd_forest_test,columns=['Scores'])
print("Mean Accuracy: %.3f%%, Standard Deviation: (%.2f%%)" % (rnd_forest_test.mean()*100.0, rnd_forest_test.std()*100.0))
```

```
Mean Accuracy: 98.575%, Standard Deviation: (1.31%)
```

Function to manually test the models:

```
#Function to manually test the models
def predd(S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,S15,S16,S17,x):
    psymptoms = [S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,S15,S16,S17]
    print(psymptoms)
    a = np.array(df1["Symptom"])
    b = np.array(df1["weight"])
    for j in range(len(psymptoms)):
        for k in range(len(a)):
            if psymptoms[j]==a[k]:
                psymptoms[j]=b[k]

    psy = [psymptoms]

    pred2 = x.predict(psy)
    print("The prediction is",pred2[0])
```

```
sympList=df1["Symptom"].to_list()
predd(sympList[7],sympList[5],sympList[2],sympList[80],0,0,0,0,0,0,0,0,0,0,0,0,rnd_forest)
```

```
['stomach pain', 'chills', 'nodal skin eruptions', 'muscle weakness', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
The prediction is Paralysis (brain hemorrhage)
```

```
sympList=df1["Symptom"].to_list()
predd(sympList[8],sympList[1],sympList[2],sympList[80],0,0,0,0,0,0,0,0,0,0,0,0,SVM_hyperd)
```

```
['acidity', 'skin rash', 'nodal skin eruptions', 'muscle weakness', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
The prediction is Paralysis (brain hemorrhage)
```

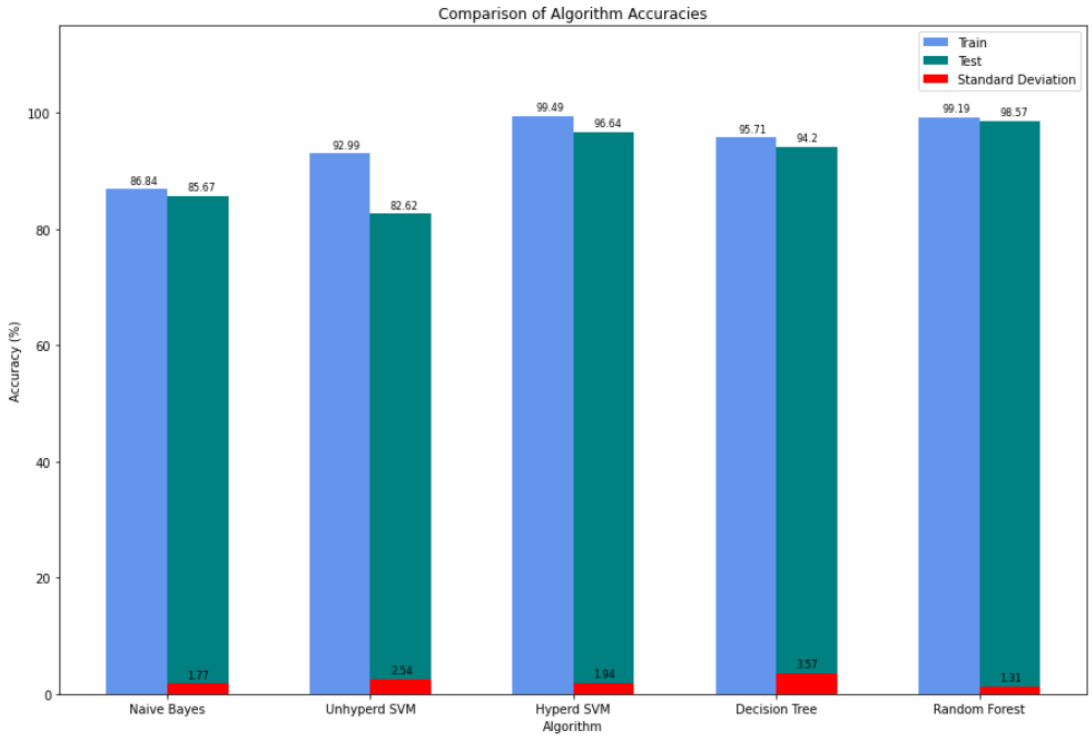
```
sympList=df1["Symptom"].to_list()
predd(sympList[8],sympList[5],sympList[2],sympList[80],0,0,0,0,0,0,0,0,0,0,0,0,SVM_unhyperd)
```

```
['acidity', 'chills', 'nodal skin eruptions', 'muscle weakness', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
The prediction is Allergy
```

COMPERISON BETWEEN ALGORITHMS TESTING AND TRAINING

```
n_groups = 5
algorithms = ('Naive Bayes','Unhyperd SVM', 'Hyperd SVM','Decision Tree', 'Random Forest')
train_accuracy = (gaussian_train.mean()*100.0,
                  SVM_unhyperd_train.mean()*100.0,
                  SVM_hyperd_train.mean()*100.0,
                  DS_train.mean()*100.0,
                  rnd_forest_train.mean()*100.0,
                  )


test_accuracy = (gaussian_test.mean()*100.0,
                  SVM_unhyperd_test.mean()*100.0,
                  SVM_hyperd_test.mean()*100.0,
                  DS_test.mean()*100.0,
                  rnd_forest_test.mean()*100.0
                 )

Standard_Deviation=(gaussian_test.std()*100.0,
                  SVM_unhyperd_test.std()*100.0,
                  SVM_hyperd_test.std()*100.0,
                  DS_test.std()*100.0,
                  rnd_forest_test.std()*100.0

                    )
```

```
# create plot
fig, ax = plt.subplots(figsize=(15, 10))
index = np.arange(n_groups)
bar_width = 0.3
opacity = 1
rects1 = plt.bar(index, train_accuracy, bar_width, alpha = opacity, color='Cornflowerblue', label='Train')
rects2 = plt.bar(index + bar_width, test_accuracy, bar_width, alpha = opacity, color='Teal', label='Test')
rects3 = plt.bar(index + bar_width, Standard_Deviation, bar_width, alpha = opacity, color='red', label='Standard Deviation')
plt.xlabel('Algorithm') # x axis label
plt.ylabel('Accuracy (%)') # y axis label
plt.ylim(0, 115)
plt.title('Comparison of Algorithm Accuracies') # plot title
plt.xticks(index + bar_width * 0.5, algorithms) # x axis data labels
plt.legend(loc = 'upper right') # show legend
for index, data in enumerate(train_accuracy):
    plt.text(x = index - 0.035, y = data + 1, s = round(data, 2), fontdict = dict(fontsize = 8))
for index, data in enumerate(test_accuracy):
    plt.text(x = index + 0.25, y = data + 1, s = round(data, 2), fontdict = dict(fontsize = 8))
for index, data in enumerate(Standard_Deviation):
    plt.text(x = index + 0.25, y = data + 1, s = round(data, 2), fontdict = dict(fontsize = 8))
```



4.6 EXPLORATORY DATA ANALYSIS

```
import numpy as np
import pandas as pd
import math
from math import factorial
import torch
import torch.nn as nn
import torch.nn.functional as F
import transformers

import matplotlib.pyplot as plt
```

```
sym_severity = pd.read_csv('Symptom-severity.csv')
di_sym = pd.read_csv('dataset.csv')
di_desc = pd.read_csv('symptom_Description.csv')
di_prec = pd.read_csv('symptom_precaution.csv')
```

```
symptoms = list(sym_severity['Symptom'])
diseases = list(di_sym['Disease'].unique())
```

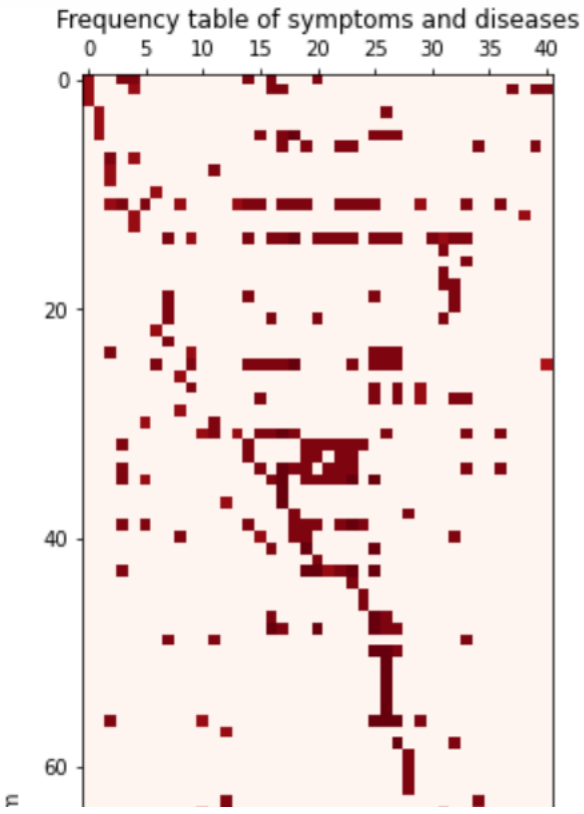
```
print('there are {} symptoms described in the dataset'.format(len(symptoms)))
print('there are {} diseases described in the dataset'.format(len(diseases)))
```

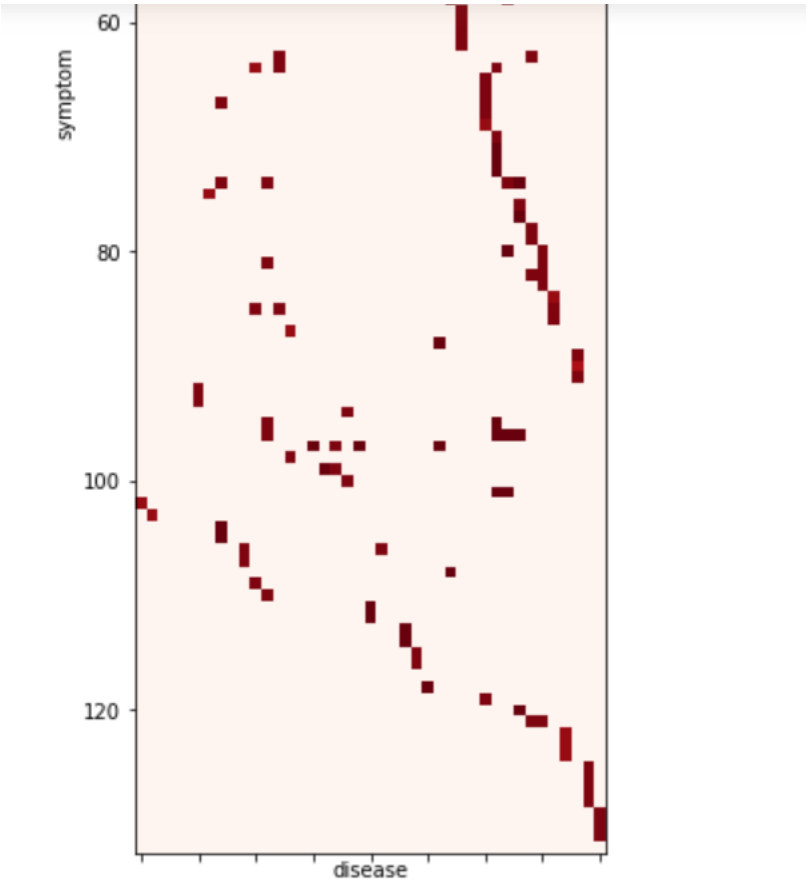
```
there are 133 symptoms described in the dataset
there are 41 diseases described in the dataset
```

```
# construct adjaciency matrix
adj_mat = np.zeros((133,41))
for i in range(len(di_sym)):
    for j in range(1, 18):
        disease = di_sym.iloc[i,0]
        symptom = di_sym.iloc[i,j]
        if pd.notnull(symptom):
            symptom = symptom.replace(' ','')
            dis_index = diseases.index(disease)
            sym_index = symptoms.index(symptom)
            adj_mat[sym_index, dis_index] += 1
        else:
            pass
```

```
fig = plt.figure(figsize=(7,14))
ax = fig.add_subplot()
ax.matshow(adj_mat, cmap='Reds')
ax.set_title('Frequency table of symptoms and diseases')
ax.set_xlabel('disease')
ax.set_ylabel('symptom')
```

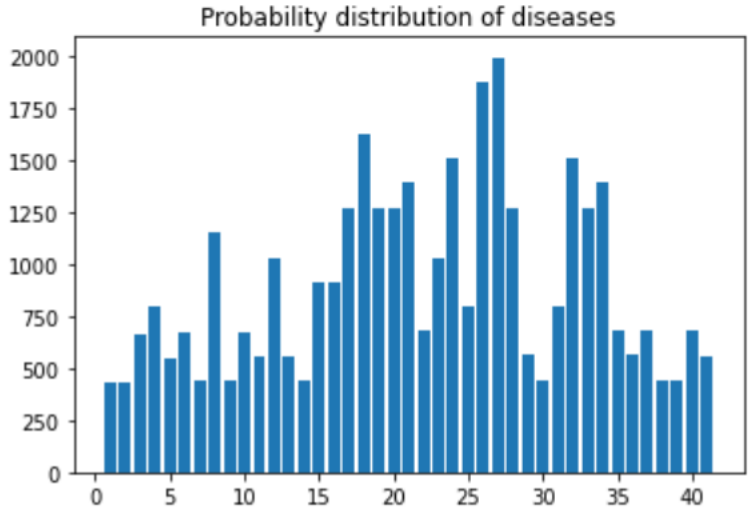
```
Text(0, 0.5, 'symptom')
```





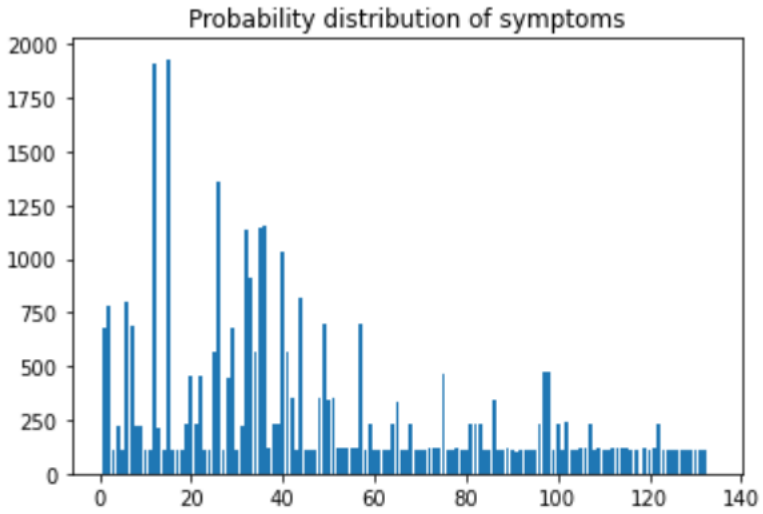

```
plt.bar(np.arange(1,42),adj_mat.sum(axis=0))
plt.title('Probability distribution of diseases')
```

```
Text(0.5, 1.0, 'Probability distribution of diseases')
```



```
plt.bar(np.arange(1,134),adj_mat.sum(axis=1))
plt.title('Probability distribution of symptoms')
```

```
Text(0.5, 1.0, 'Probability distribution of symptoms')
```



Predict disease based on Symptom Label

```
def doctor_bayes(adj_mat, symptom_list, symptoms, diseases):
    sym = [symptoms.index(s) for s in symptom_list]
    p_dis = adj_mat.sum(axis=0) / adj_mat.sum()
    p_sym = adj_mat.sum(axis=1) / adj_mat.sum()
    dist = []
    for i in range(41):
        # compute bayes probability
        prob = np.prod((adj_mat[:,i]/ adj_mat[:,i].sum())[sym])*p_dis[i] / np.prod(p_sym[sym])
        dist.append(prob)
    if sum(dist) == 0:
        return 'I have no Idea'
    else:
        idx = dist.index(max(dist))
        return diseases[idx]
```

```
doctor_bayes(adj_mat, ['fatigue', 'mood_swings'], symptoms, diseases)
```

```
'Hyperthyroidism'
```

```
doctor_bayes(adj_mat, ['itching', 'skin_rash'], symptoms, diseases)
```

```
'Fungal infection'
```

```
doctor_bayes(adj_mat, ['fatigue', 'high_fever'], symptoms, diseases)
```

```
'Bronchial Asthma'
```

```
doctor_bayes(adj_mat, ['fatigue', 'high_fever', 'itching', 'coma'], symptoms, diseases)
```

```
'I have no Idea'
```

CHAPTER 5

CONCLUSION

The manuscript presented the technique of predicting the disease based on the symptoms, almost all the ML models gave good accuracy values. As some models were dependent on the parameters, they couldn’t predict the disease and the accuracy percentage was quite low. Once the disease is predicted, we could easily manage the medical resources required for the treatment. This model would help in lowering the cost required in dealing with the disease and would also improve the recovery process.

As a future enhancement, we also look forward to executing multilingual summarization and multi-document summarization. The files which we give as input may also contain native languages, hence health records can be collected from various parts of the world and can be easily summarized using multilingual summarization. As of now, the paper proceeds with the global language (English). This paper clearly defines disease prediction using highly personalized training data sets and also some of the related tasks like fixing appointments and tracing the nearest health center.

References

1. Keniya, Rinkal and Khakharia, Aman and Shah, Vruddhi and Gada, Vrushabh and Manjalkar, Ruchi and Thaker, Tirth and Warang, Mahesh and Mehendale, Ninad and Mehendale, Ninad, Disease Prediction From Various Symptoms Using Machine Learning (July 27, 2020).

2. Talasila, Bhanuteja & Kolli, Saipoornachand & Kumar, Kilaru & Anudeep, Poonati & Ashish, Chennupati. (2021). Symptoms Based Multiple Disease Prediction Model using Machine Learning Approach. International Journal of Innovative Technology and Exploring Engineering. 10. 67-72. 10.35940/ijitee.I9364.0710921.

3. A.S. Monto, S. Gravenstein, M. Elliott, M. Colopy, J. Schweinle, Clinical signs and symptoms predicting influenza infection, Archives of internal medicine 160(21), 3243 (2000)

4. R.D.H.D.P. Sreevalli, K.P.M. Asia, Prediction of diseases using random forest classification algorithm

5. D.R. Langbehn, R.R. Brinkman, D. Falush, J.S. Paulsen, M. Hayden, an International Huntington's Disease Collaborative Group, A new model for prediction of the age of onset and penetrance for huntington's disease based on cag length, Clinical genetics 65(4), 267 (2004)

6. T. Karayılan, O. Kılı¸c, in ¨ 2017 International Conference on Computer Science and Engineering (UBMK) (IEEE, 2017), pp. 719–723

7. S. Chae, S. Kwon, D. Lee, Predicting infectious disease using deep learning and big data, International journal of environmental research and public health 15(8), 1596 (2018)