



By

Mustafa Hafeez

BCS193026

Muhammad Ahmed

BCS193054

Wasay Noor

BCS181047

A Project Report submitted to the
DEPARTMENT OF COMPUTER SCIENCE
in partial fulfillment of the requirements for the semester of
BACHELORS OF COMPUTER SCIENCE
Faculty of Computing
Capital University of Science & Technology,

Islamabad January, 2023

Copyright © 2022 by CUST Students All rights reserved. Reproduction in whole or in part in any form requires the prior written permission.

Table of Contents

Chapter 1	2
Introduction	2
Symptoms based Disease prediction	2
What is Decision Tree?	2
How Decision tree works	3
Why we use decision Tree	3
Chapter 2	4
Implementation	4
Importing Libraries	4
Data import (Preprocessing)	5
Remove ('_') underscore in the text (Preprocessing)	5
Characteristics of data (Preprocessing)	6
Check null values (Preprocessing)	6
Ratio of Null Values (Preprocessing)	7
Fill nan values with 0 (Preprocessing)	8
let's explore symptom severity (Preprocessing)	8
Overall list (Preprocessing)	9
let's encode symptoms in the data (Preprocessing)	10
Assign symptoms with no rank to zero (Preprocessing)	10
Split data into test and train	11
Implementation of decision tree and Predicted Output	12
Plotting of decision Tree	13
References	14

Chapter 1

Introduction

Symptoms based Disease prediction

This project is about disease prediction based on the symptoms given by user as an input. We are using Supervised Learning in which multiple algorithms can be used but we applied Decision Tree technique. Decision Tree classifies and predict the disease in the data set of Symptoms based Disease Symptoms and Diseases are given in our case. We have 41 unique diseases which is our target variable and 17 columns of symptoms that are independent features. Our dataset will look like the given figure below. It is also available on Kaggle.

<https://www.kaggle.com/code/chandrug/symptoms-based-disease-prediction-accuracy-99/data?select=dataset.csv>

Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6	Symptom_7	Symptom_8	Symptom_9	Symptom_10
Acne	skin_rash	blackheads	scurrying	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Acne	skin_rash	pus_filled_pimples	blackheads	scurrying	NaN	NaN	NaN	NaN	NaN	NaN
hyperthyroidism	fatigue	mood_swings	weight_loss	restlessness	sweating	diarrhoea	fast_heart_rate	excessive_hunger	muscle_weakness	irritability
AIDS	muscle_wasting	patches_in_throat	high_fever	extra_marital_contacts	NaN	NaN	NaN	NaN	NaN	NaN
Chronic_cholestasis	itching	vomiting	yellowish_skin	nausea	loss_of_appetite	abdominal_pain	yellowing_of_eyes	NaN	NaN	NaN

What is Decision Tree?

A decision tree is a type of supervised machine learning used to categorize or make predictions based on how a previous set of questions were answered. The model is a form of supervised learning, meaning that the model is trained and tested on a set of data that contains the desired categorization.

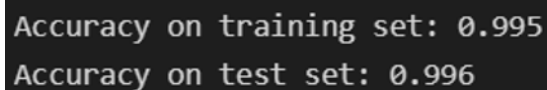
Decision Tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. Decision trees classify the examples by sorting them down the tree from the root to some leaf node, with the leaf node providing the classification to the example. Each node in the tree acts as a test case for some attribute, and each edge descending from that node corresponds to one of the possible answers to the test case. This process is recursive in nature and is repeated for every sub tree rooted at the new nodes

How Decision tree works

A tree can be “*learned*” by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called *recursive partitioning*. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions. The construction of a decision tree classifier does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery.

Why we use decision Tree

Because the main things is this that our problem is about classification and Decision Tree has ability to solve this problem. Decision trees apply a top-down approach to the dataset that is fed during training. If we talk generally decision tree classifier has good accuracy. Decision tree induction is a typical inductive approach to learn knowledge on classification. And also, in our model the accuracy by applying **Decision Tree is 0.995 on train data set and 0.996 on test dataset** as shown in given figure below.



```
Accuracy on training set: 0.995
Accuracy on test set: 0.996
```

Chapter 2

Implementation

Importing Libraries

Importing some basic libraries

```
### import packages

## lets basic import packages

import numpy as np #libraries used for arrays
import pandas as pd # mostly for data reading from csv files
import matplotlib.pyplot as plt # used for Data visualization like plotting graphs etc
import seaborn as sns #for making statistical graphics and easy to integrate with panda
from sklearn.model_selection import train_test_split #split data into train and test
from sklearn.utils import shuffle #used for shuffling attributes
from sklearn import metrics
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
```

Data import (Preprocessing)

The datafile is imported by the help of `pd.read_csv` command after importing the file. With `random_state=42`, we get the same train and test sets across different executions and `shuffle` command is used for their shuffling.

```
## 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	Symptom_9	Symptom_10
373	Acne	skin_rash	blackheads	scurrying	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4916	Acne	skin_rash	pus_filled_pimples	blackheads	scurrying	NaN	NaN	NaN	NaN	NaN	NaN
1550	Hyperthyroidism	fatigue	mood_swings	weight_loss	restlessness	sweating	diarrhoea	fast_heart_rate	excessive_hunger	muscle_weakness	irritability
3081	AIDS	muscle_wasting	patches_in_throat	high_fever	extra_marital_contacts	NaN	NaN	NaN	NaN	NaN	NaN
3857	Chronic cholestasis	itching	vomiting	yellowish_skin	nausea	loss_of_appetite	abdominal_pain	yellowing_of_eyes	NaN	NaN	NaN

Remove ('_') underscore in the text (Preprocessing)

We replace the ('_') with the empty spaces (' ').

```
## remove ('_') underscore in the text
```

```
for col in df.columns:
```

```
    df[col] = df[col].str.replace('_', ' ')
```

```
df.head()
```

	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	!
373	Acne	skin rash	blackheads	scurrying	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4916	Acne	skin rash	pus filled pimples	blackheads	scurrying	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1550	Hyperthyroidism	fatigue	mood swings	weight loss	restlessness	sweating	diarrhoea	fast heart rate	excessive hunger	muscle weakness	irritability	abnormal menstruation	NaN	NaN	NaN
3081	AIDS	muscle wasting	patches in throat	high fever	extra marital contacts	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3857	Chronic cholestasis	itching	vomiting	yellowish skin	nausea	loss of appetite	abdominal pain	yellowing of eyes	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Characteristics of data (Preprocessing)

```
## charactieristics of data
```

```
df.describe()
```

	Disease	Symptom_1	Symptom_2	Symptom_3	Symptom_4	Symptom_5	Symptom_6
count	4920	4920	4920	4920	4572	3714	2934
unique	41	34	48	54	50	38	32
top	Acne	vomiting	vomiting	fatigue	high fever	headache	nausea
freq	120	822	870	726	378	348	390

Check null values (Preprocessing)

For checking the null values, we count them and then replace it with 0.

```
## check null values
```

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

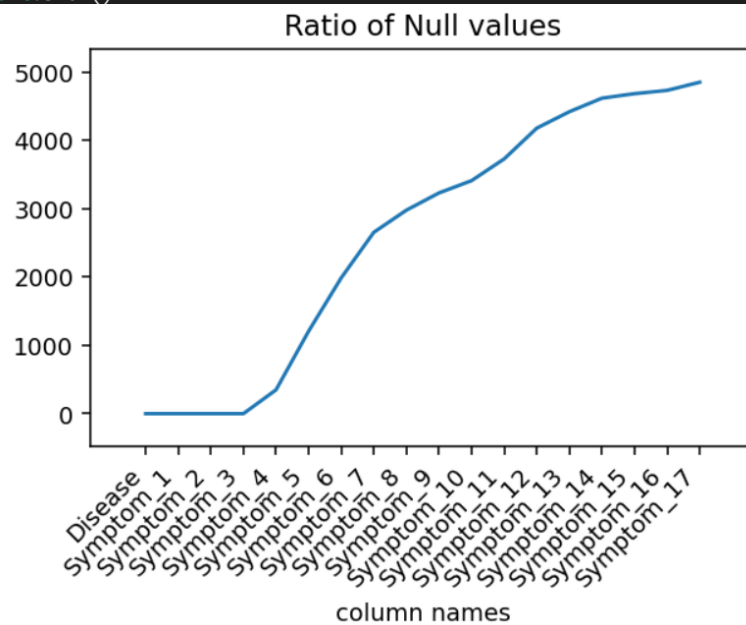
```
count  
Disease      0  
Symptom_1    0  
Symptom_2    0  
Symptom_3    0  
Symptom_4   348  
Symptom_5  1206  
Symptom_6  1986  
Symptom_7  2652  
Symptom_8  2976  
Symptom_9  3228  
Symptom_10 3408  
Symptom_11 3726  
Symptom_12 4176  
Symptom_13 4416  
Symptom_14 4614  
Symptom_15 4680  
Symptom_16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

Ratio of Null Values (Preprocessing)

Visualize null values in Graph and we find the ratio of null values.

```
## 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()
```



Fill nan values with 0 (Preprocessing)

```
## 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
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
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

let's explore symptom severity (Preprocessing)

Symptom severity is a csv file in which the weight of symptom are given.

```
## 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 (Preprocessing)

We place all the unique symptom in the list. There are 41 unique symptoms.

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

Output exceeds the [size limit](#). Open the full output data [in a text editor](#)

```
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',
```

let's encode symptoms in the data (Preprocessing)

Assigning of weight to the symptom because machine learning cannot understand string values that's why we assigned weight according to the symptom severity

```
## lets encode symptoms 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)
```

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

```
df_processed.head()
```

Assign symptoms with no rank to zero (Preprocessing)

Symptoms which has no weight we have assign them with zero. These symptoms are missed that's why we assign them with 0 separately.

```
## 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 into test and train

Split the data in to test and train. We divided it into 80 percent train and 20 percent test data

```
## 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)

print(X_train[0])
print(X_test[0])
print(y_train[0])
print (y_test[0])
```

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

```
[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]
Urinary tract infection
Migraine
```

Implementation of decision tree and Predicted Output

By using tree1.fit we have train the model and by using tree1.score we execute accuracy and by using tree1.predict we get the predicted output, which is migraine.

```
#finding Accuracy
from sklearn.tree import DecisionTreeClassifier
tree1 = DecisionTreeClassifier(criterion="entropy")

#training model
tree1.fit(X_train,y_train)

print("Accuracy on training set: {:.3f}".format(tree1.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(tree1.score(X_test, y_test)))

#prediction on model
tree1.predict([[3,5,3,5,4,4,3,2,3,0,0,0,0,0,0,0]]) #input symptoms

#tree1.predect ([[X_test[12]]])
```

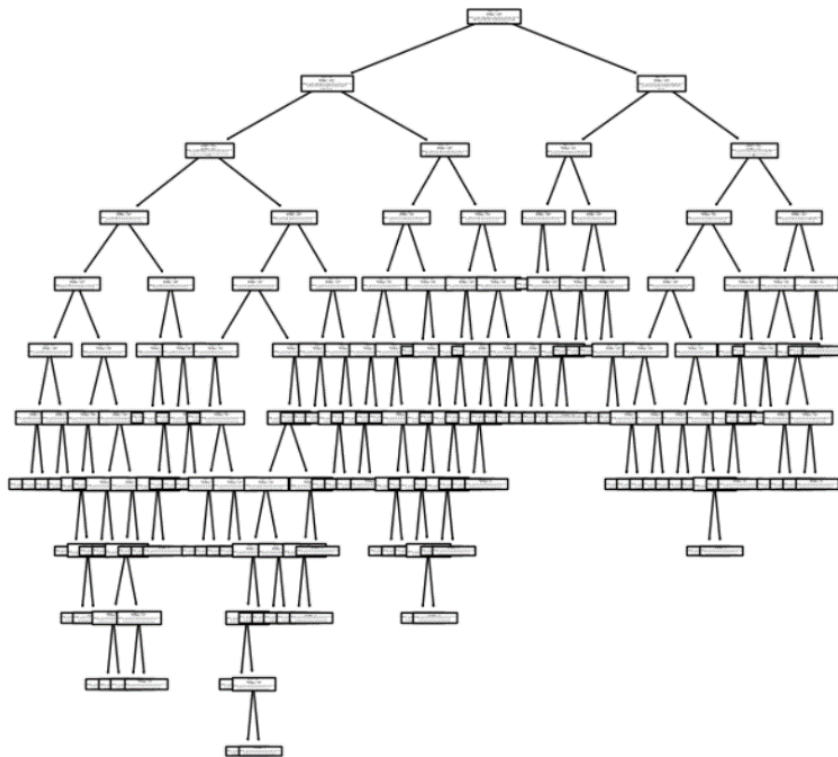
```
Accuracy on training set: 0.995
Accuracy on test set: 0.996

array(['Migraine'], dtype=object)
```

Plotting of decision Tree

Just for visualization we plot decision tree

```
from sklearn.tree import plot_tree
plt.figure(figsize=(8,8))
plot_tree(tree1)
```



References

<https://www.kaggle.com/code/chandrug/symptoms-based-disease-prediction-accuracy-99/notebook>

<https://www.kaggle.com/code/saurabhdeshtane/disease-prediction-ml-model/notebook>