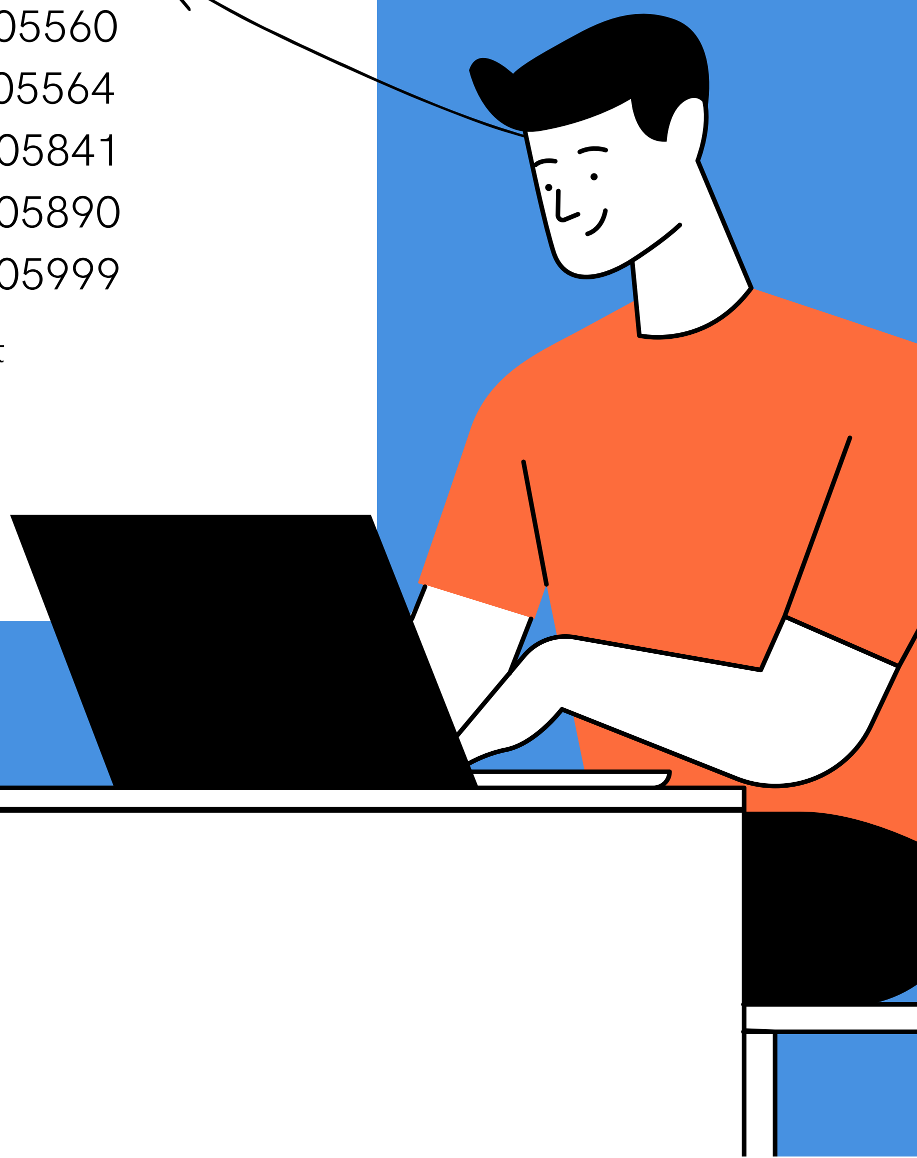
جامعة جدة
University of Jeddah

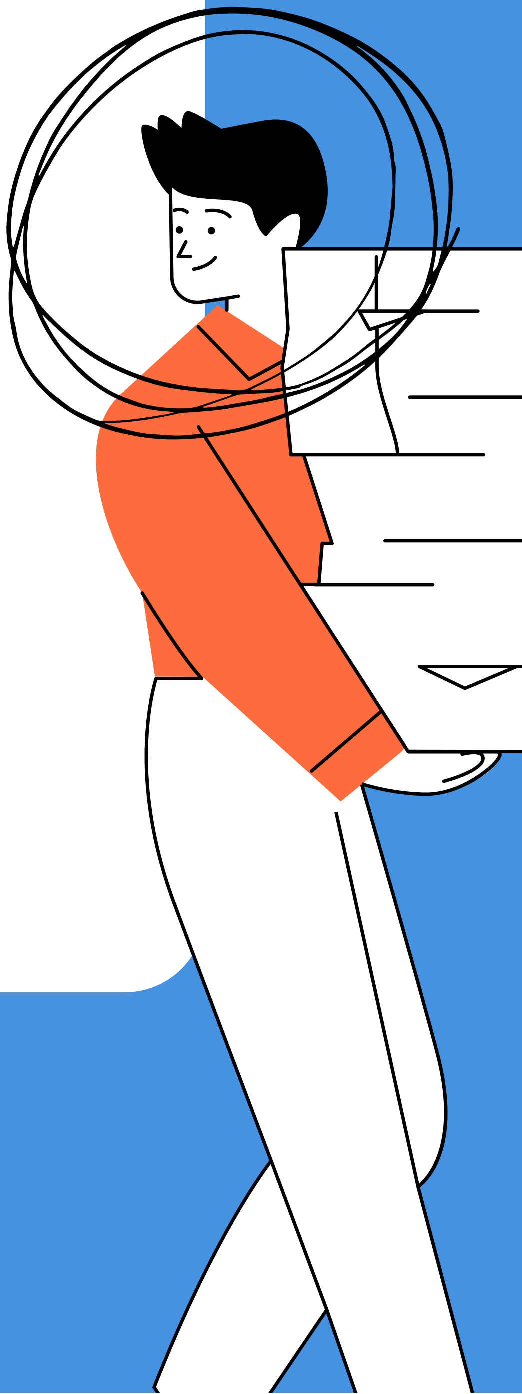
Disease Prediction Using Machine Learning

Raneen Alshehri 2005560
Ghaid Althobaity 2005564
Noor O. Alamoudi 2005841
Taif A. Basheikh 2005890
Raghad Alqithmi 2005999

Applied machine learning project
Dr. Enas Abo Fateh
2023



CONTENT



Introduction

People face various diseases due to the environmental condition and their living habits. So the prediction of disease at earlier stage becomes important task. But the accurate prediction on the basis of symptoms becomes too difficult for doctor. The correct prediction of disease is the most challenging task, and medical science has large amount of data growth per year. Due to increase amount of data growth in medical and healthcare field the accurate analysis on medical data has been benefits from early patient care. So, we proposed Disease Prediction Using Machine Learning

PROPLEM DEFINITION & OBJECTIVES

The purpose of constructing this project called "Disease Prediction Using Machine Learning" is to predict the accurate disease of the patient using all their general information's and also the symptoms. If this Prediction is completed at the first stages of the disease with the assistance of this project and everyone other necessary, measure disease is cured and generally this prediction system can even be very useful in health industry. The final purpose of this Disease prediction is to supply prediction for the assorted and customarily occurring diseases that when unchecked and sometimes ignored can turns into fatal disease and cause lot of problem to the patient and moreover as their members of the family. So, with the assistance of algorithms, techniques and methodologies we've done this project which is able to help the peoples who are within the need.

APPROCH :

DATA SET

The dataset we will use in our project is Disease Prediction consists of 2 CSV files. One of them is for training and the other is for testing your model. Each CSV file has 133 columns. 132 of these columns are symptoms that a person experiences and the last column is the prognosis. And making the task of physicians easy is the main purpose of this dataset. Determined by the type of disease using the Random Forest model.

DATA COLLECTION

We download the Disease Prediction dataset from Kaggle, and this is the website https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning Because our project is Disease Prediction Using a Machine Learning scenario, we need to choose a dataset containing various diseases. The dataset selected is the best between different data on the same subject because it contains many types of diseases and symptoms.

MACHINE LEARNING MODEL

We have tried multiple prediction models but using the Random Forest classifier model has shown the best results:

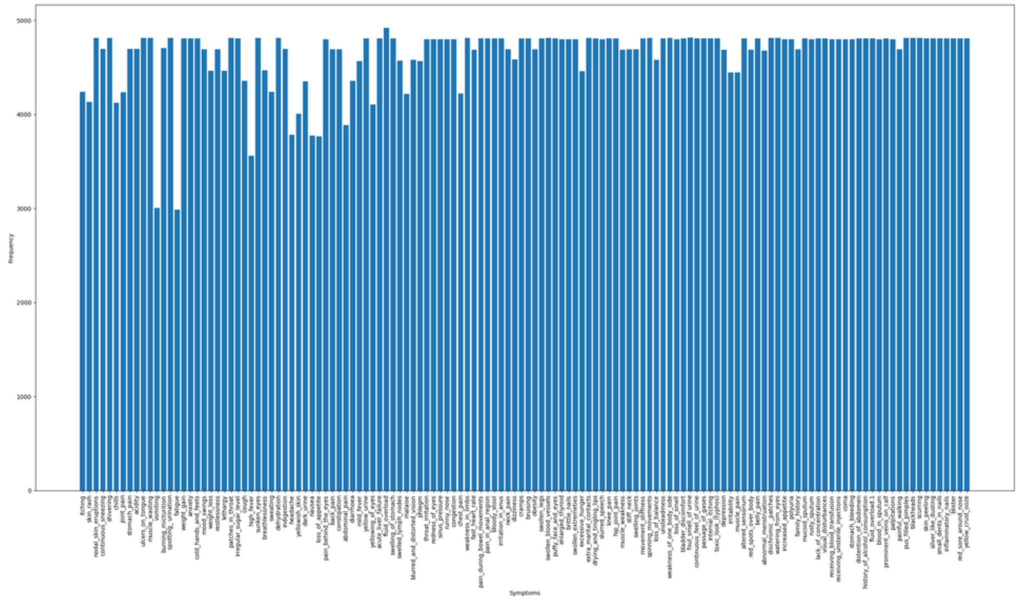
- Random Forest classifier

Random Forest classifier

The random forest classifier is a supervised learning algorithm that you can use for regression and classification problems. It is among the most popular machine learning algorithms due to its high flexibility and ease of implementation. So, The Random Forest is an ensemble learning method that can give more accurate predictions than most other machine learning algorithms. It is commonly used in decision tree learning. A forest is created using decision trees, each decision tree is a strong classifier on its own. These decision trees are used to create a forest of strong classifiers. This forest of strong classifiers gives a better prediction than decision trees or other machine learning algorithms.

Question/hypothesis

This picture shows the frequency of the symptom, we see that many of the symptoms occurred at about the same times, and the conclusion from this analysis inspired us to write these questions.



Predict whether the symptoms indicate a specific disease or not?

Is the increase in the proportion of symptoms a good indicator for the diagnosis of the disease or not?

Code and Analysis

We start with initial libraries such as:

- " Panda's library "is used in data analysis and data visualization in Python. It allows you to import data quickly from various sources, take it back to your machine to analyze it, and create compelling graphics.
- " Sklearn and Model_selection" Sklearn It is a Python library that offers various features for data processing that can be used for classification, clustering, and model selection. Model_selection is a method for setting a blueprint to analyze data and then using it to measure new data. Selecting a proper model allows you to generate accurate results when making a prediction. " train_test_split " is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data.
- " Sklearn metrics " are import metrics in SciKit Learn API to evaluate your machine learning algorithms.
- And For training the " random forest classifier " we have used sklearn RandomForestClassifier to make a classifier model. We are keeping most of its parameters as default and then pass our training data to fit.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
```

Importing the Data

Our data is divided into 2 files, Training and Testing. We used the read_csv() method provided by pandas to import our CSV files and assign it to our variables - train and test.

Import Data

```
train = pd.read_csv('Dataset/Training.csv')
test = pd.read_csv('Dataset/Testing.csv')
```

Preprocessing

Before splitting our data, we have to pre-process it to make sure the data is ready to be trained. Firstly, we check for missing (Nan) values. Using the isnull() function. It is visible that all features have 'False' as a result, meaning no missing values.

Preprocess

```
# check for null values
train.isnull().any()
```

```
itching                          False
skin_rash                        False
nodal_skin_eruptions             False
continuous_sneezing              False
shivering                        False
chills                           False
joint_pain                       False
stomach_pain                     False
acidity                          False
ulcers_on_tongue                 False
muscle_wasting                   False
vomiting                         False
burning_micturition              False
spotting_ urination              False
fatigue                          False
weight_gain                      False
anxiety                          False
cold_hands_and_feets             False
mood_swings                      False
weight_loss                      False
restlessness                     False
lethargy                         False
patches_in_throat                False
irregular_sugar_level            False
cough                            False
high_fever                       False
sunken_eyes                      False
breathlessness                   False
sweating                         False
dehydration                      False
indigestion                      False
headache                         False
yellowish_skin                   False
dark_urine                       False
nausea                           False
yellow_urine                     False
yellowing_of_eyes                False
acute_liver_failure              False
fluid_overload                   False
swelling_of_stomach              False
swelled_lymph_nodes              False
malaise                          False
blurred_and_distorted_vision     False
phlegm                           False
throat_irritation                False
redness_of_eyes                  False
sinus_pressure                   False
runny_nose                       False
congestion                       False
chest_pain                       False
weakness_in_limbs                False
fast_heart_rate                  False
pain_during_bowel_movements      False
pain_in_anal_region              False
bloody_stool                     False
irritation_in_anus               False
neck_pain                        False
dizziness                        False
cramps                           False
bruising                         False
obesity                          False
swollen_legs                     False
swollen_blood_vessels            False
puffy_face_and_eyes              False
enlarged_thyroid                 False
brittle_nails                    False
swollen_extremeties              False
excessive_hunger                 False
extra_marital_contacts           False
drying_and_tingling_lips         False
```

```
spinning_movements                False
loss_of_balance                   False
unsteadiness                      False
weakness_of_one_body_side         False
loss_of_smell                     False
bladder_discomfort                False
foul_smell_of urine               False
continuous_feel_of_urine          False
passage_of_gases                  False
internal_itching                  False
toxic_look_(typhos)               False
depression                        False
irritability                      False
muscle_pain                       False
altered_sensorium                 False
red_spots_over_body               False
belly_pain                        False
abnormal_menstruation             False
dischromic _patches               False
watering_from_eyes                False
increased_appetite                False
polyuria                          False
family_history                    False
mucoid_sputum                     False
rusty_sputum                      False
lack_of_concentration             False
visual_disturbances               False
receiving_blood_transfusion       False
receiving_unsterile_injections    False
coma                              False
stomach_bleeding                  False
distention_of_abdomen             False
history_of_alcohol_consumption    False
fluid_overload.1                  False
blood_in_sputum                   False
prominent_veins_on_calf           False
palpitations                      False
painful_walking                   False
pus_filled_pimples                False
blackheads                        False
scurring                          False
skin_peeling                      False
```

```
loss_of_appetite                  False
pain_behind_the_eyes              False
back_pain                         False
constipation                      False
abdominal_pain                    False
diarrhoea                         False
mild fever                        False
slurred_speech                    False
knee_pain                         False
hip_joint_pain                    False
muscle_weakness                   False
stiff_neck                        False
swelling_joints                   False
movement_stiffness                False
silver_like_dusting               False
small_dents_in_nails              False
inflammatory_nails                False
blister                           False
red_sore_around_nose              False
yellow_crust_ooze                 False
prognosis                         False
```

We've also displayed how much the diseases were mentioned in the set to make sure the data is unbiased. Using the value_counts() function to count the records of each disease, we observe that each disease has exactly 120 records, meaning the data is balanced and unbiased.

```
# check if balanced
train['prognosis'].value_counts()
```

```
Fungal infection                           120
Hepatitis C                                120
Hepatitis E                                120
Alcoholic hepatitis                        120
Tuberculosis                               120
Common Cold                                120
Pneumonia                                  120
Dimorphic hemmorhoids(piles)               120
Heart attack                               120
Varicose veins                             120
Hypothyroidism                             120
Hyperthyroidism                            120
Hypoglycemia                               120
Osteoarthristis                            120
Arthritis                                  120
(vertigo) Paroymsal  Positional Vertigo    120
Acne                                       120
Urinary tract infection                    120
Psoriasis                                  120
Hepatitis D                                120
Hepatitis B                                120
Allergy                                    120
hepatitis A                                120
GERD                                       120
Chronic cholestasis                        120
Drug Reaction                              120
Peptic ulcer diseae                        120
AIDS                                       120
Diabetes                                   120
Gastroenteritis                            120
Bronchial Asthma                           120
Hypertension                               120
Migraine                                   120
Cervical spondylosis                       120
Paralysis (brain hemorrhage)               120
Jaundice                                   120
Malaria                                    120
Chicken pox                                120
Dengue                                     120
Typhoid                                    120
Impetigo                                   120
```

Splitting the data

Now comes the splitting! We divide the training data to diseases and symptoms, A and B alternately. Then we split the data into 80% training and 20% testing datasets for the purpose of training and testing the model to help determine the accuracy of the machine learning algorithm. C will contain the symptoms from the testing csv file, that we will use to compare the actual and predicted value.

Split data

```
A = train[["prognosis"]] # diseases
B = train.drop(["prognosis"],axis=1) # symptoms
C = test.drop(["prognosis"],axis=1) # symptoms - testing
x_train, x_test, y_train, y_test = train_test_split(B,A,test_size=0.2)
```

Training the model

We set parameters:

- n_estimators: The number of trees in the forest = 100
- criterion: The function to measure the quality of a split = entropy (Used for information gain).
- n_jobs: The number of jobs to run in parallel = 5
- random_statet: Controls sampling and the randomness of bootstrapping = 42

Fitting the model, using fit(), to the training data is essentially the training part of the modelling process. Reval() is used to reshape the array created. Then, the model classifies the data using predict() by passing the symptoms to predict the diseases.

```
# Traning random forest model
mod = RandomForestClassifier(n_estimators = 100,n_jobs = 5, criterion= 'entropy',random_state = 42)
mod = mod.fit(x_train,y_train.values.ravel())
pred = mod.predict(x_test)
```

Performance

THE MODEL HAS AN ACCURACY OF 100%

Using metrics.accuracy_score() to calculate the accuracy, we get 100%

Accuracy

```
metrics.accuracy_score(y_test, pred)
```

```
1.0
```

We then used classification_report() to check the precision (ration of true positive to the sum of true and false positive), recall (ratio of true positive to the sum of true positive and false negative), f1-score (weighted mean of precision and recall, the closer to 1 the better expected performance) and support (number of occurrences of the class in the dataset).

```
report = classification_report(y_test, pred, output_dict=True)
pd.DataFrame(report).transpose()
```

	precision	recall	f1-score	support
(vertigo) Paroymsal Positional Vertigo	1.0	1.0	1.0	27.0
AIDS	1.0	1.0	1.0	13.0
Acne	1.0	1.0	1.0	25.0
Alcoholic hepatitis	1.0	1.0	1.0	20.0
Allergy	1.0	1.0	1.0	28.0
Arthritis	1.0	1.0	1.0	28.0
Bronchial Asthma	1.0	1.0	1.0	21.0
Cervical spondylosis	1.0	1.0	1.0	27.0
Chicken pox	1.0	1.0	1.0	17.0
Chronic cholestasis	1.0	1.0	1.0	28.0
Common Cold	1.0	1.0	1.0	26.0
Dengue	1.0	1.0	1.0	23.0
Diabetes	1.0	1.0	1.0	32.0
Dimorphic hemmorhoids(piles)	1.0	1.0	1.0	23.0
Drug Reaction	1.0	1.0	1.0	28.0
Fungal infection	1.0	1.0	1.0	18.0
GERD	1.0	1.0	1.0	20.0
Gastroenteritis	1.0	1.0	1.0	22.0
Heart attack	1.0	1.0	1.0	21.0
Hepatitis B	1.0	1.0	1.0	26.0
Hepatitis C	1.0	1.0	1.0	14.0
Hepatitis D	1.0	1.0	1.0	20.0
Hepatitis E	1.0	1.0	1.0	24.0
Hypertension	1.0	1.0	1.0	22.0
Hyperthyroidism	1.0	1.0	1.0	26.0
Hypoglycemia	1.0	1.0	1.0	28.0
Hypothyroidism	1.0	1.0	1.0	28.0
Impetigo	1.0	1.0	1.0	30.0
Jaundice	1.0	1.0	1.0	24.0
Malaria	1.0	1.0	1.0	25.0
Migraine	1.0	1.0	1.0	27.0
Osteoarthristis	1.0	1.0	1.0	26.0
Paralysis (brain hemorrhage)	1.0	1.0	1.0	25.0
Peptic ulcer diseae	1.0	1.0	1.0	22.0
Pneumonia	1.0	1.0	1.0	24.0
Psoriasis	1.0	1.0	1.0	27.0
Tuberculosis	1.0	1.0	1.0	22.0
Typhoid	1.0	1.0	1.0	24.0
Urinary tract infection	1.0	1.0	1.0	24.0
Varicose veins	1.0	1.0	1.0	21.0
hepatitis A	1.0	1.0	1.0	28.0
accuracy	1.0	1.0	1.0	1.0
macro avg	1.0	1.0	1.0	984.0
weighted avg	1.0	1.0	1.0	984.0

Performance

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	26	0	0	0	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	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	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	19	0	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	0	0	0	0	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	34	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	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
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Confusion Matrix

```
cm = confusion_matrix(y_test, pred)
pd.DataFrame(cm)
```

This is a sample view of the matrix. A full one will be provided in the Appendix.

Now to make sure that our model is not overfitted. We used the model on the testing symptoms and compared the actual values to the predicted ones. We can see that all predictions are correct. Unlike the decision tree classification model shown in the appendx.

```
test = test.join(pd.DataFrame(mod.predict(C),columns=["predicted"]))[["prognosis","predicted"]]

test['result']= ' '
for i in range(len(test)):
    if test["prognosis"][i] == test["predicted"][i]:
        test['result'].iloc[i] = 'Correct'
    else:
        test['result'].iloc[i] = 'Incorrect'
```

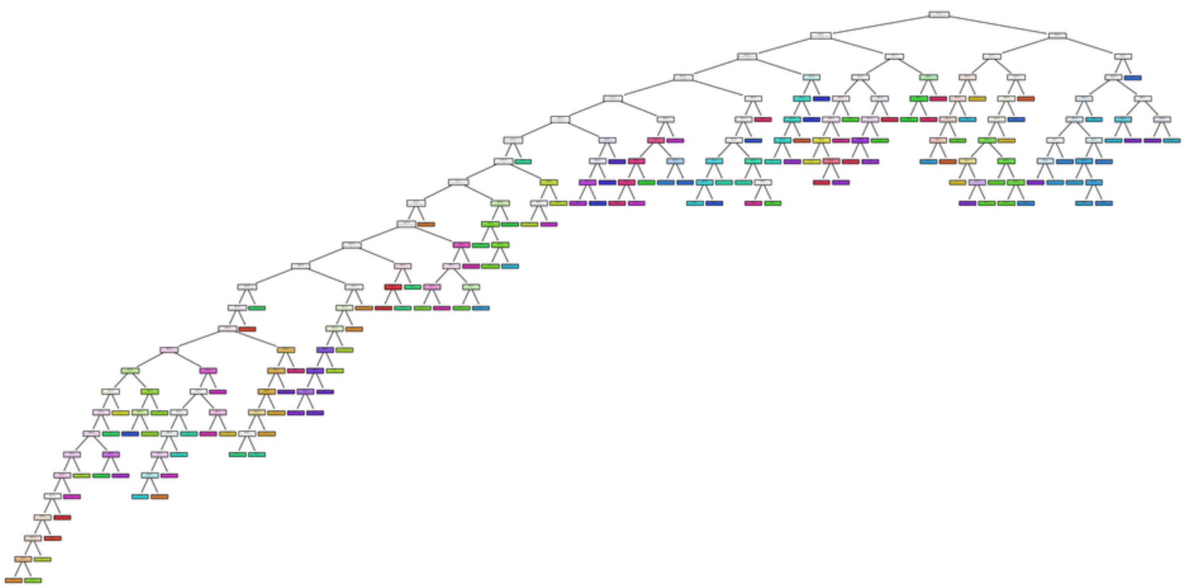
	prognosis	predicted	result
0	Fungal infection	Fungal infection	Correct
1	Allergy	Allergy	Correct
2	GERD	GERD	Correct
3	Chronic cholestasis	Chronic cholestasis	Correct
4	Drug Reaction	Drug Reaction	Correct
5	Peptic ulcer diseae	Peptic ulcer diseae	Correct
6	AIDS	AIDS	Correct
7	Diabetes	Diabetes	Correct
8	Gastroenteritis	Gastroenteritis	Correct
9	Bronchial Asthma	Bronchial Asthma	Correct
10	Hypertension	Hypertension	Correct
11	Migraine	Migraine	Correct
12	Cervical spondylosis	Cervical spondylosis	Correct
13	Paralysis (brain hemorrhage)	Paralysis (brain hemorrhage)	Correct
14	Jaundice	Jaundice	Correct
15	Malaria	Malaria	Correct
16	Chicken pox	Chicken pox	Correct
17	Dengue	Dengue	Correct

Performance

18	Typhoid	Typhoid	Correct
19	hepatitis A	hepatitis A	Correct
20	Hepatitis B	Hepatitis B	Correct
21	Hepatitis C	Hepatitis C	Correct
22	Hepatitis D	Hepatitis D	Correct
23	Hepatitis E	Hepatitis E	Correct
24	Alcoholic hepatitis	Alcoholic hepatitis	Correct
25	Tuberculosis	Tuberculosis	Correct
26	Common Cold	Common Cold	Correct
27	Pneumonia	Pneumonia	Correct
28	Dimorphic hemmorhoids(piles)	Dimorphic hemmorhoids(piles)	Correct
29	Heart attack	Heart attack	Correct
30	Varicose veins	Varicose veins	Correct
31	Hypothyroidism	Hypothyroidism	Correct
32	Hyperthyroidism	Hyperthyroidism	Correct
33	Hypoglycemia	Hypoglycemia	Correct
34	Osteoarthristis	Osteoarthristis	Correct
35	Arthritis	Arthritis	Correct
36	(vertigo) Paroymsal Positional Vertigo	(vertigo) Paroymsal Positional Vertigo	Correct
37	Acne	Acne	Correct
38	Urinary tract infection	Urinary tract infection	Correct
39	Psoriasis	Psoriasis	Correct
40	Impetigo	Impetigo	Correct
41	Fungal infection	Fungal infection	Correct

Here is a display of one of the decision trees in the forest. We have randomly chosen the tree at index 8:

```
from sklearn import tree
plt.figure(figsize=(30,15))
tree.plot_tree(mod.estimators_[8],filled = True)
```



Conclusion

The main aim of this disease prediction model is to predict the disease based on symptoms This model takes the symptoms of the user from which he or she suffers as input and generates the final output as a prediction of disease. In conclusion, for disease risk modeling, the accuracy of risk prediction depends on the diverse feature of the Health centers' data. Findings may help inform future developers of Disease Predictability Software and promote personalized patient care. The model predicts Patient Diseases through the Random Forest algorithm. Model accuracy reaches 100%. Machine learning skills are designed for Disease Prediction successfully

Resources

https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning

sklearn.ensemble.RandomForestClassifier — scikit-learn 1.2.1 documentation

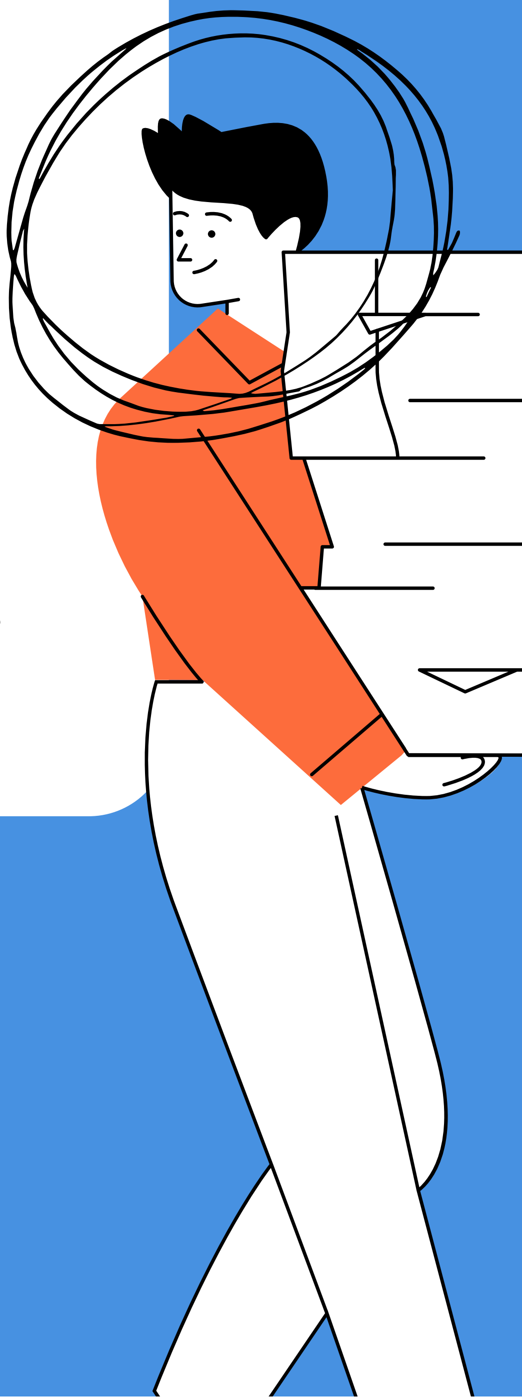
Random Forest Classifier using Scikit-learn - GeeksforGeeks

https://www.ijraset.com/research-paper/disease-prediction-using-ml

Random Forest Classifier: Overview, How Does it Work, Pros & Cons | upGrad blog

Random Forest Classifier in Python Sklearn with Example - MLK - Machine Learning Knowledge

Sklearn metrics for Machine Learning in Python - Machine Learning HD



Appendix.

Comparision of actual data and predicted values of decision tree classification model:

Import Data

```
train = pd.read_csv('Dataset/Training.csv')
test = pd.read_csv('Dataset/Testing.csv')
```

Split Data

```
A = train[["prognosis"]] #diseases
B = train.drop(["prognosis"],axis=1) #symptoms
C = test.drop(["prognosis"],axis=1) #testing - symptoms
x_train, x_test, y_train, y_test = train_test_split(B,A,test_size=0.2)
```

Model

```
mod = DecisionTreeClassifier()
mod = mod.fit(x_train,y_train)
pred = mod.predict(x_test)
```

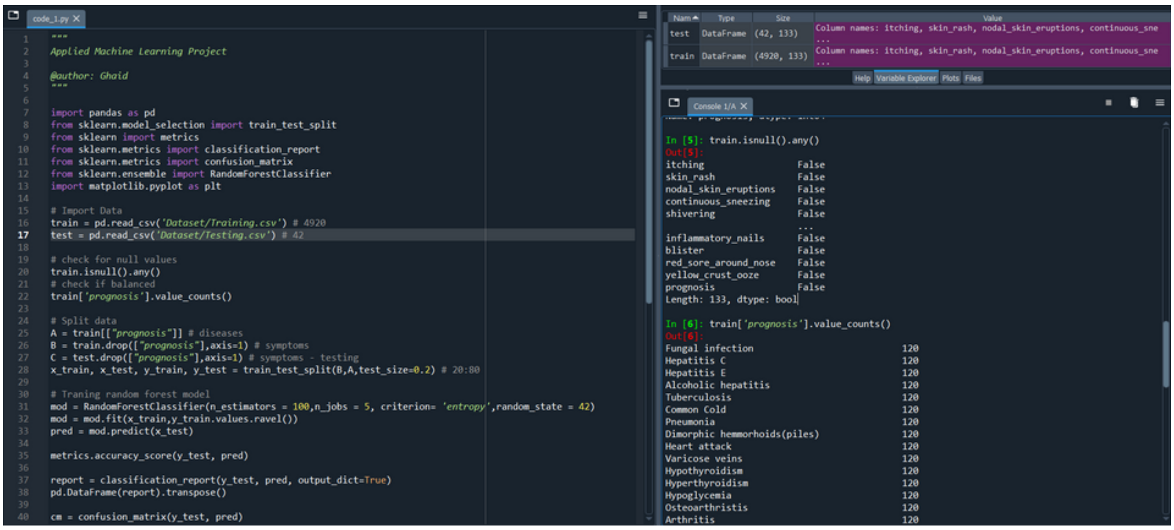
```
test = test.join(pd.DataFrame(mod.predict(C),columns=["predicted"]))[["prognosis","predicted"]]

test['result']= ' '
for i in range(len(test)):
    if test["prognosis"][i] == test["predicted"][i]:
        test['result'].iloc[i] = 'Correct'
    else:
        test['result'].iloc[i] = 'Incorrect'
```

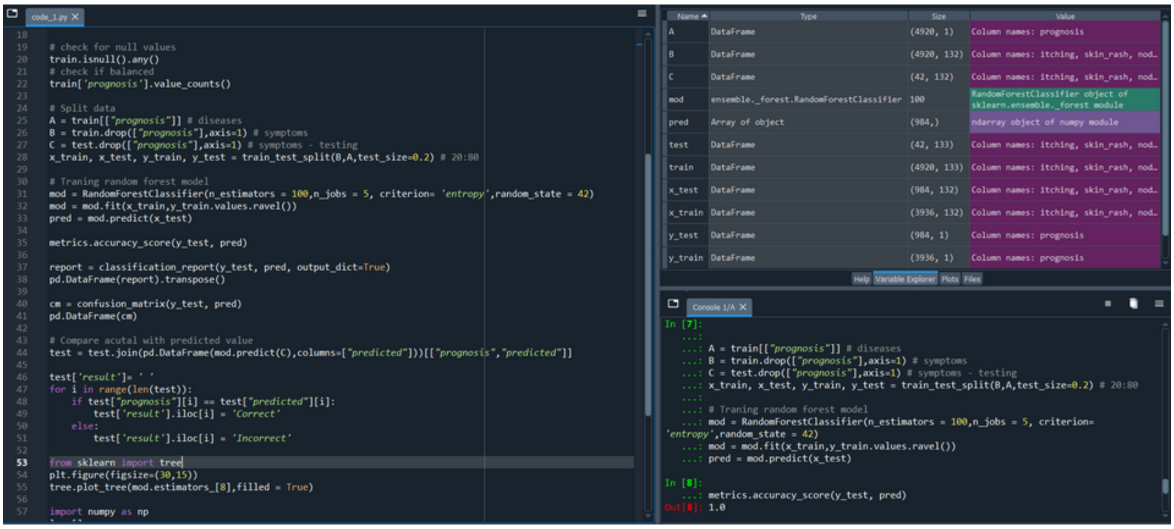
	prognosis	predicted	result
0	Fungal infection	Fungal infection	Correct
1	Allergy	Allergy	Correct
2	GERD	GERD	Correct
3	Chronic cholestasis	Chronic cholestasis	Correct
4	Drug Reaction	Drug Reaction	Correct
5	Peptic ulcer diseae	Peptic ulcer diseae	Correct
6	AIDS	AIDS	Correct
7	Diabetes	Diabetes	Correct
8	Gastroenteritis	Gastroenteritis	Correct
9	Bronchial Asthma	Bronchial Asthma	Correct
10	Hypertension	Hypertension	Correct
11	Migraine	Migraine	Correct
12	Cervical spondylosis	Cervical spondylosis	Correct
13	Paralysis (brain hemorrhage)	Paralysis (brain hemorrhage)	Correct
14	Jaundice	Jaundice	Correct
15	Malaria	Malaria	Correct
16	Chicken pox	Chicken pox	Correct

Appendix.

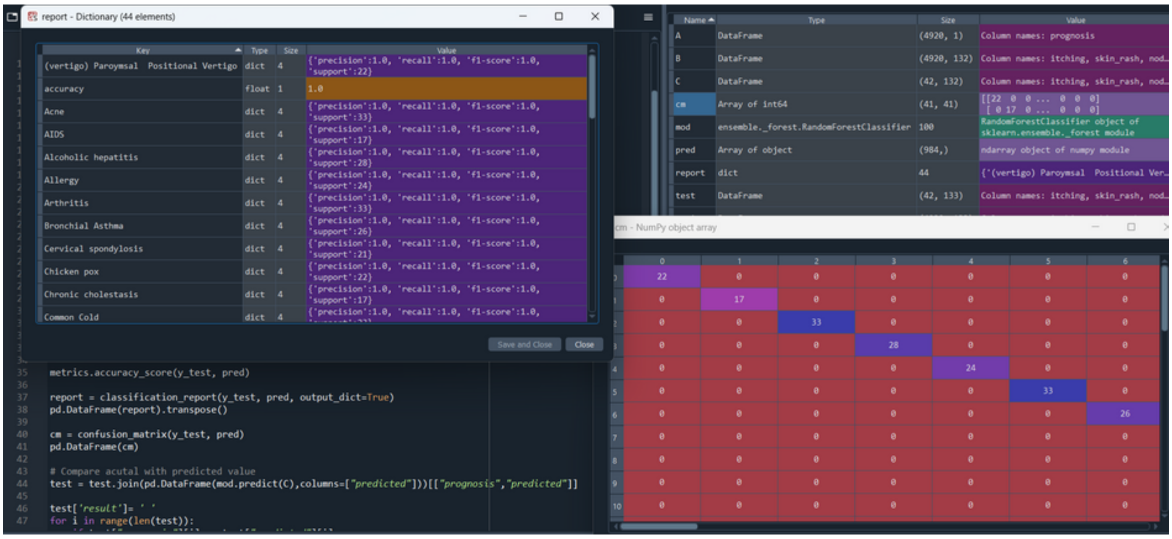
17	Dengue	Dengue	Correct
18	Typhoid	Typhoid	Correct
19	hepatitis A	hepatitis A	Correct
20	Hepatitis B	Hepatitis B	Correct
21	Hepatitis C	Hepatitis C	Correct
22	Hepatitis D	Hepatitis D	Correct
23	Hepatitis E	Hepatitis E	Correct
24	Alcoholic hepatitis	Alcoholic hepatitis	Correct
25	Tuberculosis	Tuberculosis	Correct
26	Common Cold	Common Cold	Correct
27	Pneumonia	Pneumonia	Correct
28	Dimorphic hemmorhoids(piles)	Dimorphic hemmorhoids(piles)	Correct
29	Heart attack	Heart attack	Correct
30	Varicose veins	Varicose veins	Correct
31	Hypothyroidism	Hypothyroidism	Correct
32	Hyperthyroidism	Hyperthyroidism	Correct
33	Hypoglycemia	Hypoglycemia	Correct
34	Osteoarthristis	Osteoarthristis	Correct
35	Arthritis	Arthritis	Correct
36	(vertigo) Paroymsal Positional Vertigo	(vertigo) Paroymsal Positional Vertigo	Correct
37	Acne	Acne	Correct
38	Urinary tract infection	Urinary tract infection	Correct
39	Psoriasis	Psoriasis	Correct
40	Impetigo	Impetigo	Correct
41	Fungal infection	Impetigo	Incorrect



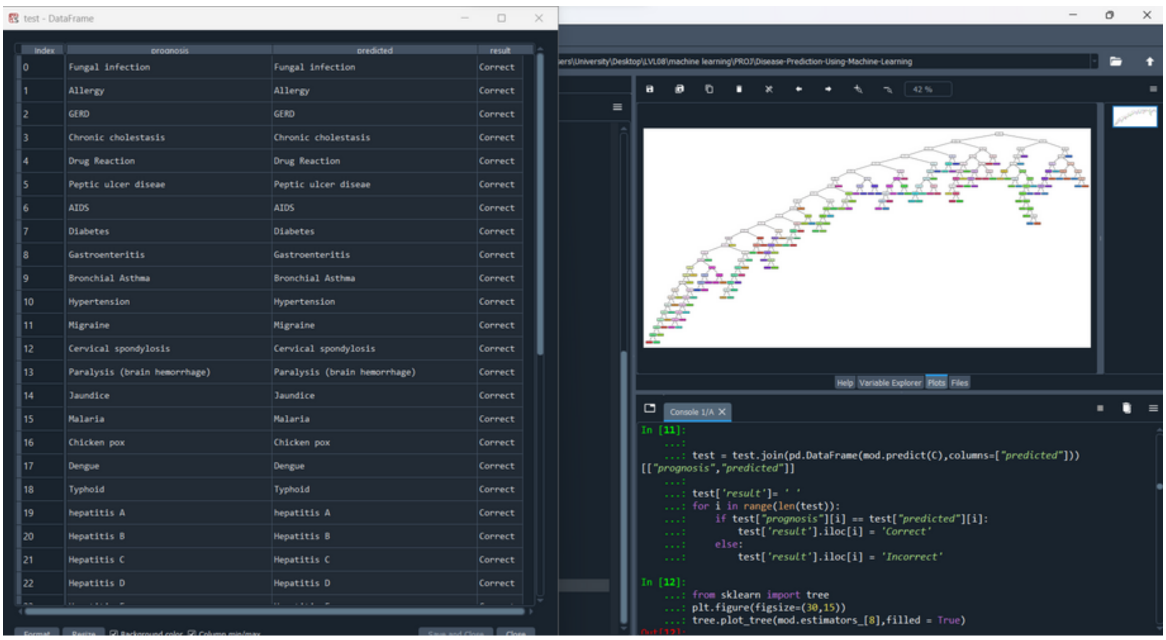
Appendix.



# check for null values
train.isnull().any()
# check if balanced
train['prognosis'].value_counts()
# Split data
A = train[["prognosis"]] # diseases
B = train.drop(["prognosis"],axis=1) # symptoms
C = test.drop(["prognosis"],axis=1) # symptoms - testing
x_train, x_test, y_train, y_test = train_test_split(B,A,test_size=0.2) # 20:80
# Traning random forest model
mod = RandomForestClassifier(n_estimators = 100,n_jobs = 5, criterion= 'entropy',random_state = 42)
mod = mod.fit(x_train,y_train.values.ravel())
pred = mod.predict(x_test)
metrics.accuracy_score(y_test, pred)
report = classification_report(y_test, pred, output_dict=True)
pd.DataFrame(report).transpose()
cm = confusion_matrix(y_test, pred)
pd.DataFrame(cm)
# Compare acutal with predicted value
test = test.join(pd.DataFrame(mod.predict(C),columns=["predicted"]))[["prognosis","predicted"]]
test['result']= ' '
for i in range(len(test)):
if test["prognosis"][i] == test["predicted"][i]:
test['result'].iloc[i] = 'Correct'
else:
test['result'].iloc[i] = 'Incorrect'
from sklearn import tree
plt.figure(figsize=(30,15))
tree.plot_tree(mod.estimators_[8],filled = True)
import numpy as np

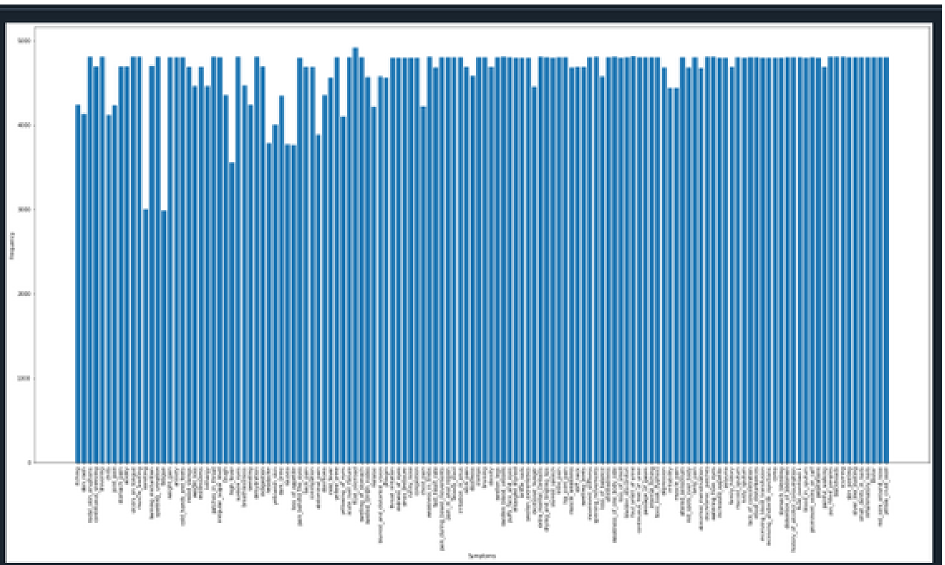


report - Dictionary (44 elements)
cm - NumPy object array



test - DataFrame

Appendix.



Help Variable Explorer Plots Files

Console 1/A

```
In [13]:
    ...: import numpy as np
    ...: l = []
    ...: sym = list(train.drop('prognosis',axis=1))
    ...: arr = np.array(sym)
    ...: arr2 = np.array(l)
    ...: plt.figure(figsize = (30,15))
    ...: for j in sym:
    ...:     h = B[j].value_counts().values
    ...:     arr2 = np.append(arr2,h[:1])
    ...:
    ...: plt.xticks(rotation = 90)
    ...: plt.ylabel("Frequency")
    ...: plt.xlabel("Symptoms")
    ...: plt.bar(arr,arr2)
Out[13]: <BarContainer object of 132 artists>
```