Disease Prediction

System Using

Machine-Learning

Project-2

**INTRODUCTION**

This document presents a project on disease prediction using machine learning techniques. The project utilizes datasets from Kaggle, specifically "Training.csv" and "Testing.csv", to train and test machine learning models for predicting various diseases based on symptoms provided by the user. Disease prediction is a critical aspect of healthcare management, allowing for early diagnosis and timely intervention. Machine learning (ML) techniques offer promising avenues for disease prediction by leveraging large datasets to identify patterns and relationships between symptoms and diseases. This document describes a project aimed at predicting diseases using machine learning techniques. The project utilizes Python programming language with libraries such as Tkinter for creating a graphical user interface (GUI) and sci-kit-learn for implementing machine learning algorithms.

**OBJECTIVES**

The primary objective of this project is to develop a system capable of accurately predicting diseases based on symptoms provided by the user. We aim to implement and compare the performance of various ML algorithms for disease prediction, including Decision Trees, Random Forests, and Naive Bayes classifiers.

**MACHINE LEARNING**

Machine learning (ML) is a subset of artificial intelligence (AI) that involves the development of algorithms and models that enable computers to learn from and make predictions or decisions based on data, without being explicitly programmed to perform specific tasks. In essence, it is about creating systems that can automatically learn and improve from experience.

**METHODS/ALGORITHMS:**

**1. Supervised Learning:**

Definition: In supervised learning, the algorithm learns from labeled data, where each input is associated with a corresponding output. The goal is to learn a mapping from inputs to outputs based on the provided examples.

Example: A common example is email spam detection. Given a dataset of emails labeled as either "spam" or "not spam," a supervised learning algorithm can be trained to classify new emails into one of these categories based on features extracted from the email content (e.g., words frequency, presence of certain keywords).

**2. Unsupervised Learning:**

Definition: Unsupervised learning involves learning from unlabeled data, where the algorithm tries to find patterns or structures within the data without explicit guidance.

Example: An example of unsupervised learning is customer segmentation in marketing. Given a dataset of customer purchase histories, an unsupervised learning algorithm can group customers with similar purchasing behavior into segments, which can then be used for targeted marketing campaigns.

**3. Semi-supervised Learning:**

Definition: Semi-supervised learning is a combination of supervised and unsupervised learning, where the algorithm learns from a small amount of labeled data along with a large amount of unlabeled data.

Example: An example of semi-supervised learning is speech recognition. By providing a small amount of labeled audio data (e.g., audio recordings with corresponding transcriptions) along with a large amount of unlabeled data, a semi-supervised learning algorithm can improve the accuracy of transcribing speech in new, unlabeled audio recordings.

**4. Reinforcement Learning:**

Definition: Reinforcement learning involves learning through interaction with an environment to achieve a goal. The algorithm learns by receiving feedback in the form of rewards or penalties based on its actions.

Example: A classic example of reinforcement learning is training an AI agent to play a game, such as chess or Go. The agent takes actions (moves) in the game environment, receives feedback (winning or losing the game), and adjusts its strategy over time to maximize its chances of winning.

**5. Deep Learning:**

Definition: Deep learning utilizes artificial neural networks with multiple layers to learn complex patterns in large amounts of data. Deep learning has shown remarkable success in various domains, including computer vision, natural language processing, and speech recognition.

Example: An example of deep learning is image classification using Convolutional Neural Networks (CNNs). CNNs can automatically learn hierarchical features from images, such as edges, textures, and object shapes, enabling accurate classification of images into different categories (e.g., cats, dogs, cars).

**Features of ML:**

1. Feature Engineering: Process of selecting, extracting, and transforming relevant features from raw data to improve model performance.

2. Feature Selection: Identifying the most relevant features for training the model, often to reduce dimensionality and computational complexity.

3. Feature Extraction: Transforming raw data into a set of meaningful features that can be used for training.

**Components of ML:**

1. Data: The foundation of machine learning, including input data used for training and testing models.

2. Model: The algorithm or set of algorithms used to learn patterns from the data and make predictions or decisions.

3. Evaluation: Assessing the performance of the model using metrics such as accuracy, precision, recall, or F1-score.

4. Deployment: Integrating the trained model into real-world applications for making predictions or decisions.

**Benefits of ML:**

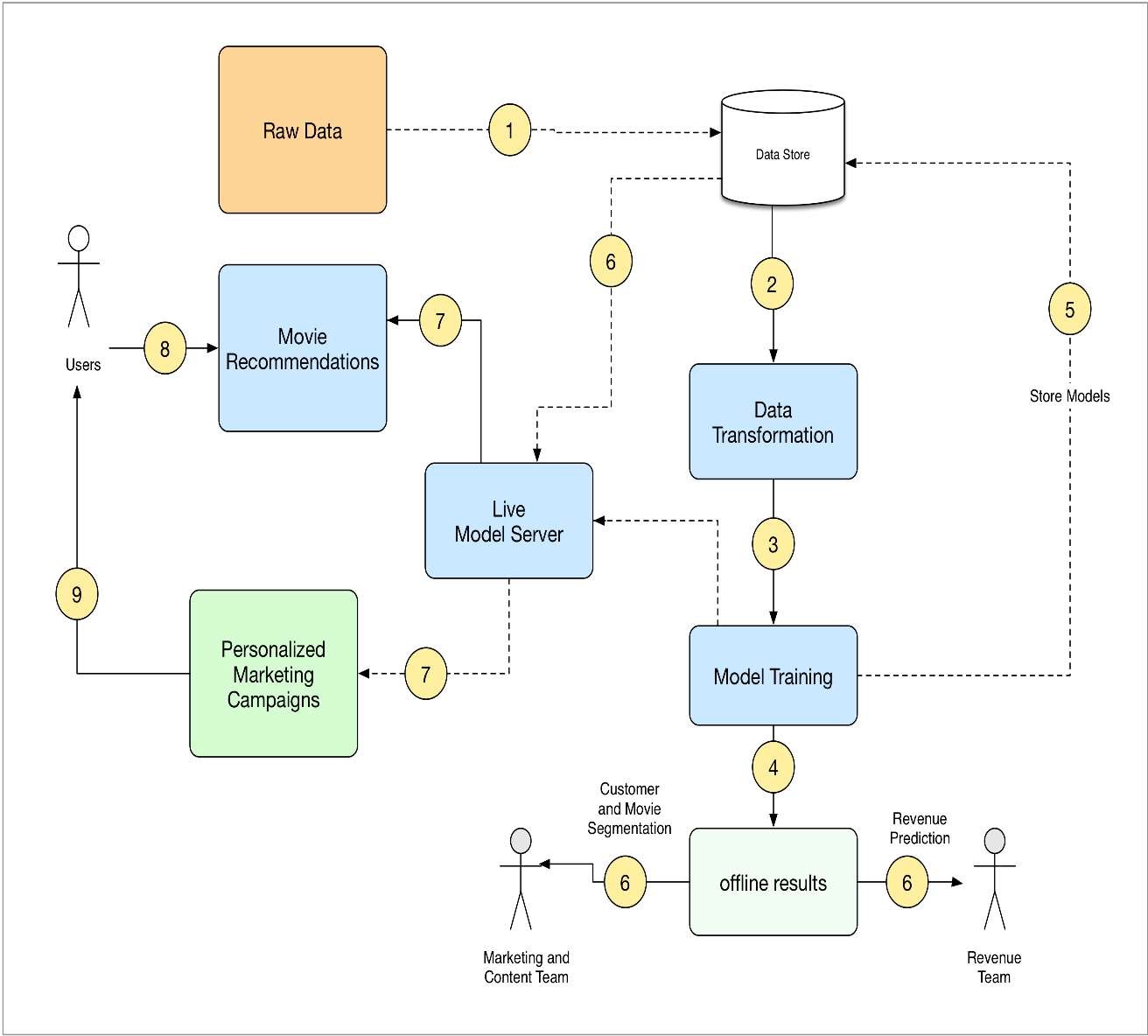
1. Predictive Analytics: Forecasting future trends or outcomes based on historical data.

2. Classification: Categorizing input data into different classes or categories.

3. Regression: Predicting continuous numerical values based on input data.

4. Clustering: Grouping similar data points together based on their features.

5. Anomaly Detection: Identifying unusual patterns or outliers in data.

**Machine Learning Architecture:**

**Human Learning vs Machine Learning:**

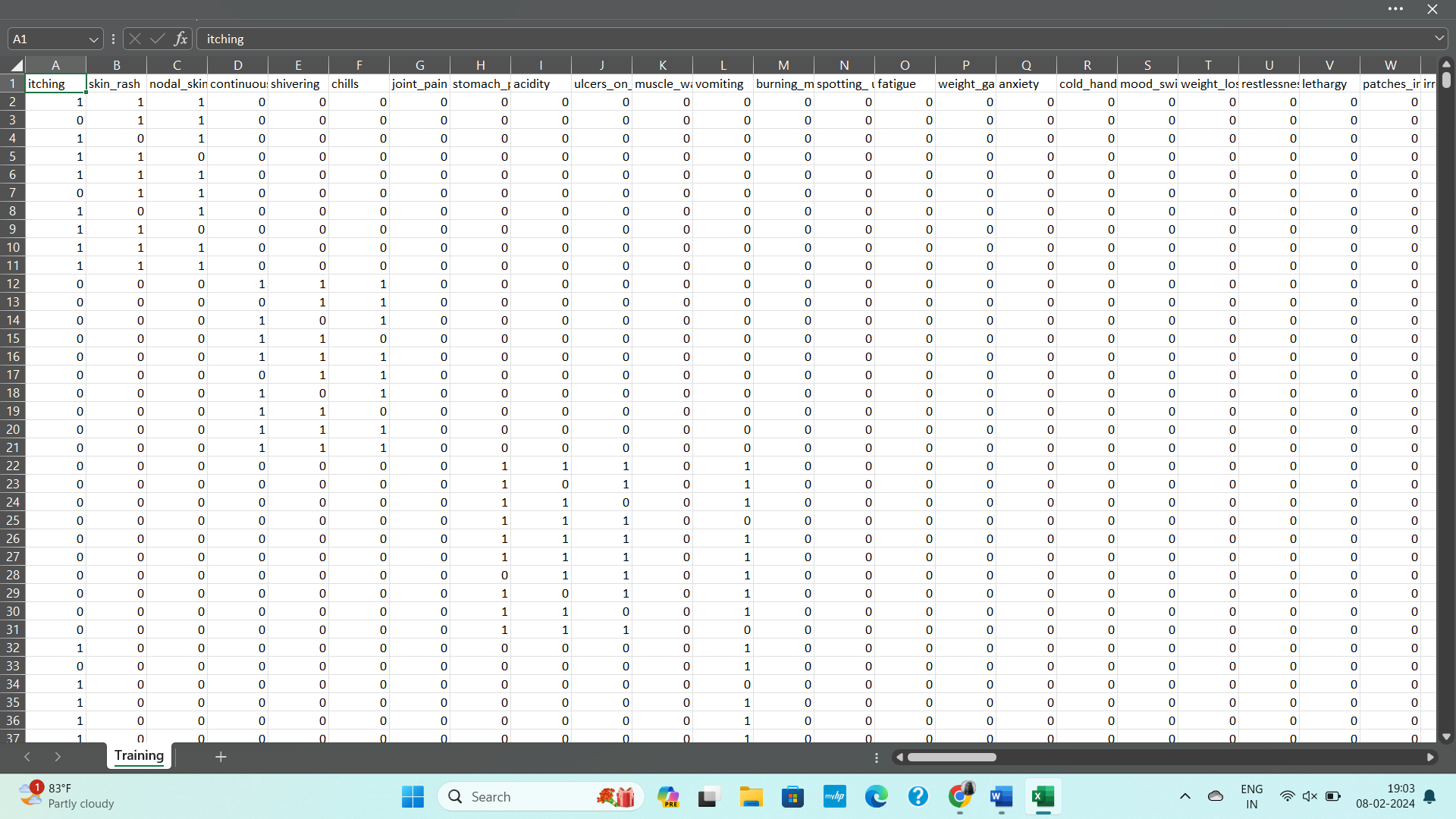
|  |  |  |
| --- | --- | --- |
| **Dimension** | **Human Learning** | **Machine Learning** |
| Speed | Slow | Fast |
| Ability to Transfer | No Copy mechanism | Easy to Copy |
| Required Repetition | Yes | Yes/No |
| Error-prone | Yes | Yes |
| Noise- tolerant | Yes | No |

**DATASET DESCRIPTION**

The datasets used in this project contain symptom data along with the corresponding diseases. Here are some key details about the datasets:

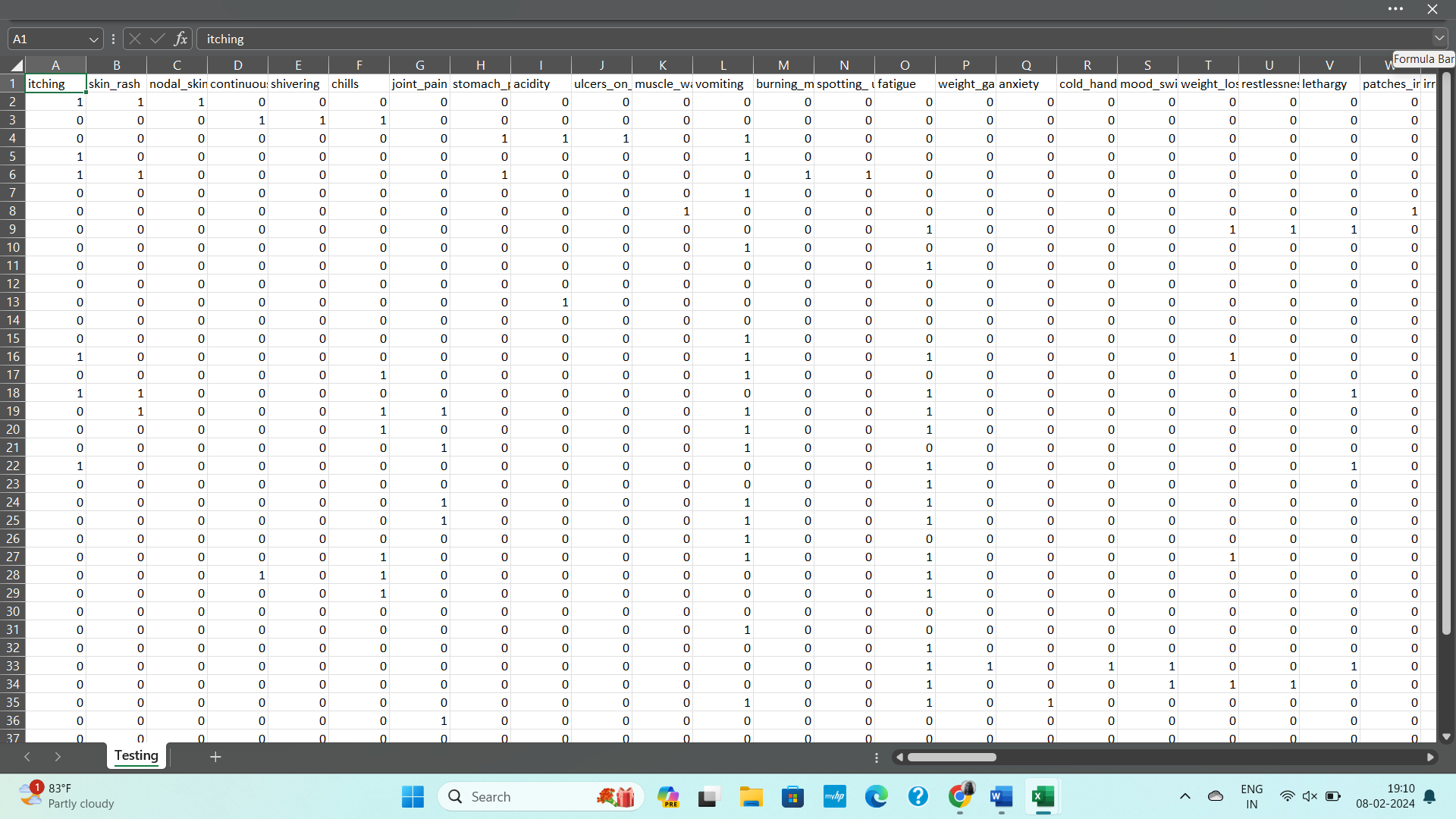
**Training Dataset(“Training.csv”):**

This dataset is used to train the machine learning models. It contains symptom data and corresponding disease labels.



**Testing Dataset(“Testing.csv”):**

This dataset is used to evaluate the performance of the trained models. Similar to the training dataset, it contains symptom data and disease labels.



**PROGRAM**

from tkinter import \*

import numpy as np

import pandas as pd

l1=['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','fluid\_overload','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']

disease=['Fungal infection','Allergy','GERD','Chronic cholestasis','Drug Reaction',

'Peptic ulcer disease',' AIDS',' Diabetes','Gastroenteritis','Bronchial Asthma','Hypertension',' Migraine',' Cervical spondylosis','Paralysis (brain hemorrhage)','Jaundice','Malaria','Chickenpox','Dengue','Typhoid','hepatitis A',

'Hepatitis B','Hepatitis C','Hepatitis D','Hepatitis E','Alcoholic hepatitis','Tuberculosis',

'Common Cold','Pneumonia','Dimorphic hemmorhoids(piles)','Heartattack','Varicoseveins','Hypothyroidism','Hyperthyroidism','Hypoglycemia','Osteoarthristis',

'Arthritis','(vertigo) Paroymsal Positional Vertigo','Acne','Urinary tract infection','Psoriasis','Impetigo']

l2=[]

for x in range(0,len(l1)):

l2.append(0)

# TESTING DATA df -------------------------------------------------------------------------------------

df=pd.read\_csv("Training.csv")

df.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,'Peptic ulcerdiseae':5,'AIDS':6,'Diabetes':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,

'Migraine':11,'Cervical spondylosis':12,

'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,

'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,

'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,

'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,

'(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,

'Impetigo':40}},inplace=True)

# print(df.head())

X= df[l1]

y = df[["prognosis"]]

np.ravel(y)

# print(y)

# TRAINING DATA tr --------------------------------------------------------------------------------

tr=pd.read\_csv("Testing.csv")

tr.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,

'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,

'Migraine':11,'Cervical spondylosis':12,

'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,

'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,

'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,

'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,

'(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,

'Impetigo':40}},inplace=True)

X\_test= tr[l1]

y\_test = tr[["prognosis"]]

np.ravel(y\_test)

# ------------------------------------------------------------------------------------------------------

def DecisionTree():

from sklearn import tree

clf3 = tree.DecisionTreeClassifier() # empty model of the decision tree

clf3 = clf3.fit(X,y)

# calculating accuracy-------------------------------------------------------------------

from sklearn.metrics import accuracy\_score

y\_pred=clf3.predict(X\_test)

print(accuracy\_score(y\_test, y\_pred))

print(accuracy\_score(y\_test, y\_pred,normalize=False))

# -----------------------------------------------------

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

for k in range(0,len(l1)):

# print (k,)

for z in psymptoms:

if(z==l1[k]):

l2[k]=1

inputtest = [l2]

predict = clf3.predict(inputtest)

predicted=predict[0]

h='no'

for a in range(0,len(disease)):

if(predicted == a):

h='yes'

break

if (h=='yes'):

t1.delete("1.0", END)

t1.insert(END, disease[a])

else:

t1.delete("1.0", END)

t1.insert(END, "Not Found")

def randomforest():

from sklearn.ensemble import RandomForestClassifier

clf4 = RandomForestClassifier()

clf4 = clf4.fit(X,np.ravel(y))

# calculating accuracy-------------------------------------------------------------------

from sklearn.metrics import accuracy\_score

y\_pred=clf4.predict(X\_test)

print(accuracy\_score(y\_test, y\_pred))

print(accuracy\_score(y\_test, y\_pred,normalize=False))

# -----------------------------------------------------

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

for k in range(0,len(l1)):

for z in psymptoms:

if(z==l1[k]):

l2[k]=1

inputtest = [l2]

predict = clf4.predict(inputtest)

predicted=predict[0]

h='no'

for a in range(0,len(disease)):

if(predicted == a):

h='yes'

break

if (h=='yes'):

t2.delete("1.0", END)

t2.insert(END, disease[a])

else:

t2.delete("1.0", END)

t2.insert(END, "Not Found")

def NaiveBayes():

from sklearn.naive\_bayes import GaussianNB

gnb = GaussianNB()

gnb=gnb.fit(X,np.ravel(y))

# calculating accuracy-------------------------------------------------------------------

from sklearn.metrics import accuracy\_score

y\_pred=gnb.predict(X\_test)

print(accuracy\_score(y\_test, y\_pred))

print(accuracy\_score(y\_test, y\_pred,normalize=False))

# -----------------------------------------------------

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

for k in range(0,len(l1)):

for z in psymptoms:

if(z==l1[k]):

l2[k]=1

inputtest = [l2]

predict = gnb.predict(inputtest)

predicted=predict[0]

h='no'

for a in range(0,len(disease)):

if(predicted == a):

h='yes'

break

if (h=='yes'):

t3.delete("1.0", END)

t3.insert(END, disease[a])

else:

t3.delete("1.0", END)

t3.insert(END, "Not Found")

# gui\_stuff------------------------------------------------------------------------------------

root = Tk()

root.title("My Doctor")

root.configure(background='grey')

# entry variables

Symptom1 = StringVar()

Symptom1.set(None)

Symptom2 = StringVar()

Symptom2.set(None)

Symptom3 = StringVar()

Symptom3.set(None)

Symptom4 = StringVar()

Symptom4.set(None)

Symptom5 = StringVar()

Symptom5.set(None)

Name = StringVar()

# Heading

w2 = Label(root, justify=LEFT, text="Disease Predictor using Machine Learning", fg="white", bg="grey")

w2.config(font=("Elephant", 30))

w2.grid(row=1, column=0, columnspan=2, padx=100)

w2 = Label(root, justify=LEFT, text="A Project by Mohit Agrawal", fg="red", bg="grey")

w2.config(font=("Aharoni", 30))

w2.grid(row=2, column=0, columnspan=2, padx=100)

# labels

NameLb = Label(root, text="Name of the Patient", fg="white", bg="black")

NameLb.grid(row=6, column=0, pady=25, sticky=W)

NameLb.config(font=("Aharoni", 15))

S1Lb = Label(root, text="Symptom 1", fg="white", bg="black")

S1Lb.grid(row=7, column=0, pady=20, sticky=W)

S1Lb.config(font=("Aharoni", 15))

S2Lb = Label(root, text="Symptom 2", fg="white", bg="black")

S2Lb.grid(row=8, column=0, pady=20, sticky=W)

S2Lb.config(font=("Aharoni", 15))

S3Lb = Label(root, text="Symptom 3", fg="white", bg="black")

S3Lb.grid(row=9, column=0, pady=20, sticky=W)

S3Lb.config(font=("Aharoni", 15))

S4Lb = Label(root, text="Symptom 4", fg="white", bg="black")

S4Lb.grid(row=10, column=0, pady=20, sticky=W)

S4Lb.config(font=("Aharoni", 15))

S5Lb = Label(root, text="Symptom 5", fg="white", bg="black")

S5Lb.grid(row=11, column=0, pady=20, sticky=W)

S5Lb.config(font=("Aharoni", 15))

lrLb = Label(root, text="DecisionTree", fg="white", bg="red")

lrLb.grid(row=15, column=0, pady=20,sticky=W)

lrLb.config(font=("Aharoni", 15))

destreeLb = Label(root, text="RandomForest", fg="white", bg="red")

destreeLb.grid(row=17, column=0, pady=20, sticky=W)

destreeLb.config(font=("Aharoni", 15))

ranfLb = Label(root, text="NaiveBayes", fg="white", bg="red")

ranfLb.grid(row=19, column=0, pady=20, sticky=W)

ranfLb.config(font=("Aharoni", 15))

# entries

OPTIONS = sorted(l1)

NameEn = Entry(root, textvariable=Name,width=20)

NameEn.grid(row=6, column=1)

S1En = OptionMenu(root, Symptom1,\*OPTIONS)

S1En.grid(row=7, column=1)

S2En = OptionMenu(root, Symptom2,\*OPTIONS)

S2En.grid(row=8, column=1)

S3En = OptionMenu(root, Symptom3,\*OPTIONS)

S3En.grid(row=9, column=1)

S4En = OptionMenu(root, Symptom4,\*OPTIONS)

S4En.grid(row=10, column=1)

S5En = OptionMenu(root, Symptom5,\*OPTIONS)

S5En.grid(row=11, column=1)

dst = Button(root, text="DecisionTree", command=DecisionTree,bg="blue",fg="white")

dst.grid(row=8, column=3,padx=10)

rnf = Button(root, text="Randomforest", command=randomforest,bg="blue",fg="white")

rnf.grid(row=9, column=3,padx=10)

lr = Button(root, text="NaiveBayes", command=NaiveBayes,bg="blue",fg="white")

lr.grid(row=10, column=3,padx=10)

#textfileds

t1 = Text(root, height=1, width=40,bg="orange",fg="white")

t1.grid(row=15, column=1, padx=10)

t2 = Text(root, height=1, width=40,bg="orange",fg="white")

t2.grid(row=17, column=1 , padx=10)

t3 = Text(root, height=1, width=40,bg="orange",fg="white")

t3.grid(row=19, column=1 , padx=10)

root.mainloop()

**Libraries Used:**

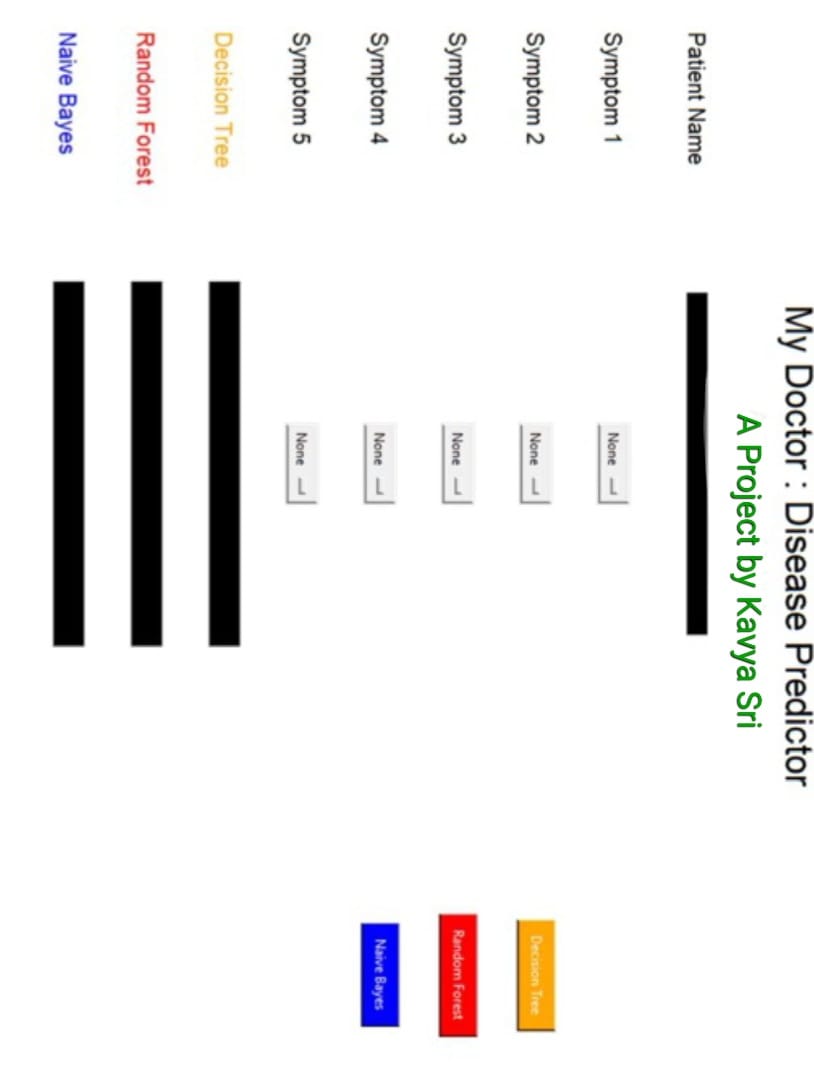
1. **Tkinter:** For building the graphical user interface (GUI) to interact with the user.
2. **Numpy and Pandas:** For data manipulation and analysis.
3. **Scikit-learn:** For implementing ML algorithms such as Decision Trees, Random Forests, and Naive Bayes classifiers.
4. **Python 3.x**

**Code Explanation:**

The provided code is written in Python using the Tkinter library for creating a graphical user interface (GUI). It utilizes machine learning models from the scikit-learn library to predict diseases based on symptoms input by the user. Here's a brief explanation of the code's functionality:

1. Data Preprocessing: The code reads the training and testing datasets ("Training.csv" and "Testing.csv") containing symptom data and corresponding disease labels. It preprocesses the data, mapping disease labels to numeric values for model training.
2. Model Training: Three machine learning models are trained: Decision Tree, Random Forest, and Naive Bayes classifiers.
3. GUI Implementation: Tkinter is used to create a GUI where users can input symptoms, and the trained models predict the most likely disease based on those symptoms.

**Output:**



**Conclusion:**

In this project, we demonstrated the application of machine learning techniques for disease prediction based on symptoms. The developed system provides an interactive interface for users to input symptoms and obtain predictions using trained ML models. By leveraging ML algorithms, healthcare professionals can potentially improve disease diagnosis and treatment planning.