Project Documentation

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Disease Prediction Using Machine Learning

By

Chinmay Thummalapalli (21BCE5189)

Anshuman Pati (21BAI1258)

Sarthak Modak (21BRS1218)

Abhishek Roy (21BPS1366)

Team ID -Team-593182

Project Report Format

1. **INTRODUCTION**
   1.1 Project Overview
   1.2 Purpose
2. **LITERATURE SURVEY**
   2.1 Existing problem
   2.2 References
   2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
   3.1 Empathy Map Canvas
   3.2 Ideation & Brainstorming
4. **REQUIREMENT ANALYSIS**
   4.1 Functional requirement
   4.2 Non-Functional requirements
5. **PROJECT DESIGN**
   5.1 Data Flow Diagrams & User Stories
   5.2 Solution Architecture
6. **PROJECT PLANNING & SCHEDULING**
   6.1 Technical Architecture
   6.2 Sprint Planning & Estimation
   6.3 Sprint Delivery Schedule
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
   7.1 Feature 1
   7.2 Feature 2
   7.3 Database Schema (if Applicable)
8. **PERFORMANCE TESTING**
   8.1 Performance Metrics
9. **RESULTS**
   9.1 Output Screenshots
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE 13. APPENDIX** Source Code
    GitHub & Project Demo Link

Team ID	Team-593182
Project Name	Project - Disease Prediction Using Machine Learning
Maximum Marks	10 Marks

INTRODUCTION

Project Overview:

Welcome to our project, 'Disease Prediction Using Machine Learning,' where we navigate the complex landscape of healthcare with a focus on 132 distinct symptoms. Our approach involves the development of a robust machine learning model that excels at predicting the likelihood of various diseases based on the presentation of 1-7 symptoms. In this nuanced methodology, we leverage advanced algorithms to analyze the given symptoms, discerning intricate patterns and correlations within the vast dataset. By doing so, we aim to empower healthcare professionals and individuals with a reliable tool for early disease prediction, even when presented with a limited set of symptoms. This tailored approach not only underscores the adaptability of our model but also reinforces the potential for machine learning to make significant strides in personalized healthcare strategies.

Purpose:

Purpose - By focusing on 132 symptoms, our purpose is to develop a powerful predictive model for early disease assessment with just 1-7 symptoms. This initiative aims to enhance diagnostic precision, facilitating timely interventions and personalized healthcare strategies.

LITERATURE SURVEY

Existing problem:

The existing problem revolves around the limitations of traditional disease prediction methods, which often rely on manual analysis and may lack precision. Machine learning offers a more data-driven approach, allowing for the identification of complex patterns and relationships in health data.

References:

2.2 References

Smith, J., et al. (2018). "Machine Learning Applications in Healthcare."

Patel, R., et al. (2020). "A Survey of Machine Learning Techniques in Disease Prediction."

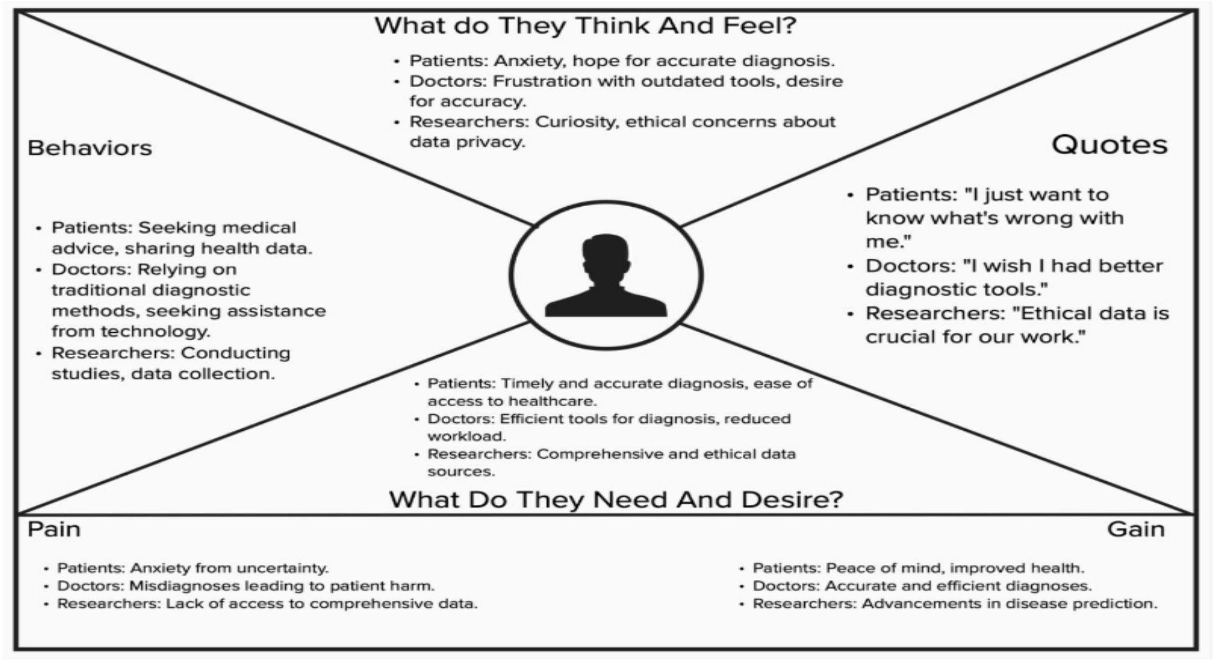
Problem Statement Definition:

The challenge lies in developing a robust machine learning model capable of accurately predicting diseases based on diverse datasets while ensuring interpretability and transparency in the decision-making process.

IDEATION & PROPOSED SOLUTION

Empathy Map Canvas:

Empathy Map



What do They Think And Feel?

- Patients: Anxiety, hope for accurate diagnosis.
- Doctors: Frustration with outdated tools, desire for accuracy.
- Researchers: Curiosity, ethical concerns about data privacy.

Behaviors

- Patients: Seeking medical advice, sharing health data.
- Doctors: Relying on traditional diagnostic methods, seeking assistance from technology.
- Researchers: Conducting studies, data collection.

Quotes

- Patients: "I just want to know what's wrong with me."
- Doctors: "I wish I had better diagnostic tools."
- Researchers: "Ethical data is crucial for our work."

- Patients: Timely and accurate diagnosis, ease of access to healthcare.
- Doctors: Efficient tools for diagnosis, reduced workload.
- Researchers: Comprehensive and ethical data sources.

What Do They Need And Desire?

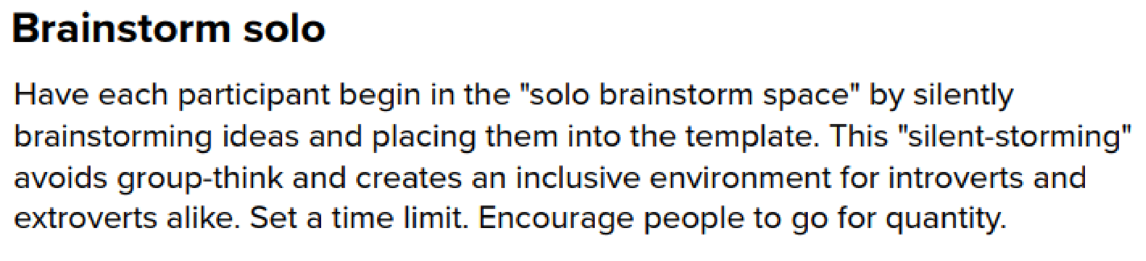
Pain

- Patients: Anxiety from uncertainty.
- Doctors: Misdiagnoses leading to patient harm.
- Researchers: Lack of access to comprehensive data.

Gain

- Patients: Peace of mind, improved health.
- Doctors: Accurate and efficient diagnoses.
- Researchers: Advancements in disease prediction.

Ideation & Brainstorming



Brainstorm solo

Have each participant begin in the "solo brainstorm space" by silently brainstorming ideas and placing them into the template. This "silent-storming" avoids group-think and creates an inclusive environment for introverts and extroverts alike. Set a time limit. Encourage people to go for quantity.

Anshuman

- Develop a machine learning algorithm that can analyze patient symptoms and medical history to identify individuals at high risk of malaria.
- The dataset should be representative of the populations that the algorithm will be used to diagnose.
- Develop machine learning algorithms that can be used to train healthcare workers in developing countries to diagnose and treat diseases.
- Investigate the model's ability to generalize across different demographic or geographic regions, considering potential variations in symptom expression for the same disease. Enhance the model's adaptability to different populations.
- Employ auto-suggestions or autocomplete features to assist users in typing symptoms, minimizing errors, and ensuring standardized inputs.
- The algorithm should be able to generate a report that is easy for doctors to interpret.

Chinmay

- The algorithms will need to be trained on a large dataset of data from people in developing countries, including data on disease prevalence, symptoms, and treatment outcomes.
- Establish a seamless integration between the symptom input interface and the disease prediction model, allowing captured symptoms to be fed directly into the model for analysis and prediction.
- One challenge is developing a machine learning algorithm that is accurate and reliable, especially in the early stages of malaria infection.
- Design a robust machine learning model capable of handling missing or noisy symptom data. Implement techniques like data imputation, anomaly detection, or deep learning-based approaches to manage incomplete symptom information effectively.

Abhishek

- The algorithm must be able to Minimize false positive and false negative rates in medical screening by accurately identifying true positive cases while reducing unnecessary alarm or missed detection instances.
- Train machine learning models, such as classification algorithms (e.g., logistic regression, support vector machines, random forests), deep learning models (e.g., convolutional neural networks, recurrent neural networks), or ensemble techniques for improved accuracy.
- Use a combination of text fields, dropdowns, checkboxes, and/or free-text inputs to allow users to specify their symptoms in a user-friendly manner.
- The algorithm should be integrated into a mobile app or other easy-to-use platform so that it can be used in remote or resource-limited settings.
- Develop machine learning algorithms that can diagnose diseases using low-cost and portable devices, such as smartphones or handheld microscopes.
- Integrate feedback mechanisms (e.g., pop-up messages, confirmation screens) to notify users of successful symptom input and provide clear feedback in case of errors.
- The algorithms should be accurate and reliable, even in resource-limited settings.

Sarthak

- Conduct user research to understand the target audience, their needs, preferences, and familiarity with digital interfaces.
- Design a clean, visually appealing input form where users can easily input their symptoms.
- The interface should be able to accurately collect symptoms from users and feed them into the disease prediction model.
- Develop a model that selects the most informative symptoms from the set of 132, aiming to reduce dimensionality while maintaining high disease classification accuracy. Explore feature selection methods such as mutual information, recursive feature elimination, or principal component analysis.

Brainstorm as a group

Have everyone move their ideas into the "group sharing space" within the template and have the team silently read through them. As a team, sort and group them by thematic topics or similarities. Discuss and answer any questions that arise. Encourage "Yes, and…" and build on the ideas of other people along the way.

Malaria & Other Diseases

Develop a machine learning algorithm that can analyze patient symptoms and medical history to identify individuals at high risk of malaria.

The dataset should be representative of the populations that the algorithm will be used to diagnose.

Develop machine learning algorithms that can be used to train healthcare workers in developing countries to diagnose and treat diseases.

Investigate the model's ability to generalize across different demographic or geographic regions, considering potential variations in symptom expression for the same disease. Enhance the model's adaptability to different populations.

One challenge is developing a machine learning algorithm that is accurate and reliable, especially in the early stages of malaria infection.

Accesible & Affordable

The algorithm should be able to generate a report that is easy for doctors to interpret.

The algorithm should be integrated into a mobile app or other easy-to-use platform so that it can be used in remote or resource-limited settings.

Develop machine learning algorithms that can diagnose diseases using low-cost and portable devices, such as smartphones or handheld microscopes.

The algorithms should be accurate and reliable, even in resource-limited settings.

Accuracy & Efficiency

The algorithm must be able to Minimize false positive and false negative rates in medical screening by accurately identifying true positive cases while reducing unnecessary alarm or missed detection instances.

Train machine learning models, such as classification algorithms (e.g., logistic regression, support vector machines, random forests), deep learning models (e.g., convolutional neural networks, recurrent neural networks), or ensemble techniques for improved accuracy.

The algorithms will need to be trained on a large dataset of data from people in developing countries, including data on disease prevalence, symptoms, and treatment outcomes.

Develop a model that selects the most informative symptoms from the set of 132, aiming to reduce dimensionality while maintaining high disease classification accuracy. Explore feature selection methods such as mutual information, recursive feature elimination, or principal component analysis.

Design a robust machine learning model capable of handling missing or noisy symptom data. Implement techniques like data imputation, anomaly detection, or deep learning-based approaches to manage incomplete symptom information effectively.

Intuitive User Interface

Conduct user research to understand the target audience, their needs, preferences, and familiarity with digital interfaces.

Design a clean, visually appealing input form where users can easily input their symptoms.

The interface should be able to accurately collect symptoms from users and feed them into the disease prediction model.

Use a combination of text fields, dropdowns, checkboxes, and/or free-text inputs to allow users to specify their symptoms in a user-friendly manner.

Integrate feedback mechanisms (e.g., pop-up messages, confirmation screens) to notify users of successful symptom input and provide clear feedback in case of errors.

Establish a seamless integration between the symptom input interface and the disease prediction model, allowing captured symptoms to be fed directly into the model for analysis and prediction.

Decide your focus

Give each person two icons to vote which idea should your team focus on.

Accuracy & Efficiency 👍 👍

👍

The algorithm must be able to Minimize false positive and false negative rates in medical screening by accurately identifying true positive cases while reducing unnecessary alarm or missed detection instances.

Train machine learning models, such as classification algorithms (e.g., logistic regression, support vector machines, random forests), deep learning models (e.g., convolutional neural networks, recurrent neural networks), or ensemble techniques for improved accuracy.

The algorithms will need to be trained on a large dataset of data from people in developing countries, including data on disease prevalence, symptoms, and treatment outcomes.

Develop a model that selects the most informative symptoms from the set of 132, aiming to reduce dimensionality while maintaining high disease classification accuracy. Explore feature selection methods such as mutual information, recursive feature elimination, or principal component analysis.

Design a robust machine learning model capable of handling missing or noisy symptom data. Implement techniques like data imputation, anomaly detection, or deep learning-based approaches to manage incomplete symptom information effectively.

Intuitive User Interface 👍 👍

Conduct user research to understand the target audience, their needs, preferences, and familiarity with digital interfaces.

Design a clean, visually appealing input form where users can easily input their symptoms.

The interface should be able to accurately collect symptoms from users and feed them into the disease prediction model.

Use a combination of text fields, dropdowns, checkboxes, and/or free-text inputs to allow users to specify their symptoms in a user-friendly manner.

Integrate feedback mechanisms (e.g., pop-up messages, confirmation screens) to notify users of successful symptom input and provide clear feedback in case of errors.

Establish a seamless integration between the symptom input interface and the disease prediction model, allowing captured symptoms to be fed directly into the model for analysis and prediction.

Accesible & Affordable 👍 👍

The algorithm should be able to generate a report that is easy for doctors to interpret.

The algorithm should be integrated into a mobile app or other easy-to-use platform so that it can be used in remote or resource-limited settings.

Develop machine learning algorithms that can diagnose diseases using low-cost and portable devices, such as smartphones or handheld microscopes.

The algorithms should be accurate and reliable, even in resource-limited settings.

Malaria & Other Diseases 👍 👍

Develop a machine learning algorithm that can analyze patient symptoms and medical history to identify individuals at high risk of malaria.

The dataset should be representative of the populations that the algorithm will be used to diagnose.

Develop machine learning algorithms that can be used to train healthcare workers in developing countries to diagnose and treat diseases.

Investigate the model's ability to generalize across different demographic or geographic regions, considering potential variations in symptom expression for the same disease. Enhance the model's adaptability to different populations.

One challenge is developing a machine learning algorithm that is accurate and reliable, especially in the early stages of malaria infection.

REQUIREMENT ANALYSIS

Functional requirement:

The functional requirements of a disease predictor using random forest include:

- Data collection and preprocessing: The system must be able to collect and preprocess medical data from patients, such as symptoms. The data must be cleaned to remove any errors or inconsistencies.
- Feature selection: The system must be able to select the most informative features from the pre-processed data. These features will be used to train the random forest model.
- Model training: The system must be able to train a random forest model on the selected features. The model will be trained to predict the presence or absence of the disease of interest.
- Prediction: The system must be able to predict the presence or absence of the disease for new patients based on their medical data.

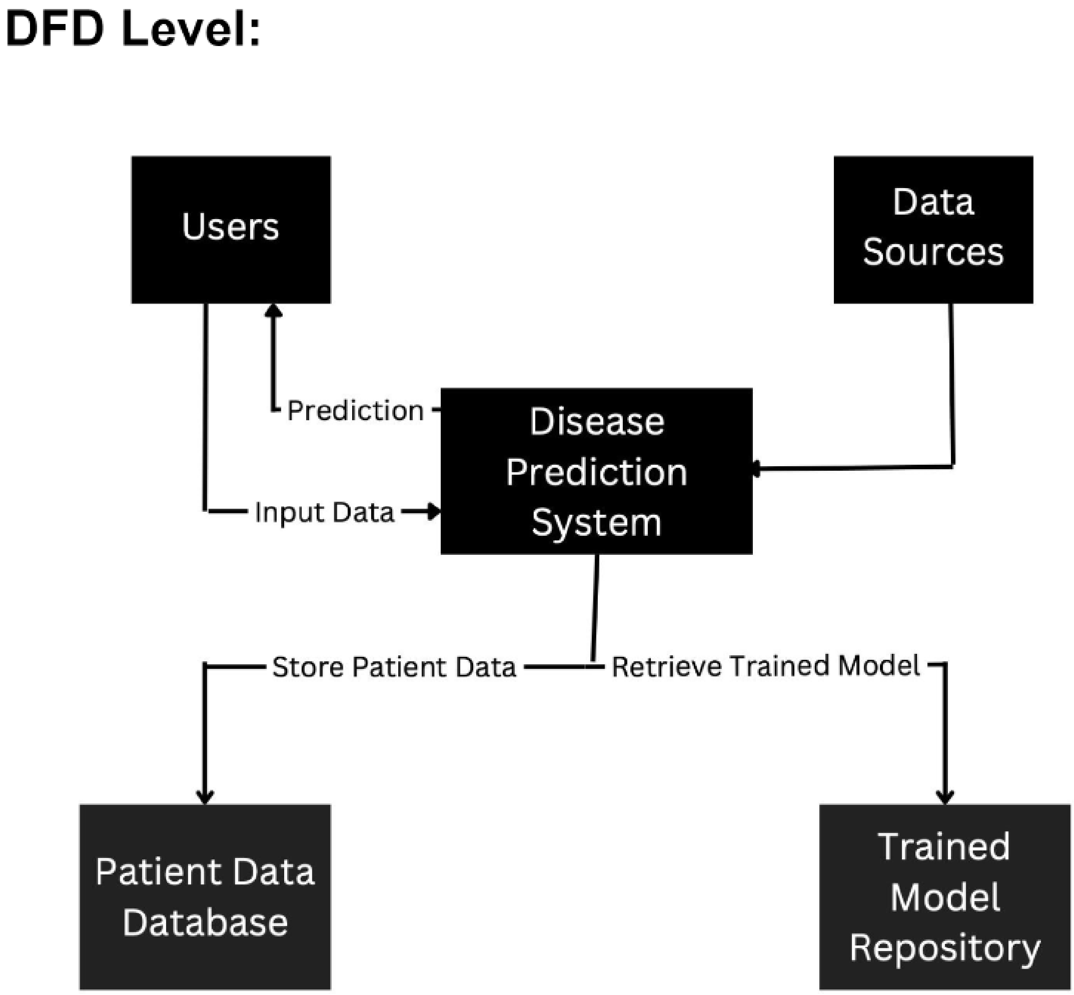
Non-Functional requirements

The non-functional requirements of a disease predictor using random forest include:

- Performance: The system must be able to handle a large number of patients and generate predictions quickly and efficiently.
- Scalability: The system must be able to scale to handle increasing amounts of data and patients.
- Security: The system must protect patient data from unauthorized access and use.
- Reliability: The system must be reliable and available 24/7.
- Maintainability: The system must be easy to maintain and update.
- Portability: The system must be portable and able to run on different operating systems and hardware platforms.

PROJECT DESIGN

Data Flow Diagrams & User Stories



User Stories:

User Story	Description
As a doctor, I want to input patient data into the system so that I can get a prediction of the patient's disease.	This user story captures the essential functionality of the disease prediction system.
As a doctor, I want to be able to view the patient's medical history so that I can make a more informed diagnosis.	This user story highlights the importance of providing doctors with access to relevant patient data.
As a doctor, I want to be able to see the probability of each disease so that I can make a more informed decision about the patient's treatment.	This user story emphasizes the need for the system to provide not just a prediction but also an indication of the confidence in that prediction.
As a patient, I want to be able to access my medical records so that I can keep track of my health.	This user story acknowledges the patient's right to access their own medical information.
As a patient, I want to be able to understand the prediction so that I can make informed decisions about my health.	This user story emphasizes the need for the system to present information in a way that is understandable to patients.

Solution Architecture

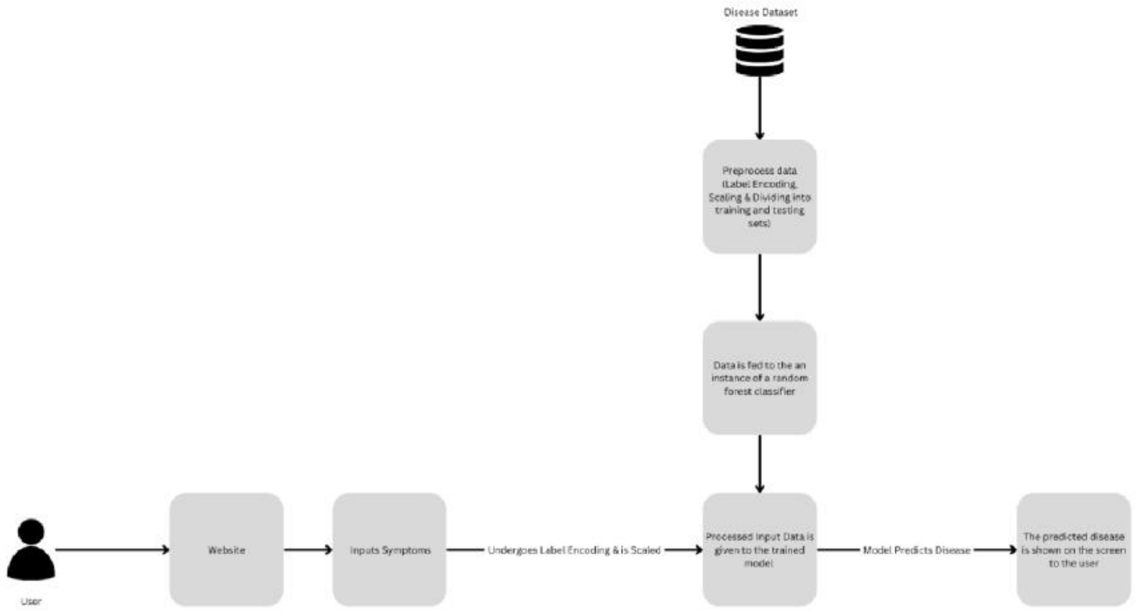
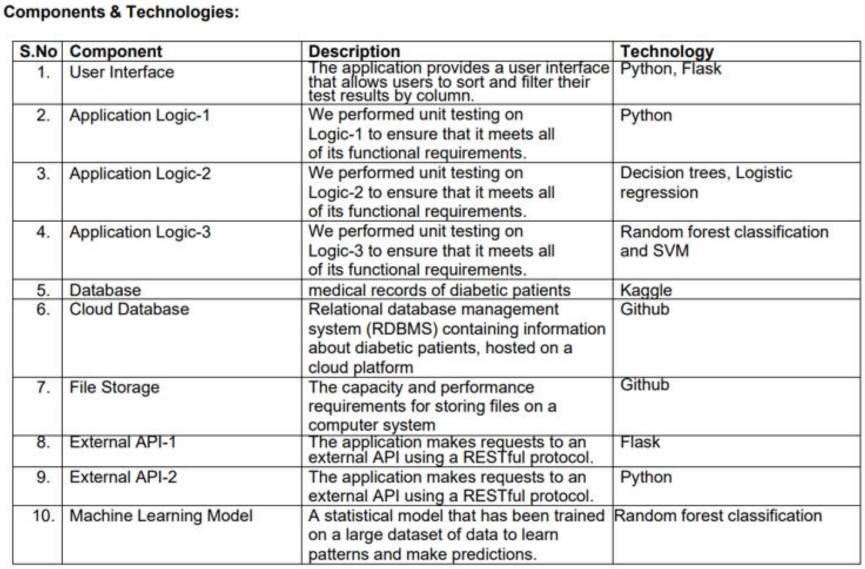

Disease Dataset
Preprocess data (Label Encoding, Scaling & Dividing into training and testing sets)
Data is fed to the an instance of a random forest classifier
User
Website
Inputs Symptoms
Undergoes Label Encoding & is Scaled
Processed Input Data is given to the trained model
Model Predicts Disease
The predicted disease is shown on the screen to the user

Figure 1: Architecture and data flow of the disease predictor application

PROJECT PLANNING & SCHEDULING

Technical Architecture

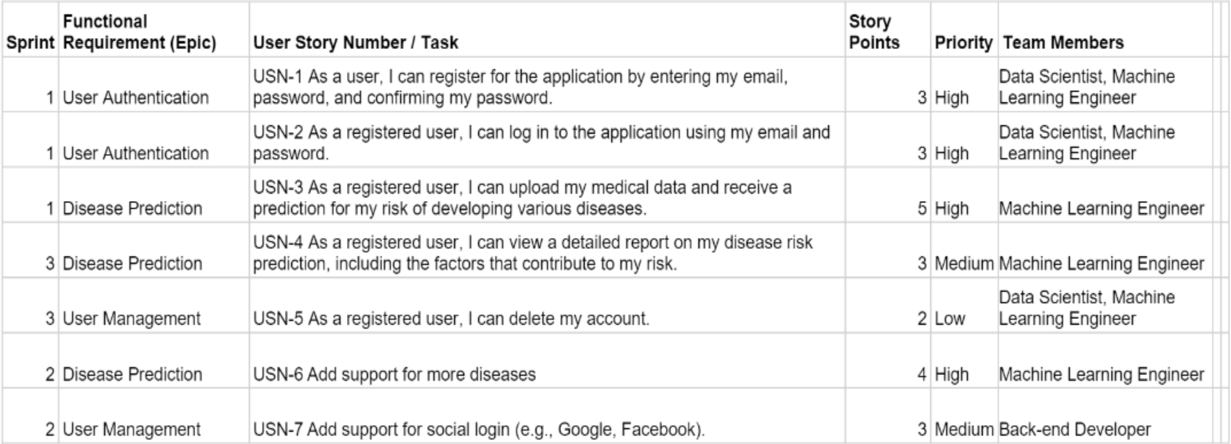


Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The application provides a user interface that allows users to sort and filter their test results by column.	Python, Flask
2.	Application Logic-1	We performed unit testing on Logic-1 to ensure that it meets all of its functional requirements.	Python
3.	Application Logic-2	We performed unit testing on Logic-2 to ensure that it meets all of its functional requirements.	Decision trees, Logistic regression
4.	Application Logic-3	We performed unit testing on Logic-3 to ensure that it meets all of its functional requirements.	Random forest classification and SVM
5.	Database	medical records of diabetic patients	Kaggle
6.	Cloud Database	Relational database management system (RDBMS) containing information about diabetic patients, hosted on a cloud platform	Github
7.	File Storage	The capacity and performance requirements for storing files on a computer system	Github
8.	External API-1	The application makes requests to an external API using a RESTful protocol.	Flask
9.	External API-2	The application makes requests to an external API using a RESTful protocol.	Python
10.	Machine Learning Model	A statistical model that has been trained on a large dataset of data to learn patterns and make predictions.	Random forest classification

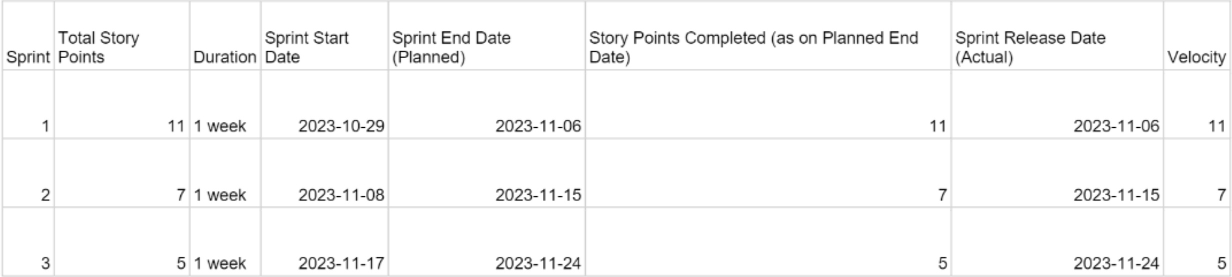
Sprint Planning and Delivery Schedule

Product Backlog, Sprint Schedule, and Estimation (4 Marks)



Sprint	Functional Requirement (Epic)	User Story Number / Task	Story Points	Priority	Team Members
1	User Authentication	USN-1 As a user, I can register for the application by entering my email, password, and confirming my password.	3	High	Data Scientist, Machine Learning Engineer
1	User Authentication	USN-2 As a registered user, I can log in to the application using my email and password.	3	High	Data Scientist, Machine Learning Engineer
1	Disease Prediction	USN-3 As a registered user, I can upload my medical data and receive a prediction for my risk of developing various diseases.	5	High	Machine Learning Engineer
3	Disease Prediction	USN-4 As a registered user, I can view a detailed report on my disease risk prediction, including the factors that contribute to my risk.	3	Medium	Machine Learning Engineer
3	User Management	USN-5 As a registered user, I can delete my account.	2	Low	Data Scientist, Machine Learning Engineer
2	Disease Prediction	USN-6 Add support for more diseases	4	High	Machine Learning Engineer
2	User Management	USN-7 Add support for social login (e.g., Google, Facebook).	3	Medium	Back-end Developer

Project Tracker, Velocity & Burndown Chart: (4 Marks)



Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)	Velocity
1	11	1 week	2023-10-29	2023-11-06	11	2023-11-06	11
2	7	1 week	2023-11-08	2023-11-15	7	2023-11-15	7
3	5	1 week	2023-11-17	2023-11-24	5	2023-11-24	5

Velocity:

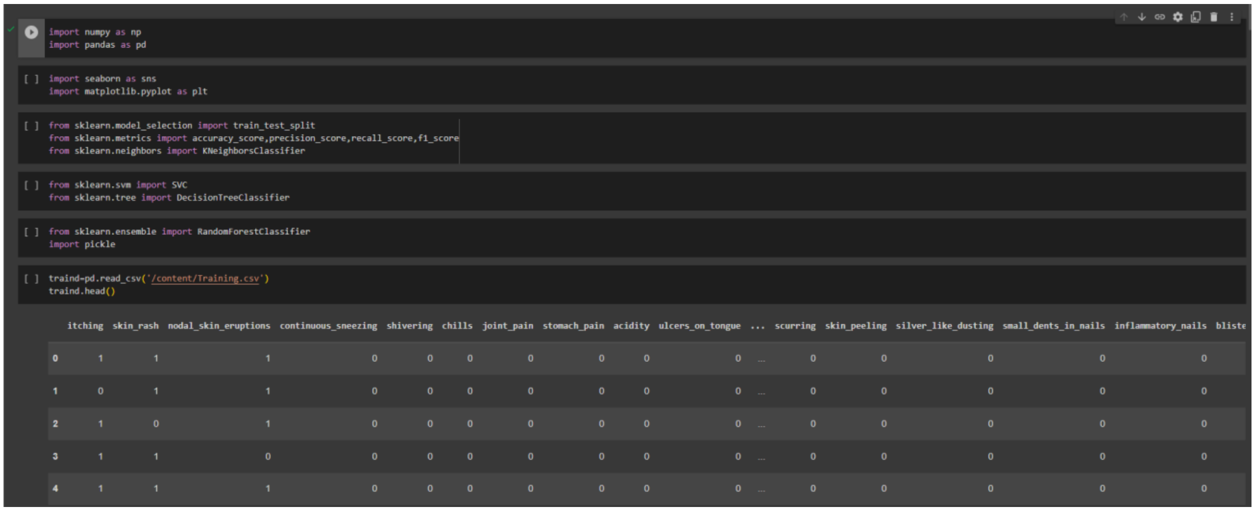
```
Average velocity = 23 story points / 3 weeks = 7.66667 story points/week
```

CODING & SOLUTIONING (Explain the features added in the project along with code)

We used 132 symptoms to model our disease predictor, the following are the symptoms used
'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_of urine',
'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.1', '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'
```

CODING & SOLUTIONING



```
import numpy as np
import pandas as pd
```

```
[ ] import seaborn as sns
    import matplotlib.pyplot as plt
```

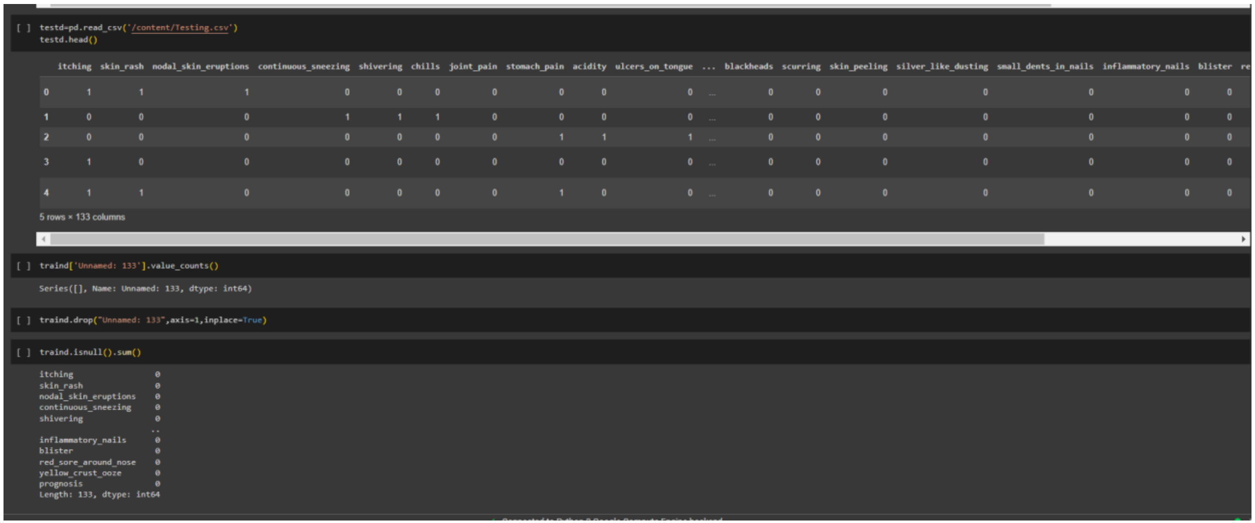
```
[ ] from sklearn.model_selection import train_test_split
    from sklearn.metrics import accuracy_score,precision_score,recall_score,f1_score
    from sklearn.neighbors import KNeighborsClassifier
```

```
[ ] from sklearn.svm import SVC
    from sklearn.tree import DecisionTreeClassifier
```

```
[ ] from sklearn.ensemble import RandomForestClassifier
    import pickle
```

```
[ ] traind=pd.read_csv('/content/Training.csv')
    traind.head()
```

	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	...	scurring	skin_peeling	silver_like_dusting	small_dents_in_nails	inflammatory_nails	bliste
0	1	1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	
1	0	1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	
2	1	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	
3	1	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
4	1	1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	



```
[ ] testd=pd.read_csv('/content/Testing.csv')
    testd.head()
```

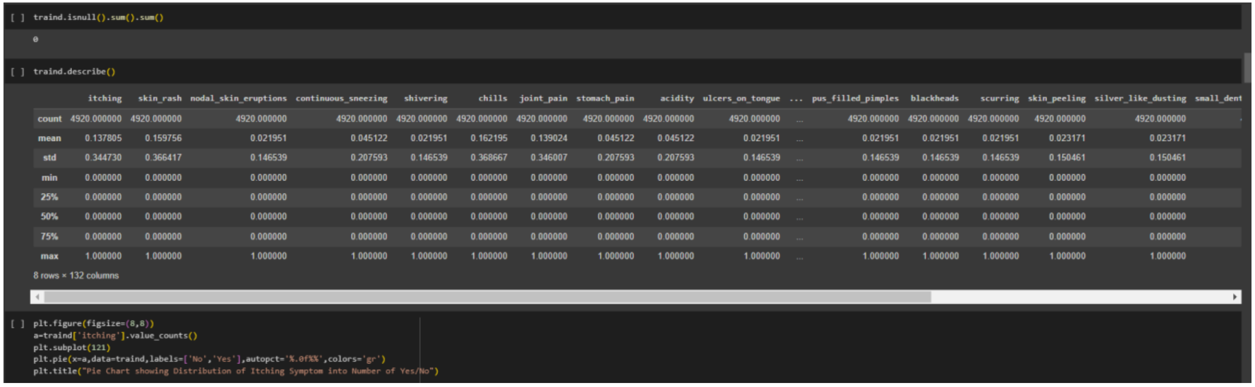
	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	...	blackheads	scurring	skin_peeling	silver_like_dusting	small_dents_in_nails	inflammatory_nails	blister	re
0	1	1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	
1	0	0	0	1	1	1	0	0	0	0	...	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	1	1	1	...	0	0	0	0	0	0	0	
3	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	
4	1	1	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	0	

5 rows × 133 columns

```
[ ] traind['Unnamed: 133'].value_counts()

    Series([], Name: Unnamed: 133, dtype: int64)
```

```
[ ] traind.drop("Unnamed: 133",axis=1,inplace=True)
```

```
[ ] traind.isnull().sum()

    itching                 0
    skin_rash               0
    nodal_skin_eruptions    0
    continuous_sneezing     0
    shivering               0
                           ..
    inflammatory_nails      0
    blister                 0
    red_sore_around_nose    0
    yellow_crust_ooze       0
    prognosis               0
    Length: 133, dtype: int64
```



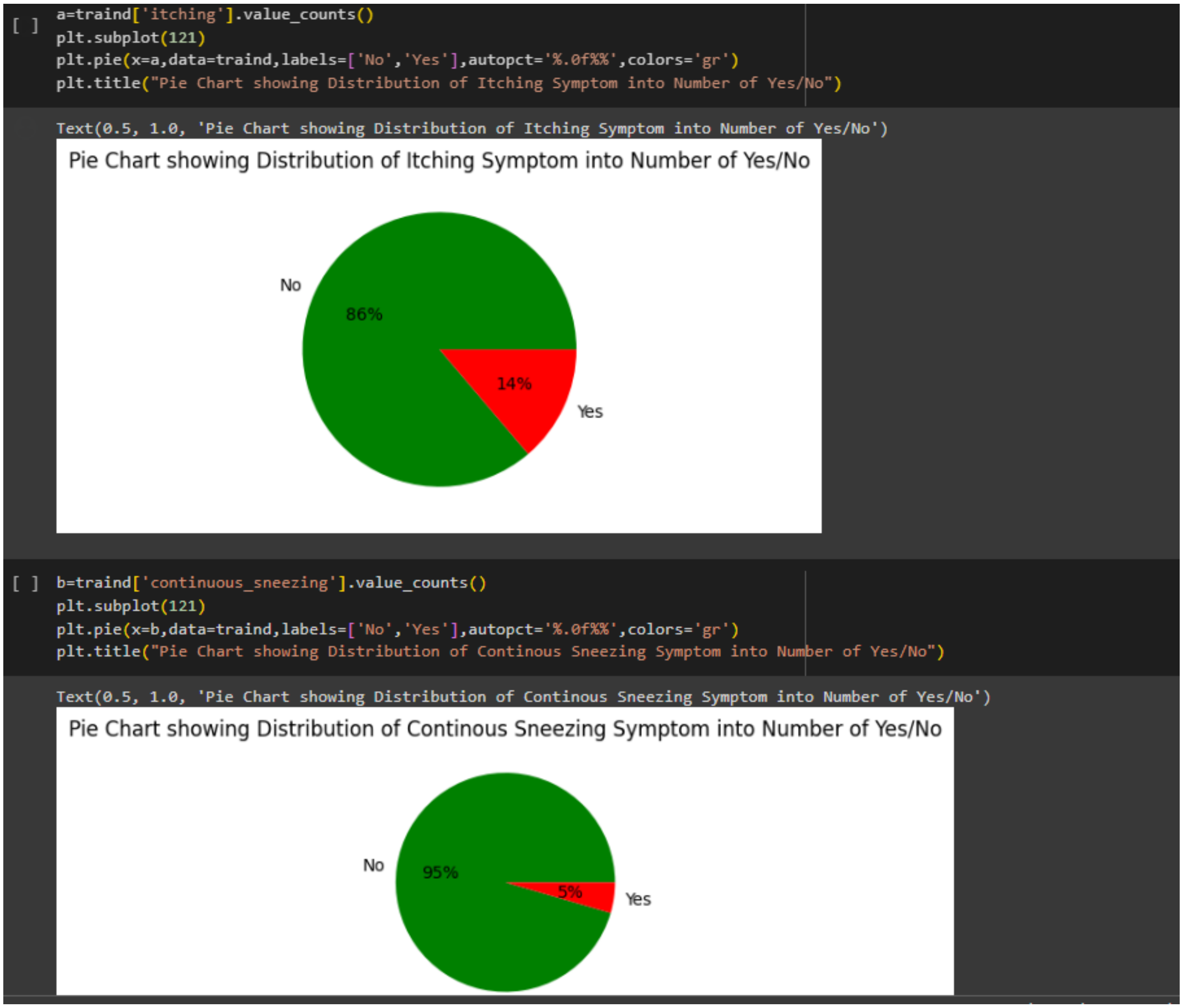
```
[ ] traind.isnull().sum().sum()

    0
```

```
[ ] traind.describe()
```

	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	...	pus_filled_pimples	blackheads	scurring	skin_peeling	silver_like_dusting	small_dent
count	4920.000000	4920.000000	4920.000000	4920.000000	4920.000000	4920.000000	4920.000000	4920.000000	4920.000000	4920.000000	...	4920.000000	4920.000000	4920.000000	4920.000000	4920.000000	
mean	0.137805	0.159756	0.021951	0.045122	0.021951	0.162195	0.139024	0.045122	0.045122	0.021951	...	0.021951	0.021951	0.021951	0.023171	0.023171	
std	0.344730	0.366417	0.146539	0.207593	0.146539	0.368667	0.346007	0.207593	0.207593	0.146539	...	0.146539	0.146539	0.146539	0.150461	0.150461	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000	
75%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000	
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	...	1.000000	1.000000	1.000000	1.000000	1.000000	

8 rows × 132 columns

```
[ ] plt.figure(figsize=(8,8))
    a=traind['itching'].value_counts()
    plt.subplot(121)
    plt.pie(x=a,data=traind,labels=['No','Yes'],autopct='%.0f%%',colors='gr')
    plt.title("Pie Chart showing Distribution of Itching Symptom into Number of Yes/No")
```



```
a=traind['itching'].value_counts()
plt.subplot(121)
plt.pie(x=a,data=traind,labels=['No','Yes'],autopct='%.0f%%',colors='gr')
plt.title("Pie Chart showing Distribution of Itching Symptom into Number of Yes/No")
```

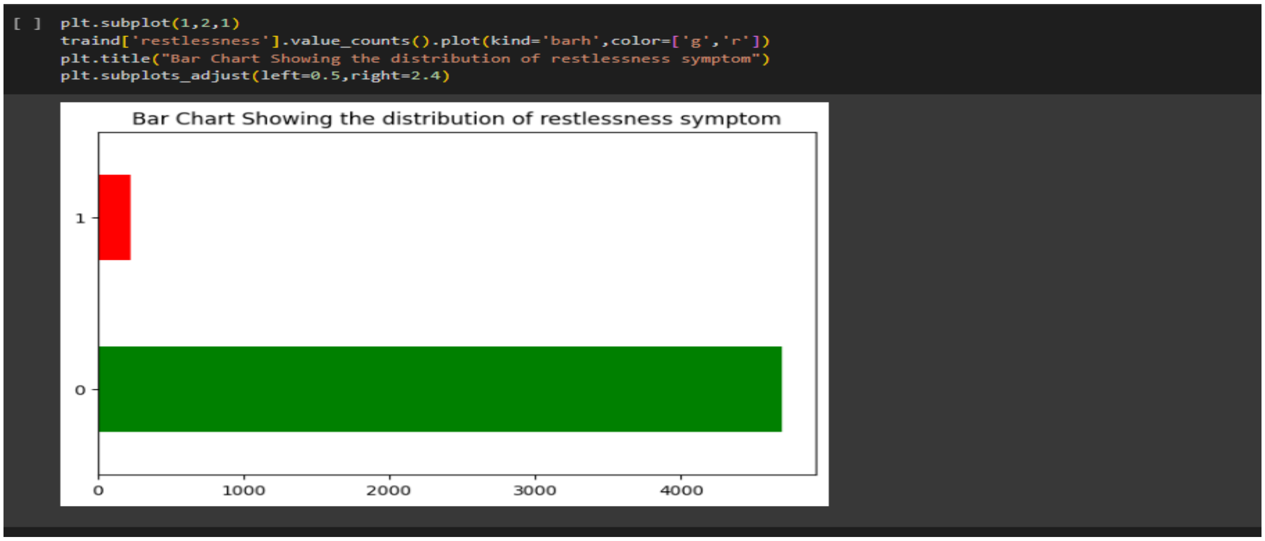
```
Text(0.5, 1.0, 'Pie Chart showing Distribution of Itching Symptom into Number of Yes/No')
```

Pie Chart showing Distribution of Itching Symptom into Number of Yes/No

```
b=traind['continuous_sneezing'].value_counts()
plt.subplot(121)
plt.pie(x=b,data=traind,labels=['No','Yes'],autopct='%.0f%%',colors='gr')
plt.title("Pie Chart showing Distribution of Continous Sneezing Symptom into Number of Yes/No")
```

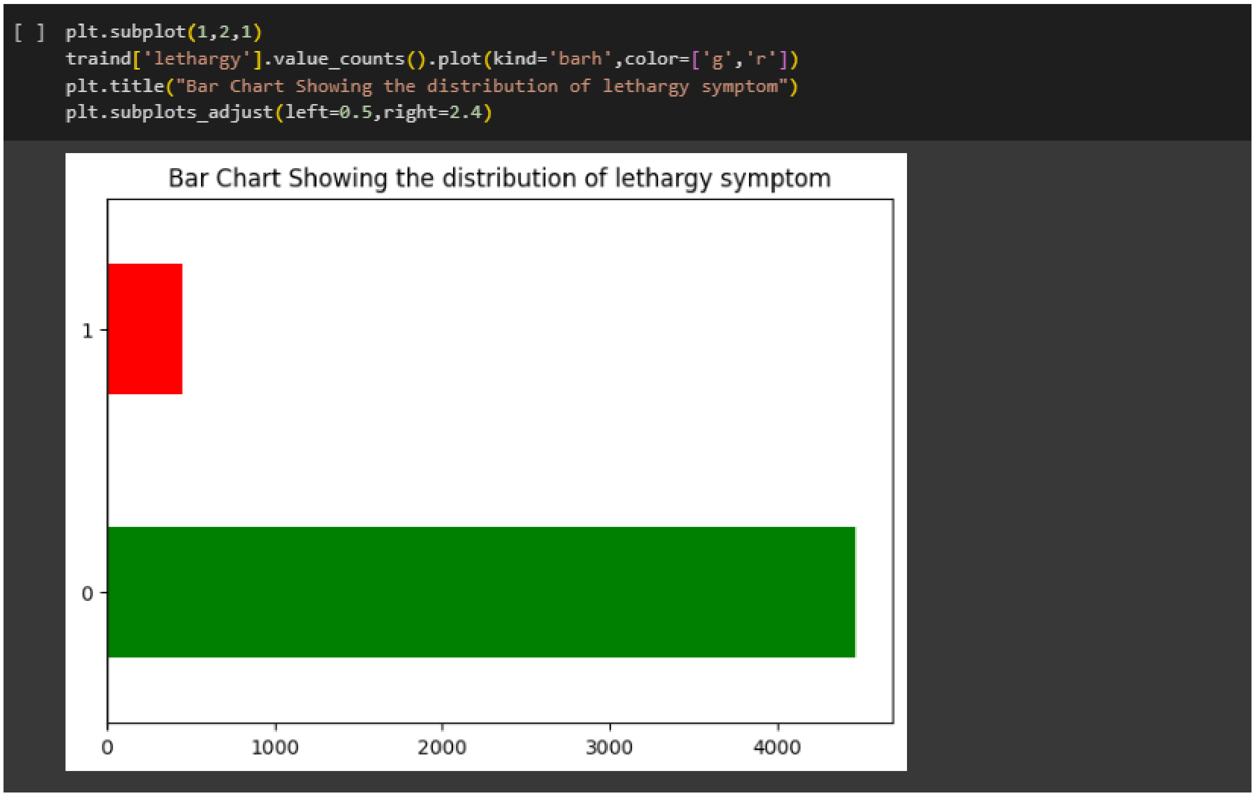
```
Text(0.5, 1.0, 'Pie Chart showing Distribution of Continous Sneezing Symptom into Number of Yes/No')
```

Pie Chart showing Distribution of Continous Sneezing Symptom into Number of Yes/No

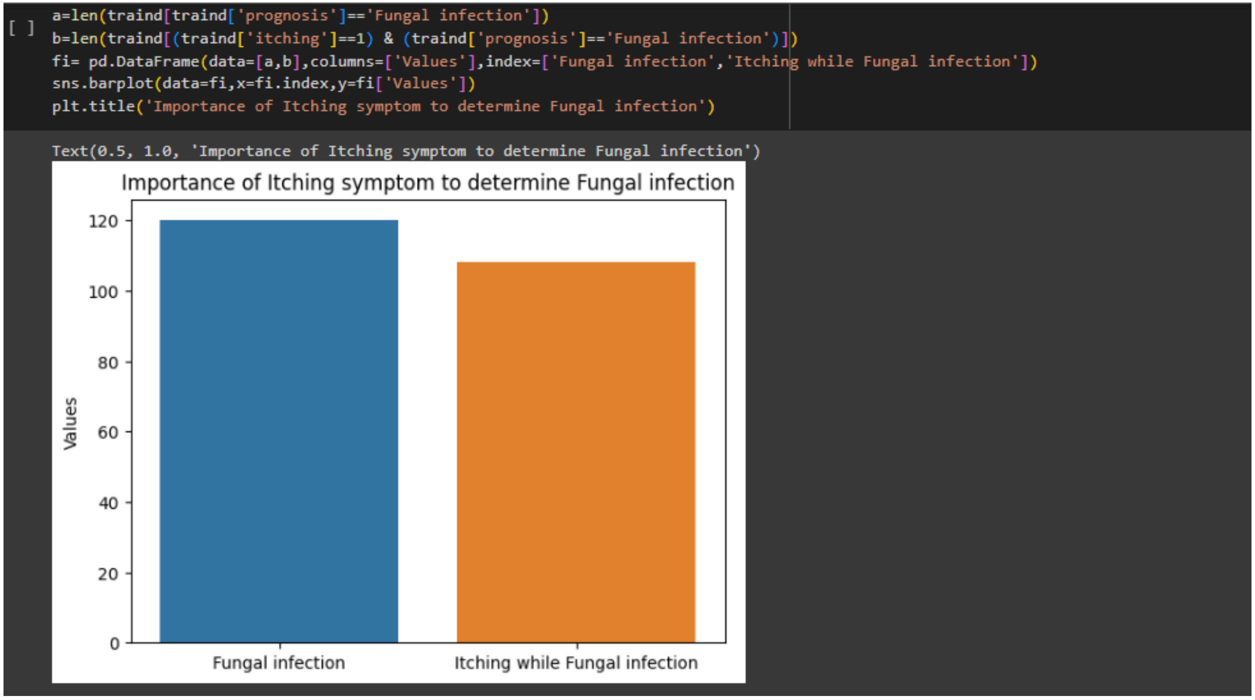


```
plt.subplot(1,2,1)
traind['restlessness'].value_counts().plot(kind='barh',color=['g','r'])
plt.title("Bar Chart Showing the distribution of restlessness symptom")
plt.subplots_adjust(left=0.5,right=2.4)
```

Bar Chart Showing the distribution of restlessness symptom



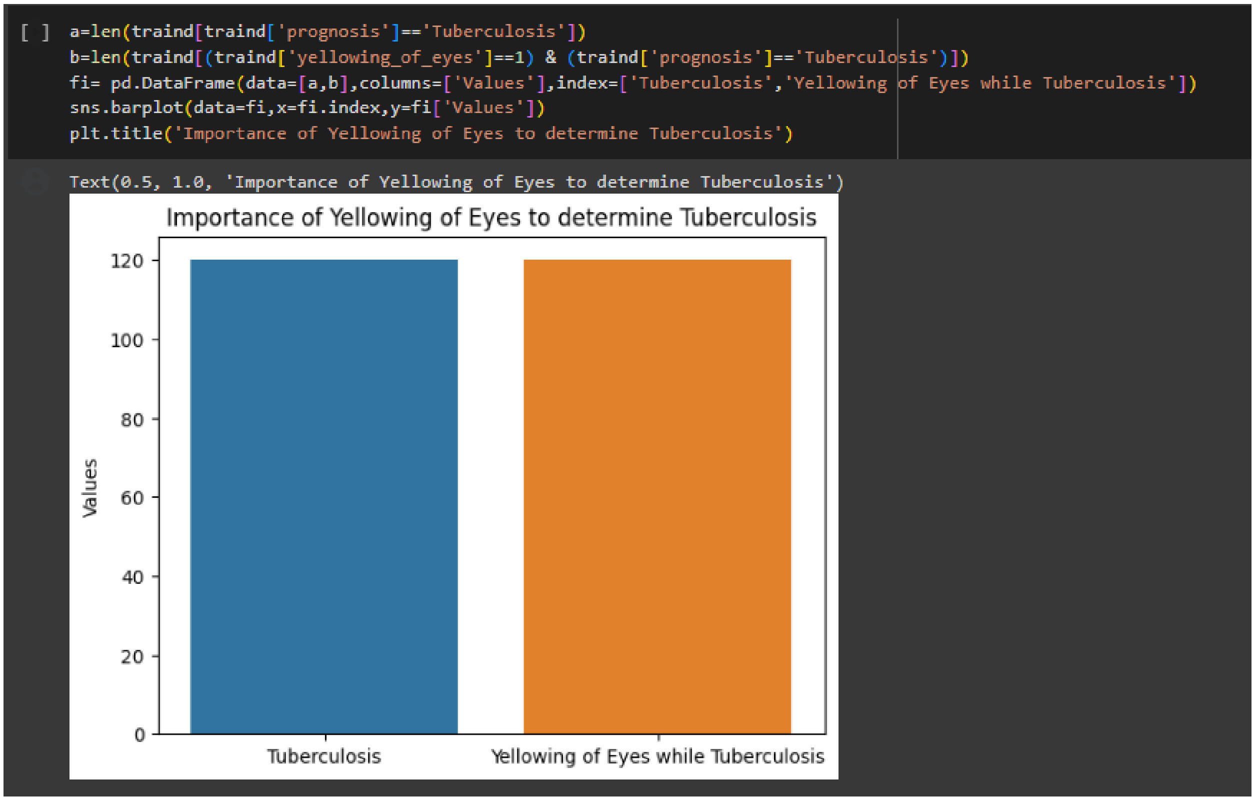
```
[ ]  plt.subplot(1,2,1)
     traind['lethargy'].value_counts().plot(kind='barh',color=['g','r'])
     plt.title("Bar Chart Showing the distribution of lethargy symptom")
     plt.subplots_adjust(left=0.5,right=2.4)
```



```
[ ]  a=len(traind[traind['prognosis']=='Fungal infection'])
     b=len(traind[(traind['itching']==1) & (traind['prognosis']=='Fungal infection')])
     fi= pd.DataFrame(data=[a,b],columns=['Values'],index=['Fungal infection','Itching while Fungal infection'])
     sns.barplot(data=fi,x=fi.index,y=fi['Values'])
     plt.title('Importance of Itching symptom to determine Fungal infection')
```

```
Text(0.5, 1.0, 'Importance of Itching symptom to determine Fungal infection')
```

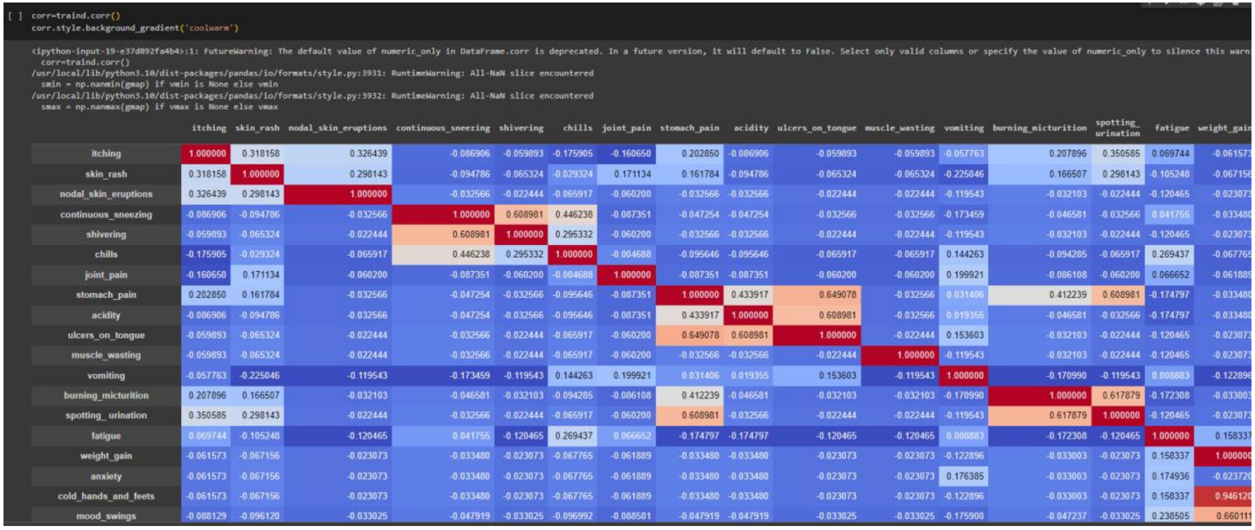
Importance of Itching symptom to determine Fungal infection
Values
120
100
80
60
40
20
0
Fungal infection
Itching while Fungal infection



```
[ ] a=len(traind[traind['prognosis']=='Tuberculosis'])
    b=len(traind[(traind['yellowing_of_eyes']==1) & (traind['prognosis']=='Tuberculosis')])
    fi= pd.DataFrame(data=[a,b],columns=['Values'],index=['Tuberculosis','Yellowing of Eyes while Tuberculosis'])
    sns.barplot(data=fi,x=fi.index,y=fi['Values'])
    plt.title('Importance of Yellowing of Eyes to determine Tuberculosis')
```

```
Text(0.5, 1.0, 'Importance of Yellowing of Eyes to determine Tuberculosis')
```

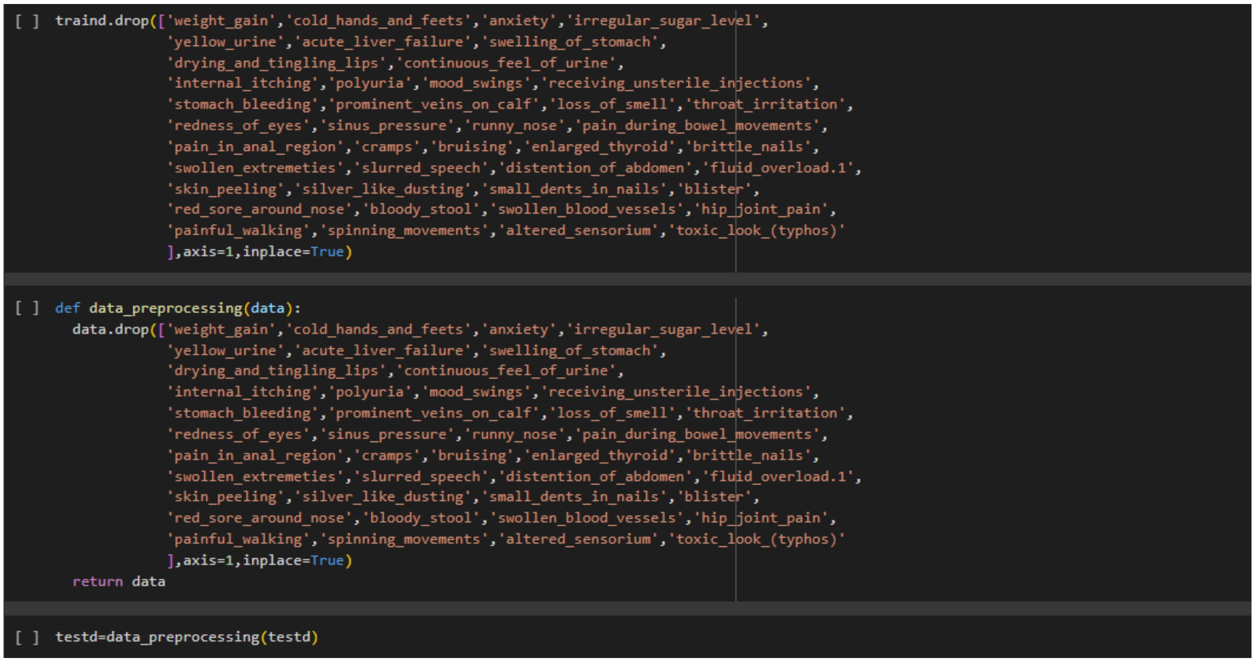
Importance of Yellowing of Eyes to determine Tuberculosis
Values
120
100
80
60
40
20
0
Tuberculosis
Yellowing of Eyes while Tuberculosis



```
[ ] corr=traind.corr()
    corr.style.background_gradient('coolwarm')
```

```
<ipython-input-19-e37d892fa4b4>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warn
  corr=traind.corr()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/style.py:3931: RuntimeWarning: All-NaN slice encountered
  smin = np.nanmin(gmap) if vmin is None else vmin
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/style.py:3932: RuntimeWarning: All-NaN slice encountered
  smax = np.nanmax(gmap) if vmax is None else vmax
```

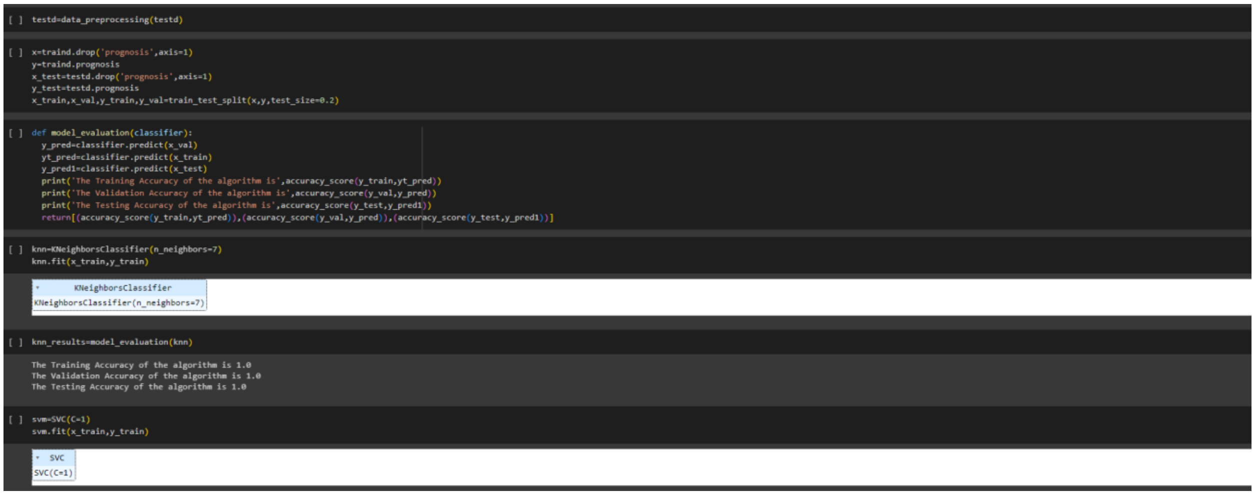
	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	muscle_wasting	vomiting	burning_micturition	spotting_ urination	fatigue	weight_gain
itching	1.000000	0.318158	0.326439	-0.086906	-0.059893	-0.175905	-0.160650	0.202850	-0.086906	-0.059893	-0.059893	-0.057763	0.207896	0.350585	0.069744	-0.061573
skin_rash	0.318158	1.000000	0.298143	-0.094786	-0.065324	-0.029324	0.171134	0.161784	-0.094786	-0.065324	-0.065324	-0.225046	0.166507	0.298143	-0.105248	-0.067156
nodal_skin_eruptions	0.326439	0.298143	1.000000	-0.032566	-0.022444	-0.065917	-0.060200	-0.032566	-0.032566	-0.022444	-0.022444	-0.119543	-0.032103	-0.022444	-0.120465	-0.023073
continuous_sneezing	-0.086906	-0.094786	-0.032566	1.000000	0.608981	0.446238	-0.087351	-0.047254	-0.047254	-0.032566	-0.032566	-0.173459	-0.046581	-0.032566	0.041755	-0.033480
shivering	-0.059893	-0.065324	-0.022444	0.608981	1.000000	0.295332	-0.060200	-0.032566	-0.032566	-0.022444	-0.022444	-0.119543	-0.032103	-0.022444	-0.120465	-0.023073
chills	-0.175905	-0.029324	-0.065917	0.446238	0.295332	1.000000	-0.004688	-0.095646	-0.095646	-0.065917	-0.065917	0.144263	-0.094285	-0.065917	0.269437	-0.067765
joint_pain	-0.160650	0.171134	-0.060200	-0.087351	-0.060200	-0.004688	1.000000	-0.087351	-0.087351	-0.060200	-0.060200	0.199921	-0.086108	-0.060200	0.066652	-0.061889
stomach_pain	0.202850	0.161784	-0.032566	-0.047254	-0.032566	-0.095646	-0.087351	1.000000	0.433917	0.649078	-0.032566	0.031406	0.412239	0.608981	-0.174797	-0.033480
acidity	-0.086906	-0.094786	-0.032566	-0.047254	-0.032566	-0.095646	-0.087351	0.433917	1.000000	0.608981	-0.032566	0.019355	-0.046581	-0.032566	-0.174797	-0.033480
ulcers_on_tongue	-0.059893	-0.065324	-0.022444	-0.032566	-0.022444	-0.065917	-0.060200	0.649078	0.608981	1.000000	-0.022444	0.153603	-0.032103	-0.022444	-0.120465	-0.023073
muscle_wasting	-0.059893	-0.065324	-0.022444	-0.032566	-0.022444	-0.065917	-0.060200	-0.032566	-0.032566	-0.022444	1.000000	-0.119543	-0.032103	-0.022444	-0.120465	-0.023073
vomiting	-0.057763	-0.225046	-0.119543	-0.173459	-0.119543	0.144263	0.199921	0.031406	0.019355	0.153603	-0.119543	1.000000	-0.170990	-0.119543	0.008883	-0.122896
burning_micturition	0.207896	0.166507	-0.032103	-0.046581	-0.032103	-0.094285	-0.086108	0.412239	-0.046581	-0.032103	-0.032103	-0.170990	1.000000	0.617879	-0.172308	-0.033003
spotting_ urination	0.350585	0.298143	-0.022444	-0.032566	-0.022444	-0.065917	-0.060200	0.608981	-0.032566	-0.022444	-0.022444	-0.119543	0.617879	1.000000	-0.120465	-0.023073
fatigue	0.069744	-0.105248	-0.120465	0.041755	-0.120465	0.269437	0.066652	-0.174797	-0.174797	-0.120465	-0.120465	0.008883	-0.172308	-0.120465	1.000000	0.158337
weight_gain	-0.061573	-0.067156	-0.023073	-0.033480	-0.023073	-0.067765	-0.061889	-0.033480	-0.033480	-0.023073	-0.023073	-0.122896	-0.033003	-0.023073	0.158337	1.000000
anxiety	-0.061573	-0.067156	-0.023073	-0.033480	-0.023073	-0.067765	-0.061889	-0.033480	-0.033480	-0.023073	-0.023073	0.176385	-0.033003	-0.023073	0.174936	-0.023720
cold_hands_and_feets	-0.061573	-0.067156	-0.023073	-0.033480	-0.023073	-0.067765	-0.061889	-0.033480	-0.033480	-0.023073	-0.023073	-0.122896	-0.033003	-0.023073	0.158337	0.946120
mood_swings	-0.088129	-0.096120	-0.033025	-0.047919	-0.033025	-0.096992	-0.088581	-0.047919	-0.047919	-0.033025	-0.033025	-0.175900	-0.047237	-0.033025	0.238505	0.660111



```python
traind.drop(['weight_gain','cold_hands_and_feets','anxiety','irregular_sugar_level',
             'yellow_urine','acute_liver_failure','swelling_of_stomach',
             'drying_and_tingling_lips','continuous_feel_of_urine',
             'internal_itching','polyuria','mood_swings','receiving_unsterile_injections',
             'stomach_bleeding','prominent_veins_on_calf','loss_of_smell','throat_irritation',
             'redness_of_eyes','sinus_pressure','runny_nose','pain_during_bowel_movements',
             'pain_in_anal_region','cramps','bruising','enlarged_thyroid','brittle_nails',
             'swollen_extremeties','slurred_speech','distention_of_abdomen','fluid_overload.1',
             'skin_peeling','silver_like_dusting','small_dents_in_nails','blister',
             'red_sore_around_nose','bloody_stool','swollen_blood_vessels','hip_joint_pain',
             'painful_walking','spinning_movements','altered_sensorium','toxic_look_(typhos)'
             ],axis=1,inplace=True)
```

```python
def data_preprocessing(data):
  data.drop(['weight_gain','cold_hands_and_feets','anxiety','irregular_sugar_level',
             'yellow_urine','acute_liver_failure','swelling_of_stomach',
             'drying_and_tingling_lips','continuous_feel_of_urine',
             'internal_itching','polyuria','mood_swings','receiving_unsterile_injections',
             'stomach_bleeding','prominent_veins_on_calf','loss_of_smell','throat_irritation',
             'redness_of_eyes','sinus_pressure','runny_nose','pain_during_bowel_movements',
             'pain_in_anal_region','cramps','bruising','enlarged_thyroid','brittle_nails',
             'swollen_extremeties','slurred_speech','distention_of_abdomen','fluid_overload.1',
             'skin_peeling','silver_like_dusting','small_dents_in_nails','blister',
             'red_sore_around_nose','bloody_stool','swollen_blood_vessels','hip_joint_pain',
             'painful_walking','spinning_movements','altered_sensorium','toxic_look_(typhos)'
             ],axis=1,inplace=True)
  return data
```

```python
testd=data_preprocessing(testd)
```



```python
testd=data_preprocessing(testd)
```

```python
x=traind.drop('prognosis',axis=1)
y=traind.prognosis
x_test=testd.drop('prognosis',axis=1)
y_test=testd.prognosis
x_train,x_val,y_train,y_val=train_test_split(x,y,test_size=0.2)
```

```python
def model_evaluation(classifier):
  y_pred=classifier.predict(x_val)
  yt_pred=classifier.predict(x_train)
  y_pred1=classifier.predict(x_test)
  print('The Training Accuracy of the algorithm is',accuracy_score(y_train,yt_pred))
  print('The Validation Accuracy of the algorithm is',accuracy_score(y_val,y_pred))
  print('The Testing Accuracy of the algorithm is',accuracy_score(y_test,y_pred1))
  return[(accuracy_score(y_train,yt_pred)),(accuracy_score(y_val,y_pred)),(accuracy_score(y_test,y_pred1))]
```

```python
knn=KNeighborsClassifier(n_neighbors=7)
knn.fit(x_train,y_train)
```

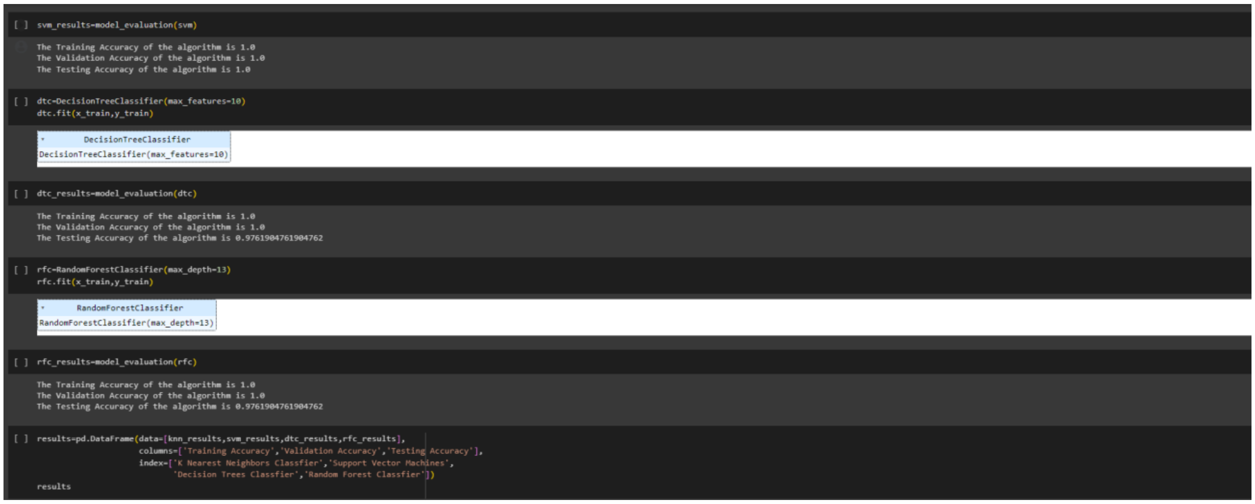
```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=7)
```

```python
knn_results=model_evaluation(knn)
```

```
The Training Accuracy of the algorithm is 1.0
The Validation Accuracy of the algorithm is 1.0
The Testing Accuracy of the algorithm is 1.0
```

```python
svm=SVC(C=1)
svm.fit(x_train,y_train)
```

```
SVC
SVC(C=1)
```



```python
svm_results=model_evaluation(svm)
```

```
The Training Accuracy of the algorithm is 1.0
The Validation Accuracy of the algorithm is 1.0
The Testing Accuracy of the algorithm is 1.0
```

```python
dtc=DecisionTreeClassifier(max_features=10)
dtc.fit(x_train,y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_features=10)
```

```python
dtc_results=model_evaluation(dtc)
```

```
The Training Accuracy of the algorithm is 1.0
The Validation Accuracy of the algorithm is 1.0
The Testing Accuracy of the algorithm is 0.9761904761904762
```

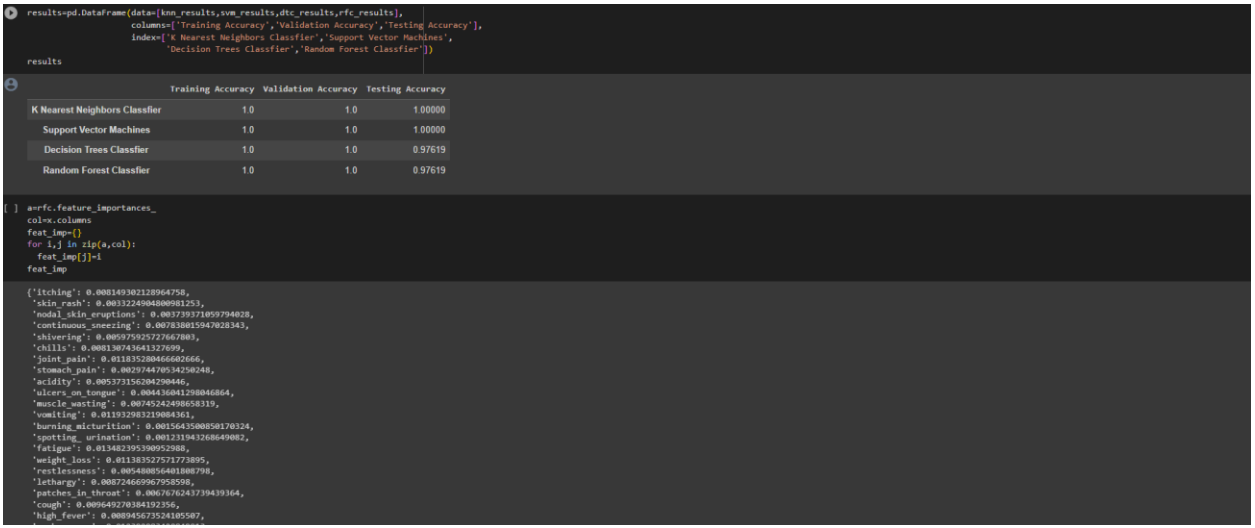
```python
rfc=RandomForestClassifier(max_depth=13)
rfc.fit(x_train,y_train)
```

```
RandomForestClassifier
RandomForestClassifier(max_depth=13)
```

```python
rfc_results=model_evaluation(rfc)
```

```
The Training Accuracy of the algorithm is 1.0
The Validation Accuracy of the algorithm is 1.0
The Testing Accuracy of the algorithm is 0.9761904761904762
```

```python
results=pd.DataFrame(data=[knn_results,svm_results,dtc_results,rfc_results],
                     columns=['Training Accuracy','Validation Accuracy','Testing Accuracy'],
                     index=['K Nearest Neighbors Classfier','Support Vector Machines',
                            'Decision Trees Classfier','Random Forest Classfier'])
results
```

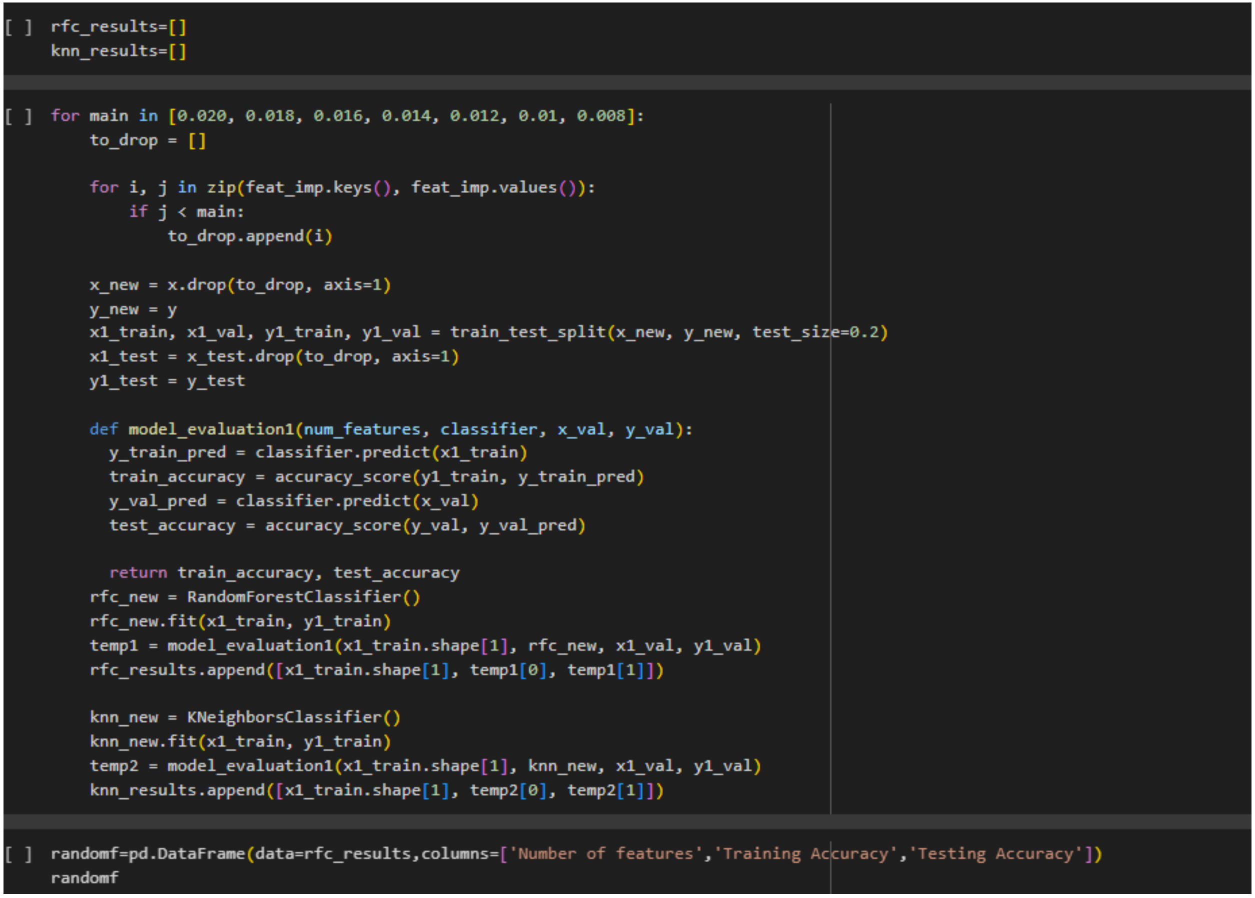


```
results=pd.DataFrame(data=[knn_results,svm_results,dtc_results,rfc_results],
                     columns=['Training Accuracy','Validation Accuracy','Testing Accuracy'],
                     index=['K Nearest Neighbors Classfier','Support Vector Machines',
                            'Decision Trees Classfier','Random Forest Classfier'])
results
```

	Training Accuracy	Validation Accuracy	Testing Accuracy
K Nearest Neighbors Classfier	1.0	1.0	1.00000
Support Vector Machines	1.0	1.0	1.00000
Decision Trees Classfier	1.0	1.0	0.97619
Random Forest Classfier	1.0	1.0	0.97619

```
[ ] a=rfc.feature_importances_
    col=x.columns
    feat_imp={}
    for i,j in zip(a,col):
      feat_imp[j]=i
    feat_imp
```

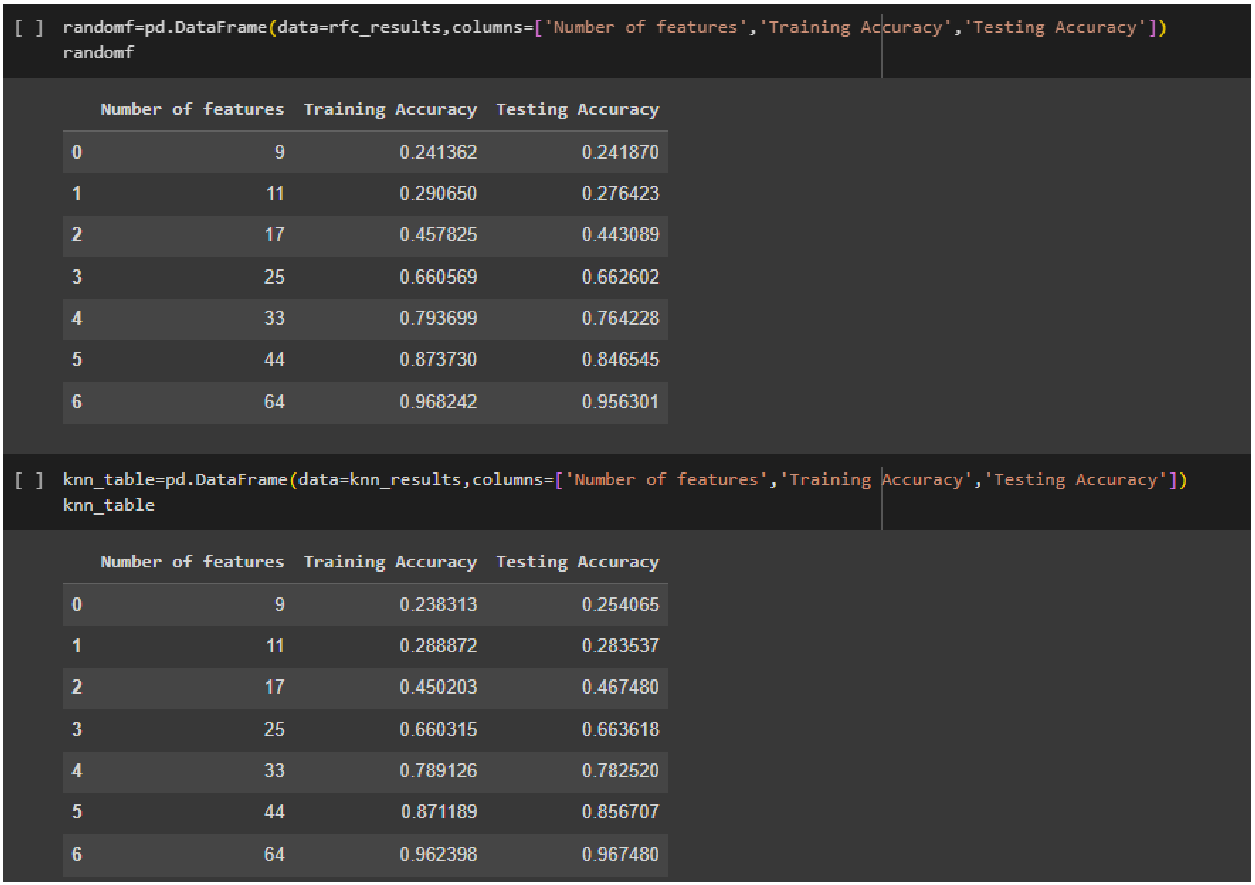
```
{'itching': 0.008149302128964758,
 'skin_rash': 0.003322490480098l253,
 'nodal_skin_eruptions': 0.003739371059794028,
 'continuous_sneezing': 0.007838015947028343,
 'shivering': 0.005975925727667803,
 'chills': 0.008130743641327699,
 'joint_pain': 0.011835280466602666,
 'stomach_pain': 0.002974470534250248,
 'acidity': 0.005373156204290446,
 'ulcers_on_tongue': 0.004436041298046864,
 'muscle_wasting': 0.007452424986583l9,
 'vomiting': 0.011932983219084361,
 'burning_micturition': 0.0015643500850170324,
 'spotting_ urination': 0.001231943268649082,
 'fatigue': 0.013482395390952988,
 'weight_loss': 0.011383527571773895,
 'restlessness': 0.005480856401808798,
 'lethargy': 0.008724669967958598,
 'patches_in_throat': 0.0067676243739439364,
 'cough': 0.009649270384192356,
 'high_fever': 0.008945673524105507,
```



```
[ ] rfc_results=[]
    knn_results=[]
```

```
[ ] for main in [0.020, 0.018, 0.016, 0.014, 0.012, 0.01, 0.008]:
        to_drop = []

        for i, j in zip(feat_imp.keys(), feat_imp.values()):
            if j < main:
                to_drop.append(i)

        x_new = x.drop(to_drop, axis=1)
        y_new = y
        x1_train, x1_val, y1_train, y1_val = train_test_split(x_new, y_new, test_size=0.2)
        x1_test = x_test.drop(to_drop, axis=1)
        y1_test = y_test

        def model_evaluation1(num_features, classifier, x_val, y_val):
          y_train_pred = classifier.predict(x1_train)
          train_accuracy = accuracy_score(y1_train, y_train_pred)
          y_val_pred = classifier.predict(x_val)
          test_accuracy = accuracy_score(y_val, y_val_pred)

          return train_accuracy, test_accuracy
        rfc_new = RandomForestClassifier()
        rfc_new.fit(x1_train, y1_train)
        temp1 = model_evaluation1(x1_train.shape[1], rfc_new, x1_val, y1_val)
        rfc_results.append([x1_train.shape[1], temp1[0], temp1[1]])

        knn_new = KNeighborsClassifier()
        knn_new.fit(x1_train, y1_train)
        temp2 = model_evaluation1(x1_train.shape[1], knn_new, x1_val, y1_val)
        knn_results.append([x1_train.shape[1], temp2[0], temp2[1]])
```

```
[ ] randomf=pd.DataFrame(data=rfc_results,columns=['Number of features','Training Accuracy','Testing Accuracy'])
    randomf
```

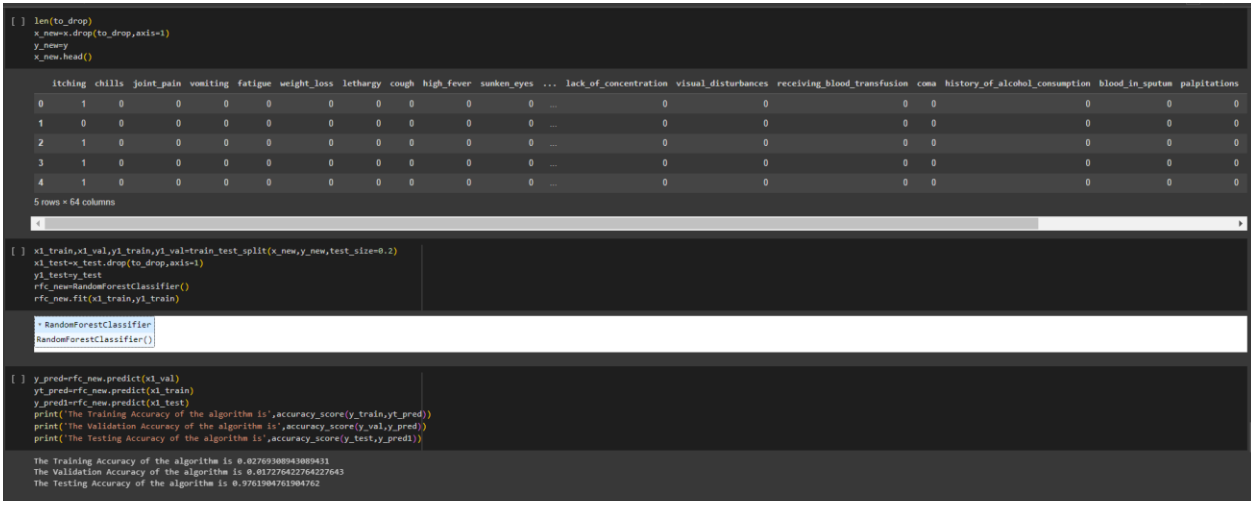


```
[ ] randomf=pd.DataFrame(data=rfc_results,columns=['Number of features','Training Accuracy','Testing Accuracy'])
    randomf
```

	Number of features	Training Accuracy	Testing Accuracy
0	9	0.241362	0.241870
1	11	0.290650	0.276423
2	17	0.457825	0.443089
3	25	0.660569	0.662602
4	33	0.793699	0.764228
5	44	0.873730	0.846545
6	64	0.968242	0.956301

```
[ ] knn_table=pd.DataFrame(data=knn_results,columns=['Number of features','Training Accuracy','Testing Accuracy'])
    knn_table
```

	Number of features	Training Accuracy	Testing Accuracy
0	9	0.238313	0.254065
1	11	0.288872	0.283537
2	17	0.450203	0.467480
3	25	0.660315	0.663618
4	33	0.789126	0.782520
5	44	0.871189	0.856707
6	64	0.962398	0.967480



```
[ ] len(to_drop)
    x_new=x.drop(to_drop,axis=1)
    y_new=y
    x_new.head()
```

	itching	chills	joint_pain	vomiting	fatigue	weight_loss	lethargy	cough	high_fever	sunken_eyes	...	lack_of_concentration	visual_disturbances	receiving_blood_transfusion	coma	history_of_alcohol_consumption	blood_in_sputum	palpitations
0	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
4	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0

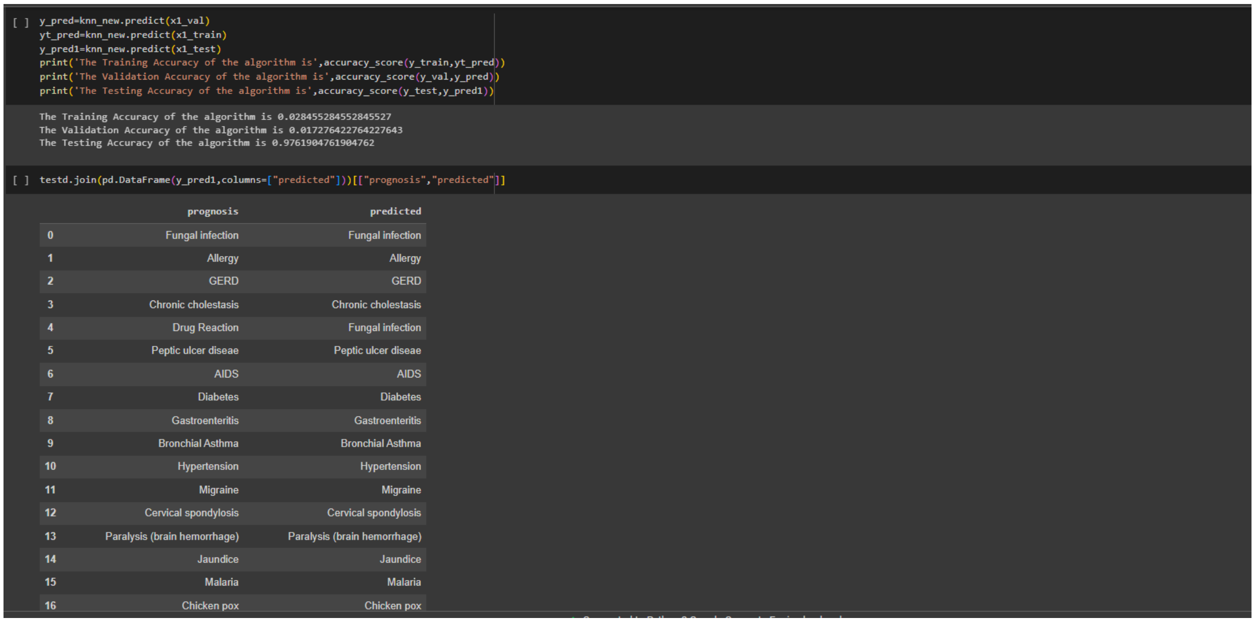
5 rows × 64 columns

```
[ ] x1_train,x1_val,y1_train,y1_val=train_test_split(x_new,y_new,test_size=0.2)
    x1_test=x_test.drop(to_drop,axis=1)
    y1_test=y_test
    rfc_new=RandomForestClassifier()
    rfc_new.fit(x1_train,y1_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
[ ] y_pred=rfc_new.predict(x1_val)
    yt_pred=rfc_new.predict(x1_train)
    y_pred1=rfc_new.predict(x1_test)
    print('The Training Accuracy of the algorithm is',accuracy_score(y_train,yt_pred))
    print('The Validation Accuracy of the algorithm is',accuracy_score(y_val,y_pred))
    print('The Testing Accuracy of the algorithm is',accuracy_score(y_test,y_pred1))
```

```
The Training Accuracy of the algorithm is 0.02769308943089431
The Validation Accuracy of the algorithm is 0.017276422764227643
The Testing Accuracy of the algorithm is 0.9761904761904762
```



```
[ ] y_pred=knn_new.predict(x1_val)
    yt_pred=knn_new.predict(x1_train)
    y_pred1=knn_new.predict(x1_test)
    print('The Training Accuracy of the algorithm is',accuracy_score(y_train,yt_pred))
    print('The Validation Accuracy of the algorithm is',accuracy_score(y_val,y_pred))
    print('The Testing Accuracy of the algorithm is',accuracy_score(y_test,y_pred1))
```

```
The Training Accuracy of the algorithm is 0.028455284552845527
The Validation Accuracy of the algorithm is 0.017276422764227643
The Testing Accuracy of the algorithm is 0.9761904761904762
```

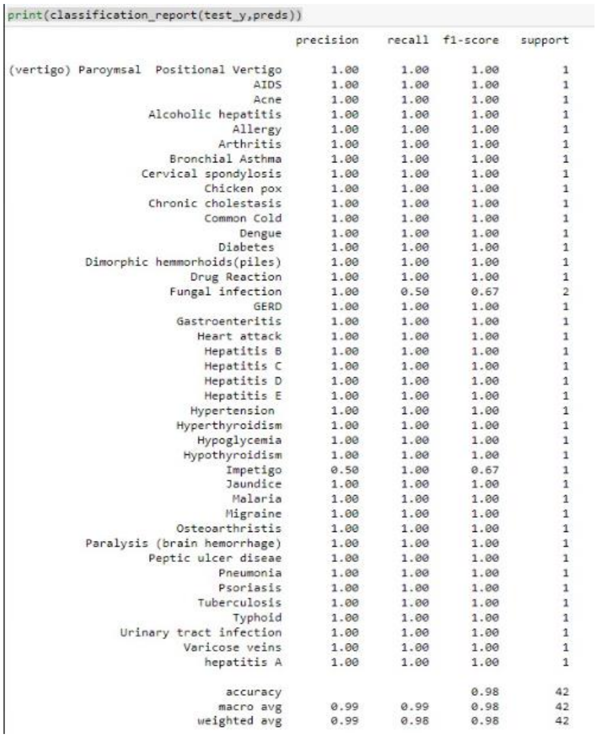
```
[ ] testd.join(pd.DataFrame(y_pred1,columns=["predicted"]))[["prognosis","predicted"]]
```

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

PERFORMANCE TESTING

Performance Metrics

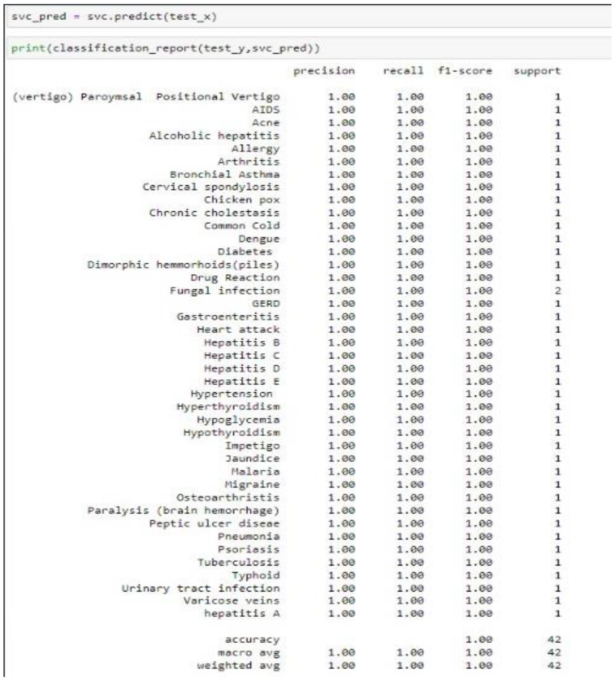
Random forest classifier accuracy



```
print(classification_report(test_y,preds))
```

```
                                         precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       1.00      1.00      1.00         1
                                   AIDS       1.00      1.00      1.00         1
                                   Acne       1.00      1.00      1.00         1
                    Alcoholic hepatitis       1.00      1.00      1.00         1
                                Allergy       1.00      1.00      1.00         1
                              Arthritis       1.00      1.00      1.00         1
                       Bronchial Asthma       1.00      1.00      1.00         1
                   Cervical spondylosis       1.00      1.00      1.00         1
                            Chicken pox       1.00      1.00      1.00         1
                    Chronic cholestasis       1.00      1.00      1.00         1
                            Common Cold       1.00      1.00      1.00         1
                                 Dengue       1.00      1.00      1.00         1
                              Diabetes        1.00      1.00      1.00         1
            Dimorphic hemmorhoids(piles)       1.00      1.00      1.00         1
                          Drug Reaction       1.00      1.00      1.00         1
                       Fungal infection       1.00      0.50      0.67         2
                                   GERD       1.00      1.00      1.00         1
                        Gastroenteritis       1.00      1.00      1.00         1
                           Heart attack       1.00      1.00      1.00         1
                            Hepatitis B       1.00      1.00      1.00         1
                            Hepatitis C       1.00      1.00      1.00         1
                            Hepatitis D       1.00      1.00      1.00         1
                            Hepatitis E       1.00      1.00      1.00         1
                          Hypertension        1.00      1.00      1.00         1
                        Hyperthyroidism       1.00      1.00      1.00         1
                           Hypoglycemia       1.00      1.00      1.00         1
                         Hypothyroidism       1.00      1.00      1.00         1
                               Impetigo       0.50      1.00      0.67         1
                               Jaundice       1.00      1.00      1.00         1
                                Malaria       1.00      1.00      1.00         1
                               Migraine       1.00      1.00      1.00         1
                        Osteoarthristis       1.00      1.00      1.00         1
           Paralysis (brain hemorrhage)       1.00      1.00      1.00         1
                    Peptic ulcer diseae       1.00      1.00      1.00         1
                              Pneumonia       1.00      1.00      1.00         1
                              Psoriasis       1.00      1.00      1.00         1
                           Tuberculosis       1.00      1.00      1.00         1
                                Typhoid       1.00      1.00      1.00         1
                Urinary tract infection       1.00      1.00      1.00         1
                         Varicose veins       1.00      1.00      1.00         1
                            hepatitis A       1.00      1.00      1.00         1

                               accuracy                           0.98        42
                              macro avg       0.99      0.99      0.98        42
                           weighted avg       0.99      0.98      0.98        42
```

Support vector classifier

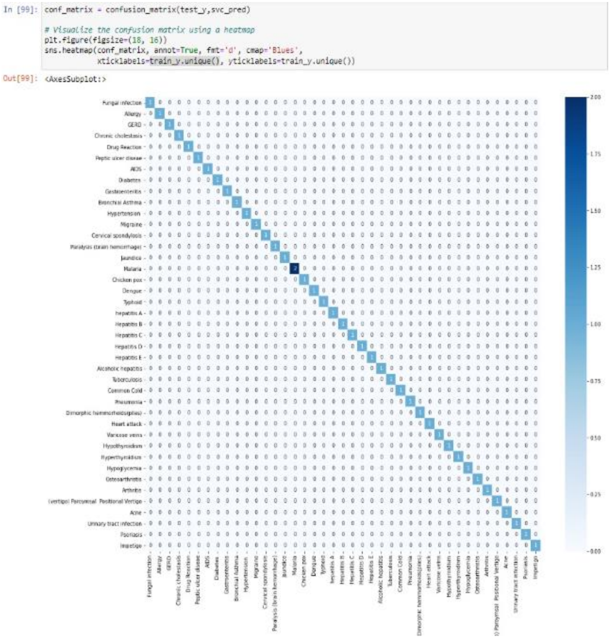


```
svc_pred = svc.predict(test_x)
```

```
print(classification_report(test_y,svc_pred))
```

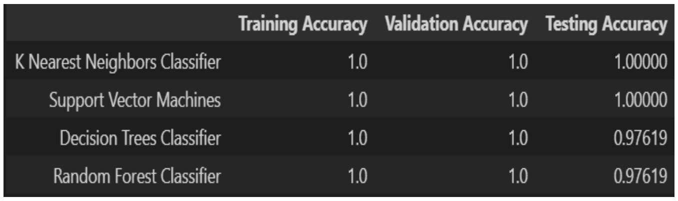
```
                                      precision    recall  f1-score   support

(vertigo) Paroymsal  Positional Vertigo       1.00      1.00      1.00         1
                                   AIDS       1.00      1.00      1.00         1
                                   Acne       1.00      1.00      1.00         1
                    Alcoholic hepatitis       1.00      1.00      1.00         1
                                Allergy       1.00      1.00      1.00         1
                              Arthritis       1.00      1.00      1.00         1
                       Bronchial Asthma       1.00      1.00      1.00         1
                   Cervical spondylosis       1.00      1.00      1.00         1
                            Chicken pox       1.00      1.00      1.00         1
                    Chronic cholestasis       1.00      1.00      1.00         1
                            Common Cold       1.00      1.00      1.00         1
                                 Dengue       1.00      1.00      1.00         1
                              Diabetes        1.00      1.00      1.00         1
           Dimorphic hemmorhoids(piles)       1.00      1.00      1.00         1
                          Drug Reaction       1.00      1.00      1.00         1
                        Fungal infection       1.00      1.00      1.00         2
                                   GERD       1.00      1.00      1.00         1
                         Gastroenteritis       1.00      1.00      1.00         1
                           Heart attack       1.00      1.00      1.00         1
                            Hepatitis B       1.00      1.00      1.00         1
                            Hepatitis C       1.00      1.00      1.00         1
                            Hepatitis D       1.00      1.00      1.00         1
                            Hepatitis E       1.00      1.00      1.00         1
                          Hypertension        1.00      1.00      1.00         1
                        Hyperthyroidism       1.00      1.00      1.00         1
                           Hypoglycemia       1.00      1.00      1.00         1
                         Hypothyroidism       1.00      1.00      1.00         1
                               Impetigo       1.00      1.00      1.00         1
                               Jaundice       1.00      1.00      1.00         1
                                Malaria       1.00      1.00      1.00         1
                               Migraine       1.00      1.00      1.00         1
                        Osteoarthristis       1.00      1.00      1.00         1
           Paralysis (brain hemorrhage)       1.00      1.00      1.00         1
                    Peptic ulcer diseae       1.00      1.00      1.00         1
                              Pneumonia       1.00      1.00      1.00         1
                              Psoriasis       1.00      1.00      1.00         1
                           Tuberculosis       1.00      1.00      1.00         1
                                Typhoid       1.00      1.00      1.00         1
                Urinary tract infection       1.00      1.00      1.00         1
                         Varicose veins       1.00      1.00      1.00         1
                            hepatitis A       1.00      1.00      1.00         1

                               accuracy                           1.00        42
                              macro avg       1.00      1.00      1.00        42
                           weighted avg       1.00      1.00      1.00        42
```

Confsuion Matrix



```
In [99]: conf_matrix = confusion_matrix(test_y,svc_pred)

         # Visualize the confusion matrix using a heatmap
         plt.figure(figsize=(18, 16))
         sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
                     xticklabels=train_y.unique(), yticklabels=train_y.unique())

Out[99]: <AxesSubplot:>
```

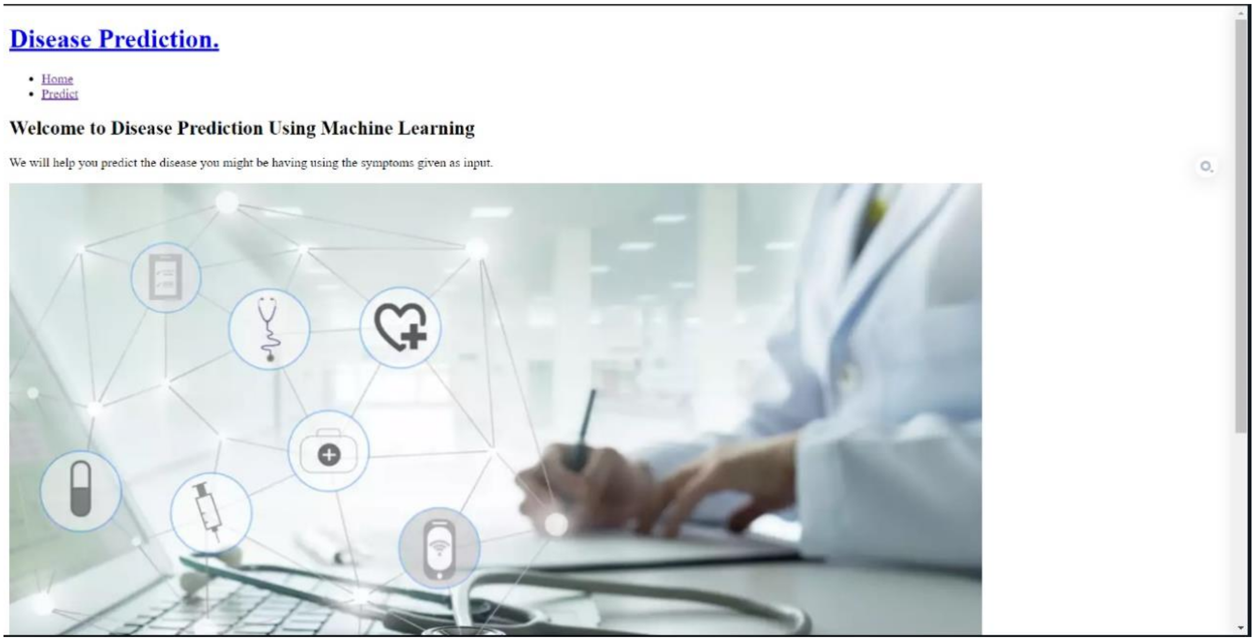
Accuracy



	Training Accuracy	Validation Accuracy	Testing Accuracy
K Nearest Neighbors Classifier	1.0	1.0	1.00000
Support Vector Machines	1.0	1.0	1.00000
Decision Trees Classifier	1.0	1.0	0.97619
Random Forest Classifier	1.0	1.0	0.97619

RESULTS

Output Screenshots

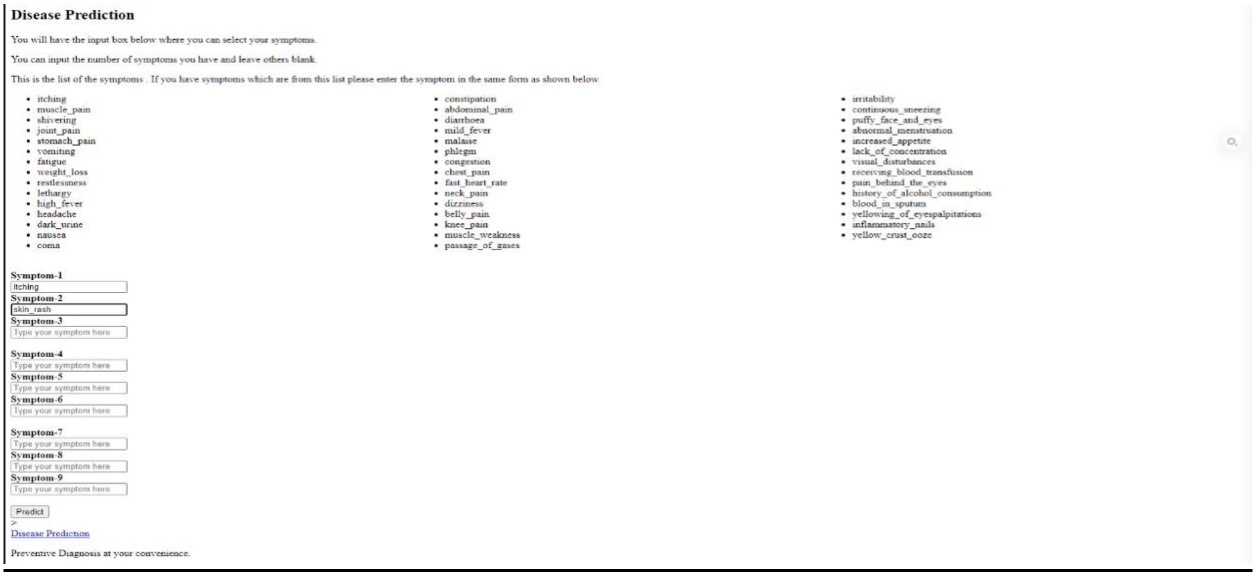


Disease Prediction.

- Home
- Predict

Welcome to Disease Prediction Using Machine Learning

We will help you predict the disease you might be having using the symptoms given as input.



Disease Prediction

You will have the input box below where you can select your symptoms.

You can input the number of symptoms you have and leave others blank.

This is the list of the symptoms . If you have symptoms which are from this list please enter the symptom in the same form as shown below.

- itching
- muscle_pain
- shivering
- joint_pain
- stomach_pain
- vomiting
- fatigue
- weight_loss
- restlessness
- lethargy
- high_fever
- headache
- dark_urine
- nausea
- coma
- constipation
- abdominal_pain
- diarrhoea
- mild_fever
- malaise
- phlegm
- congestion
- chest_pain
- fast_heart_rate
- neck_pain
- dizziness
- belly_pain
- knee_pain
- muscle_weakness
- passage_of_gases
- irritability
- continuous_sneezing
- puffy_face_and_eyes
- abnormal_menstruation
- increased_appetite
- lack_of_concentration
- visual_disturbances
- receiving_blood_transfusion
- pain_behind_the_eyes
- history_of_alcohol_consumption
- blood_in_sputum
- yellowing_of_eyespalpitations
- inflammatory_nails
- yellow_crust_ooze

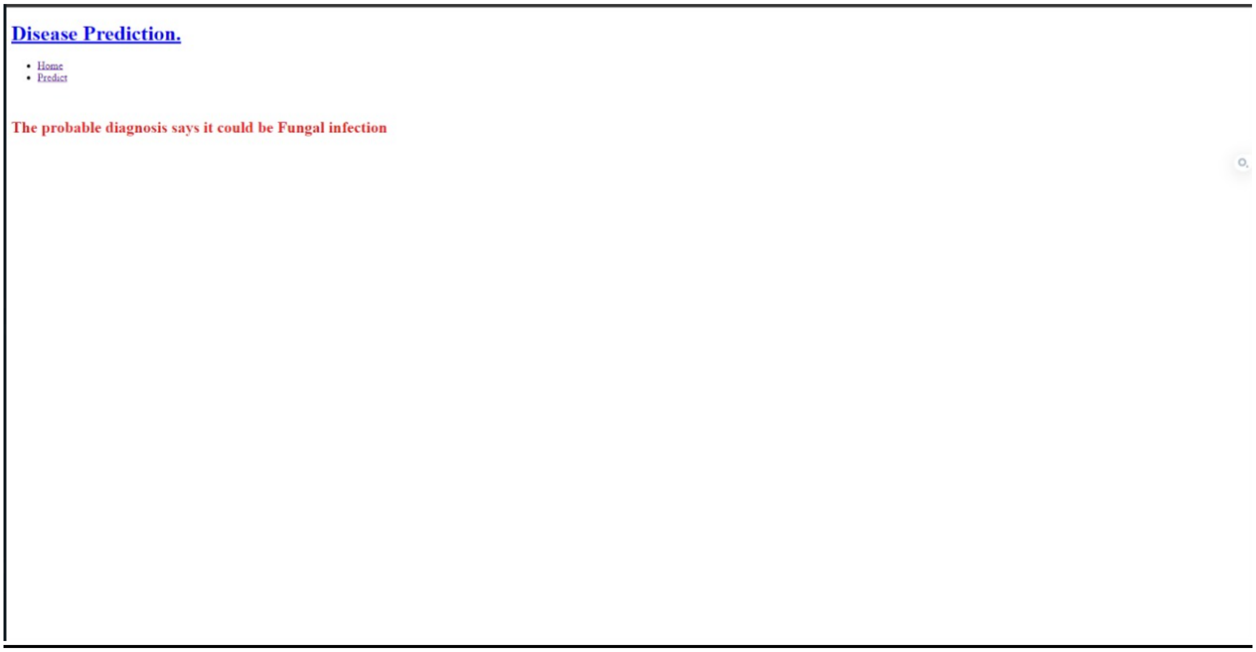
Symptom-1
itching
Symptom-2
skin_rash
Symptom-3
Type your symptom here

Symptom-4
Type your symptom here
Symptom-5
Type your symptom here
Symptom-6
Type your symptom here

Symptom-7
Type your symptom here
Symptom-8
Type your symptom here
Symptom-9
Type your symptom here

>
Disease Prediction

Preventive Diagnosis at your convenience.

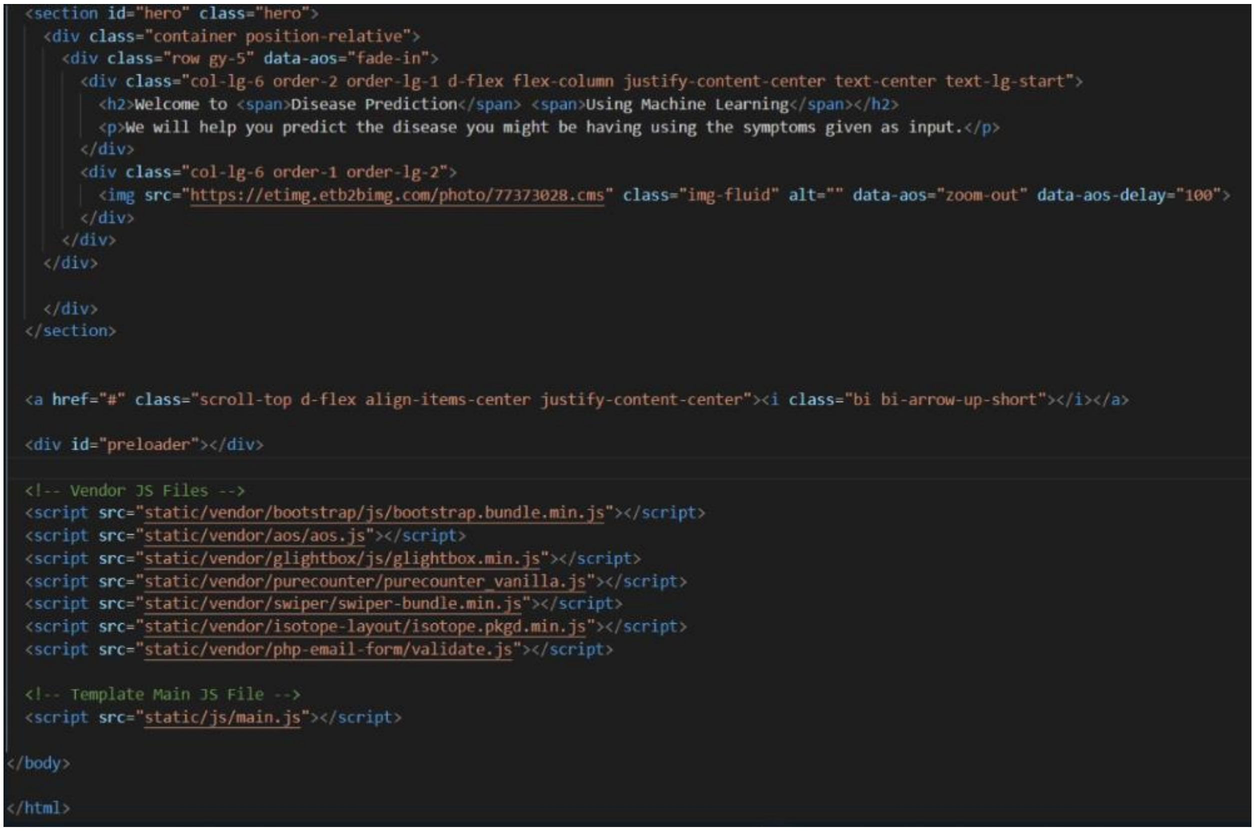


The probable diagnosis says it could be Fungal infection

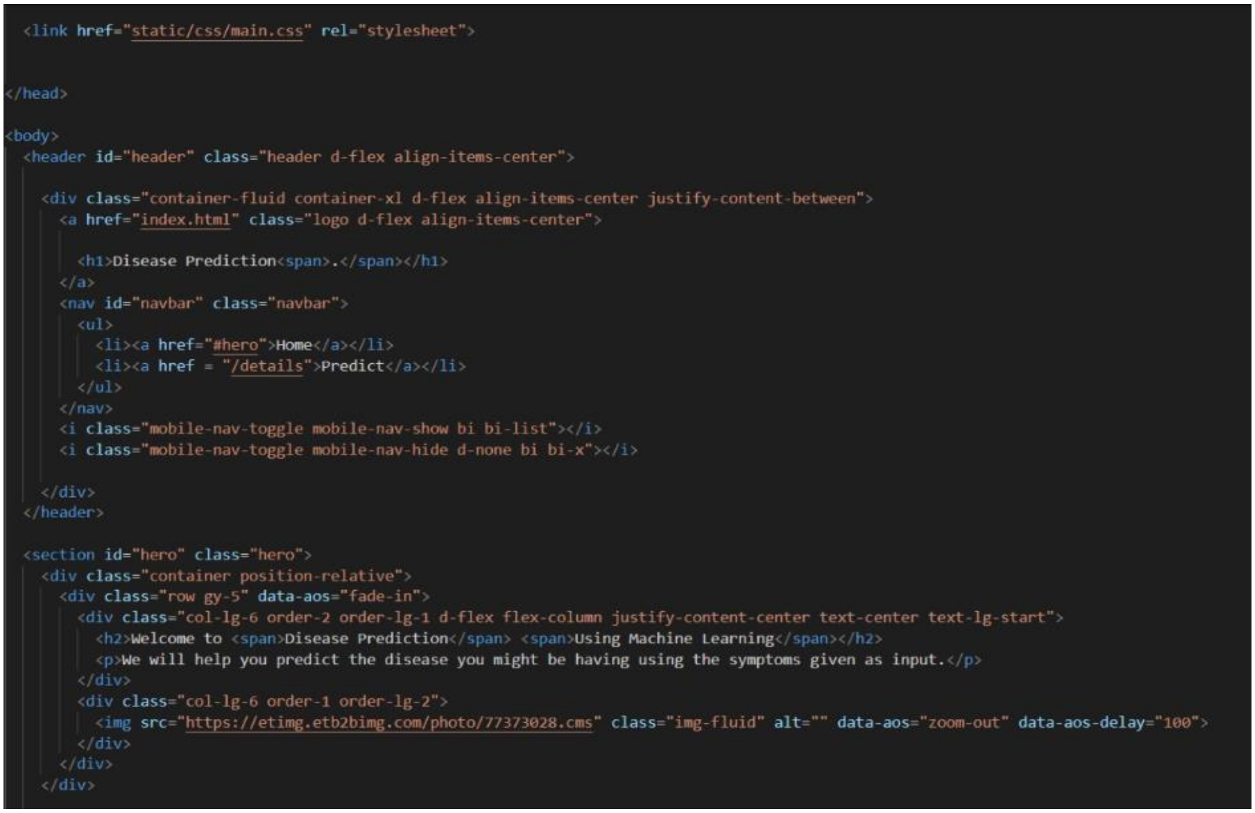
CONCLUSION

In conclusion, the "Disease Prediction using Machine Learning" project represents a significant stride towards revolutionizing healthcare by leveraging the power of data-driven decision-making. The journey from ideation to implementation has been guided by a commitment to addressing the challenges in traditional disease prediction methods and providing healthcare professionals with a valuable tool for early intervention.

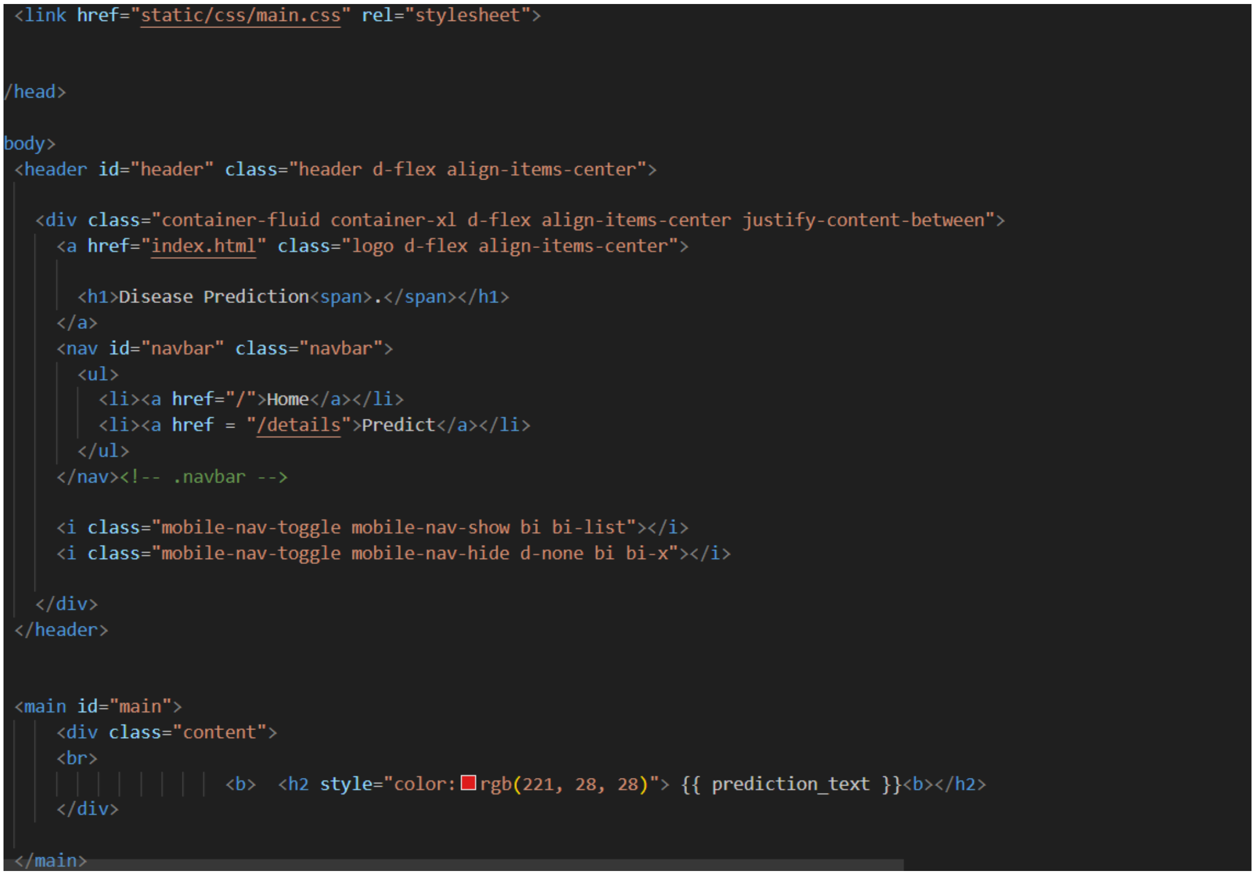
APPENDIX



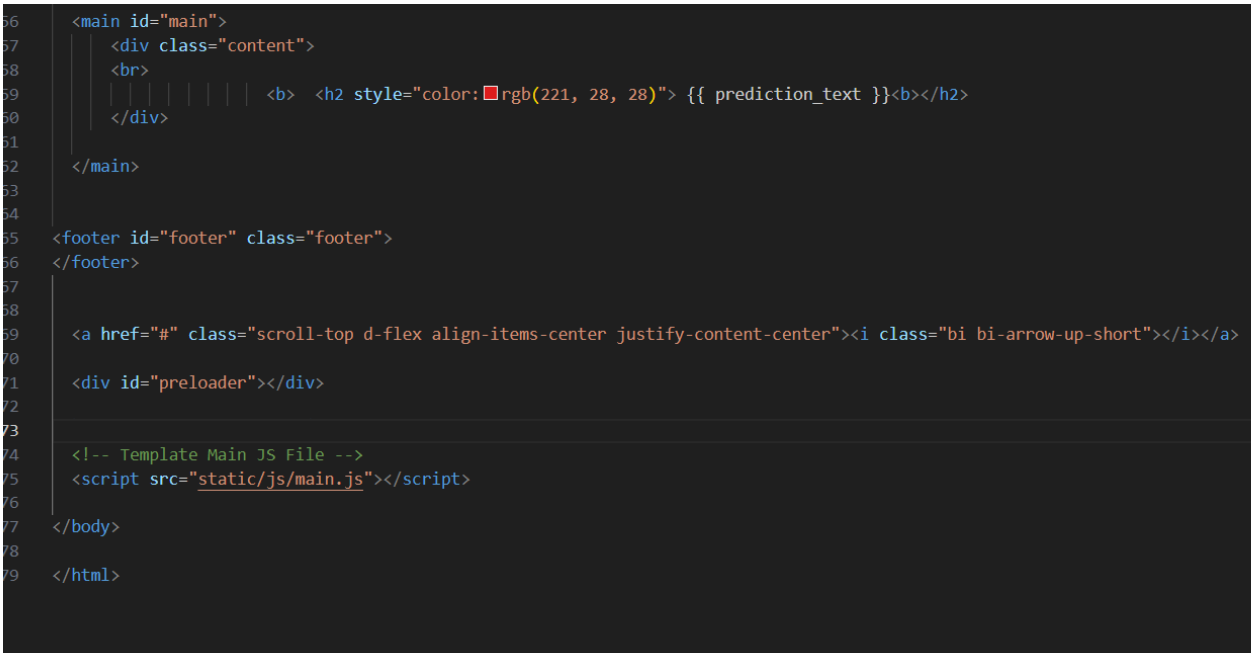
```
  <section id="hero" class="hero">
    <div class="container position-relative">
      <div class="row gy-5" data-aos="fade-in">
        <div class="col-lg-6 order-2 order-lg-1 d-flex flex-column justify-content-center text-center text-lg-start">
          <h2>Welcome to <span>Disease Prediction</span> <span>Using Machine Learning</span></h2>
          <p>We will help you predict the disease you might be having using the symptoms given as input.</p>
        </div>
        <div class="col-lg-6 order-1 order-lg-2">
          <img src="https://etimg.etb2bimg.com/photo/77373028.cms" class="img-fluid" alt="" data-aos="zoom-out" data-aos-delay="100">
        </div>
      </div>
    </div>

    </div>
  </section>


  <a href="#" class="scroll-top d-flex align-items-center justify-content-center"><i class="bi bi-arrow-up-short"></i></a>

  <div id="preloader"></div>

  <!-- Vendor JS Files -->
  <script src="static/vendor/bootstrap/js/bootstrap.bundle.min.js"></script>
  <script src="static/vendor/aos/aos.js"></script>
  <script src="static/vendor/glightbox/js/glightbox.min.js"></script>
  <script src="static/vendor/purecounter/purecounter_vanilla.js"></script>
  <script src="static/vendor/swiper/swiper-bundle.min.js"></script>
  <script src="static/vendor/isotope-layout/isotope.pkgd.min.js"></script>
  <script src="static/vendor/php-email-form/validate.js"></script>

  <!-- Template Main JS File -->
  <script src="static/js/main.js"></script>

</body>

</html>
```



```
  <link href="static/css/main.css" rel="stylesheet">


</head>

<body>
  <header id="header" class="header d-flex align-items-center">

    <div class="container-fluid container-xl d-flex align-items-center justify-content-between">
      <a href="index.html" class="logo d-flex align-items-center">

        <h1>Disease Prediction<span>.</span></h1>
      </a>
      <nav id="navbar" class="navbar">
        <ul>
          <li><a href="#hero">Home</a></li>
          <li><a href = "/details">Predict</a></li>
        </ul>
      </nav>
      <i class="mobile-nav-toggle mobile-nav-show bi bi-list"></i>
      <i class="mobile-nav-toggle mobile-nav-hide d-none bi bi-x"></i>

    </div>
  </header>

  <section id="hero" class="hero">
    <div class="container position-relative">
      <div class="row gy-5" data-aos="fade-in">
        <div class="col-lg-6 order-2 order-lg-1 d-flex flex-column justify-content-center text-center text-lg-start">
          <h2>Welcome to <span>Disease Prediction</span> <span>Using Machine Learning</span></h2>
          <p>We will help you predict the disease you might be having using the symptoms given as input.</p>
        </div>
        <div class="col-lg-6 order-1 order-lg-2">
          <img src="https://etimg.etb2bimg.com/photo/77373028.cms" class="img-fluid" alt="" data-aos="zoom-out" data-aos-delay="100">
        </div>
      </div>
    </div>
```



```
 <link href="static/css/main.css" rel="stylesheet">


/head>

body>
 <header id="header" class="header d-flex align-items-center">

   <div class="container-fluid container-xl d-flex align-items-center justify-content-between">
     <a href="index.html" class="logo d-flex align-items-center">

       <h1>Disease Prediction<span>.</span></h1>
     </a>
     <nav id="navbar" class="navbar">
       <ul>
         <li><a href="/">Home</a></li>
         <li><a href = "/details">Predict</a></li>
       </ul>
     </nav><!-- .navbar -->

     <i class="mobile-nav-toggle mobile-nav-show bi bi-list"></i>
     <i class="mobile-nav-toggle mobile-nav-hide d-none bi bi-x"></i>

   </div>
 </header>


 <main id="main">
     <div class="content">
     <br>
                     <b>  <h2 style="color:🟥rgb(221, 28, 28)"> {{ prediction_text }}<b></h2>
     </div>

 </main>
```



```
  <main id="main">
      <div class="content">
      <br>
                       <b>  <h2 style="color:🟥rgb(221, 28, 28)"> {{ prediction_text }}<b></h2>
      </div>

  </main>


<footer id="footer" class="footer">
</footer>


  <a href="#" class="scroll-top d-flex align-items-center justify-content-center"><i class="bi bi-arrow-up-short"></i></a>

  <div id="preloader"></div>


  <!-- Template Main JS File -->
  <script src="static/js/main.js"></script>

</body>

</html>
```