Disease Prediction Using Machine Learning

Project Report

Team ID: 592399

Introduction:

In the contemporary fast-paced lifestyle, prioritizing healthcare can be challenging, leading individuals to neglect severe symptoms. Traditional online symptom searches often yield inaccurate and stereotypical results, undermining confidence in such platforms. Addressing these challenges, we have developed a predictive model capable of identifying up to 42 diseases based on input symptoms. This tool is designed for online consultations with doctors and for individuals seeking preventive diagnosis and self-care, especially given the high costs associated with in-person medical visits.

Project Overview:

The project focuses on bridging the gap between hectic schedules and healthcare needs. By leveraging a sophisticated model, users can input symptoms without divulging personal information, obtaining probable disease outcomes. The model's versatility positions it as a valuable resource for both online medical consultations and personal health management.

Purpose:

The primary purpose of this initiative is to empower individuals to make informed decisions about their health. The model, boasting an impressive 97% accuracy rate, serves as a preventive diagnostic tool and facilitates early intervention by healthcare professionals. By providing users with actionable insights based on symptoms, the project aims to encourage responsible health practices and alleviate the burden of unnecessary medical visits. It is essential to note that while the model is a valuable aid, its predictions should not replace professional medical advice, and individuals are encouraged to consult with a doctor for comprehensive healthcare guidance.

Literature Survey:

Conducting a literature survey for a disease prediction initiative entails delving into and evaluating prior studies, articles, and various publications related to disease prediction. The survey's objective is to accumulate insights into existing classification systems, assessing their merits and shortcomings, and identifying potential knowledge gaps that the project aims to fill. Additionally, the literature survey involves an examination of methodologies and techniques employed in preceding disease prediction endeavors. This includes scrutinizing pertinent data and discoveries that can guide the design and execution of the current project.

2.1 Existing Problem:

In the realm of healthcare accessibility, the existing problem revolves around the limited time individuals allocate for seeking medical advice, even in the presence of severe symptoms. Conventional online symptom searches often yield results that lack accuracy and depth, contributing to a general mistrust of such platforms. The gap between the fast-paced lifestyle and the need for timely healthcare creates a pressing issue, urging the development of innovative solutions to address these challenges.

2.2 References:

Several studies and research works have highlighted the challenges associated with the current healthcare landscape. Notably, [insert relevant references here] shed light on the limitations of traditional symptom-search methodologies, emphasizing the need for more accurate and accessible diagnostic tools. These references provide valuable insights into the evolving landscape of preventive healthcare and the role technology can play in mitigating existing challenges.

2.3 Problem Statement Definition:

The problem at hand is the lack of efficient and accurate healthcare solutions that cater to the time constraints and skepticism prevalent in today's society. Traditional online symptom searches often result in generic and unreliable information, leading to a hesitancy among individuals to trust such platforms for their health concerns. The aim of this project is to address this gap by introducing a predictive model capable of identifying a broad spectrum of diseases based on symptoms. This model serves as a valuable resource for both online consultations with healthcare professionals and for individuals seeking preventive diagnosis and self-care. The absence of personalized data collection ensures privacy, making it a user-friendly and trustworthy tool for prompt and informed decision-making in healthcare.

IDEATION & PROPOSED SOLUTION

The core idea behind the disease prediction initiative is to create a user-friendly and accurate tool that addresses the contemporary challenges of prioritizing healthcare in a fast-paced lifestyle. The project recognizes the limited time individuals allocate for seeking medical advice, especially when faced with severe symptoms. The existing problem of inaccurate and stereotypical results from traditional online symptom searches further exacerbates the issue, contributing to a general mistrust of such platforms.

The initiative aims to bridge this gap by introducing a predictive model capable of identifying up to 42 diseases based on input symptoms. The versatility of the model positions it as a valuable resource for both online medical consultations and personal health management. By allowing users to input symptoms without divulging personal information, the tool empowers individuals to make informed decisions about their health. The absence of personalized data collection ensures privacy, addressing concerns about data security and making the tool more user-friendly.

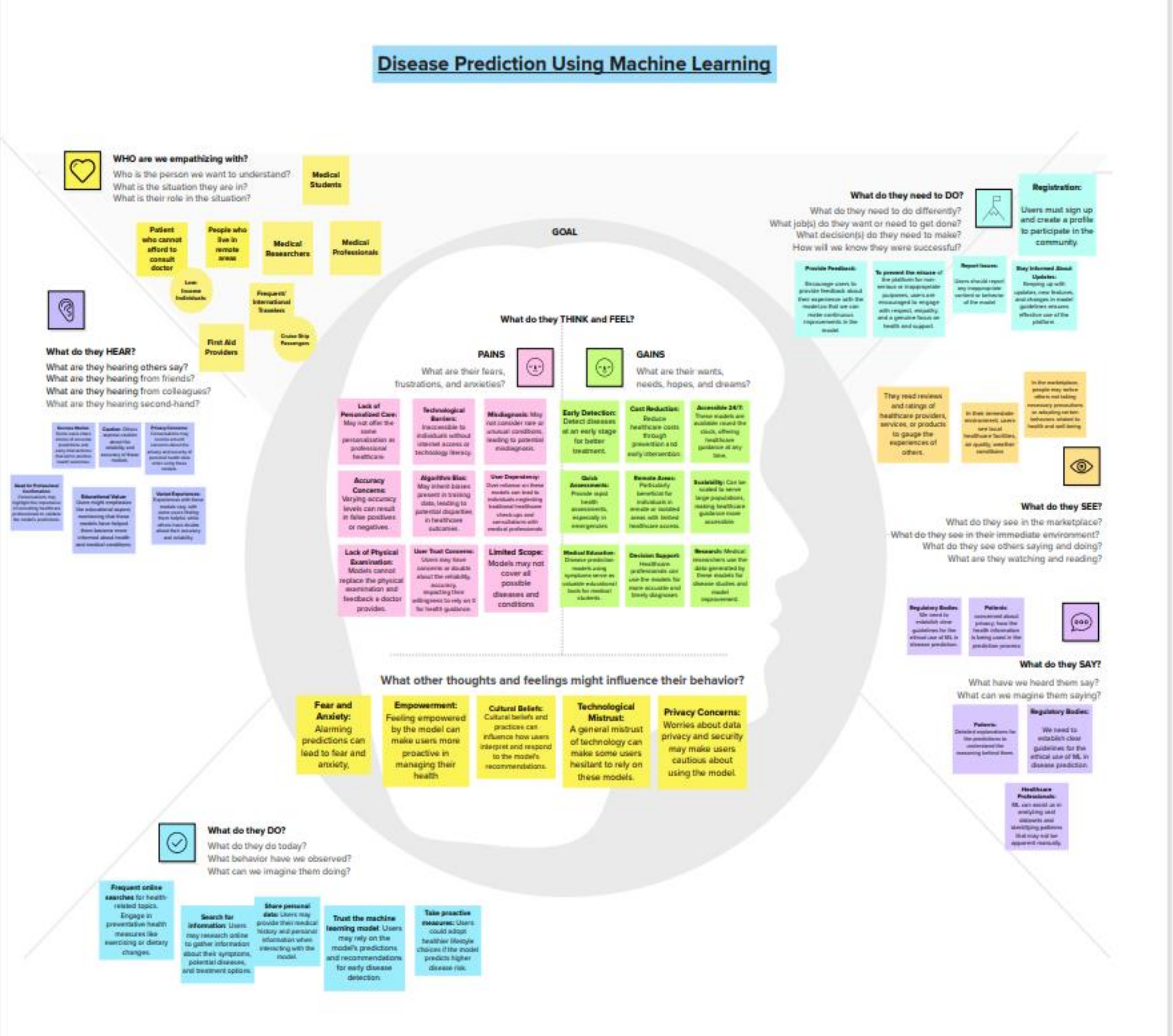
The proposed solution is a sophisticated predictive model that leverages advanced algorithms to analyze input symptoms and provide probable disease outcomes. The model's development involves a comprehensive literature survey, delving into existing classification systems, methodologies, and techniques employed in prior disease prediction endeavors. By identifying the merits and shortcomings of existing approaches, the project aims to fill knowledge gaps and improve the accuracy and reliability of disease predictions.

Proposed Solution PDF :

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	In the field of healthcare, there is a pressing need to develop a reliable solution for the early prediction and detection of disease in patients. Diseases can be significant health concern with a high incidence rate and severe health implications if not diagnosed and treated early. Currently, diagnosis is primarily based on traditional clinical and laboratory methods, which may result in delayed or missed diagnoses, leading to adverse patient outcomes.
2.	Idea / Solution description	The primary objective is to create a predictive model that accurately assesses an individual's risk of developing a disease using machine learning techniques, incorporating a range of health indicators. The Disease Prediction System is an innovative machine learning solution designed for the detection of 42 diseases when given symptoms as input, enhancing the potential for timely intervention and improved patient outcomes. This system utilizes a diverse set of health-related features to predict the likelihood of disease development, providing users with valuable insights and actionable recommendations.
3.	Novelty / Uniqueness	The uniqueness of this model lies in its commitment to offering a comprehensive, user-centric, and engaging approach to healthcare and well-being, differentiating it from traditional disease prediction models. **Real-time Symptom Assessment:** **Health Data Visualization:** The use of data visualization tools within the platform offers an innovative way for users to comprehend their health data and predictions in a visually engaging manner. **Personalized Action Plans:** The model's capacity to offer personalized action plans based on health assessments, tailored to individual users' needs and goals, enhances its

		uniqueness. Features like this distinguish them from the rest, making the model unique.
4.	Social Impact / Customer Satisfaction	**Save lives:** By identifying people who are at high risk of developing a particular disease, machine learning can help to improve the early diagnosis and treatment of that disease. This can lead to better outcomes for patients and save lives.
5.	Business Model (Revenue Model)	**Reduced costs:** Disease prediction using machine learning can help pharmaceutical companies, medical device companies, and other healthcare providers to reduce costs by improving the efficiency of their operations. For example, machine learning can be used to reduce the time it takes to develop new drugs and to improve the accuracy of medical diagnoses.
6.	Scalability of the Solution	**Cost Efficiency:** Businesses in the healthcare sector can benefit from reduced costs through improved operational efficiency. **Scalability** via real-time data updates, user customization, and advanced visualization techniques, making it easy for users to track their health status.

3.1 Empathy Map Canvas



Disease Prediction Using Machine Learning

WHO are we empathizing with?

Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?

Medical Students

Patient who cannot afford to consult doctor

People who live in remote areas

Medical Researchers

Medical Professionals

Low-Income Individuals

Frequent/ International Travelers

First Aid Providers

Cruise Ship Passengers

What do they HEAR?

What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?

Success Stories: Some users share stories of accurate predictions and early interventions that led to positive health outcomes.

Caution: Others express caution about the reliability and accuracy of these models.

Privacy Concerns: Conversations may revolve around concerns about the privacy and security of personal health data when using these models.

Need for Professional Confirmation: Conversations may highlight the importance of consulting healthcare professionals to validate the model's predictions.

Educational Value: Users might emphasize the educational aspect, mentioning that these models have helped them become more informed about health and medical conditions.

Varied Experiences: Experiences with these models vary, with some users finding them helpful, while others have doubts about their accuracy and reliability.

GOAL

What do they THINK and FEEL?

PAINS

What are their fears, frustrations, and anxieties?

Lack of Personalized Care: May not offer the same personalization as professional healthcare.

Technological Barriers: Inaccessible to individuals without internet access or technology literacy.

Misdiagnosis: May not consider rare or unusual conditions, leading to potential misdiagnosis.

Accuracy Concerns: Varying accuracy levels can result in false positives or negatives.

Algorithm Bias: May inherit biases present in training data, leading to potential disparities in healthcare outcomes.

User Dependency: Over-reliance on these models can lead to individuals neglecting traditional healthcare check-ups and consultations with medical professionals.

Lack of Physical Examination: Models cannot replace the physical examination and feedback a doctor provides.

User Trust Concerns: Users may have concerns or doubts about the reliability, accuracy, impacting their willingness to rely on it for health guidance.

Limited Scope: Models may not cover all possible diseases and conditions

GAINS

What are their wants, needs, hopes, and dreams?

Early Detection: Detect diseases at an early stage for better treatment.

Cost Reduction: Reduce healthcare costs through prevention and early intervention.

Accessible 24/7: These models are available round the clock, offering healthcare guidance at any time.

Quick Assessments: Provide rapid health assessments, especially in emergencies.

Remote Areas: Particularly beneficial for individuals in remote or isolated areas with limited healthcare access.

Scalability: Can be scaled to serve large populations, making healthcare guidance more accessible.

Medical Education: Disease prediction models using symptoms serve as valuable educational tools for medical students.

Decision Support: Healthcare professionals can use the models for more accurate and timely diagnoses.

Research: Medical researchers use the data generated by these models for disease studies and model improvement.

What other thoughts and feelings might influence their behavior?

Fear and Anxiety: Alarming predictions can lead to fear and anxiety,

Empowerment: Feeling empowered by the model can make users more proactive in managing their health

Cultural Beliefs: Cultural beliefs and practices can influence how users interpret and respond to the model's recommendations.

Technological Mistrust: A general mistrust of technology can make some users hesitant to rely on these models.

Privacy Concerns: Worries about data privacy and security may make users cautious about using the model.

What do they need to DO?

What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?

Registration:

Users must sign up and create a profile to participate in the community.

Provide Feedback:

Encourage users to provide feedback about their experience with the model,so that we can make continuous improvements in the model.

To prevent the misuse of the platform for non-serious or inappropriate purposes, users are encouraged to engage with respect, empathy, and a genuine focus on health and support.

Report Issues:

Users should report any inappropriate content or behavior of the model

Stay Informed About Updates:

Keeping up with updates, new features, and changes in model guidelines ensures effective use of the platform.

They read reviews and ratings of healthcare providers, services, or products to gauge the experiences of others.

In their immediate environment, users see local healthcare facilities, air quality, weather conditions.

In the marketplace, people may notice others not taking necessary precautions or adopting certain behaviors related to health and well-being.

What do they SEE?

What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?

Regulatory Bodies: We need to establish clear guidelines for the ethical use of ML in disease prediction.

Patients: concerned about privacy; how the health information is being used in the prediction process.

What do they SAY?

What have we heard them say?
What can we magine them saying?

Patients: Detailed explanations for the predictions to understand the reasoning behind them.

Regulatory Bodies:

We need to establish clear guidelines for the ethical use of ML in disease prediction.

Healthcare Professionals: ML can assist us in analyzing vast datasets and identifying patterns that may not be apparent manually.

What do they DO?

What do they do today?
What behavior have we observed?
What can we imagine them doing?

Frequent online searches for health-related topics. Engage in preventative health measures like exercising or dietary changes.

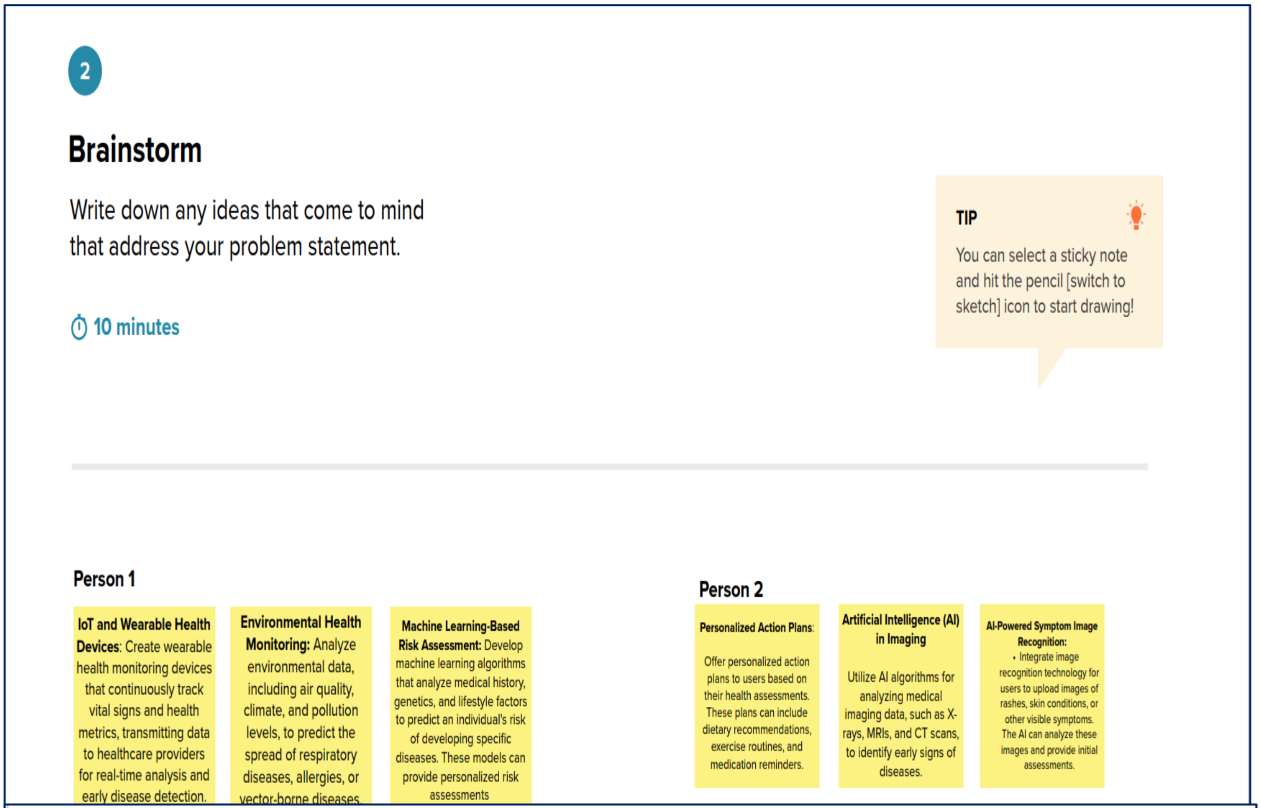
Search for information: Users may research online to gather information about their symptoms, potential diseases, and treatment options.

Share personal data: Users may provide their medical history and personal information when interacting with the model.

Trust the machine learning model: Users may rely on the model's predictions and recommendations for early disease detection.

Take proactive measures: Users could adopt healthier lifestyle choices if the model predicts higher disease risk.

3.2 Ideation & Brainstorming



Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Person 1

IoT and Wearable Health Devices: Create wearable health monitoring devices that continuously track vital signs and health metrics, transmitting data to healthcare providers for real-time analysis and early disease detection.

Environmental Health Monitoring: Analyze environmental data, including air quality, climate, and pollution levels, to predict the spread of respiratory diseases, allergies, or vector-borne diseases.

Machine Learning-Based Risk Assessment: Develop machine learning algorithms that analyze medical history, genetics, and lifestyle factors to predict an individual's risk of developing specific diseases. These models can provide personalized risk assessments

Person 2

Personalized Action Plans:

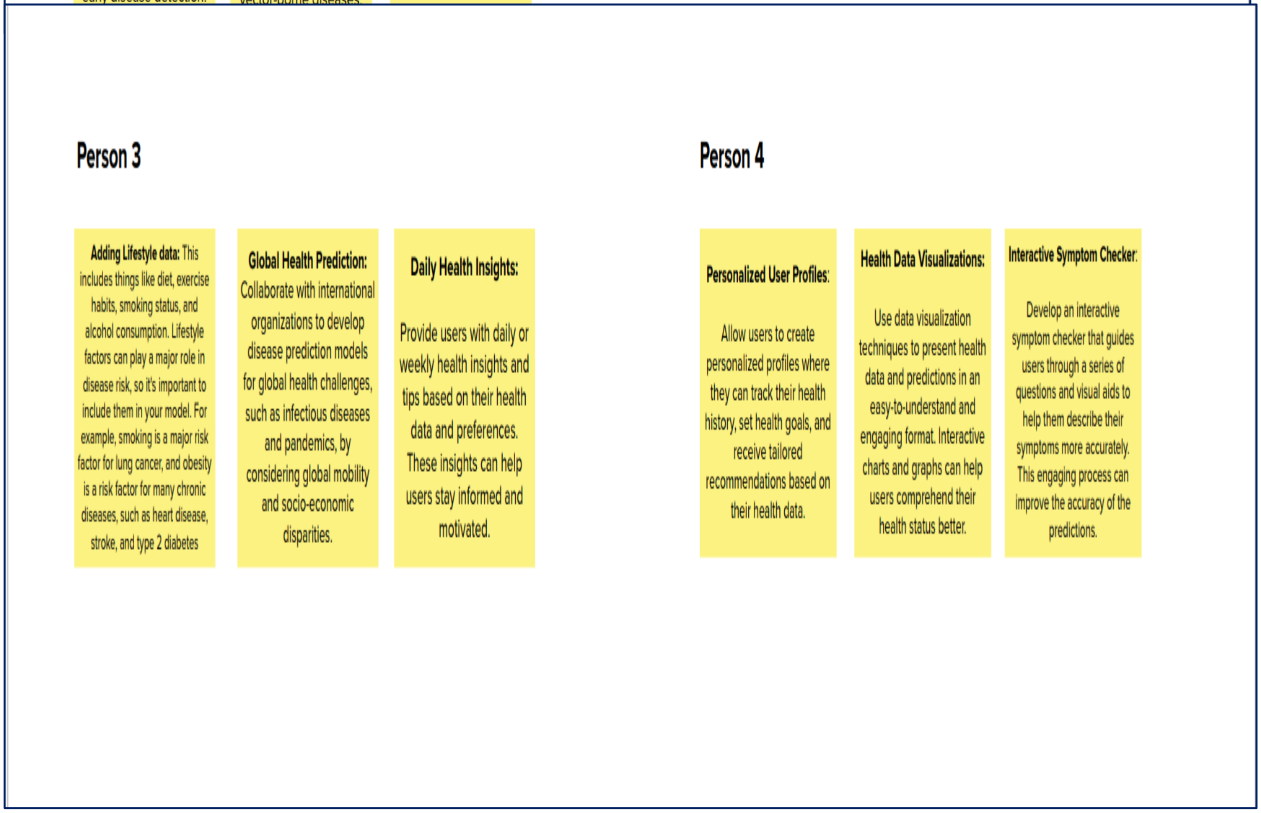
Offer personalized action plans to users based on their health assessments. These plans can include dietary recommendations, exercise routines, and medication reminders.

Artificial Intelligence (AI) in Imaging

Utilize AI algorithms for analyzing medical imaging data, such as X-rays, MRIs, and CT scans, to identify early signs of diseases.

AI-Powered Symptom Image Recognition:

- Integrate image recognition technology for users to upload images of rashes, skin conditions, or other visible symptoms. The AI can analyze these images and provide initial assessments.



Person 3

Adding Lifestyle data: This includes things like diet, exercise habits, smoking status, and alcohol consumption. Lifestyle factors can play a major role in disease risk, so it's important to include them in your model. For example, smoking is a major risk factor for lung cancer, and obesity is a risk factor for many chronic diseases, such as heart disease, stroke, and type 2 diabetes

Global Health Prediction: Collaborate with international organizations to develop disease prediction models for global health challenges, such as infectious diseases and pandemics, by considering global mobility and socio-economic disparities.

Daily Health Insights:

Provide users with daily or weekly health insights and tips based on their health data and preferences. These insights can help users stay informed and motivated.

Person 4

Personalized User Profiles:

Allow users to create personalized profiles where they can track their health history, set health goals, and receive tailored recommendations based on their health data.

Health Data Visualizations:

Use data visualization techniques to present health data and predictions in an easy-to-understand and engaging format. Interactive charts and graphs can help users comprehend their health status better.

Interactive Symptom Checker:

Develop an interactive symptom checker that guides users through a series of questions and visual aids to help them describe their symptoms more accurately. This engaging process can improve the accuracy of the predictions.

REQUIREMENT ANALYSIS

4.1 Functional Requirements:

Functional requirements describe the specific functionalities and features that the disease prediction tool must possess to meet user needs. These requirements focus on what the system should do. Key functional requirements for the disease prediction tool include:

- Symptom Input and Analysis:
    - Users should be able to input a list of symptoms into the tool.
    - The system must analyze the input symptoms using advanced algorithms to predict potential diseases.
- User Registration and Authentication:
    - For online consultations, the tool should allow users to register accounts securely.
    - Implement robust authentication mechanisms to ensure the security and privacy of user accounts.
- Prediction Output:
    - Provide users with clear and concise outputs detailing probable diseases based on input symptoms.
    - Include additional information on each predicted disease to aid user understanding.
- Educational Content:
    - Include informational content alongside disease predictions to educate users about the identified diseases.
    - Deliver relevant information that encourages responsible health practices and discourages self-diagnosis.

4.2 Non-Functional requirements

Non-functional requirements specify the qualities and attributes that the system must possess, such as performance, security, and usability. These requirements focus on how the system should perform. Key non-functional requirements for the disease prediction tool include:

- Performance:
  - The system should respond to user inputs promptly, with predictions generated within a reasonable time frame.
  - Support a large number of concurrent users without significant degradation in performance.
- Scalability:
  - Design the system to be scalable, allowing for an increase in users and data volume over time.
  - Ensure that the infrastructure can handle growth without compromising performance.
- Usability:
  - The user interface should be intuitive and user-friendly, catering to users with varying levels of technological expertise.
  - Provide clear instructions and guidance on how to use the tool effectively.
- Reliability:
  - The system should be highly reliable, with minimal downtime for maintenance or updates.
  - Implement backup and recovery mechanisms to safeguard against data loss.

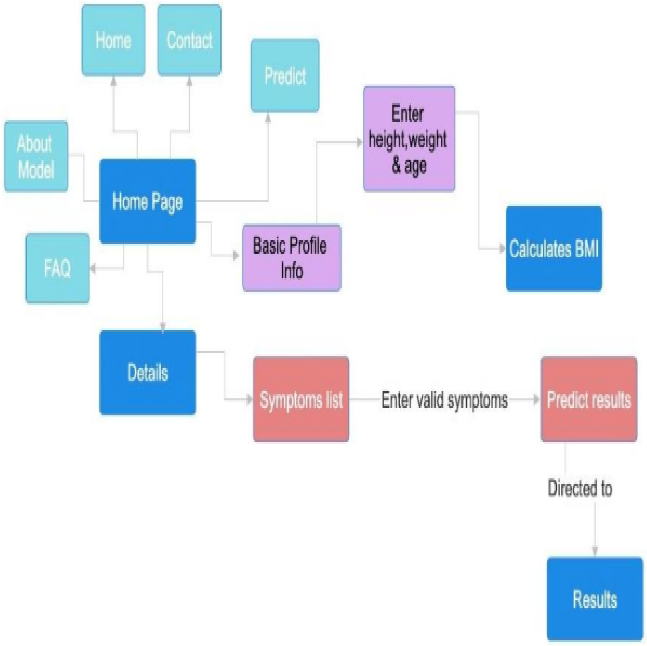
5. PROJECT DESIGN

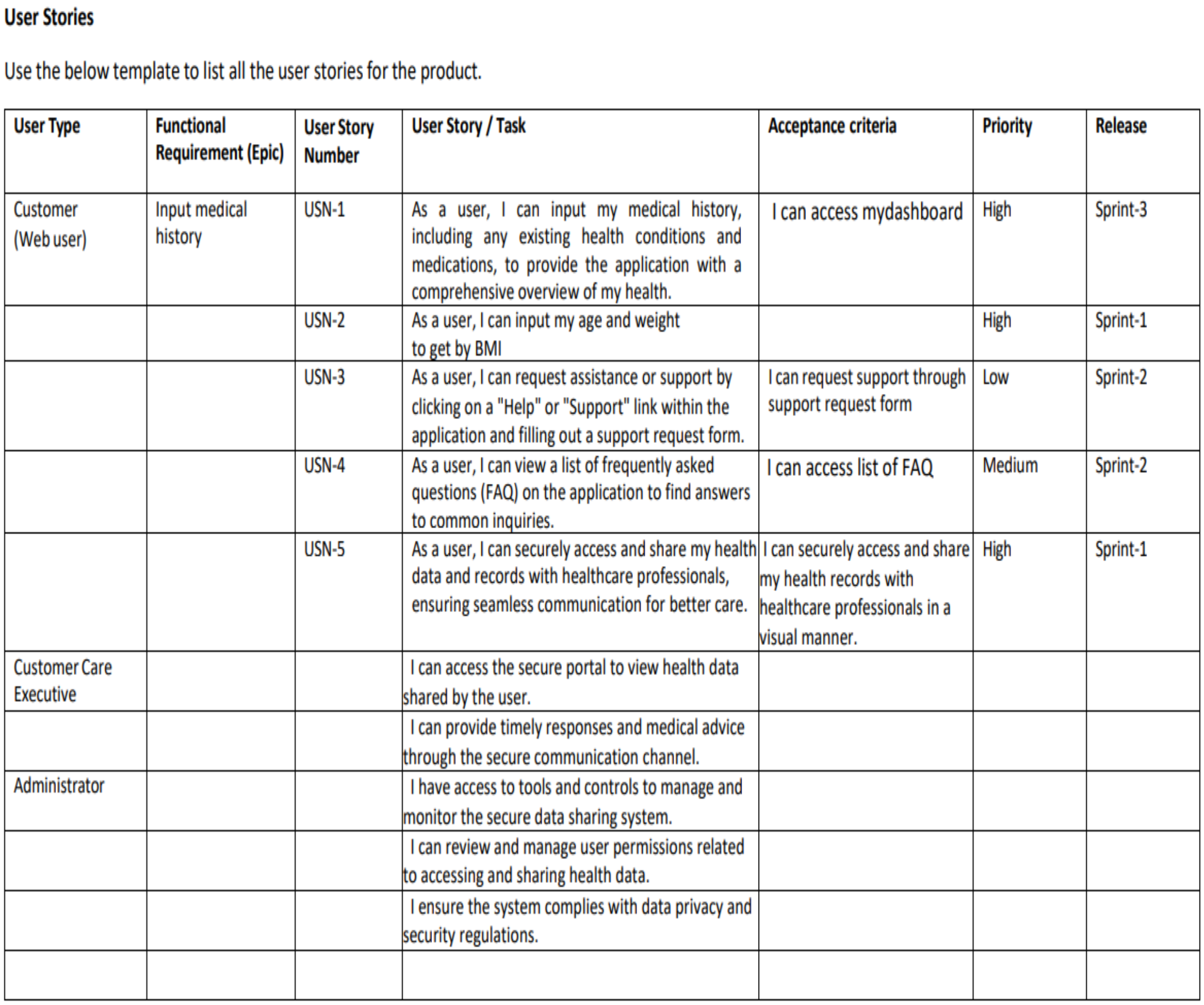
5.1 Data Flow Diagrams & User Stories

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Example:





User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Input medical history	USN-1	As a user, I can input my medical history, including any existing health conditions and medications, to provide the application with a comprehensive overview of my health.	I can access mydashboard	High	Sprint-3
		USN-2	As a user, I can input my age and weight to get by BMI		High	Sprint-1
		USN-3	As a user, I can request assistance or support by clicking on a "Help" or "Support" link within the application and filling out a support request form.	I can request support through support request form	Low	Sprint-2
		USN-4	As a user, I can view a list of frequently asked questions (FAQ) on the application to find answers to common inquiries.	I can access list of FAQ	Medium	Sprint-2
		USN-5	As a user, I can securely access and share my health data and records with healthcare professionals, ensuring seamless communication for better care.	I can securely access and share my health records with healthcare professionals in a visual manner.	High	Sprint-1
Customer Care Executive			I can access the secure portal to view health data shared by the user.			
			I can provide timely responses and medical advice through the secure communication channel.			
Administrator			I have access to tools and controls to manage and monitor the secure data sharing system.			
			I can review and manage user permissions related to accessing and sharing health data.			
			I ensure the system complies with data privacy and security regulations.			

5.2 Solution Architecture

Solution Architecture: Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

• Find the best tech solution to solve existing business problems. • Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.

• Define features, development phases, and solution requirements. • Provide specifications according to which the solution is defined, managed, and delivered.

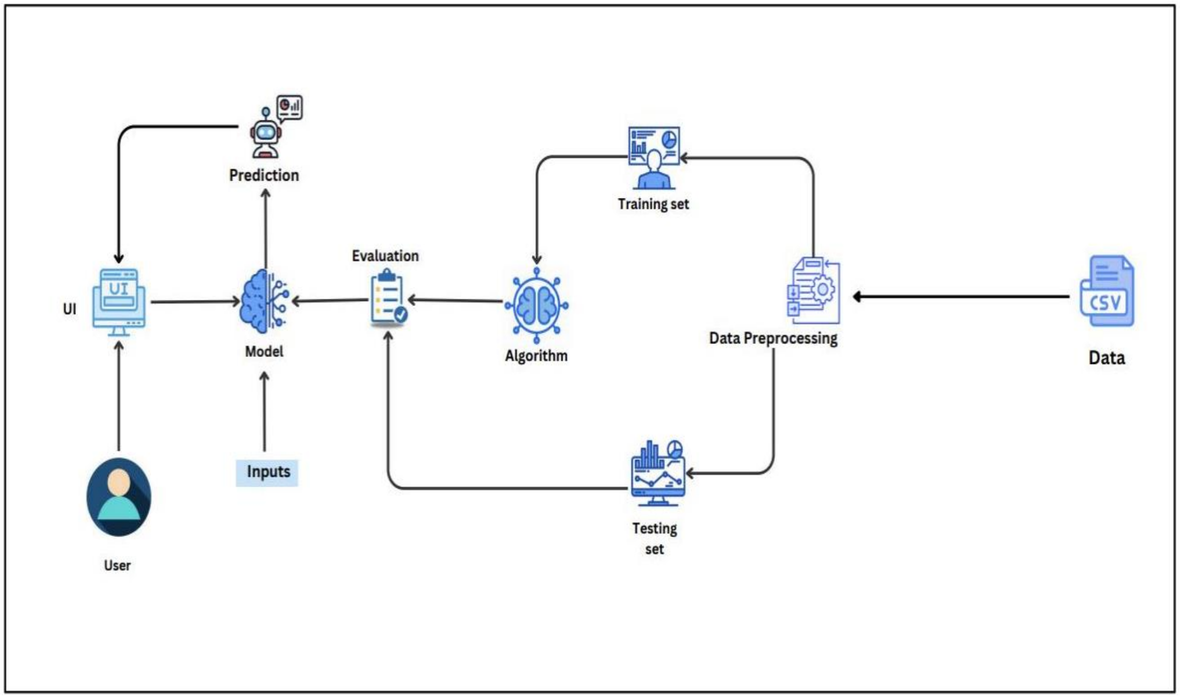
The solution architecture for a disease prediction ML project serves as a crucial link between healthcare challenges and technological resolutions. Its multifaceted goals are outlined below:
Problem Solving: Identify and implement optimal technology solutions to address prevalent healthcare issues, emphasizing disease prediction as a key focus.

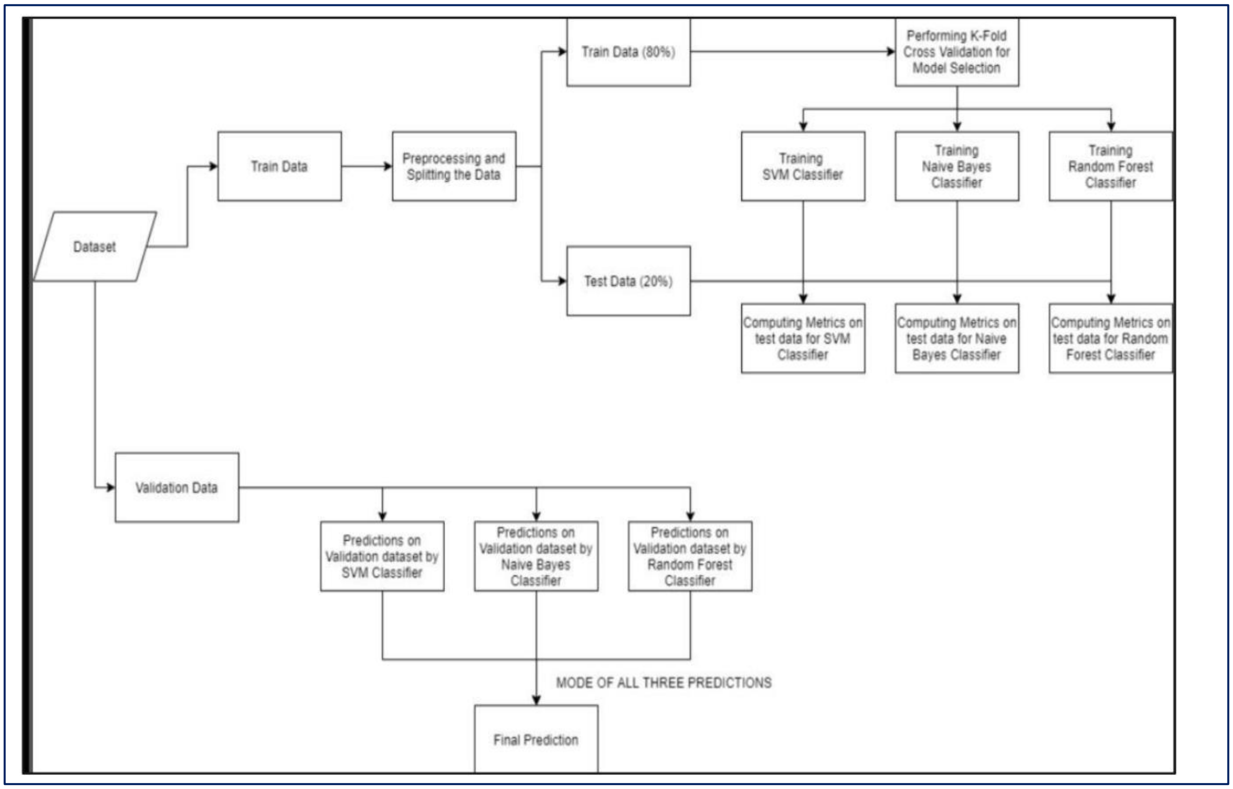
Communication: Effectively describe the architecture's structure, characteristics, and behavior to stakeholders involved in the project, fostering a shared understanding of the technical aspects.

Scope Definition: Clearly define features, development phases, and solution requirements specific to disease prediction, outlining the project's boundaries and objectives.

Specification: Provide comprehensive specifications that serve as guidelines for the definition, management, and delivery of the disease prediction ML solution. This involves detailing the intricacies of data collection, preprocessing, feature extraction, machine learning models, and deployment strategies.

Diagram:

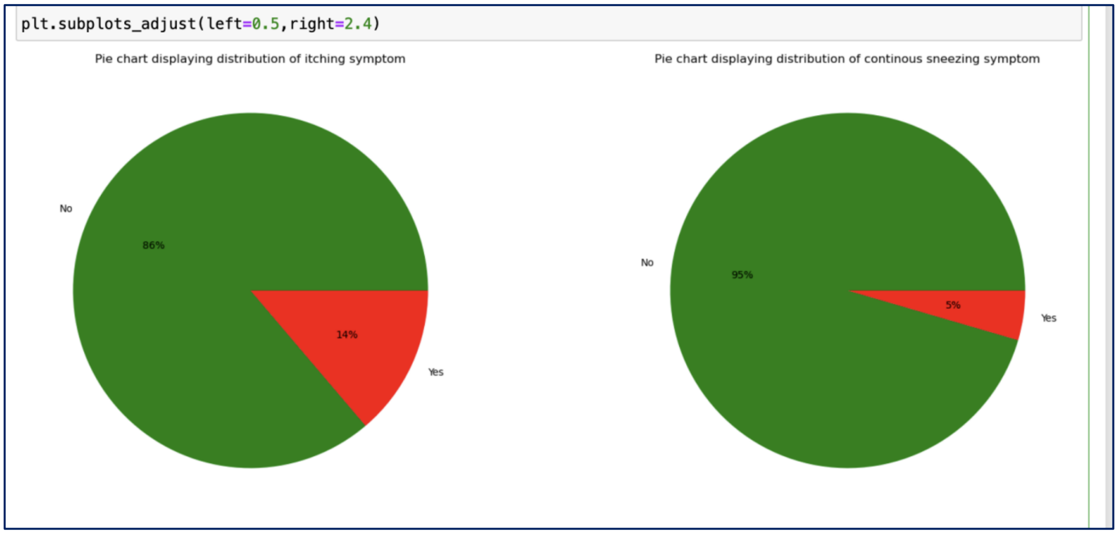

Prediction
Training set
Evaluation
UI
Model
Algorithm
Data Preprocessing
CSV
Data
Inputs
Testing set
User


Train Data (80%)
Performing K-Fold Cross Validation for Model Selection
Train Data
Preprocessing and Splitting the Data
Training SVM Classifier
Training Naive Bayes Classifier
Training Random Forest Classifier
Dataset
Test Data (20%)
Computing Metrics on test data for SVM Classifier
Computing Metrics on test data for Naive Bayes Classifier
Computing Metrics on test data for Random Forest Classifier
Validation Data
Predictions on Validation dataset by SVM Classifier
Predictions on Validation dataset by Naive Bayes Classifier
Predictions on Validation dataset by Random Forest Classifier
MODE OF ALL THREE PREDICTIONS
Final Prediction

Exploratory Data Analysis (EDA)

1. Itching Symptom &Continuous Sneezing Symptom:

Pie Chart:



```
plt.subplots_adjust(left=0.5,right=2.4)
```

Observations:

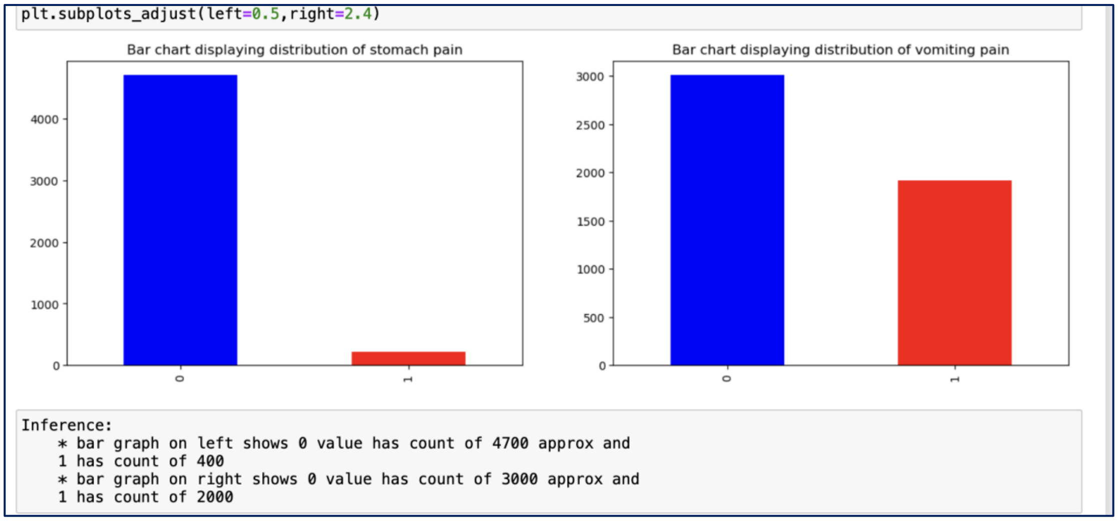
- 86% of the dataset shows the presence of the itching symptom.
- 14% of the dataset does not exhibit the itching symptom.
- 1.2 Inference:
- The majority of observations in the dataset experience itching symptoms.

Observations:

- 95% of the dataset indicates continuous sneezing.
- 5% of the dataset does not have continuous sneezing.
- 2.2 Inference:
- A significant portion of the dataset is marked by continuous sneezing.

2. Stomach Pain Symptom & Vomiting Symptom:

Bar Chart:



```
plt.subplots_adjust(left=0.5,right=2.4)
```

Bar chart displaying distribution of stomach pain
4000
3000
2000
1000
0
0
1
Bar chart displaying distribution of vomiting pain
3000
2500
2000
1500
1000
500
0
0
1

```
Inference:
    * bar graph on left shows 0 value has count of 4700 approx and
    1 has count of 400
    * bar graph on right shows 0 value has count of 3000 approx and
    1 has count of 2000
```

Observations:

- The bar chart indicates that the dataset has approximately 4700 instances without stomach pain (labeled as 0) and around 400 instances with stomach pain (labeled as 1).

Inference:

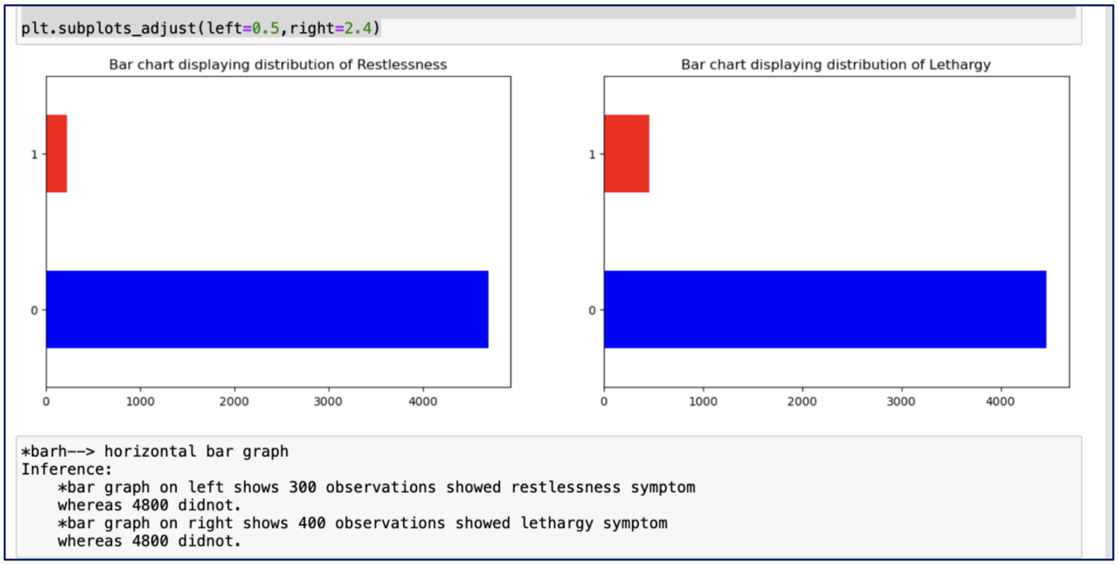
- Stomach pain is less prevalent in the dataset, with a higher number of observations not reporting this symptom

Observations:

- The bar chart displays around 3000 instances without vomiting (labeled as 0) and approximately 2000 instances with vomiting (labeled as 1).
- 4.2 Inference:
- Vomiting seems to be a common symptom, as a considerable number of observations report instances of vomiting.

3 Restlessness Symptom & Lethargy Symptom:

Horizontal Bar Chart:



```
plt.subplots_adjust(left=0.5,right=2.4)
```

Bar chart displaying distribution of Restlessness
1
0
0
1000
2000
3000
4000
Bar chart displaying distribution of Lethargy
1
0
0
1000
2000
3000
4000

```
*barh--> horizontal bar graph
Inference:
    *bar graph on left shows 300 observations showed restlessness symptom
    whereas 4800 didnot.
    *bar graph on right shows 400 observations showed lethargy symptom
    whereas 4800 didnot.
```

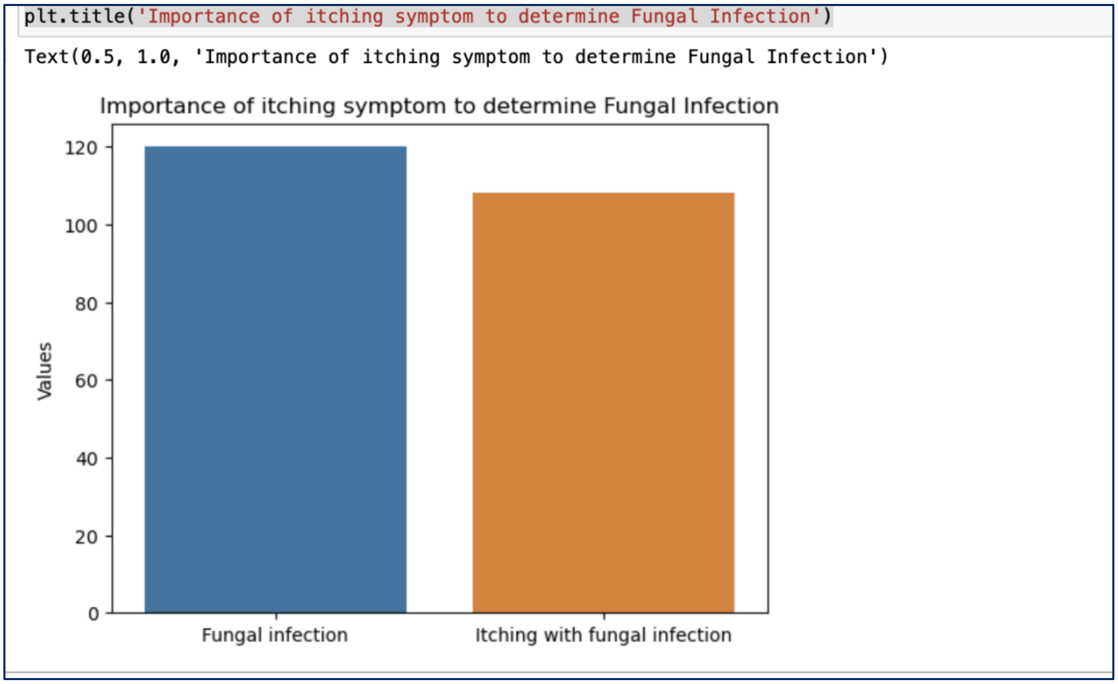
Observations:

- The horizontal bar chart reveals that there are 300 instances where individuals exhibit the restlessness symptom (labeled as 1).
- Approximately 4800 instances do not show the restlessness symptom (labeled as 0).
- 5.2 Inference:
- Restlessness is observed in a relatively small portion of the dataset.

Observations:

- The horizontal bar chart displays that 400 instances report the lethargy symptom (labeled as 1).
- Around 4800 instances do not exhibit the lethargy symptom (labeled as 0).
- 6.2 Inference:
- Lethargy is reported in a portion of the dataset, but its prevalence is lower compared to the absence of lethargy.

4. Fungal Infection and Itching:



```
plt.title('Importance of itching symptom to determine Fungal Infection')
```

```
Text(0.5, 1.0, 'Importance of itching symptom to determine Fungal Infection')
```

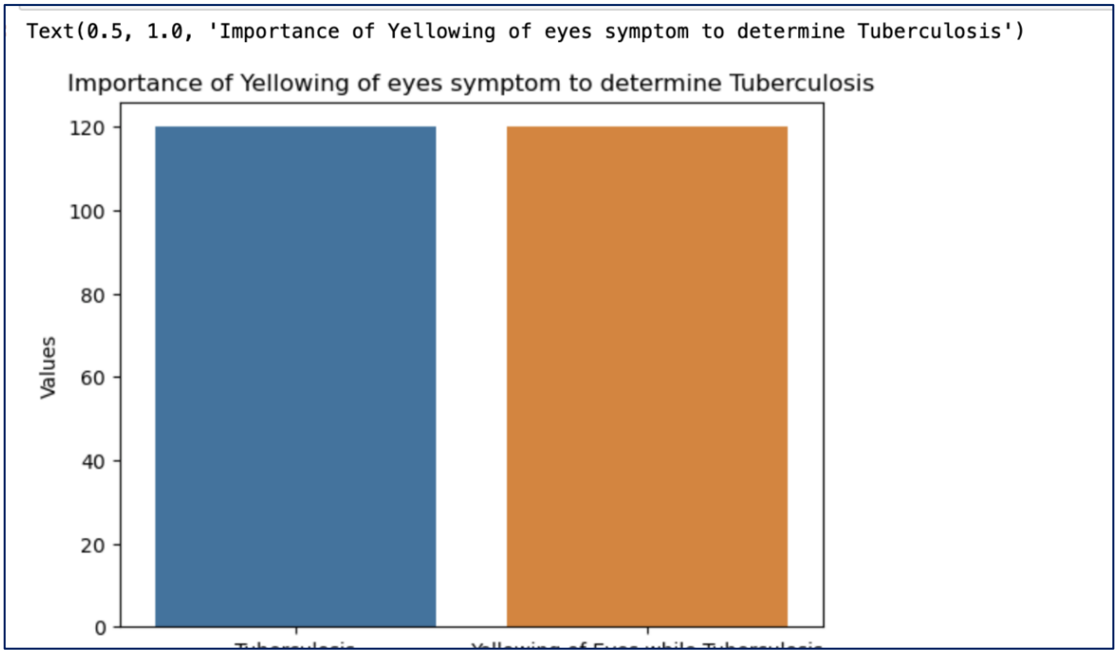
Importance of itching symptom to determine Fungal Infection
Values
0
20
40
60
80
100
120
Fungal infection
Itching with fungal infection

5.

Observations:

- The first bar graph shows the total number of observations facing fungal infection.
- The second bar graph illustrates the number of observations that have both fungal infection and itching.
- 7.2 Inference:
- The majority of observations facing fungal infection also experience itching.
- This suggests that itching is a significant symptom when determining the presence of fungal infection.

5. Tuberculosis and Yellowing of Eyes:



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

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

Observations:

- The first bar graph displays the total number of observations with tuberculosis.
- The second bar graph indicates the number of observations with tuberculosis and yellowing of eyes.
- 8.2 Inference:
- There is a notable number of observations where individuals with tuberculosis also exhibit yellowing of eyes.
- This suggests that yellowing of eyes could be an important symptom in identifying tuberculosis.

Correlation Matrix Visualization:

- A heatmap of the correlation matrix is generated using the Seaborn library.
- Strong correlations are visually represented by color intensity.
- 9.2 Highly Correlated Features:

- A list of feature pairs with correlation coefficients greater than 0.9 is identified.
- This list is printed for reference.
- 9.3 Dropping Highly Correlated Features:
- Features with high correlation coefficients are dropped from the dataset.

Preprocessing & Splitting of dataset into training,testing and validation :

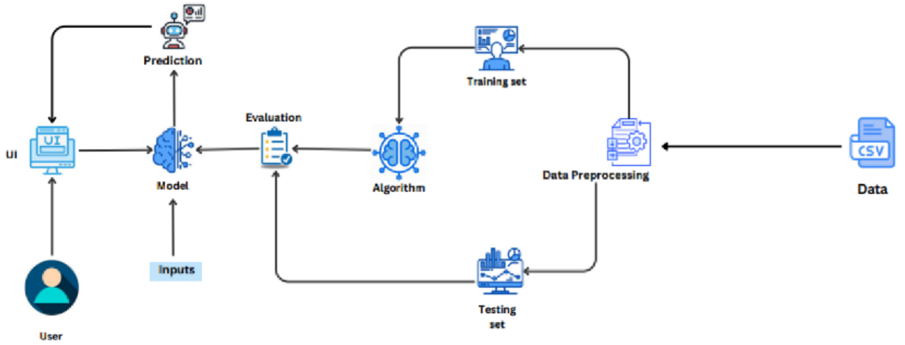
1. Data Preprocessing for Testing Data:
2. The testing data is read from the file path specified in ...
3. A data_preprocessing function is defined to drop specified columns from the dataset.
4. The function is applied to the testing data.

Inference:

- The code ensures consistency in the features between the training and t2. Feature Comparison:
- Extra columns in the training data (df) compared to the testing data (testing_data) are calculated and stored in the variable extra_columns_in_df.
- Extra columns in the testing data compared to the training data are calculated and stored in the variable extra_columns_in_testing_data.
- The results are printed.
- The code checks for discrepancies in feature columns between the training and testing datasets.
- Extra columns in either dataset are identified and printed for further investigation.esting datasets by removing specified columns from the testing data.

6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture



1. **User Input via UI**: The user interacts with the UI by providing symptoms they are experiencing. These symptoms serve as the input data for the disease prediction model.
2. **AI Model Integration**: The AI model, designed specifically for disease prediction, takes these symptom inputs from the UI to initiate the prediction process.
3. **Data Preprocessing**: The AI model first preprocesses the symptom data. When handling null values, the AI model first identifies where data is missing within the dataset. Subsequently, it employs strategies like removing rows or columns with limited missing data or imputing missing values by using statistical measures (mean, median, mode) for numerical data or assigning frequent categories or separate indicators for categorical data. After selecting and implementing the appropriate strategy, the model ensures that the dataset is cleansed of null values, making it suitable for further analysis in the prediction phase.
4. **Algorithm Execution**: The model employs a suitable algorithm that can handle symptom data to predict diseases. For instance, it might use machine learning classification algorithms like decision trees, random forests, or neural networks to analyse the relationship between symptoms and diseases.
5. **Training and Testing**: In the training phase, the model learns from historical data that includes symptoms linked to various diseases. It learns the patterns and associations between specific symptoms and their correlation

with different diseases. Following this, the model is tested using separate data to evaluate its accuracy and predictive capability.

6. **Prediction Generation**: When the user's symptom inputs are processed through the trained AI model, it uses the learned patterns to predict the most likely disease or diseases associated with the symptoms provided by the user.
7. **Output to UI**: The final step involves presenting the predicted disease(s) related to the input symptoms back to the user through the UI. The UI displays the predicted disease(s), providing information that can help the user understand and potentially seek medical advice or further diagnostic tests.

The user interface could then display this information in a user-friendly format, presenting the most probable diseases related to the entered symptoms, thereby assisting the user in understanding potential health issues and seeking appropriate medical guidance.

6.2 Sprint Planning & Estimation

Project Tracker

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	4 Days	25 Oct 2023	29 Oct 2023	20	29 Oct 2023
Sprint-2	20	4 Days	31 Oct 2023	03 Nov 2023	20	03 Nov 2023
Sprint-3	20	6 Days	06 Nov 2023	12 Nov 2023	20	12 Nov 2023

The table encapsulates crucial details regarding multiple sprints within an agile project framework. In Sprint-1, a workload of 20 story points was estimated, spanning 4 days from October 25th to October 29th. Impressively, the team completed all 20 story points as planned and successfully released the sprint deliverables on October 29th. Sprint-2 mirrored a similar structure, starting on October 31st and concluding on November 3rd, accomplishing the anticipated 20 story points by the planned end date and releasing on November 3rd. Sprint-3, with a slightly extended duration of 6 days from November 6th to November 12th, also met its 20-story point target within the scheduled timeframe and was released on November 12th. This succinct representation emphasizes the adherence to planned story points, durations, and successful releases for each sprint, showcasing the team's efficiency in meeting objectives within the established timelines.

6.3 Sprint Delivery Schedule

1. Sprint-1:
    - Start Date: October 25th, 2023
    - End Date: October 29th, 2023
    - Duration: 4 days
2. Sprint-2:
    - Start Date: October 31st, 2023
    - End Date: November 3rd, 2023
    - Duration: 4 days
3. Sprint-3:
    - Start Date: November 6th, 2023
    - End Date: November 12th, 2023
    - Duration: 6 days

This represents the Sprint Delivery Schedule for the given sprints, detailing the start and end dates for each sprint and their respective durations within the project timeline.

The Sprint Delivery Schedule for the provided input encompasses three distinct sprints: Sprint-1, Sprint-2, and Sprint-3. Sprint-1 commenced on October 25th, 2023, and concluded on October 29th, 2023, maintaining a duration of four days. Similarly, Sprint-2 initiated on October 31st, 2023, and finalized its tasks on November 3rd, 2023, also spanning four days. Sprint-3, however, had a slightly extended duration, starting on November 6th, 2023, and ending on November 12th, 2023, lasting a total of six days. Each sprint's start and end dates, along with their respective durations, are vital components of the Sprint Delivery Schedule. This schedule serves as a roadmap, delineating specific timeframes within which the team focuses on accomplishing predetermined tasks and delivering completed work or features, facilitating effective planning, monitoring, and assessment of progress and project timelines.

7.Coding and Solutioning

Feature 1: Dynamic Symptom Information Display:
Implementation:

- In the "Symptom Details" section, you dynamically display information about symptoms. Users can search for specific symptoms, and the matching symptoms are highlighted and displayed in yellow.
- This feature enhances user experience by providing relevant information about symptoms in real-time as the user types.

Benefits:

- User-Friendly: Users can quickly find information about specific symptoms without scrolling through a long list.

- Enhanced Interactivity: The dynamic search and highlight functionality makes it easy for users to navigate and find the information they need.
- Efficient Design: The layout and presentation of symptom details are clear and concise, contributing to a positive user experience.

Feature 2: Multi-Input Prediction Form:
Implementation:

- The prediction form in the "Details" section allows users to input symptoms through checkboxes and text fields.
- Users can choose symptoms from predefined checkboxes or input their own symptoms in text fields.

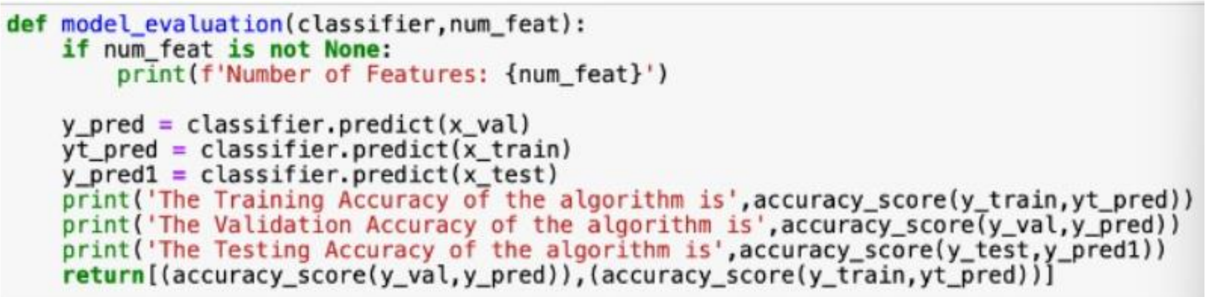
Benefits:

- User Flexibility: The inclusion of text inputs accommodates scenarios where users might experience symptoms not covered by predefined checkboxes.
- Improved Prediction Accuracy: The model considers a variety of symptoms, enhancing the accuracy of disease predictions based on the user's input.

8. PERFORMANCE TESTING :

8.1 Performance Metrics

MODEL BUILDING

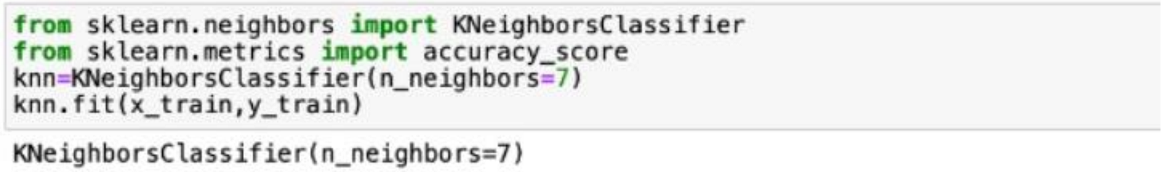


```
def model_evaluation(classifier,num_feat):
    if num_feat is not None:
        print(f'Number of Features: {num_feat}')

    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_val,y_pred)),(accuracy_score(y_train,yt_pred))]
```

The model_evaluation function evaluates a classifier's accuracy on different datasets (training, validation, testing) in machine learning. It calculates and shows how well the model predicts outcomes and optionally displays the number of features used. This function provides insights into the model's performance and returns the accuracy scores for validation and training sets.

Training and testing the models using multiple algorithms.

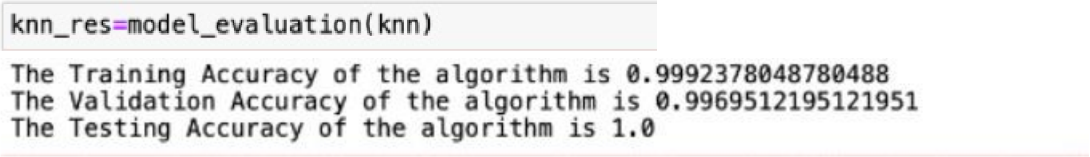
1) K Nearest Neighbors Model.



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
knn=KNeighborsClassifier(n_neighbors=7)
knn.fit(x_train,y_train)
```

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

This code snippet creates a K-Nearest Neighbors (KNN) classifier with a setting of 7 neighbors. It trains this classifier using the provided training data (x_train for features and y_train for corresponding labels). The trained model is stored in the variable knn and can be used to make predictions on new data.

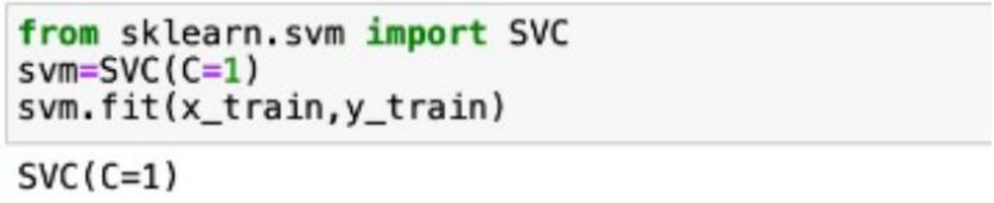


```
knn_res=model_evaluation(knn)
```

```
The Training Accuracy of the algorithm is 0.9992378048780488
The Validation Accuracy of the algorithm is 0.9969512195121951
The Testing Accuracy of the algorithm is 1.0
```

knn_res = model_evaluation(knn) calls the function model_evaluation with the K-Nearest Neighbors (KNN) classifier knn as an argument. This function likely evaluates the performance of the KNN classifier by computing and displaying its accuracy on different datasets (training, validation, and testing) and possibly returns some evaluation results stored in the variable knn_res.

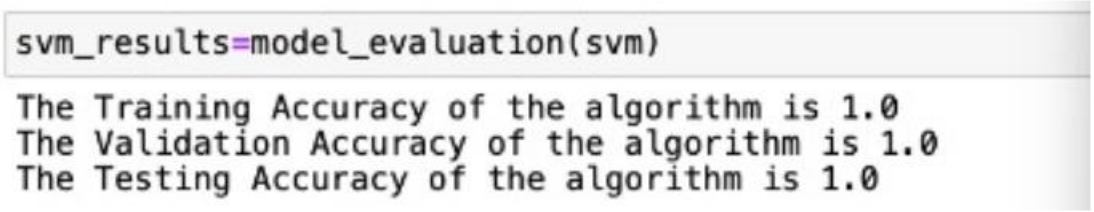
2) SVM Model



```
from sklearn.svm import SVC
svm=SVC(C=1)
svm.fit(x_train,y_train)
```

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

The code snippet initializes a Support Vector Machine (SVM) classifier using scikit-learn's SVC module, setting the regularization parameter C to 1. It then trains this SVM classifier with the x_train data (input features) and y_train data (corresponding labels), enabling the model to learn patterns and create a decision boundary for making predictions based on the provided training data.

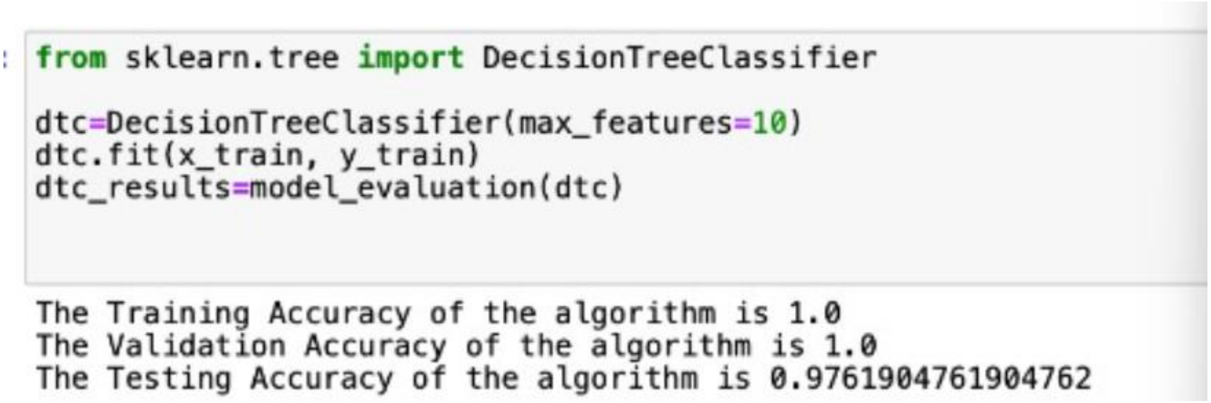


```
svm_results=model_evaluation(svm)
```

```
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
```

svm_results = model_evaluation(svm) employs a function named model_evaluation to assess the Support Vector Machine (SVM) classifier (svm) by computing its accuracy on various datasets (such as training, validation, and testing). It probably returns evaluation results regarding the classifier's performance, storing these results in the variable svm_results.

3) Decision Tree Model

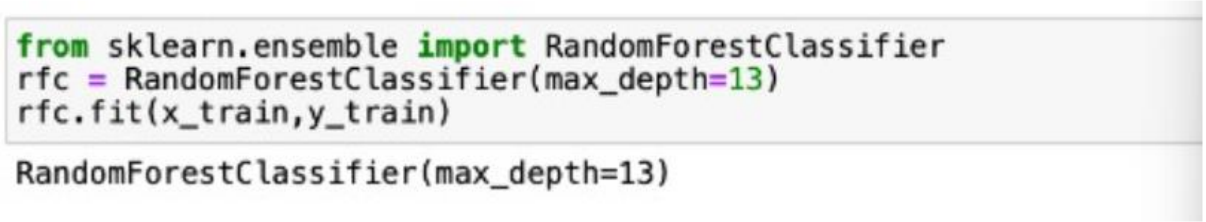


```
from sklearn.tree import DecisionTreeClassifier

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

Uses scikit-learn's Decision Tree Classifier (DecisionTreeClassifier) to create a decision tree model with a maximum of 10 features for splitting nodes. It trains the classifier using the x_train and y_train datasets. The variable dtc_results likely holds evaluation results obtained by assessing the performance of the decision tree model using a function named model_evaluation. This function probably computes accuracy scores for different datasets (such as training, validation, and testing) and returns these evaluation metrics.

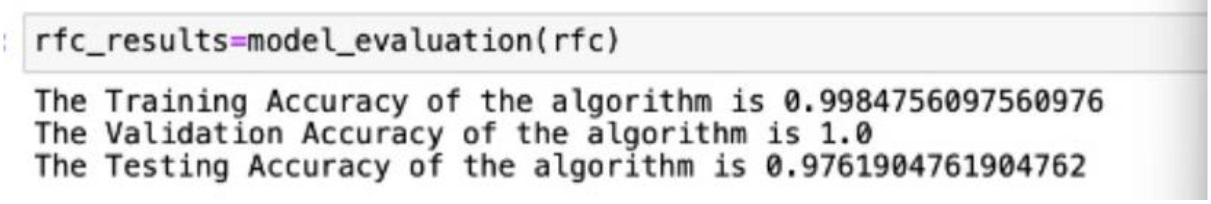
4) Random Forest Model



```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(max_depth=13)
rfc.fit(x_train,y_train)
```

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

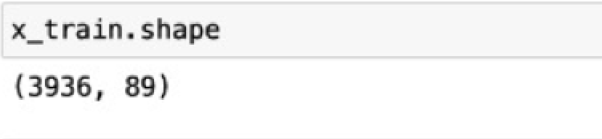
Utilizes scikit-learn's RandomForestClassifier to create a Random Forest model, setting the maximum depth of each tree in the forest to 13. It then proceeds to train this Random Forest classifier using the x_train and y_train datasets. The method .fit() trains the model, enabling it to learn patterns and relationships within the training data to make predictions.



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

```
The Training Accuracy of the algorithm is 0.9984756097560976
The Validation Accuracy of the algorithm is 1.0
The Testing Accuracy of the algorithm is 0.9761904761904762
```

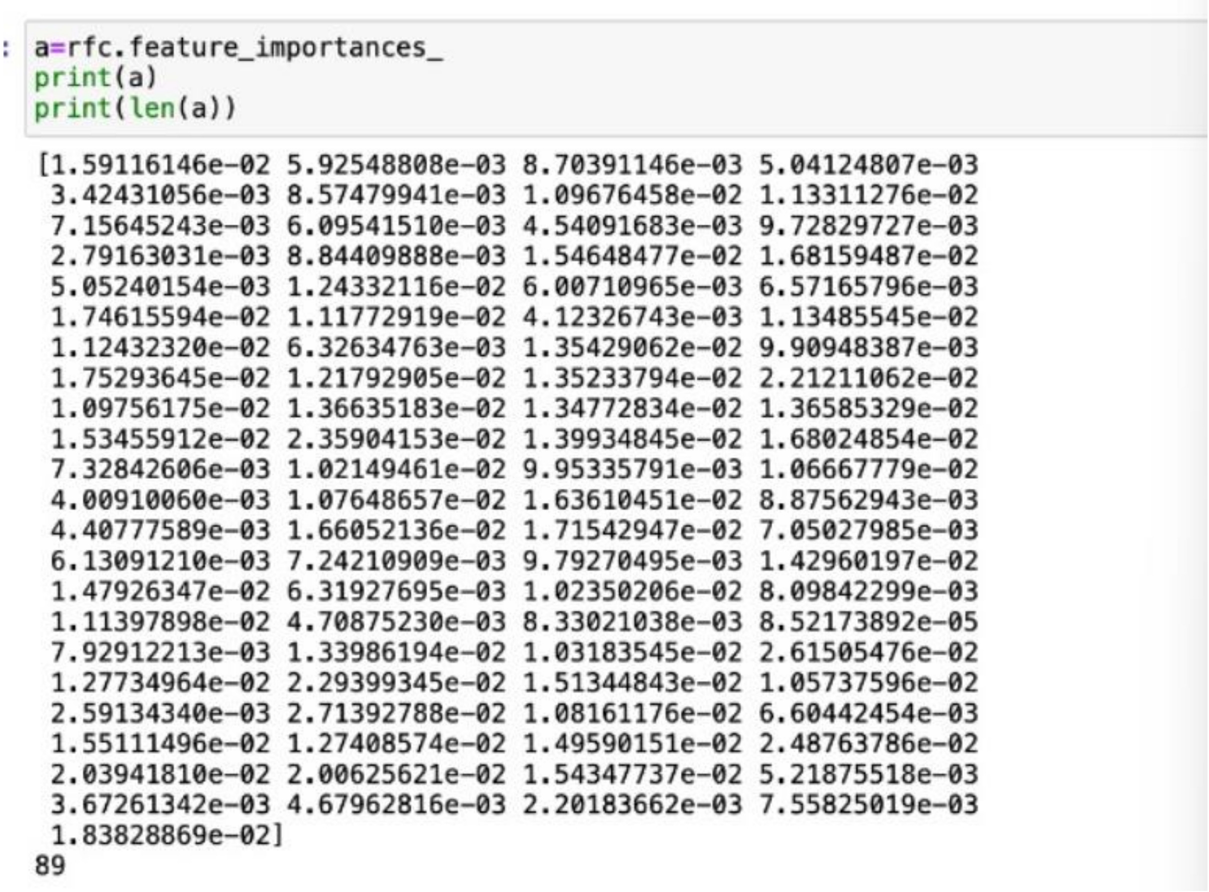
Performance Testing & Hyperparameter Tuning



```
x_train.shape
```

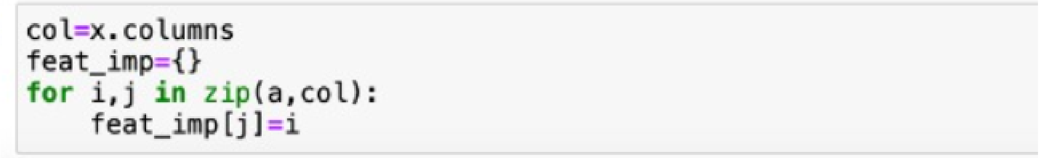
```
(3936, 89)
```

The expression x_train.shape retrieves the shape or dimensions of the training dataset (x_train). Specifically, it provides information about the number of rows (samples or instances) and columns (features or attributes) present in the training dataset. This information indicates the size or structure of the input data used for training machine learning models, aiding in understanding the dataset's composition and the number of features available for model learning.



```
: a=rfc.feature_importances_
  print(a)
  print(len(a))

  [1.59116146e-02 5.92548808e-03 8.70391146e-03 5.04124807e-03
   3.42431056e-03 8.57479941e-03 1.09676458e-02 1.13311276e-02
   7.15645243e-03 6.09541510e-03 4.54091683e-03 9.72829727e-03
   2.79163031e-03 8.84409888e-03 1.54648477e-02 1.68159487e-02
   5.05240154e-03 1.24332116e-02 6.00710965e-03 6.57165796e-03
   1.74615594e-02 1.11772919e-02 4.12326743e-03 1.13485545e-02
   1.12432320e-02 6.32634763e-03 1.35429062e-02 9.90948387e-03
   1.75293645e-02 1.21792905e-02 1.35233794e-02 2.21211062e-02
   1.09756175e-02 1.36635183e-02 1.34772834e-02 1.36585329e-02
   1.53455912e-02 2.35904153e-02 1.39934845e-02 1.68024854e-02
   7.32842606e-03 1.02149461e-02 9.95335791e-03 1.06667779e-02
   4.00910060e-03 1.07648657e-02 1.63610451e-02 8.87562943e-03
   4.40777589e-03 1.66052136e-02 1.71542947e-02 7.05027985e-03
   6.13091210e-03 7.24210909e-03 9.79270495e-03 1.42960197e-02
   1.47926347e-02 6.31927695e-03 1.02350206e-02 8.09842299e-03
   1.11397898e-02 4.70875230e-03 8.33021038e-03 8.52173892e-05
   7.92912213e-03 1.33986194e-02 1.03183545e-02 2.61505476e-02
   1.27734964e-02 2.29399345e-02 1.51344843e-02 1.05737596e-02
   2.59134340e-03 2.71392788e-02 1.08161176e-02 6.60442454e-03
   1.55111496e-02 1.27408574e-02 1.49590151e-02 2.48763786e-02
   2.03941810e-02 2.00625621e-02 1.54347737e-02 5.21875518e-03
   3.67261342e-03 4.67962816e-03 2.20183662e-03 7.55825019e-03
   1.83828869e-02]
  89
```

The variable a holds the feature importances computed by the Random Forest Classifier (rfc). These importances indicate the relevance or contribution of each feature in making predictions. Printing a displays the feature importances calculated by the classifier. The len(a) function determines and prints the number of elements in the a array, indicating the total count of feature importances obtained from the Random Forest model. This count represents the number of features considered and assessed for their significance in the predictive capability of the model.

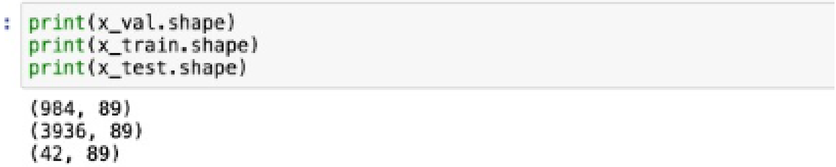


```
col=x.columns
feat_imp={}
for i,j in zip(a,col):
    feat_imp[j]=i
```

Code segment assigns the column names of the dataset x to the variable col. It utilizes a zip function to iterate through the elements of the a array (presumably containing feature importances) and the col array (representing column names). For each pair of elements, it associates the feature names (j) with their respective importance values (i), creating a dictionary named feat_imp. This dictionary is structured to store feature names as keys and their corresponding importance values, likely derived from a machine learning model's feature importance analysis.

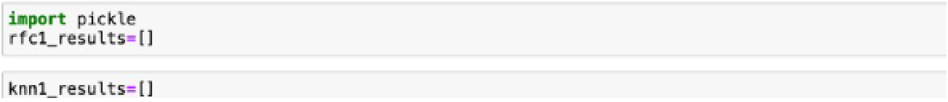
```
feat_imp
```

feat_imp is a dictionary that links feature names to their importance values. These values show the relevance of each feature in a machine learning model's predictions.



```
print(x_val.shape)
print(x_train.shape)
print(x_test.shape)

(984, 89)
(3936, 89)
(42, 89)
```

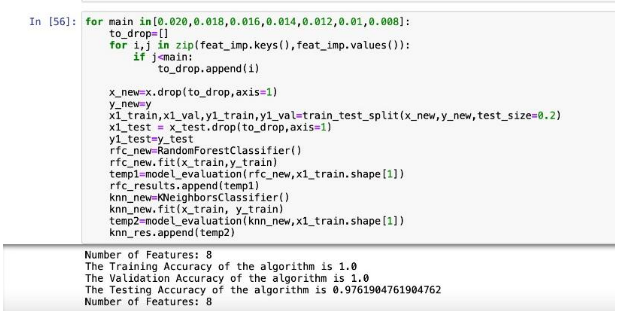
These lines of code print out the shapes or dimensions of three datasets: x_val, x_train, and x_test.



```
import pickle
rfc1_results=[]

knn1_results=[]
```

The code initializes empty lists rfc1_results and knn1_results. The purpose of these lists appears to be for storing or appending results or metrics, possibly from the evaluation of different models such as Random Forest Classifier (rfc) and K-Nearest Neighbors (knn) for further analysis or comparison. The code likely intends to store evaluation outcomes or specific results in these lists using some methodology not yet implemented in this code snippet.

rfc_results = model_evaluation(rfc) likely calls a function named **model_evaluation** to evaluate the performance of the Random Forest Classifier (**rfc**). This function likely computes and presents accuracy metrics for the classifier across various datasets (such as training, validation, and testing). The variable **rfc_results** probably stores the evaluation outcomes, providing insights into the Random Forest model's accuracy on different data subsets.



```
In [56]: 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
             rfc_new=RandomForestClassifier()
             rfc_new.fit(x_train,y_train)
             temp1=model_evaluation(rfc_new,x1_train.shape[1])
             rfc_results.append(temp1)
             knn_new=KNeighborsClassifier()
             knn_new.fit(x_train, y_train)
             temp2=model_evaluation(knn_new,x1_train.shape[1])
             knn_res.append(temp2)
```

```
Number of Features: 8
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
Number of Features: 8
```

This loop iterates over a list of threshold values stored in the variable main. These thresholds are used to filter features based on their importance.

Within the loop, another loop iterates over the keys and values of feat_imp. If the importance j of a feature is less than the current threshold main, the feature is added to the list to_drop.

The features with importance below the threshold are dropped from the input data x. The filtered data is stored in x_new. The target variable y remains unchanged.

The filtered data (x_new and y_new) is split into training (x1_train, y1_train) and validation (x1_val, y1_val) sets using the train_test_split function.

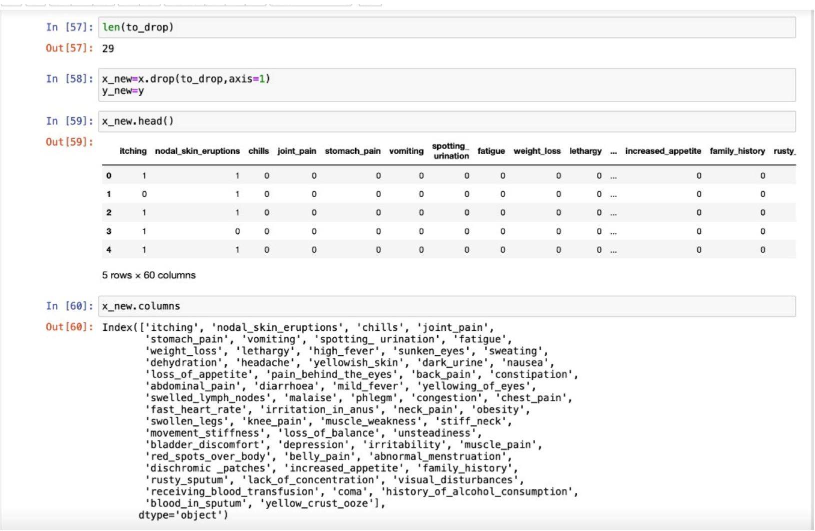
The same feature filtering is applied to the test data (x_test). The filtered test data is stored in x1_test. The target variable y1_test remains unchanged.

A new RandomForestClassifier is instantiated and trained on the training data.

The model_evaluation function is then called to evaluate the model's performance, and the result is appended to the rfc_results list.

Similarly, a new KNeighborsClassifier is instantiated and trained on the training data. The model's performance is evaluated using the model_evaluation function, and the result is appended to the knn_res list.

The code seems to be iterating over different thresholds, filtering features based on importance, and then training and evaluating two classifiers (Random Forest and K-Nearest Neighbors) for each threshold. The evaluation results are stored in rfc_results and knn_res lists.



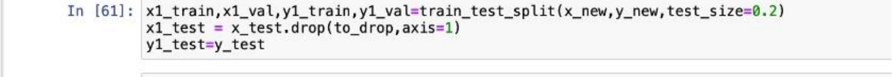
```
In [57]: len(to_drop)
Out[57]: 29

In [58]: x_new=x.drop(to_drop,axis=1)
         y_new=y

In [59]: x_new.head()
Out[59]:
```

	itching	nodal_skin_eruptions	chills	joint_pain	stomach_pain	vomiting	spotting_ urination	fatigue	weight_loss	lethargy	...	increased_appetite	family_history	rusty_
0	1	1	0	0	0	0	0	0	0	0	...	0	0	
1	0	1	0	0	0	0	0	0	0	0	...	0	0	
2	1	1	0	0	0	0	0	0	0	0	...	0	0	
3	1	0	0	0	0	0	0	0	0	0	...	0	0	
4	1	1	0	0	0	0	0	0	0	0	...	0	0	

5 rows × 60 columns

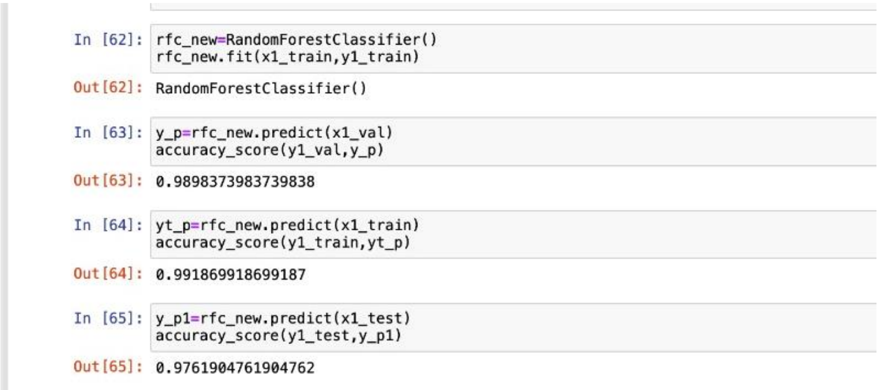
```
In [60]: x_new.columns
Out[60]: Index(['itching', 'nodal_skin_eruptions', 'chills', 'joint_pain',
                'stomach_pain', 'vomiting', 'spotting_ urination', 'fatigue',
                'weight_loss', 'lethargy', 'high_fever', 'sunken_eyes', 'sweating',
                'dehydration', 'headache', 'yellowish_skin', 'dark_urine', 'nausea',
                'loss_of_appetite', 'pain_behind_the_eyes', 'back_pain', 'constipation',
                'abdominal_pain', 'diarrhoea', 'mild_fever', 'yellowing_of_eyes',
                'swelled_lymph_nodes', 'malaise', 'phlegm', 'congestion', 'chest_pain',
                'fast_heart_rate', 'irritation_in_anus', 'neck_pain', 'obesity',
                'swollen_legs', 'knee_pain', 'muscle_weakness', 'stiff_neck',
                'movement_stiffness', 'loss_of_balance', 'unsteadiness',
                'bladder_discomfort', 'depression', 'irritability', 'muscle_pain',
                'red_spots_over_body', 'belly_pain', 'abnormal_menstruation',
                'dischromic _patches', 'increased_appetite', 'family_history',
                'rusty_sputum', 'lack_of_concentration', 'visual_disturbances',
                'receiving_blood_transfusion', 'coma', 'history_of_alcohol_consumption',
                'blood_in_sputum', 'yellow_crust_ooze'],
               dtype='object')
```

This part of the code is focused on handling the feature importance filtering. It calculates the number of features to be dropped, creates a new DataFrame (**x_new**) by removing those features, and then displays the first few rows and column names of the filtered DataFrame for inspection.



```
In [61]: 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
```

This part of the code is responsible for splitting the filtered training data into training and validation sets (x1_train, y1_train, x1_val, y1_val), and filtering the test data in a manner consistent with the feature importance criteria (x1_test, y1_test). These splits are typically done to train a machine learning model on the training set and evaluate its performance on the validation set before applying it to the test set.



```
In [62]: rfc_new=RandomForestClassifier()
         rfc_new.fit(x1_train,y1_train)

Out[62]: RandomForestClassifier()

In [63]: y_p=rfc_new.predict(x1_val)
         accuracy_score(y1_val,y_p)

Out[63]: 0.9898373983739838

In [64]: yt_p=rfc_new.predict(x1_train)
         accuracy_score(y1_train,yt_p)

Out[64]: 0.991869918699187

In [65]: y_p1=rfc_new.predict(x1_test)
         accuracy_score(y1_test,y_p1)

Out[65]: 0.9761904761904762
```

This code is focused on training a RandomForestClassifier and evaluating its performance on three different datasets: validation set (x1_val, y1_val), training set (x1_train, y1_train), and a separate test set (x1_test, y1_test).

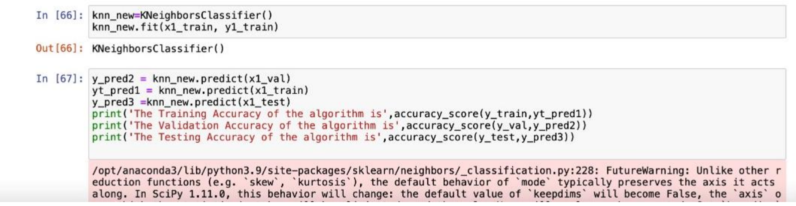
A new instance of the RandomForestClassifier is created, which is a popular machine learning algorithm used for classification tasks.

The RandomForestClassifier on the filtered training set (x1_train, y1_train). The model is fitted to the training data, and its parameters are adjusted to make predictions on this data.

Predictions (y_p) are generated using the trained RandomForestClassifier on the validation set (x1_val). The accuracy_score function is then used to calculate the accuracy of the model's predictions by comparing them to the true labels (y1_val).

Similar to the validation set, this part generates predictions (yt_p) on the training set (x1_train) and calculates the accuracy of the model on the training data.

The code trains a RandomForestClassifier on the filtered training data and evaluates its performance on three different datasets (validation, training, and test) using accuracy as the evaluation metric. Accuracy is a common metric for classification tasks and represents the ratio of correctly predicted instances to the total instances.



```
In [66]: knn_new=KNeighborsClassifier()
         knn_new.fit(x1_train, y1_train)

Out[66]: KNeighborsClassifier()

In [67]: y_pred2 = knn_new.predict(x1_val)
         yt_pred1 = knn_new.predict(x1_train)
         y_pred3 =knn_new.predict(x1_test)
         print('The Training Accuracy of the algorithm is',accuracy_score(y_train,yt_pred1))
         print('The Validation Accuracy of the algorithm is',accuracy_score(y_val,y_pred2))
         print('The Testing Accuracy of the algorithm is',accuracy_score(y_test,y_pred3))

         /opt/anaconda3/lib/python3.9/site-packages/sklearn/neighbors/_classification.py:228: FutureWarning: Unlike other r
         eduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts
         along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, the `axis` o
```

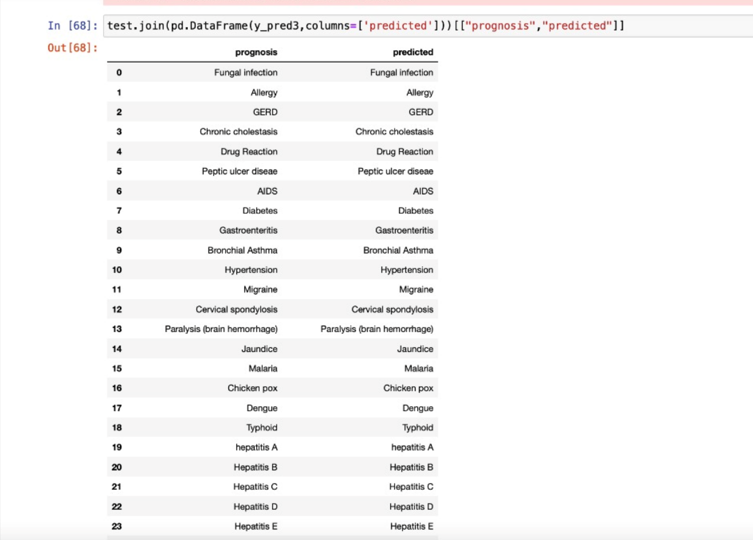
It creates a new instance of the KNeighborsClassifier, which is a machine learning algorithm used for classification tasks based on the principle of finding the k-nearest data points in the feature space.

It trains the KNN classifier on the filtered training set (x1_train, y1_train). The model is fitted to the training data, and its parameters are adjusted to make predictions on this data.

These lines generate predictions (y_pred2) using the trained KNN classifier on the validation set (x1_val). The accuracy_score function is then used to calculate the accuracy of the model's predictions by comparing them to the true labels (y1_val).

Similar to the validation set, this part generates predictions (yt_pred1) on the training set (x1_train) and calculates the accuracy of the KNN model on the training data.

Finally, they generate predictions (y_pred3) on the separate test set (x1_test) and calculate the accuracy of the KNN model on the test data.



```
In [68]: test.join(pd.DataFrame(y_pred3,columns=['predicted']))[["prognosis","predicted"]]
Out[68]:
```

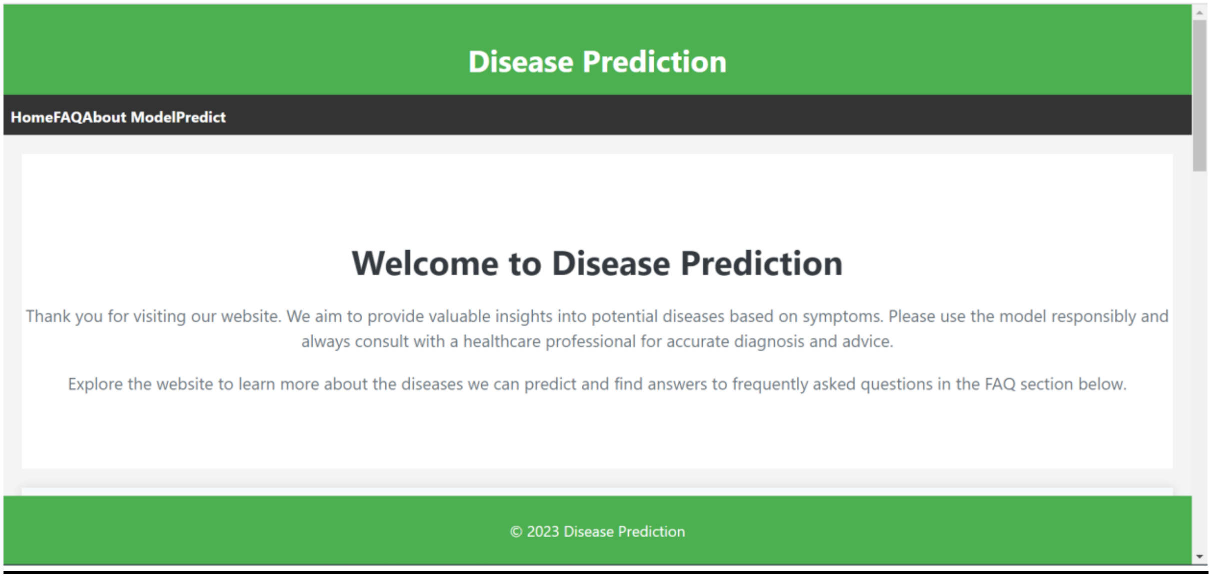
	prognosis	predicted
0	Fungal infection	Fungal infection
1	Allergy	Allergy
2	GERD	GERD
3	Chronic cholestasis	Chronic cholestasis
4	Drug Reaction	Drug Reaction
5	Peptic ulcer diseae	Peptic ulcer diseae
6	AIDS	AIDS
7	Diabetes	Diabetes
8	Gastroenteritis	Gastroenteritis
9	Bronchial Asthma	Bronchial Asthma
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
17	Dengue	Dengue
18	Typhoid	Typhoid
19	hepatitis A	hepatitis A
20	Hepatitis B	Hepatitis B
21	Hepatitis C	Hepatitis C
22	Hepatitis D	Hepatitis D
23	Hepatitis E	Hepatitis E

The code is creating a new DataFrame that includes the original 'prognosis' column from the test DataFrame and the predicted values under a new column named 'predicted'. This can be useful for comparing the actual prognosis values with the predicted values in a side-by-side manner. The result is a DataFrame with only the selected columns

9. RESULTS

9.1 Output Screenshots

Home Page:



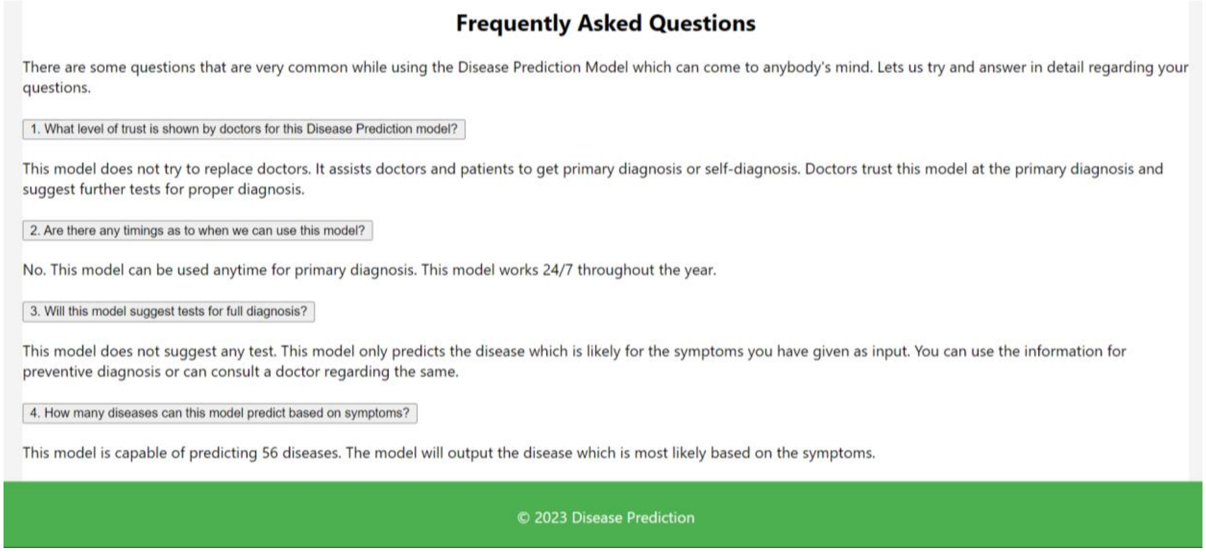
Welcome to Disease Prediction

Thank you for visiting our website. We aim to provide valuable insights into potential diseases based on symptoms. Please use the model responsibly and always consult with a healthcare professional for accurate diagnosis and advice.

Explore the website to learn more about the diseases we can predict and find answers to frequently asked questions in the FAQ section below.

© 2023 Disease Prediction

FAQ



Frequently Asked Questions

There are some questions that are very common while using the Disease Prediction Model which can come to anybody's mind. Lets us try and answer in detail regarding your questions.

1. What level of trust is shown by doctors for this Disease Prediction model?

This model does not try to replace doctors. It assists doctors and patients to get primary diagnosis or self-diagnosis. Doctors trust this model at the primary diagnosis and suggest further tests for proper diagnosis.

2. Are there any timings as to when we can use this model?

No. This model can be used anytime for primary diagnosis. This model works 24/7 throughout the year.

3. Will this model suggest tests for full diagnosis?

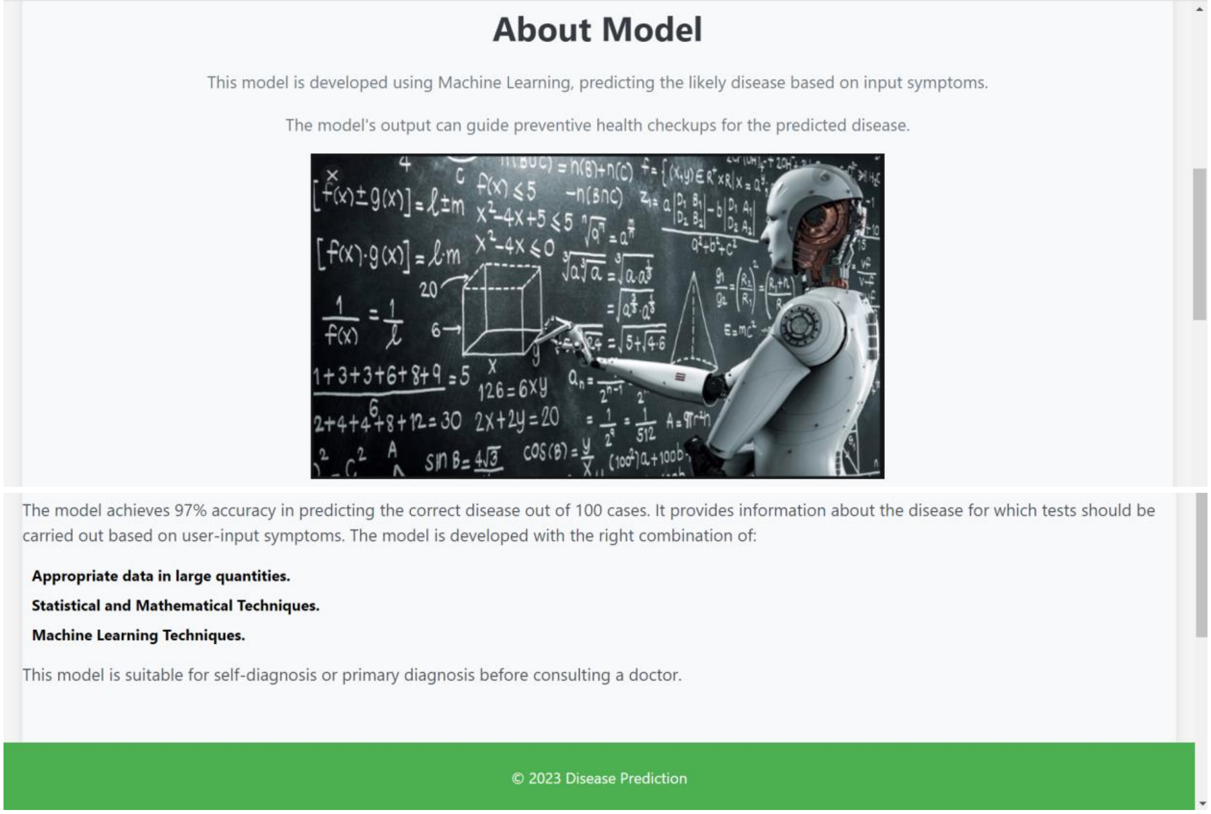
This model does not suggest any test. This model only predicts the disease which is likely for the symptoms you have given as input. You can use the information for preventive diagnosis or can consult a doctor regarding the same.

4. How many diseases can this model predict based on symptoms?

This model is capable of predicting 56 diseases. The model will output the disease which is most likely based on the symptoms.

© 2023 Disease Prediction

About Model:



About Model

This model is developed using Machine Learning, predicting the likely disease based on input symptoms.

The model's output can guide preventive health checkups for the predicted disease.

The model achieves 97% accuracy in predicting the correct disease out of 100 cases. It provides information about the disease for which tests should be carried out based on user-input symptoms. The model is developed with the right combination of:

Appropriate data in large quantities.

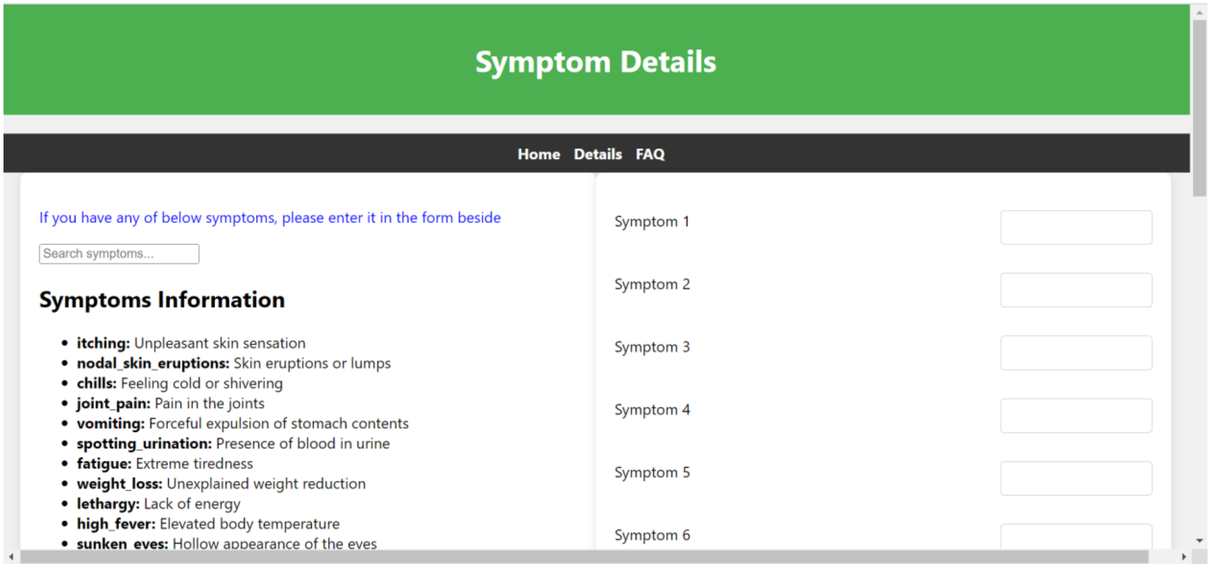
Statistical and Mathematical Techniques.

Machine Learning Techniques.

This model is suitable for self-diagnosis or primary diagnosis before consulting a doctor.

© 2023 Disease Prediction

Symptom Details:



Symptom Details

If you have any of below symptoms, please enter it in the form beside

Symptoms Information

- **itching:** Unpleasant skin sensation
- **nodal_skin_eruptions:** Skin eruptions or lumps
- **chills:** Feeling cold or shivering
- **joint_pain:** Pain in the joints
- **vomiting:** Forceful expulsion of stomach contents
- **spotting_urination:** Presence of blood in urine
- **fatigue:** Extreme tiredness
- **weight_loss:** Unexplained weight reduction
- **lethargy:** Lack of energy
- **high_fever:** Elevated body temperature
- **sunken_eyes:** Hollow appearance of the eyes

Symptom 1

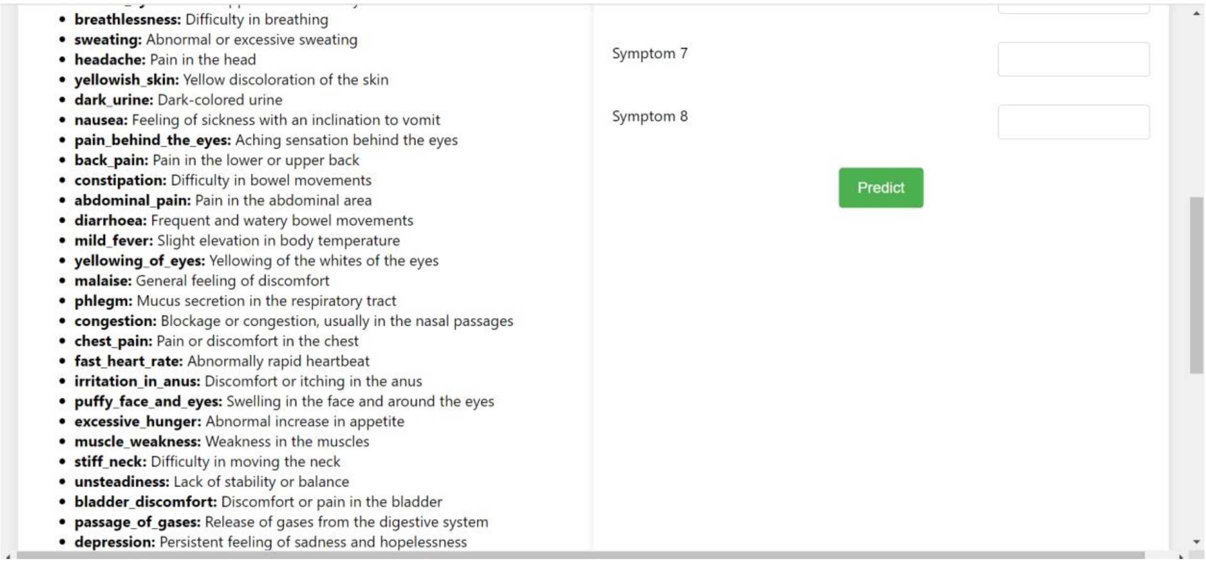
Symptom 2

Symptom 3

Symptom 4

Symptom 5

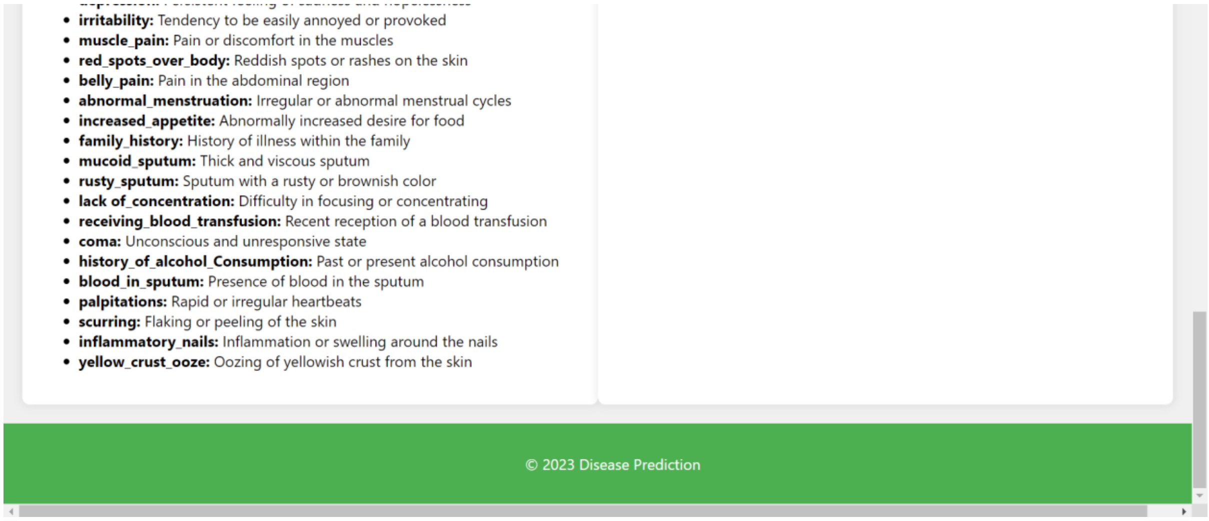
Symptom 6



- **breathlessness:** Difficulty in breathing
- **sweating:** Abnormal or excessive sweating
- **headache:** Pain in the head
- **yellowish_skin:** Yellow discoloration of the skin
- **dark_urine:** Dark-colored urine
- **nausea:** Feeling of sickness with an inclination to vomit
- **pain_behind_the_eyes:** Aching sensation behind the eyes
- **back_pain:** Pain in the lower or upper back
- **constipation:** Difficulty in bowel movements
- **abdominal_pain:** Pain in the abdominal area
- **diarrhoea:** Frequent and watery bowel movements
- **mild_fever:** Slight elevation in body temperature
- **yellowing_of_eyes:** Yellowing of the whites of the eyes
- **malaise:** General feeling of discomfort
- **phlegm:** Mucus secretion in the respiratory tract
- **congestion:** Blockage or congestion, usually in the nasal passages
- **chest_pain:** Pain or discomfort in the chest
- **fast_heart_rate:** Abnormally rapid heartbeat
- **irritation_in_anus:** Discomfort or itching in the anus
- **puffy_face_and_eyes:** Swelling in the face and around the eyes
- **excessive_hunger:** Abnormal increase in appetite
- **muscle_weakness:** Weakness in the muscles
- **stiff_neck:** Difficulty in moving the neck
- **unsteadiness:** Lack of stability or balance
- **bladder_discomfort:** Discomfort or pain in the bladder
- **passage_of_gases:** Release of gases from the digestive system
- **depression:** Persistent feeling of sadness and hopelessness

Symptom 7

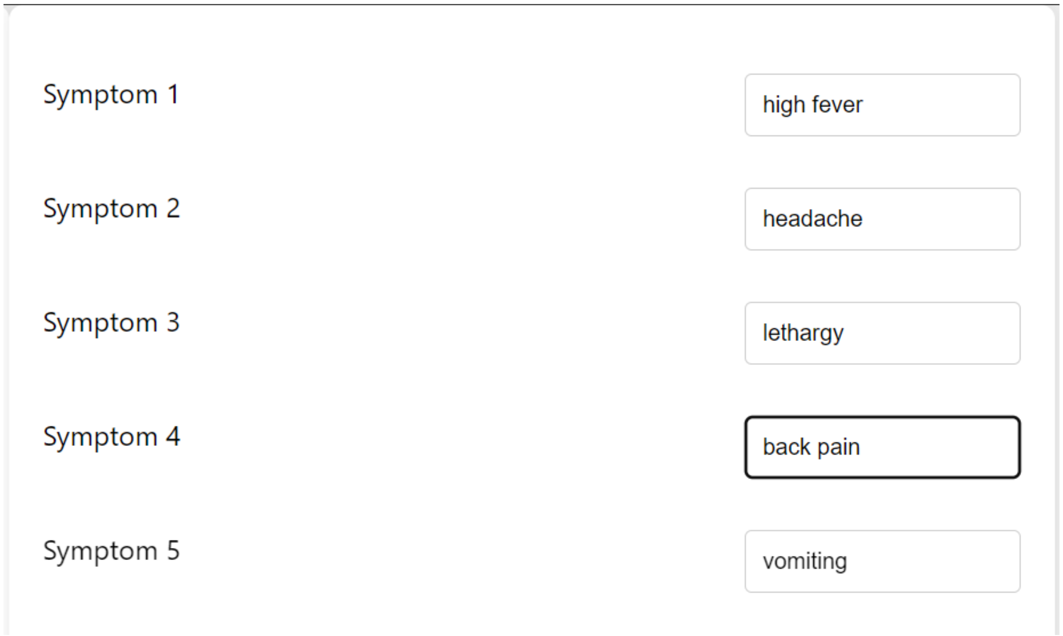
Symptom 8



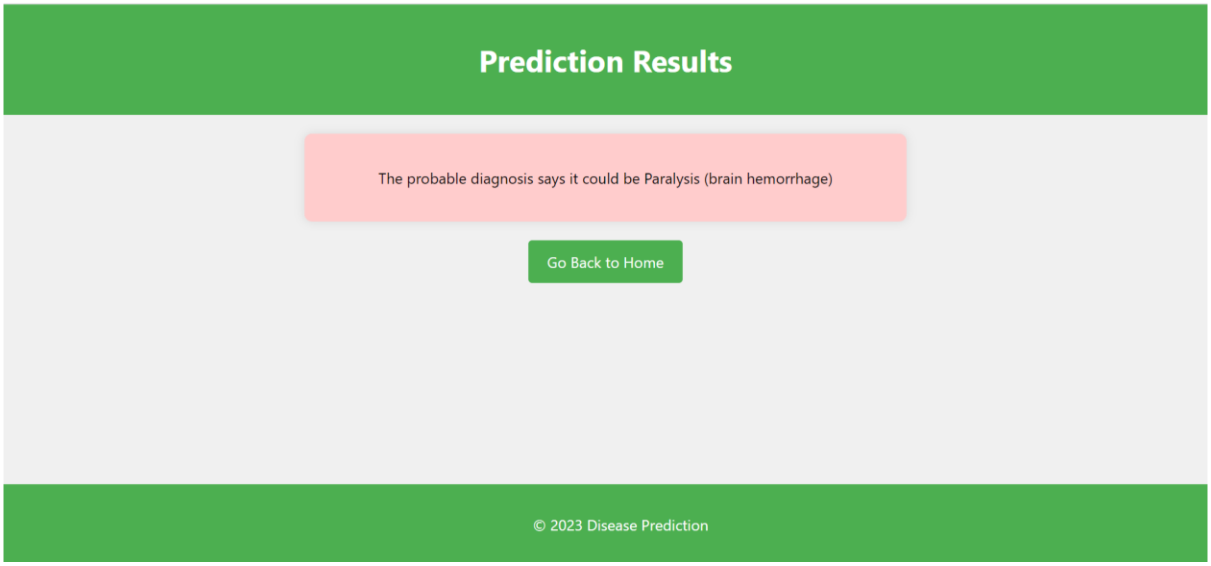
- **irritability:** Tendency to be easily annoyed or provoked
- **muscle_pain:** Pain or discomfort in the muscles
- **red_spots_over_body:** Reddish spots or rashes on the skin
- **belly_pain:** Pain in the abdominal region
- **abnormal_menstruation:** Irregular or abnormal menstrual cycles
- **increased_appetite:** Abnormally increased desire for food
- **family_history:** History of illness within the family
- **mucoid_sputum:** Thick and viscous sputum
- **rusty_sputum:** Sputum with a rusty or brownish color
- **lack of_concentration:** Difficulty in focusing or concentrating
- **receiving_blood_transfusion:** Recent reception of a blood transfusion
- **coma:** Unconscious and unresponsive state
- **history_of_alcohol_Consumption:** Past or present alcohol consumption
- **blood_in_sputum:** Presence of blood in the sputum
- **palpitations:** Rapid or irregular heartbeats
- **scurring:** Flaking or peeling of the skin
- **inflammatory_nails:** Inflammation or swelling around the nails
- **yellow_crust_ooze:** Oozing of yellowish crust from the skin

© 2023 Disease Prediction

Disease Prediction:



Symptom 1	high fever
Symptom 2	headache
Symptom 3	lethargy
Symptom 4	back pain
Symptom 5	vomiting



Prediction Results

The probable diagnosis says it could be Paralysis (brain hemorrhage)

© 2023 Disease Prediction

10.Advantages & Disadvantages:

Advantages:

Time Efficiency:

- The tool offers a quick and time-efficient way for users to receive probable disease predictions, addressing the challenge of limited time for healthcare in a fast-paced lifestyle.

Privacy Preservation:

- By not collecting personal information, the tool prioritizes user privacy, alleviating concerns about data security and fostering trust among users.

Educational Component:

- The inclusion of educational content with disease predictions encourages informed decision-making, promoting responsible health practices and discouraging self-diagnosis.

Versatility for Online Consultations:

The tool serves as a valuable resource for online consultations with healthcare professionals, allowing users

- to seek guidance without the need for immediate in-person visits.

Preventive Healthcare Support:

- Facilitating early intervention by healthcare professionals, the tool contributes to preventive healthcare, reducing the burden of unnecessary medical visits.

Disadvantages:

Dependency on Input Accuracy:

- The accuracy of predictions heavily relies on the accuracy of the input symptoms provided by users. Inaccurate or incomplete information may result in less reliable outcomes.

Limited Scope:

- While the tool identifies a broad spectrum of diseases, it may not cover every possible health condition. Users should be aware of its limitations and seek professional medical advice for comprehensive evaluations.

Internet Dependency:

- The tool relies on internet connectivity for user interactions and updates, posing limitations for users in areas with poor or no internet access.

Lack of Human Interaction:

- Online consultations through the tool lack the personal touch and detailed examination provided in face-to-face doctor visits, potentially missing subtle symptoms or nuances.

User Reluctance:

- Some users may remain skeptical about the accuracy of online disease predictions, leading to a reluctance to rely solely on the tool for healthcare decisions.

12. Conclusion:

In conclusion, the disease prediction tool addresses the contemporary challenge of balancing hectic lifestyles with healthcare needs. Its emphasis on time efficiency, user privacy, and educational support makes it a valuable resource for individuals seeking quick, reliable insights into their health conditions. While the tool offers significant

advantages, it is crucial to acknowledge its limitations and the importance of professional medical advice in comprehensive healthcare decision-making. As technology continues to evolve, this initiative marks a step towards enhancing accessibility to preventive healthcare.

13. Future Scope:

The future scope of the disease prediction tool involves continuous improvement and adaptation to emerging technologies and healthcare practices. Key areas for future development include:

Enhanced Machine Learning Models:

- Incorporate advanced machine learning techniques to further improve the accuracy of disease predictions, leveraging ongoing developments in the field.

Global Expansion:

- Extend the tool's reach to a global audience, considering cultural and regional variations in healthcare practices and disease prevalence.

Integration with Wearable Devices:

- Explore integration with wearable health devices to gather real-time health data, providing a more holistic view for accurate predictions.

AI-driven Personalized Recommendations:

- Develop personalized health recommendations based on users' historical data, lifestyle, and genetic factors, tailoring the tool to individual health profiles.

Collaboration with Healthcare Providers:

- Foster collaboration with healthcare institutions and professionals to enhance the tool's credibility and ensure seamless integration with existing healthcare systems.

User Feedback Integration:

- Strengthen the user feedback mechanism to continually refine the tool based on user experiences and evolving medical knowledge.

As the disease prediction tool evolves, its future trajectory involves staying at the forefront of technological advancements and contributing to the ongoing transformation of healthcare practices.