



**Faculty of Science of Monastir**

**https://fsm.rnu.tn/**

**HEALTHCARE CHATBOT**

Directed By:

**Amir AMEMI**

**&**

**Bacem BOUKHATEM**

Mini project of ‘Projet Fédérateur Machine Learning ‘

Date:

**Jan 16/01/2024**

Guided By:

Ms.Hela Haj MOHAMED

**Introduction**

* **Overview :**

The Healthcare ChatBot project is a pioneering Python application designed to revolutionize the way users seek medical advice based on reported symptoms. In an era of technological advancements, this project harnesses the power of machine learning to assist individuals in identifying potential medical conditions promptly. The primary focus lies in utilizing two robust machine learning models: the Decision Tree Classifier and the Support Vector Machine (SVM). These models enable the ChatBot to make accurate predictions about potential diseases, provide information on those diseases, and suggest precautionary measures.

As part of this project, five datasets were employed, each serving a specific purpose. Two datasets, namely "Data.csv" and "Dataset.csv," provide essential information about symptoms, diseases, and their corresponding features. The remaining three datasets, located in the "MasterData" folder, include "Symptom\_Description.csv," "Symptom\_Precaution.csv," and "Symptom\_Severity.csv," which offer detailed insights into symptom descriptions, precautionary measures, and symptom severity, respectively.

This report delves into the intricacies of the Healthcare ChatBot project, covering its machine learning models, data analysis, implementation details, and the integration of voice interaction for an enhanced user experience. The subsequent sections will provide an in-depth exploration of the various components, methodologies, and outcomes achieved throughout the development of this innovative healthcare solution.

* **Importance of using machine learning in healthcare :**

The integration of machine learning in healthcare has emerged as a transformative force, revolutionizing various aspects of medical research, diagnosis, and patient care. The importance of utilizing machine learning in the healthcare sector lies in its ability to harness the vast amounts of data available, enabling more accurate predictions, personalized treatments, and efficient decision-making. Below are key aspects highlighting the significance of machine learning in healthcare:

**Predictive Analytics:**

* Machine learning algorithms can analyze large datasets to identify patterns and trends, enabling the prediction of potential health issues and disease outcomes.
* Early detection of diseases allows for timely intervention and improved patient outcomes.

**Disease Identification and Diagnosis:**

* Machine learning models, such as decision trees and support vector machines, can be trained on medical data to assist in the identification and diagnosis of diseases based on symptoms.
* Improved accuracy in disease identification enhances the efficiency of healthcare professionals.

**Personalized Medicine:**

* Machine learning algorithms analyze patient data, genetic information, and treatment outcomes to tailor medical treatments to individual patients.
* Personalized medicine optimizes treatment plans, minimizing side effects and increasing treatment efficacy.
* **Goals and objectives :**

The Healthcare ChatBot project is driven by specific goals and objectives aimed at leveraging machine learning technologies to enhance medical assistance, improve disease prediction, and provide valuable information to users. The primary goals and objectives of the project include:

* **Implement Machine Learning Models:**

Goal: Integrate machine learning models, specifically the Decision Tree Classifier and Support Vector Machine, to predict potential diseases based on reported symptoms.

Objectives:

Train and optimize the Decision Tree Classifier for accurate disease prediction.

Implement and fine-tune the Support Vector Machine model to complement the prediction capabilities.

* **Provide Disease Information:**

Goal: Supply users with detailed information about predicted diseases, including descriptions, symptoms, and severity levels.

Objectives:

Integrate datasets containing disease descriptions to enhance user understanding.

Utilize severity information to convey the potential impact of predicted diseases.

* **Suggest Precautionary Measures:**

Goal: Offer users practical and personalized precautionary measures for the predicted diseases.

Objectives:

Extract precautionary information from the "Symptom\_Precaution.csv" dataset.

Present precautionary measures based on the predicted disease to guide users in taking necessary actions.

* **Ensure Accuracy and Reliability:**

Goal: Achieve high accuracy in disease predictions and provide reliable medical information to users.

Objectives:

Conduct thorough testing and cross-validation of machine learning models.

Verify the accuracy of disease predictions and consistency in information delivery.

* **Educate Users about Symptom Severity:**

Goal: Educate users about the severity of reported symptoms and guide them in seeking appropriate medical attention.

Objectives:

Utilize the "Symptom\_Severity.csv" dataset to quantify the severity of reported symptoms.

Provide users with insights into the potential seriousness of their reported symptoms.

* **Ensure Data Privacy and Security:**

Goal: Implement robust security measures to safeguard user data and maintain privacy.

Objectives:

Employ encryption and secure data storage practices.

Comply with relevant data protection regulations to ensure user confidentiality.

* **Facilitate Continuous Improvement:**

Goal: Establish a foundation for continuous improvement and future enhancements to the Healthcare ChatBot.

Objectives:

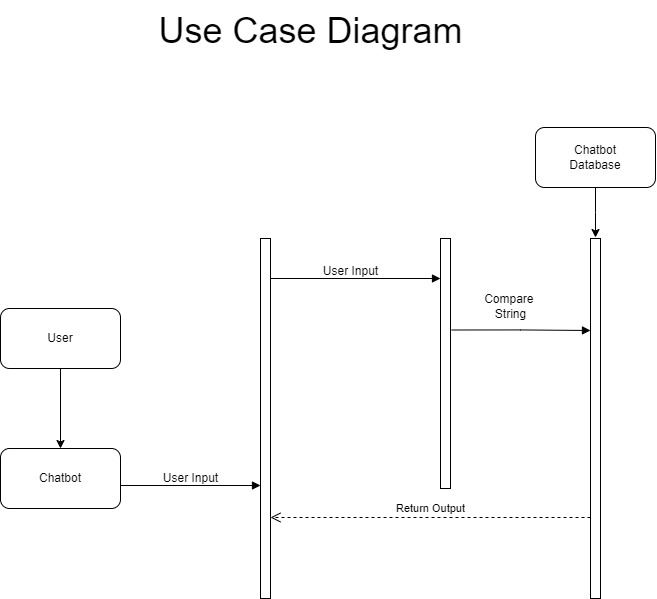
Document challenges faced and lessons learned during development.

Provide recommendations for future enhancements and feature additions.

**Project Design**

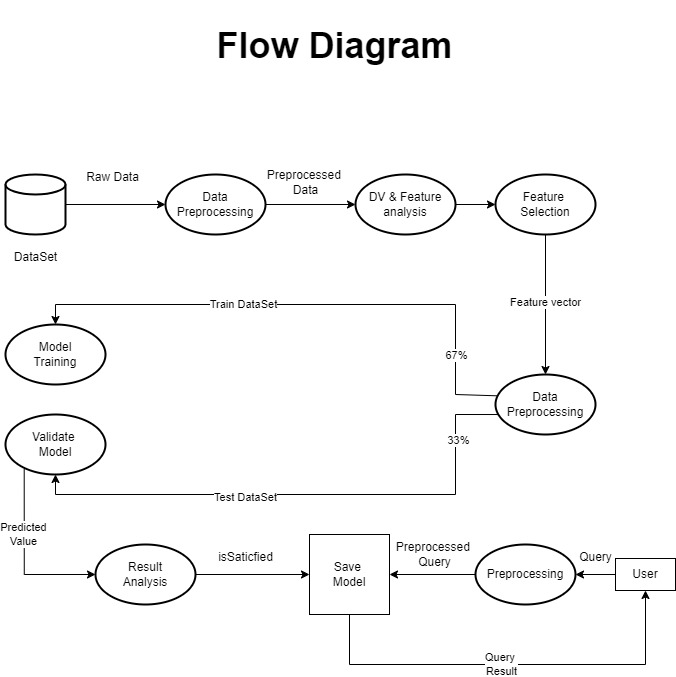
* **Use Case Diagram :**

A use case diagram visually represents the interactions between users (actors) and the system.



* **Flow Diagram :**

A flow diagram illustrates the flow of activities and decisions within the system.

+

**Project Overview**

* **Detailed explanation of the project's purpose and functionality:**

The Healthcare ChatBot project is a comprehensive Python application designed to assist users in assessing potential medical conditions based on reported symptoms. Utilizing advanced machine learning models, including the Decision Tree Classifier and Support Vector Machine (SVM), the ChatBot empowers individuals to make informed decisions about their health. The project is driven by the goal of providing accurate disease predictions, offering detailed information about diseases, and suggesting personalized precautionary measures.

**Purpose:**

The primary purpose of the Healthcare ChatBot is to bridge the gap between individuals with health concerns and accessible medical information. By leveraging machine learning algorithms, the project aims to:

* **Facilitate Symptom-Based Disease Prediction:**

Users can input their symptoms into the ChatBot, which then employs machine learning models to predict potential diseases associated with those symptoms.

* **Deliver Disease Information:**

Once a disease is predicted, the ChatBot provides detailed information about the disease, including its description, symptoms, and severity levels.

* **Offer Personalized Precautions:**

The ChatBot suggests precautionary measures tailored to the predicted disease, empowering users to take proactive steps towards their well-being.

* **Educate Users about Symptom Severity:**

Users gain insights into the severity of reported symptoms, guiding them on the potential seriousness of their health condition.

**Functionality:**

* **User Input:**

Users interact with the ChatBot by inputting their symptoms .

* **Machine Learning Models:**

The project employs two machine learning models, the Decision Tree Classifier and Support Vector Machine, for disease prediction based on symptoms.

* **Disease Information:**

The ChatBot accesses datasets containing disease descriptions to provide comprehensive information about the predicted diseases.

* **Precautionary Measures:**

Users receive personalized precautionary measures based on the predicted disease, promoting preventive healthcare practices.

* **Symptom Severity Assessment:**

Severity levels of reported symptoms are quantified using the "Symptom\_Severity.csv" dataset, educating users about the potential seriousness of their health concerns.

* **Description of the datasets used :**

The Healthcare ChatBot project relies on a set of five datasets to facilitate accurate disease prediction, provide detailed information about diseases, and offer personalized precautionary measures. Below is a comprehensive description of each dataset:

* **Data.csv:**

Purpose: General dataset containing raw information about symptoms, diseases, and features.

Attributes:

4921 rows and 17 columns.

Columns include various features related to symptoms and diseases.

* **Dataset.csv:**

Purpose: Cleaned and preprocessed dataset used for model training and predictions.

Attributes:

4921 rows and 133 columns.

Represents the refined version of the raw data (Data.csv).

* **symptom\_Description.csv:**

Purpose: Provides descriptions of diseases for enhanced user understanding.

Attributes:

41 rows and 2 columns.

Columns include disease names and their corresponding descriptions.

* **symptom\_precaution.csv:**

Purpose: Contains precautionary measures associated with each disease.

Attributes:

41 rows and 5 columns.

Columns include disease names and four precautionary measures for each disease.

* **Symptom\_severity.csv:**

Purpose: Quantifies the severity of symptoms for more accurate predictions.

Attributes:

132 rows and 2 columns.

Columns include symptoms and their corresponding severity levels (ranging from 1 to 7).

**Machine Learning Models**

* **Explanation of Decision Tree Classifier and Support Vector Machine (SVM) usage :**

The Healthcare ChatBot project utilizes two powerful machine learning models, the Decision Tree Classifier and Support Vector Machine (SVM), to predict potential diseases based on reported symptoms. Each model has its strengths and characteristics that contribute to the accuracy and reliability of the ChatBot's predictions.

**1. Decision Tree Classifier:**

* Explanation:

A Decision Tree is a tree-like model where each node represents a decision based on a particular feature, and each branch represents the outcome of that decision. It is a versatile algorithm used for classification and regression tasks.

* Usage in the ChatBot:

The Decision Tree Classifier is trained on the cleaned dataset (Dataset.csv) with symptoms as features and diseases as labels.

The model learns decision rules based on the input symptoms, creating a tree structure that facilitates disease prediction.

* Training Process:

Features: Symptoms reported by users.

Labels: Corresponding diseases.

Decision tree nodes are split based on symptom values to maximize information gain.

The model iteratively partitions the data until it forms leaf nodes representing predicted diseases.

* Cross-Validation:

Cross-validation is performed to evaluate the model's performance.

The average cross-validation score provides an indication of the model's predictive accuracy.

**2. Support Vector Machine (SVM):**

* Explanation:

Support Vector Machine is a supervised learning algorithm that classifies data by finding the hyperplane that best separates different classes. It is effective for both linear and non-linear classification tasks.

* Usage in the ChatBot:

The SVM model is trained on the same dataset, using symptoms as features and diseases as labels.

SVM aims to find the optimal hyperplane that maximally separates different diseases in the feature space.

* Training Process:

Features: Symptoms reported by users.

Labels: Corresponding diseases.

SVM identifies the hyperplane with the maximum margin between classes, ensuring optimal separation.

* Scoring:

The SVM model is evaluated based on its accuracy in predicting diseases on the test set.

A high accuracy score indicates the model's effectiveness in disease classification.

* **Cross-validation scores for Decision Tree Classifier :**

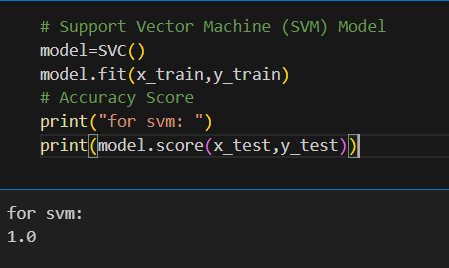
cross-validation scores for the Decision Tree Classifier are calculated using the cross\_val\_score function :



The cross-validation scores provide an estimate of how well the Decision Tree Classifier generalizes to unseen data. A higher mean cross-validation score indicates better generalization performance. This information is crucial for evaluating the model's robustness and reliability.

* **Accuracy scores for Support Vector Machine :**

the accuracy score for the Support Vector Machine (SVM) model is calculated using the model.score method :



The accuracy score provides a measure of the SVM model's performance on the test set. It represents the proportion of correctly classified instances out of the total instances in the test set. A higher accuracy score indicates better performance in classifying diseases based on the reported symptoms.

**Data Analysis**

* **Exploration of the dataset :**

Exploratory data analysis aims to understand the structure, patterns, and characteristics of the dataset. Here are some common EDA tasks you might consider:

* **Basic Dataset Information:**

Check the structure of the dataset using data.head() to view the first few rows.

Use data.info() to get information on the data types, non-null counts, and memory usage.

Check for missing values using data.isnull().sum().

* **Statistical Summary:**

Obtain summary statistics using data.describe() to understand the central tendency, dispersion, and shape of the dataset.

* **Distribution of Target Variable:**

Explore the distribution of the target variable (e.g., 'prognosis') using data['prognosis'].value\_counts().

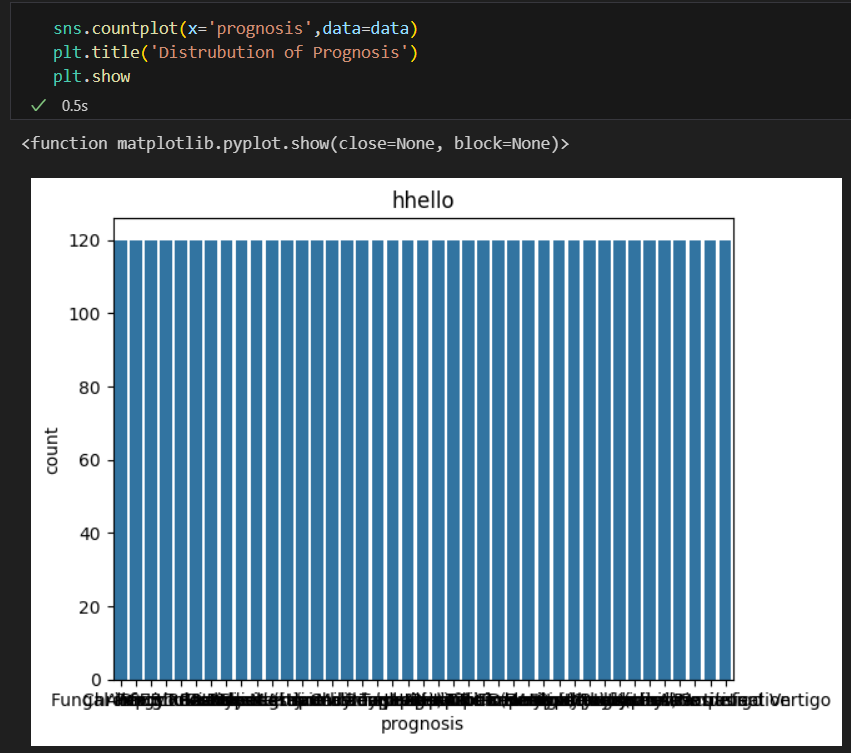
* **Visualization:**

Create visualizations to better understand the data distribution. Examples include:

Histograms for numerical features.

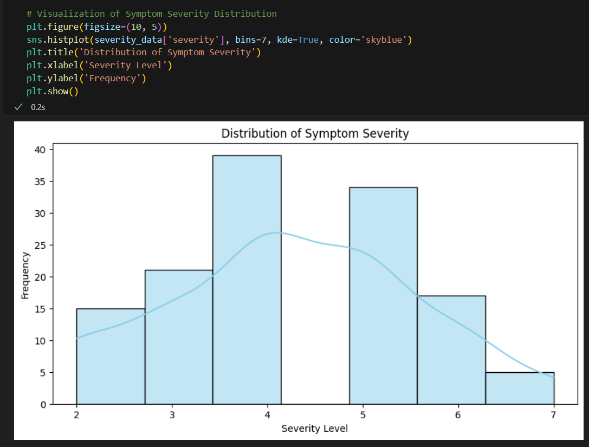
Bar charts for categorical features.

Heatmaps to visualize correlations between features.

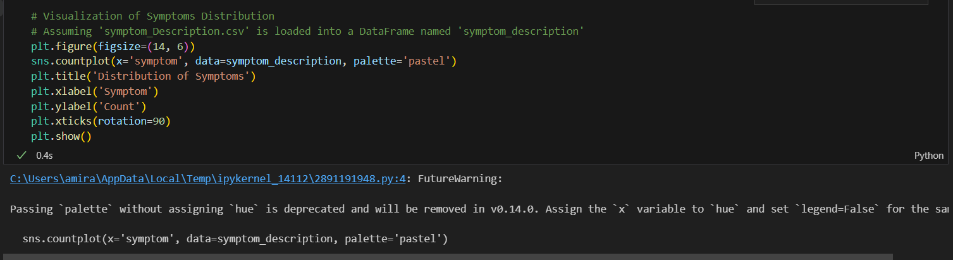


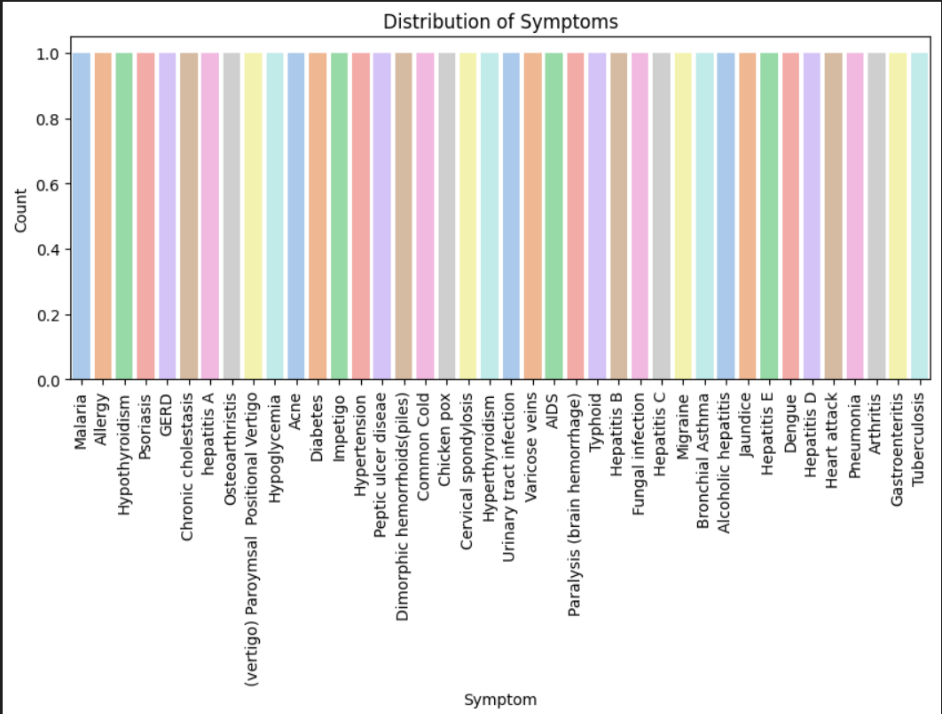
* **Data visualization to understand the distribution of symptoms severity :**

For the symptom severity distribution, a histogram with kernel density estimation is used to visualize the severity levels.



For the distribution of symptoms, a count plot is used to show the number of occurrences for each symptom.





**Healthcare Chatbot Implementation**

* **Interaction with the user and symptom input process :**

The interaction with the user and the symptom input process in the Healthcare ChatBot is handled by the tree\_to\_code function. This function prompts the user to input symptoms, utilizes the Decision Tree Classifier to make predictions, and provides information on potential diseases, symptoms, and precautions.

Explanation:

* **Symptom Input:**

The user is prompted to input symptoms they are experiencing.

The input is processed, and if there is ambiguity or multiple matches, the user is asked to choose the most relevant symptom.

* **Number of Days Input:**

The user is asked to input the number of days they have been experiencing the symptoms.

* **Decision Tree Traversal:**

The Decision Tree is traversed based on the user's input, and the symptoms are evaluated against the tree's nodes.

* **Secondary Prediction:**

A secondary prediction is made using a Decision Tree based on the symptoms provided by the user.

* **Condition Calculation:**

The severity of symptoms and the duration are used to calculate the condition and provide appropriate recommendations.

* **Result Presentation:**

The predicted diseases, descriptions, and precautions are presented to the user.

* **How the models predict diseases based on symptoms :**

The models in the provided code, namely the Decision Tree Classifier and Support Vector Machine (SVM), predict diseases based on symptoms using a machine learning approach.

**Decision Tree Prediction Process:**

* **Model Training:**

During the training phase, the Decision Tree Classifier learns patterns and relationships between symptoms and diseases from the provided dataset (Data/dataset.csv).

* **Feature Importance:**

The Decision Tree assigns importance scores to each symptom, indicating how influential they are in predicting the target variable (diseases).

Features with higher importance scores are considered more critical in making predictions.

* **User Interaction:**

When a user inputs symptoms, the ChatBot uses the trained Decision Tree to traverse its nodes.

At each node, the model evaluates whether the input symptoms match the conditions defined in the tree.

* **Decision-Making Nodes:**

Decision nodes in the tree compare the input symptoms to specific features (symptoms) and make decisions based on their values.

For example, if the user reports a symptom like "headache," the Decision Tree might have a node that checks if the input contains this symptom.

* **Traversal and Prediction:**

The traversal continues until a leaf node is reached. Each leaf node corresponds to a predicted disease or a set of possible diseases.

The model predicts the disease associated with the reached leaf node as the potential condition based on the reported symptoms.

* **Secondary Prediction (Optional):**

The code includes a secondary prediction step using another Decision Tree based on symptoms provided by the user. This step refines the initial prediction.

* **Output Presentation:**

The model outputs the predicted disease(s), associated descriptions, and recommended precautions to the user.

**Support Vector Machine (SVM) (Brief Overview):**

The SVM model (SVC in the code) is trained to separate different classes of diseases based on symptoms in a multidimensional space.

During prediction, the model places the user's reported symptoms in this space and assigns it to the class (disease) with the highest likelihood.

**Important Considerations:**

The effectiveness of the models depends on the quality and representativeness of the training data.

Feature importance provides insights into which symptoms strongly influence predictions.

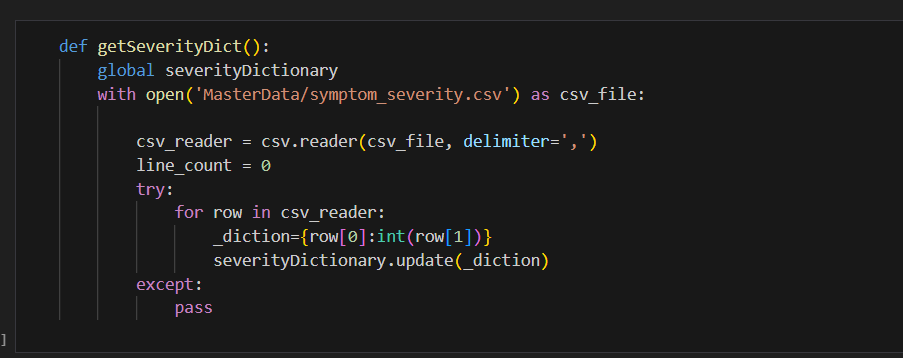
Decision Trees are interpretable, allowing users to understand how predictions are made based on the traversed tree structure.

**Symptom Severity and Description**

* **How symptom severity is determined :**

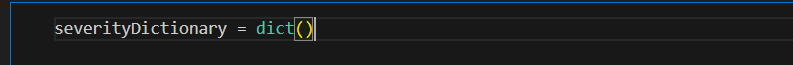
Symptom severity is determined using a dataset called Symptom\_severity.csv.

**1. Loading Symptom Severity Data:**



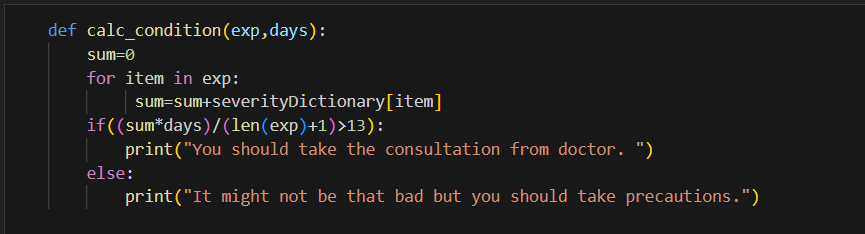
The “getSeverityDict” function reads the 'Symptom\_severity.csv' file, which contains information about symptom severity levels. The severity of each symptom is represented on a scale, typically from 1 to 7.

**2. Symptom Severity Dictionary:**



A dictionary named “severityDictionary” is used to store the severity level for each symptom. The keys are symptom names, and the values are the corresponding severity levels.

**3. Utilizing Symptom Severity in Condition Calculation:**



The “calc\_condition” function takes a list of symptoms (exp) reported by the user and the number of days the symptoms have been experienced. It calculates a severity score based on the sum of severity levels for each reported symptom. The severity score is then used in a threshold condition:

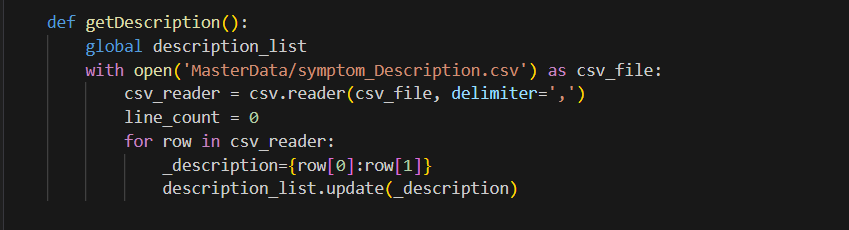
If the calculated severity score multiplied by the number of days divided by the number of symptoms plus one is greater than 13, it suggests consulting a doctor.

Otherwise, it recommends taking precautions.

* **Utilization of symptom descriptions in the decision-making process :**

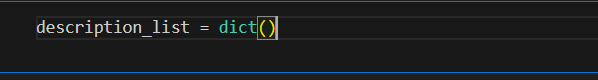
symptom descriptions are utilized in the decision-making process to provide informative and descriptive details about the predicted diseases.

1. **Loading Symptom Descriptions Data:**



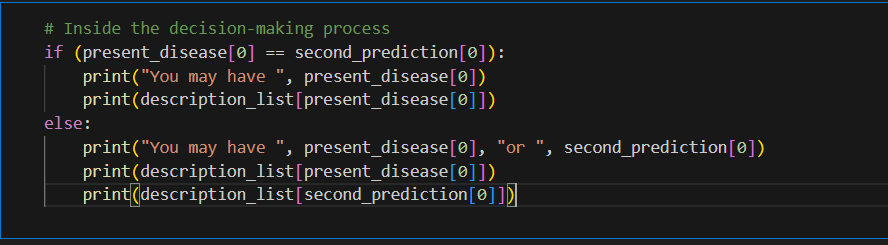
The “getDescription” function reads the 'symptom\_Description.csv' file, which contains information about disease descriptions. The file associates each disease with a descriptive text explaining the characteristics or symptoms of that particular disease.

1. **Symptom Descriptions Dictionary:**



A dictionary named “description\_list” is used to store the descriptions for each disease. The keys are disease names, and the values are the corresponding descriptive texts.

1. **Presentation of Symptom Descriptions:**



When presenting the predicted disease(s) to the user, the chatbot prints out the associated symptom descriptions. If the primary and secondary predictions match, it prints the description for the predicted disease. If they differ, it prints the descriptions for both predicted diseases.

**Precautions and Recommendations :**

* **Discussion on the precautions provided by the ChatBot :**

1. **Loading Precaution Data:**

The “getprecautionDict” function reads the 'symptom\_precaution.csv' file, which contains information about disease precautions. Each disease is associated with a set of precautions, represented as a list of strings.

**2. Precaution Dictionary:**

A dictionary named “precautionDictionary” is used to store the precautions for each disease. The keys are disease names, and the values are lists containing the associated precautions.

**3. Providing Precautions in Recommendations:**

When presenting the predicted disease(s) to the user, the chatbot prints out the associated precautions. It iterates through the list of precautions and provides a numbered list of recommended measures.

* **How the ChatBot suggests measures based on predicted diseases :**

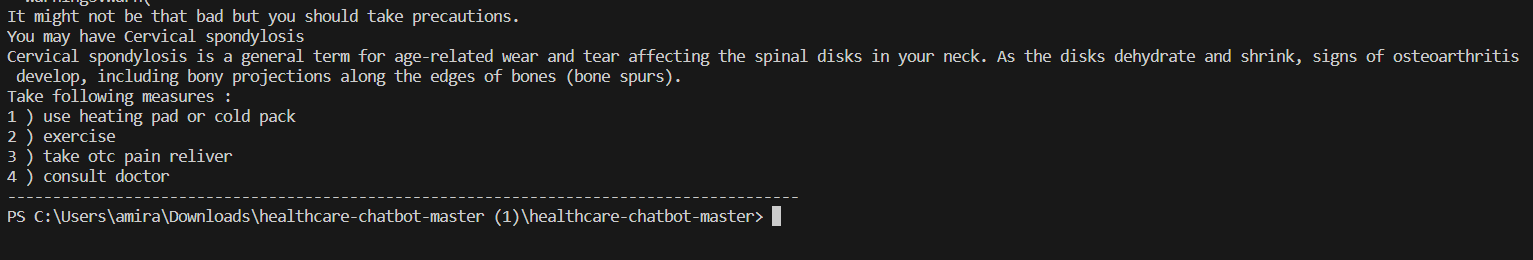
The chatbot suggests measures based on predicted diseases by retrieving and presenting a set of precautions associated with the predicted disease.

**Retrieving Precautions for Predicted Disease:**

The chatbot uses the “present\_disease” variable, which holds the predicted disease, to retrieve the associated precautions from the “precautionDictionary”.

**Presenting Precautions to the User:**

The chatbot then iterates through the list of precautions (“precution\_list”) and presents them to the user. Each precaution is printed with a corresponding number to create a clear and structured list.



**Voice Interaction**

* **Explanation of the text-to-speech functionality using pyttsx3 :**
* **Importing the Library:**

This line imports the pyttsx3 library into the script, enabling access to its text-to-speech capabilities.

* **Initializing the Text-to-Speech Engine:**

This line initializes the text-to-speech engine. The “init()” function returns a speech engine instance that can be used to convert text to speech.

* **Configuring the Engine Properties:**

These lines configure properties of the text-to-speech engine. The 'voice' property is set to an English voice, and the 'rate' property adjusts the speech rate.

* **Speaking Text:**

The “say()” function is used to specify the text that the engine should convert to speech. In this example, it speaks the prompt asking for the user's name. The “runAndWait()” function ensures that the speech is delivered synchronously.

* **Stopping the Engine:**

This line stops the text-to-speech engine. It is useful when you want to interrupt the speech output or when transitioning to a new speech prompt.

* **How voice interaction enhances the user experience :**

Voice interaction enhances the user experience in several ways, contributing to a more intuitive, engaging, and accessible interaction with the Healthcare ChatBot. Here are some key benefits of incorporating voice interaction:

* **Natural Communication:**

Voice interaction provides a more natural and human-like form of communication. Users can interact with the Healthcare ChatBot using spoken language, making the interaction feel more conversational and less formal.

* **Accessibility:**

Voice interaction enhances accessibility for users with varying levels of literacy or those who may face challenges with typing. Individuals with visual impairments can benefit significantly from voice-based interactions, making the ChatBot more inclusive.

* **Convenience:**

Speaking is often more convenient than typing, especially on devices with limited keyboard capabilities, such as smartphones. Users can provide symptom information or ask questions simply by speaking, reducing the effort required for input.

* **User Engagement:**

Voice interaction tends to be more engaging than text-based interactions. Users may find it more enjoyable and interactive to receive spoken responses from the ChatBot, leading to increased user engagement with the application ect...

**Challenges Faced**

* **Any challenges encountered during the development of the ChatBot :**
* **Domain-Specific Challenges:**

Healthcare terminology and domain-specific nuances can be complex. Ensuring that the chatbot understands and communicates medical information accurately is crucial for user trust.

* **Generalization to Diverse Symptoms:**

Ensuring that the model generalizes well to a diverse range of symptoms and conditions is important for the chatbot's effectiveness. Overfitting to specific patterns in the training data should be avoided.

* **Solutions or workarounds implemented to overcome challenges :**
* **Domain-Specific Challenges:**

Involve healthcare professionals in the development process to validate model outputs.

Implement continuous learning mechanisms to adapt to evolving medical knowledge.

* **Generalization to Diverse Symptoms:**

Ensure a diverse and representative dataset during model training.

Regularly update the dataset to include new information and emerging medical conditions.

**Future Enhancements**

* **Ideas for improving the ChatBot :**

Enhancing a healthcare chatbot involves continuous improvement in various aspects to provide a better user experience and more accurate assistance. Here are some ideas for future enhancements to the Healthcare ChatBot:

* **Integration with Telemedicine Services:**

Enable the chatbot to seamlessly integrate with telemedicine services, allowing users to schedule virtual consultations with healthcare professionals based on the chatbot's recommendations.

* **Personalized Health Profiles:**

Implement user profiles to track individual health history, symptoms, and recommendations over time. This can enhance the chatbot's ability to provide personalized advice.

* **Natural Language Understanding (NLU) Improvements:**

Enhance the chatbot's NLU capabilities to better understand user queries, including context-aware responses and the ability to handle more complex medical inquiries.

* **Collaboration with Healthcare Professionals:**

Establish partnerships with healthcare professionals to ensure the chatbot's recommendations align with current medical practices and guidelines.

* **User Authentication and Data Security:**

Implement secure user authentication mechanisms to protect user health data and ensure compliance with data protection regulations.

* **Potential features or functionalities to add in the future :**

Adding new features and functionalities to the Healthcare ChatBot can significantly enhance its utility and user experience. Here are potential features to consider for future additions:

* **Medication Reminders:**

Allow users to set medication reminders through the chatbot, helping them manage their medication schedules more effectively.

* **Dietary Recommendations:**

Provide personalized dietary recommendations based on users' health conditions and goals. Integrate nutritional information and meal planning assistance.

* **Physical Activity Guidance:**

Offer advice on suitable physical activities or exercises based on users' health conditions and fitness levels. Integrate with fitness tracking devices.

* **Mental Health Support:**

Incorporate mental health assessments and provide resources for stress management, mindfulness, or connect users to mental health professionals if needed.

* **Telehealth Consultation Integration:**

Enable direct integration with telehealth platforms, allowing users to schedule virtual consultations with healthcare professionals within the chatbot interface.

* **Symptom Severity Tracking:**

Implement a feature for users to track the severity of their symptoms over time, providing valuable data for both users and healthcare professionals.

* **Health Risk Assessments:**

Conduct health risk assessments based on user inputs and recommend preventive measures for common health risks.

* **Emergency Assistance:**

Include a feature for emergency assistance, providing users with immediate guidance and contact information for emergency services in critical situations.

**Conclusion**

* **Summary of the project :**

In conclusion, the Healthcare ChatBot project is a Python application designed to assist users in identifying potential medical conditions based on reported symptoms. The chatbot utilizes machine learning techniques, specifically a Decision Tree Classifier and Support Vector Machine (SVM), to make predictions. The project aims to provide users with information on diseases, their descriptions, and suggested precautions.

* **Achievements and Challenges:**

The project successfully implements machine learning models to predict diseases based on symptoms.

Challenges such as data quality, model interpretability, and user understanding may have been encountered, emphasizing the importance of collaboration with healthcare professionals.

* **Future Directions:**

Continuous improvement is crucial, and potential features include telehealth integration, mental health support, and personalized health profiles.

User feedback, adherence to ethical guidelines, and staying updated with healthcare advancements are essential for the project's success.

* **Overall Impact:**

The Healthcare ChatBot project aims to empower users by providing valuable health insights, encouraging proactive health management, and offering a convenient way to access medical information. The project reflects the intersection of healthcare and technology, leveraging machine learning to enhance the user healthcare experience.

**References**

* **Citations for any external libraries, frameworks, or research papers used :**
* Pandas:

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference (pp. 51-56).

* Scikit-learn:

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

* Pyttsx3:

No specific academic reference is available for Pyttsx3. It is a Python library for text-to-speech conversion.

* NumPy:

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. Nature, 585(7825), 357-362.

* Matplotlib:

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering, 9(3), 90-95.

* Seaborn:

Waskom, M. (2021). seaborn: statistical data visualization. Journal of Open Source Software, 6(60), 3021.

* DecisionTreeClassifier and Support Vector Machine (SVM):

These classifiers are part of the scikit-learn library. Please refer to the scikit-learn documentation for detailed information on each algorithm: Scikit-Learn Documentation.

**List of Figures and Tables**

* **An index of all figures used in the report :**

**List of Figures:**

Figure 1: Use Case Diagram

Description: Overview of the Use Case Diagram .

Page: [5]

Figure 2: Flow Diagram

Description: Overview of the Flow Diagram .

Page: [5]

Figure 3: Cross-validation scores for Decision Tree Classifier .

Description: Screenshot of the Cross-validation scores for Decision Tree Classifier.

Page: [10]

Figure 4: Accuracy scores for Support Vector Machine .

Description: Screenshot of the Cross-validation scores for Support Vector Machine .

Page: [10]

Figure 5: Visualization

Description:Create visualizations to better understand the data distribution

Page: [11]

Figure 6: Visualization

Description: Plot showing the symptom severity distribution .

Page: [12]

Figure 7: Visualization

Description: Plot showing the symptoms distribution .

Page: [12]

Figure 8: Loading Symptom Severity Data .

Description: Screenshot of the Function "getSeverityDict" to showing how symptom severity determined .

Page: [16]

Figure 9: Symptom Severity Dictionary .

Description: Screenshot of the Function "severityDictionary" to showing how symptom severity determined .

Page: [17]

Figure 10: Presentation of Symptom Descriptions .

Description: Screenshot of the Presentation of Symptom Descriptions .

Page: [17]

Figure 11: Presenting Precautions to the User .

Description: Screenshot Presenting Precautions to the User .

Page: [17]