Symptom-Based Medication Suggestions

Farzana Sayeda Sumaiya
Dept. of Electrical and Computer Engineering
North South University
Plot #15, Block B, Bashundhara R/A, Dhaka, 1229, Bangladesh
farzana.sumaiya@northsouth.edu

Riazul Zannat
Dept. of Electrical and Computer Engineering
North South University
Plot #15, Block B, Bashundhara R/A, Dhaka, 1229, Bangladesh
riazul.zannat@northsouth.edu

***Abstract*—This paper presents a Symptom-Based Medication System, an intelligent healthcare solution designed to deliver personalized treatment plans based on user-input symptoms. The system utilizes a comprehensive symptom-prognosis dataset to identify probable health conditions and recommends tailored prescriptions, including medication details, dosage schedules, potential side effects, suggestion, and precautions. An integrated alarm system ensures timely medication reminders, enhancing user adherence to prescribed treatments. The dataset, curated manually from the medical reference book QIMP17 by Dr. Ridwan Ullah Shahidi, ensures high reliability and domain accuracy.The system achieves an overall accuracy of 88%, with an F1-score of 88% and a precision of 92% by using SVC model, validated through rigorous testing and evaluation.By incorporating machine learning and domain expertise, this framework optimizes the decision-making process, offering accurate and actionable insights. This project addresses the increasing demand for accessible healthcare solutions, particularly in underserved regions, by providing a scalable and user-friendly platform. The proposed system demonstrates its potential as a critical tool for preliminary healthcare management, reducing reliance on in-person consultations and promoting proactive health management.**

***Index Terms*—Symptom-Based Medication System, healthcare, machine learning, personalized treatment, medication adherence**

I. Introduction

Access to accurate and timely medical advice remains a significant challenge, particularly in remote or underserved areas with limited healthcare facilities. Individuals frequently encounter difficulties in obtaining reliable preliminary diagnoses or medication recommendations, resulting in delayed treatment and potential health risks. This issue is exacerbated by the absence of integrated systems that combine disease prediction with actionable health advice, such as suggestions , precautions and medication schedules.

We propose a Symptom-Based Medication System that utilizes machine-learning algorithms to predict diseases based on user-reported symptoms. The system provides users with tailored health recommendations, including pharmacological interventions, suggestions , precautions, and side effect warnings. It also incorporates an alarm feature to remind users to adhere to medication schedules. By implementing models such as Support Vector Classifier (SVC), Random Forest Classifier, K-Nearest Neighbors (KNN), Multinomial Naïve Bayes (MultinomialNB), and Gradient Boosting, we aim to optimize prediction reliability. The highest-performing model based on evaluation metrics is deployed to ensure dependable results.

Researchers have extensively used machine learning techniques such as Random Forest, SVM, and Naïve Bayes for symptom-based disease prediction, proving their ability to handle structured medical datasets effectively. These methods have improved diagnostic accuracy by analyzing patterns in symptoms and diseases, providing an alternative to primary medical screening [1], [2], [5]. Other studies have gone beyond simple disease prediction to include additional features. For example, integrating medication suggestions and hospital recommendations to enhance usability has been a focus of some systems. These features make healthcare systems more comprehensive and practical for patients in remote or underserved areas [3], [4], [7] . Comparative analyses have highlighted the strengths and weaknesses of different machine learning algorithms. Studies comparing models like Random Forest, SVM, and Decision Trees have found that algorithm selection significantly impacts performance, depending on the dataset's complexity and size [5], [6], [8].Some systems incorporate demographic and clinical data alongside symptoms to improve disease prediction. By combining structured and unstructured data, these models deliver more robust predictions, particularly for diseases with overlapping symptoms [9]–[11].Advanced approaches such as LightGBM and Gradient Boosting have been introduced for handling larger datasets and achieving scalability, making them suitable for practical healthcare applications. These systems have demonstrated the potential to predict multiple diseases from a single input, enhancing their usability in resource-constrained settings [8], [10], [11].

Despite these advancements, most existing systems focus on disease prediction alone and do not include integrated health management features such as medication schedules, suggestions , precautions, or adherence tools like alarms. Many existing systems are restricted to disease prediction and fail to offer actionable recommendations for managing conditions. For instance, while some systems provide hospital recommendations or basic medication suggestions, they do not include features such as suggestions , precautions, or personalized advice for ongoing health management [3], [4], [6]. While these systems predict diseases effectively, they often overlook the importance of ensuring users follow the recommended treatments. Features such as alarms for medication reminders are rarely implemented, which can lead to non-compliance and reduced effectiveness [4], [7]. Many studies use datasets

limited to a small number of diseases and symptoms, which restricts the generalizability of the models. Expanding the dataset to cover a broader range of conditions remains an open challenge [2], [8], [10]. Few studies validate their systems in real-world healthcare environments, leaving a gap in understanding how these systems perform under practical conditions. This hinders their readiness for deployment in clinical or remote settings [9], [10]. Most systems lack user-friendly designs, making them inaccessible to non-technical users, especially in areas with limited technological literacy [5], [7], [11].

The novelties of this paper can be summarized as:

• **User-Friendly Interface:** A web-based platform designed for ease of use, even for users with limited technical skills, ensuring accessibility for underserved areas.

• **Integrated Health Guidance:** The system not only predicts diseases but also provides medication schedules, suggestions , precautions to aid users in managing their health.

• **Alarm Feature:** Timely reminders for medication ensure better adherence to prescribed treatments.

• **Comprehensive Model Evaluation:** Multiple machine learning models are implemented, and the best-performing model is deployed based on evaluation metrics, ensuring reliability and accuracy.

• **Curated Dataset:** The system leverages a dataset manually compiled from the medical reference QIMP17, ensuring accuracy and domain-specific reliability.

• **Scalable Application:** Designed for use in remote and underserved areas, the system bridges healthcare gaps effectively.

By addressing the limitations identified in previous studies, our system provides a holistic and practical solution for managing diseases and improving healthcare accessibility and reliability.

II. Literature Review

Machine learning techniques have demonstrated significant potential in disease prediction by analyzing user-reported symptoms and mapping them to probable medical conditions. Studies employing Random Forest, SVM, and Naïve Bayes have shown the effectiveness of these models in structured datasets, enabling early diagnosis and healthcare delivery in resource- constrained areas [1]–[3]. Further advancements include efforts to enhance model efficiency through feature engineering and optimization techniques such as Recursive Feature Elimination, as seen in studies comparing Gradient Boosting and Decision Trees [5], [15]. Recent research has also highlighted the integration of machine learning with natural language processing to allow for more intuitive user inputs, improving accessibility and adoption in real-world applications [16], [17].

The usability of disease prediction systems is critical for their widespread adoption, especially in underserved regions. Several studies have combined machine learning models with intuitive web-based platforms to enable non-technical users to input symptoms and receive predictions. For example, systems integrating KNN and Logistic Regression were specifically designed to cater to diverse populations by offering multilingual support and simplified interfaces [6], [7]. Another noteworthy approach is integrating disease prediction with hospital recommendations, providing users with actionable next steps based on diagnosis, a feature that enhances the practical utility of these platforms [18], [19]. Furthermore, research focusing on smartphone-based applications with offline functionality has expanded the reach of these systems to rural and remote areas with limited internet access [4], [19].

The scalability of machine learning models is essential to accommodate diverse datasets and ensure broader applicability. Studies utilizing large-scale datasets with over 200,000 samples demonstrated the robustness of Weighted KNN and LightGBM models in predicting diseases across varied conditions and populations [8], [10] . Additionally, techniques such as federated learning have been introduced to train models on distributed datasets, ensuring privacy preservation while improving performance [15], [20]. Efforts to include unstructured data, such as clinical notes and electronic health records, have also been undertaken, with Transformer-based architectures showing promise in processing such data types effectively [17], [20].

Comparative evaluations have played a crucial role in determining the most effective algorithms for disease prediction. A review of over 50 studies concluded that Random Forest consistently outperformed other models, particularly in scenarios requiring high accuracy and reliability [10], [11]. Gradient Boosting and XGBoost were found to be highly effective for datasets with complex relationships, whereas simpler models like Logistic Regression performed well on smaller, homogeneous datasets [12], [15], [18]. Additionally, the use of feature reduction methods, such as Principal Component Analysis (PCA), has been shown to improve the performance of algorithms like Decision Trees and LightGBM by addressing overfitting and enhancing computational efficiency [9], [15].

Beyond prediction, modern systems are increasingly incorporating health management features to provide holistic support to users. For example, one study developed a platform that combined symptom-based prediction with medication schedules, suggestions , precautions. This system also included alarms for medication reminders, ensuring adherence and enhancing treatment outcomes [13], [14]. Another study explored the integration of wearable devices for real-time health monitoring, allowing for dynamic updates to recommendations based on users' physiological data [14], [20]. Furthermore, systems capable of suggesting alternative treatments and monitoring side effects are emerging, reflecting a shift toward more personalized healthcare solutions [18], [19].

Future research should focus on expanding datasets to include a wider range of diseases, symptoms, and demographics. Federated learning and distributed data processing can be further explored to maintain privacy while leveraging large, diverse datasets [15], [20]. Real-world deployment and validation in clinical and community settings are essential to assess

the usability and effectiveness of these systems in practice [9], [17]. Moreover, incorporating advanced technologies like deep reinforcement learning and generative models could enable more adaptive and personalized healthcare solutions [18], [20]. Finally, the integration of real-time monitoring tools, wearable devices, and multi-modal data inputs can significantly enhance the utility and scalability of machine learning-based disease prediction systems.

III. Methodology

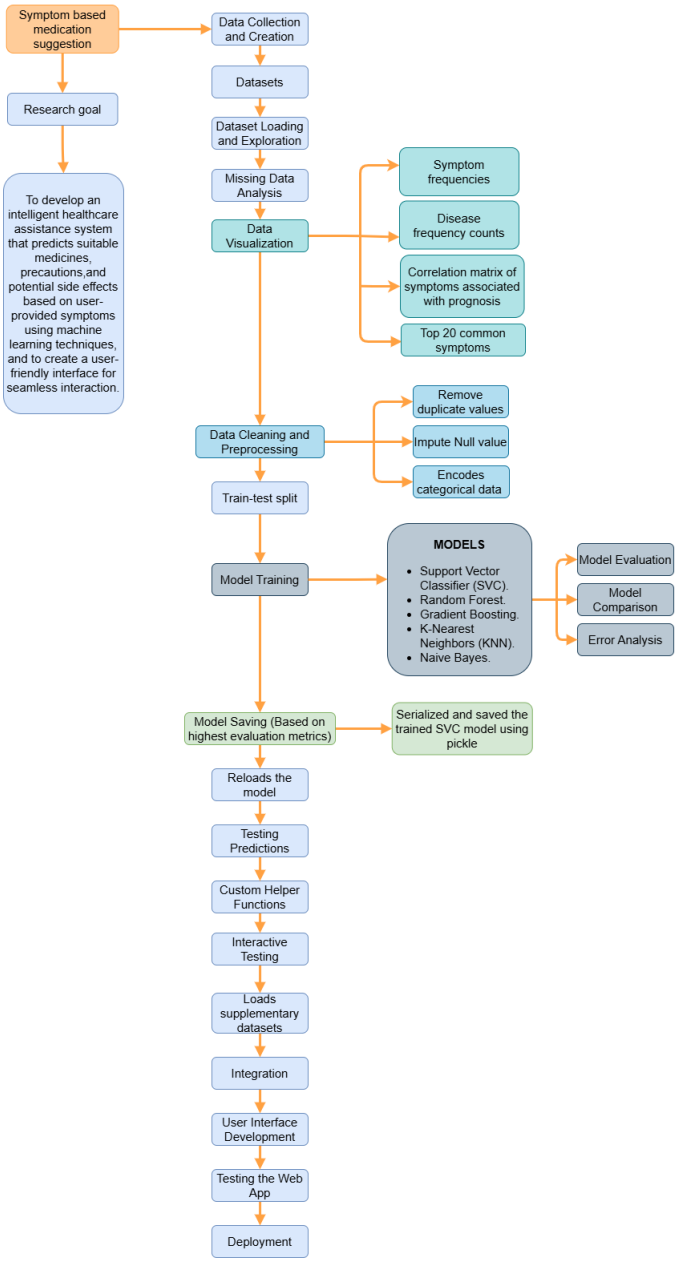

Fig 1: Flowdiagram of Methodology

The system is designed to predict diseases based on user symptoms, providing personalized recommendations for medication, workouts, diets, and precautions. It integrates machine learning models, data preprocessing, and a user-friendly interface to facilitate disease prediction and improve patient compliance through medication reminders. The complete process involves data collection, model training, disease prediction, and providing tailored recommendations for users. Below is the detailed flow of the system:

• **Data Collection and Preprocessing** The first step in the system involves gathering comprehensive data that can be processed and used by the machine learning models. The goal is to collect accurate and relevant symptom data, along with disease-specific information, to ensure the models can make reliable predictions.The dataset includes over 400 samples with 134 symptom features, representing a variety of symptoms and 46 diseases. Categorical disease labels are encoded using LabelEncoder to convert them into a format suitable for machine learning algorithms. Symptom features are scaled using RobustScaler to handle outliers and standardize feature magnitudes. The dataset is split into training (70%) and testing (30%) subsets, ensuring that the models can be evaluated accurately and tested for generalization.

• **Model Training and Evaluation** Once the data is preprocessed, the next step is to train various machine learning models to predict diseases based on the input symptoms. The objective is to select the best-performing model based on multiple evaluation metrics to ensure reliable predictions. Model Selection: Multiple machine learning models are trained, including Support Vector Classifier (SVC), Random Forest Classifier, Gradient Boosting Classifier, K-Nearest Neighbors (KNN), and Multinomial Naïve Bayes. The models are evaluated using key metrics such as accuracy, precision, recall, and F1-score. This allows for a comprehensive assessment of each model's performance. The SVC model, which provides the best performance, is selected for disease prediction. The trained model is then serialized and saved using pickle, making it reusable for future predictions without retraining.

• **Disease Prediction and User Interaction** After selecting the model, the next goal is to integrate the system with a user interface that allows individuals to enter their symptoms and receive disease predictions along with personalized recommendations. This phase ensures that users can easily interact with the system to get valuable insights. The user inputs symptoms via the web interface in a comma-separated format. Once the disease is predicted, the system retrieves related information, including precautions, medications, workouts, and diet recommendations, from predefined datasets and displays them to the user.

• **Medication Reminder System** The final objective is to improve patient compliance by integrating a medication reminder system. This feature ensures that users not only receive accurate predictions but also adhere to their prescribed treatment regimens. After predicting the disease, the system offers users the option to set alarms for taking their medications based on the prescribed schedule. The alarm system notifies users at the appropriate times to take their medications, ensuring compliance and promoting better health outcomes.

The user-friendly interface allows users to manage their symptoms, receive personalized advice, and set reminders, ensuring a seamless and comprehensive experience for disease management.

A. *Dataset Details*

The dataset used in this project is inspired by symptom-disease associations from various journals and articles on disease prediction, and the data itself was manually curated from the QIMP17 medical reference book. It consists of 414 samples, each corresponding to a specific disease and a set of symptoms. The dataset contains 134 columns, with 133 features representing individual symptoms, and a target column for the disease prognosis. The symptoms are encoded as binary values (0 or 1), indicating the presence or absence of each symptom for a given patient. The target variable is the disease diagnosis, which is categorized into 46 different diseases.

Aspect	Description
Source	The dataset is derived from symptom-disease relationships found in disease prediction journals and articles, manually compiled from the QIMP17 medical reference.
Samples	414
Features	134 total: 133 symptom columns and 1 target column for disease diagnosis (prognosis)
Diseases	46 diseases
Symptom Encoding	Symptoms are represented as binary values (0 or 1), indicating presence or absence of each symptom.
Target Variable	Disease diagnosis (encoded as numerical values)

B. *Data Preprocessing*

*1) **Handling Missing Values**:* The dataset was thoroughly examined for missing values using the isnull().sum() function, and it was determined that no missing values were present across any of the columns. This confirms that the dataset is complete and does not require any imputation or removal of data. The absence of missing values ensures the integrity of the data, facilitating a streamlined machine learning workflow without the need for additional data cleaning processes. Consequently, this allows the model to be trained on a full and consistent dataset, avoiding any potential issues related to data loss, bias, or inaccuracies in the prediction process.

*2) **Categorical Encoding**:* To prepare the dataset for machine learning modeling, the prognosis column, which represents the disease diagnosis, was treated as a categorical feature. Label Encoding was applied to transform the categorical disease labels into corresponding numerical values. The LabelEncoder from scikit-learn was utilized to assign a unique integer to each disease category. For instance, "(Vertigo) Paroxysmal Positional Vertigo" was encoded as 0, "AIDS" as 1, and so on, covering all 46 distinct diseases in the dataset. This encoding method ensures that each disease is represented by a distinct numerical identifier, facilitating compatibility with machine learning algorithms that do not inherently understand categorical data. This transformation of the target variable into numerical values allows the model to effectively learn and predict the disease class based on the given symptoms. The encoded values are stored in the variable Y, which is then used for model training, validation, and evaluation.

*3) **Visualization of Disease Distribution**:* A bar plot was generated to visualize the frequency of each disease in the prognosis column, providing insights into the dataset's composition. The value_counts() function was used to calculate the occurrences of each disease, and the results were plotted using Seaborn with a clear and professional layout. The bar plot **[Fig:2]** revealed that all diseases in the prognosis column are represented with equal frequency, indicating a perfectly balanced dataset. Such uniform distribution is highly advantageous for machine learning, as it eliminates the risk of class imbalance, ensuring that the model is not biased toward any particular disease category. This balanced representation enhances the fairness and reliability of the training process, leading to more accurate and equitable predictions across all disease classes.

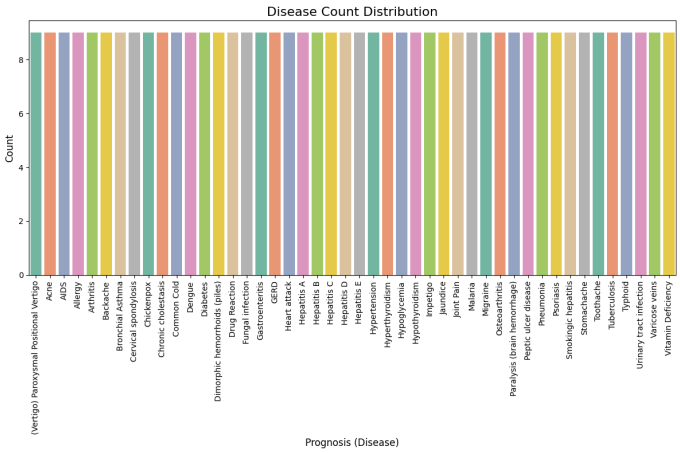

Fig 2: Disease count Distribution

*4) **Visualization of Symptom Frequencies**:* A bar plot was generated to visualize the 20 most frequently occurring symptoms across all diseases**[Fig:3]**. By summing the binary values of each symptom column, the symptoms with the highest occurrences were identified. This helps in understanding which symptoms are most prevalent in the dataset. A

heatmap was created to highlight the top 20 most common symptoms associated with Allergy**[Fig:4]**. This visualization offers insights into the symptom profile for a specific disease, providing valuable context for both dataset understanding and model interpretation.

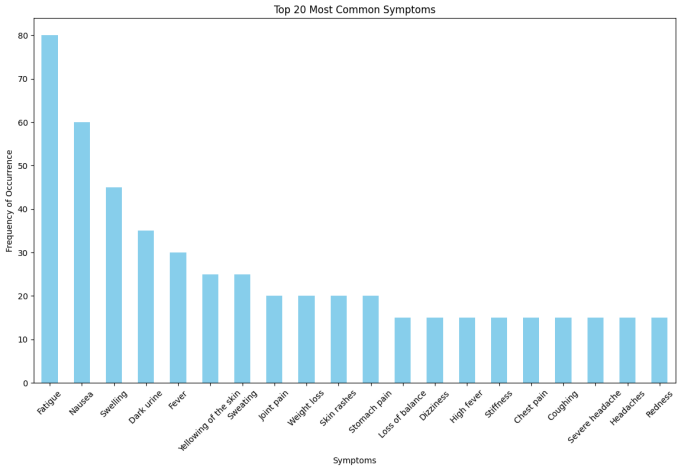

Fig 3: Top 20 Common Symptoms

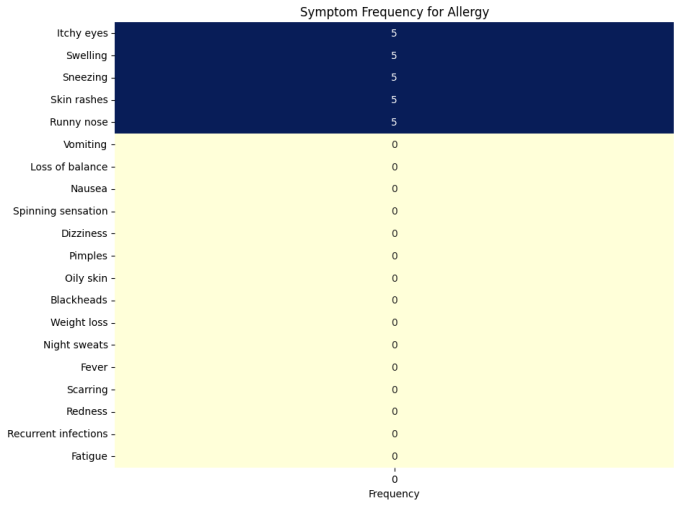

Fig 4: Symptom frequency for Allergy

*5) **Correlation Analysis of Features with Prognosis**:* The analysis investigates the relationships between symptoms and the target variable prognosis using correlation metrics. An initial heatmap of selected features, including Dizziness and Nausea, revealed weak correlations, indicating limited linear dependence**[Fig:5]**. A focused correlation analysis identified the top three features most associated with prognosis, including Diarrhea or constipation (correlation: 0.21) and Frequent infections or slow healing (correlation: 0.19)**[Fig:6]**. While these features show moderate relevance, the overall correlations are relatively weak, suggesting the need for additional feature engineering or consideration of non-linear relationships for better predictive modeling.

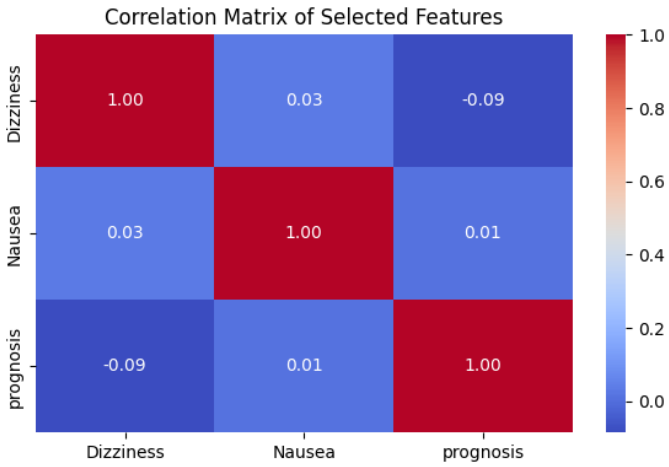

Fig 5: Correlation Matrix of selected Features

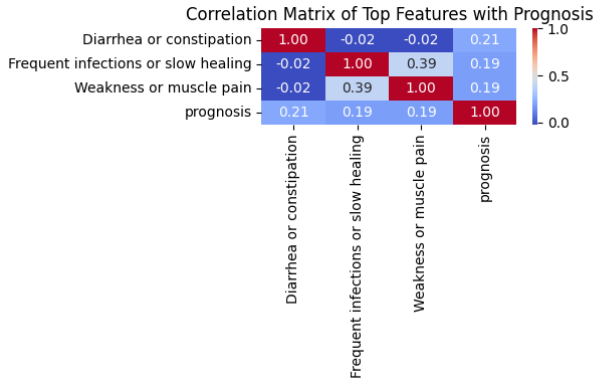

Fig 6: Correlation matrix of Top features with prognosis

Top Correlated Features with Prognosis:

prognosis	1.000000
Diarrhea or constipation	0.213313
Frequent infections or slow healing	0.187388
Weakness or muscle pain	0.187388
Hair loss or brittle nails	0.187388
Fatigue or low energy	0.187388
Dry or pale skin	0.187388
Swollen veins	0.179059
Aching legs	0.179059
Skin discoloration	0.179059

*6) **Normalization**:* Using the MinMaxScaler, each feature was scaled to a range between 0 and 1. This transformation ensures that the data is proportionally adjusted without altering its distribution, making it suitable for algorithms sensitive to feature magnitude, such as distance-based methods.

$$X_{\text{normalized}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{1}$$

*7) **Standardization**:* With the StandardScaler, the features were transformed to have a mean of 0 and a standard deviation

of 1. This process centers the data and removes scaling differences, which is particularly useful for algorithms that assume normality in the input data.

$$X_{\text{standardized}} = \frac{X - \mu}{\sigma} \tag{2}$$

- X: The original value of the feature.
- μ: The mean of the feature in the dataset.
- σ: The standard deviation of the feature in the dataset.

This formula adjusts the feature values to a standard normal distribution (mean = 0, standard deviation = 1), which is useful for algorithms assuming Gaussian distribution.

C. *Train-Test Split*

To evaluate the performance of machine learning models, the dataset was split into training and testing subsets using the train_test_split function from scikitlearn. This process ensures that the model is trained on one portion of the data while being validated on a separate unseen portion, mimicking real-world scenarios. The feature set (X) and the target variable (y) were first scaled using the RobustScaler to handle outliers effectively. Subsequently, the dataset was divided into 70% training data and 30% testing data, with a fixed random state for reproducibility. This split allows for a balanced and reliable assessment of the model's accuracy, precision, recall, and F1 score while ensuring that the model generalizes well to unseen data.

D. *Machine learning Algorithms*

*1) **Support Vector Classifier (SVC)**:* The Support Vector Classifier (SVC) is a robust and widely used machine learning algorithm designed for both linear and non-linear classification tasks. In this project, an SVC with a linear kernel was employed to classify diseases based on symptoms. SVC is particularly effective in handling high-dimensional datasets, making it suitable for the multi-feature symptom dataset used in this study. The SVC model achieved high performance, with strong metrics across accuracy, precision, recall, and F1 score. These results indicate the model's effectiveness in correctly classifying the disease based on the symptoms. A heatmap of the confusion matrix provided a visual representation of true positives, false positives, and false negatives across all classes. The majority of predictions were accurate, with minimal misclassifications. The detailed classification report highlighted the per-class performance of the model, showcasing its balanced handling of multiple disease categories. Metrics such as precision and recall were consistent, further confirming the model's reliability.

*2) **Random Forest Classifier**:* The Random Forest Classifier is an ensemble learning method that operates by constructing multiple decision trees during training. Each tree is trained on a random subset of the data, and their predictions are aggregated to improve accuracy and reduce overfitting. In this project, the Random Forest model was trained on the dataset with 100 estimators (trees). The model demonstrated strong performance, as indicated by its high accuracy, precision, recall, and F1 score. Additionally, the confusion matrix and classification report further confirmed its effective handling of multiple disease categories, making it a reliable choice for disease prediction.

*3) **Gradient Boosting Classifier**:* The Gradient Boosting Classifier is an ensemble machine learning algorithm that builds strong models by combining the predictions of several weak learners, typically decision trees. It works by iteratively training each tree to correct the errors made by the previous one, thus improving the overall model's accuracy. In this project, the Gradient Boosting model, with 100 estimators, was trained on the scaled dataset and achieved good performance across accuracy, precision, recall, and F1 score metrics.The confusion matrix and classification report further demonstrated the model's ability to handle multi-class classification, making it an effective tool for disease prediction.

*4) **K-Nearest Neighbors (KNN) Classifier**:* The K-Nearest Neighbors (KNN) classifier is a simple yet powerful algorithm that classifies data points based on the majority vote of their nearest neighbors in the feature space. In this project, the KNN model was trained with 5 neighbors and evaluated on the scaled dataset. The model demonstrated reasonable performance, achieving good metrics across accuracy, precision, recall, and F1 score. The confusion matrix highlighted the model's ability to correctly classify multiple disease categories, while the classification report provided a detailed performance overview. KNN is particularly useful for its simplicity and ease of implementation, though it may be less efficient on larger datasets compared to more complex models like SVC or Random Forest.

*5) **Multinomial Naive Bayes (NB) Classifier**:* The Multinomial Naive Bayes (NB) classifier is a probabilistic model commonly used for classification tasks involving discrete features. It is particularly effective when dealing with multi-class classification problems, such as disease prediction in this case. The model was trained on the scaled dataset and evaluated based on various metrics, including accuracy, precision, recall, and F1 score. The performance metrics indicated the model's effectiveness in predicting diseases based on symptom data. Additionally, the confusion matrix provided insights into the model's classification accuracy across different disease categories. The classification report further highlighted the model's performance, making it a valuable tool in predicting medical conditions. Despite being a simpler model, Naive Bayes offers a good baseline for text or symptom-based classification tasks.

E. *Model Evaluation*

The SVC (Support Vector Classifier) model, after being trained, was saved and loaded using the pickle module for further use. The model achieved an impressive accuracy of 88%, indicating it correctly predicted the outcomes in 88% of the test data. Additionally, the precision was 92%, demonstrating that 92% of the positive predictions made by the model were correct. The F1 score further supports the model's performance, balancing both precision and recall. These high

evaluation metrics showcase the model's robustness and effectiveness in making accurate predictions.

F. User Interfaces

The system features multiple user interfaces corresponding to the HTML templates used in various routes. Below is a detailed breakdown of each interface:

- **Index Page (`index.html`):** Serves as the homepage where users can input symptoms.
  - **Navigation Bar:** Provides links to Home, About Us, and Contact Us page.
  - **Search Box:** Allows users to input symptoms for predictions and includes user-friendly placeholder text to guide input.
  - **Buttons and Forms:** A search button for submitting symptoms and forms for data submission to the backend.

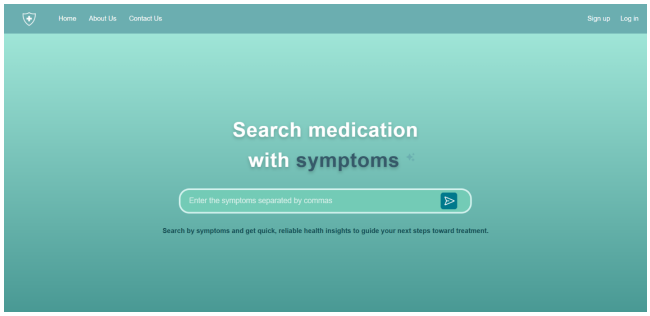

Fig 7: Index Page

- **Result Page (`result.html`):** Displays the predicted disease and related medication suggestions.
  - **Search Box:** Similar functionality to the index page for re-entering symptoms.
  - **Results Section:** Displays information about the predicted disease, medicines, precautions, suggestions, and side effects.

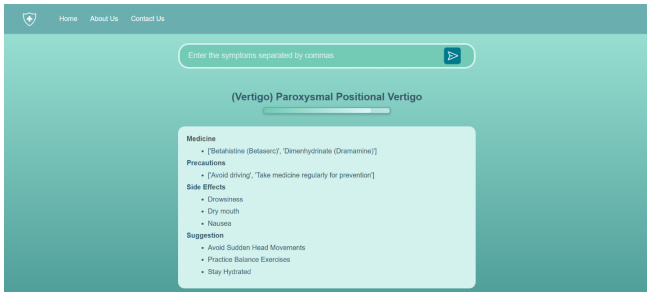

Fig 8: Result Page

- **Signup and Login-Related Pages:**
  - **`aftersignup.html`:** Dashboard for users after signing up or logging in.
  - **`aftersignupresult.html`:** Results displayed for signed-up or logged-in users.

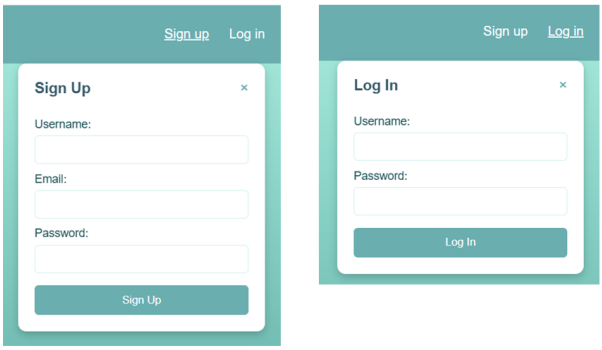

Fig 9: Signup& Login Page

- **About Us Pages:**
  - **`aboutus.html`:** Provides information about the project.
  - **`afteraboutus.html`:** A logged-in user version of the About Us page.

Fig 10: About Us

- **Contact Us Pages:**
  - **`contact.html`:** A page for users to contact the team.
  - **`aftercontact.html`:** A logged-in user version of the Contact Us page.

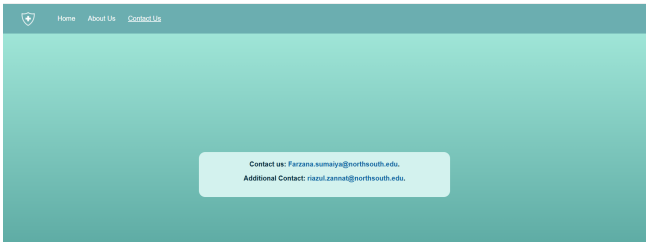

Fig 11: Contact Us

- **Check My Symptoms Page (`checkmysymptoms.html`):**
  Designed for logged-in users to review their symptoms and predictions.
  - **Symptoms Display Section:** Lists user-entered symptoms. Also, displays a table with predicted diseases, medicines, precautions, side effects, and suggestions.

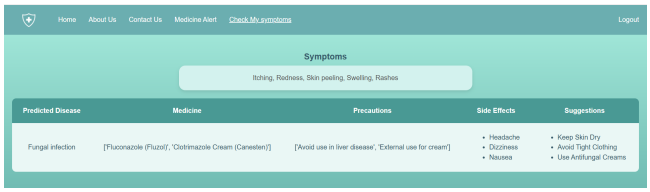

Fig 12: Symptoms Display

- **Medicine Alert Page (`medicinealert.html`):** A page for logged-in users to manage medicine alerts.
  - **Prescription Display Section:** Displays diseases and prescribed medicines and uses conditionally rendered elements (`% if %`) to repeat data or show messages.
  - **Set Alarms Button:** Allows users to set medicine schedule alarms.
  - **Alarm Message Display:** A placeholder for showing alarm-related messages.
  - **Custom Popup:** For medicine alarm notifications.
  - **Dynamic JavaScript Content:** Handles alarm-related interactions.

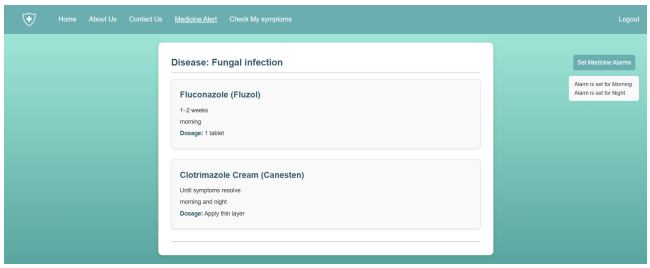

Fig 13: Medicine Alert

- **Feasibility of User Interfaces:**
  - **Usability:** The UI design is well-suited for a medical consultation system, with intuitive features like symptom input fields and clear result displays. User-friendliness remains key to ensuring effectiveness.
  - **Performance:** Flask enables lightweight and efficient dynamic page rendering, making it suitable for small to medium-sized applications.
  - **Maintenance:** Separate templates for pages such as "About Us" and "Contact" ensure clarity but could benefit from shared components (e.g., headers, footers) to minimize redundancy and simplify updates.
  - **Scalability:** To support a larger user base, implementing caching for frequently accessed pages or integrating a front-end framework could optimize response times and improve scalability.

G. *Tech Stack*

- **Backend**
  1) **Flask (Python Framework):** Serves as the backend framework, enabling routing and smooth interaction with HTML through the Jinja templating engine.
  2) **MySQL:** Used as the database for storing user data, symptoms, predictions, and other relevant information.
  3) **Flask-MySQLdb:** Provides integration between Flask and MySQL.
  4) **Flask-Bcrypt:** Ensures secure password hashing for user authentication.
- **Frontend**
  1) **HTML/CSS:** Used for structuring and styling the web pages, and templates are dynamically rendered via Flask.
  2) **JavaScript:** Likely used for handling dynamic elements and user interactions.
- **Machine Learning**
  1) **Scikit-learn:** Utilized for loading and applying the saved Support Vector Classifier (SVC) model (`svc.pkl`).
  2) **NumPy:** Enables efficient vectorized operations required for model predictions.
  3) **Pandas:** Used for reading and processing datasets.
- **Other Libraries and Tools**
  1) **Pickle:** Handles serialization and deserialization of the saved machine learning model.
  2) **Jinja2:** Flask's built-in templating engine for dynamic HTML rendering.
  3) **Audio API:** Plays alarm sounds with specific restrictions based on user interactions.
  4) **DOM API:** Interacts with HTML elements for tasks such as fetching times, displaying messages, and managing popup controls.
  5) **JavaScript Date Object:** Facilitates time calculations for alarm scheduling and determining delays.
  6) **Fetch API:** Used for making asynchronous HTTP requests to the Flask backend.

***Feasibility of the Tech Stack**:*

- **Usability:** The stack is well-suited for developing a moderately complex web application, offering rapid development and deployment capabilities.
  - The use of Flask for the backend ensures seamless integration with Python-based machine learning models.
  - HTML, CSS, and JavaScript provide an intuitive structure for the frontend.
- **Performance:** Flask, coupled with MySQL, delivers sufficient performance for small-to-medium workloads.
- **Maintenance:**
  - Flask offers simple and modular routing.
  - MySQL supports structured, relational data management with robust community resources.
  - Libraries like Scikit-learn and Pandas are industry standards with strong support for machine learning workflows.
- **Security:** Security is a strong aspect of this stack:
  - Flask-Bcrypt ensures secure password hashing.

  - Flask provides built-in methods for CSRF protection.
  - Additional measures, such as input validation, session security, and HTTPS implementation, are essential to reduce sensitivity.

Overall Strengths and Challenges:

- **Strengths:**
  - Lightweight stack and simple deployment process.
  - Modular design with separate UIs for different functionalities.
  - Secure user authentication using Bcrypt.
  - Easy integration of the predictive model using Scikit-learn and Pickle.
- **Challenges:**
  - **Scalability:** Flask may not handle heavy concurrent loads effectively.
  - **Database Optimization:** Without proper indexing, the MySQL database could slow down as the user base grows.
  - **Frontend Experience:** Limited to basic HTML, CSS, and JS, which might not provide a modern user experience.
  - **Model Updates:** If the SVC model needs regular updates, manual intervention might be required unless automated retraining pipelines are implemented.

Reasons for Choosing the Tech Stack

- **Backend: Flask:**
  - **Lightweight:** Ideal for small-to-medium web applications, with minimal overhead.
  - **Flexibility:** The absence of a predefined structure enables custom design and architecture.
  - **Rich Ecosystem:** Access to plugins such as Flask-Bcrypt (security), Flask-MySQLdb (database integration), and Flask-WTF (form handling) enhances functionality.
- **MySQL:** Well-suited for structured data types like user profiles, symptoms, and predictions.
- **Flask-Bcrypt:** Provides reliable password hashing to safeguard user credentials.
- **Machine Learning: NumPy, Pandas, and Scikit-learn:**
  - **Efficient Data Handling:** Enables seamless data manipulation and machine learning pipeline creation.
  - **Integration:** Easily integrates with Flask for backend data processing and analysis.
- **Frontend: HTML, CSS, and JavaScript:**
  - **Simplicity:** Supports the creation of server-rendered templates with minimal complexity.
  - **Flexibility:** When combined with frameworks, ensures the design of modern, responsive user interfaces.
- **Other Libraries and Tools:**
  - **Pickle:** Facilitates saving and loading trained models, eliminating the need for re-training.
  - **Jinja2:** Allows seamless integration of backend logic into the frontend for dynamic template rendering.
  - **DOM API:** Enables fast and direct manipulation of the Document Object Model (DOM) for real-time user interface updates without backend communication.
  - **Audio Alerts:** Utilizes the native Audio API to integrate sound notifications for better usability.
  - **Fetch API:** Facilitates real-time data exchange between the frontend and backend without requiring full-page reloads.

Additional Benefits of the Tech Stack:

- Enables quick development, testing, and iteration of features.
- Relies on open-source libraries, significantly reducing development and deployment costs.

IV. Results

Evaluation metrics

- **Accuracy:** It measures the overall correctness of the model (i.e., the proportion of correct predictions). It is calculated as:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Population}} \tag{3}$$

- **Precision:** Precision is the ratio of correctly predicted positive observations to the total predicted positives. It tells us how many of the predicted positive cases were actually positive. It is calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{4}$$

- **Recall:** Recall is the ratio of correctly predicted positive observations to all observations in the actual class. It tells us how many actual positive cases were correctly identified by the model. It is calculated as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{5}$$

- **F1 Score:** The F1 score is the weighted average of precision and recall. It balances both the false positives and false negatives, making it a good metric when we want to balance precision and recall. It is calculated as:

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{6}$$

Comparison Table of Evaluation Metrics:
Comparative evaluation of all the machine learning models used in this study.

Models	Accuracy	F1 Score	Precision	Recall
SVC	0.88	0.88	0.92	0.88
Random Forest Classifier	0.79	0.80	0.86	0.79
Gradient Boosting	0.78	0.79	0.87	0.78
KNN	0.73	0.75	0.87	0.73
Multinomial NB	0.84	0.84	0.87	0.84

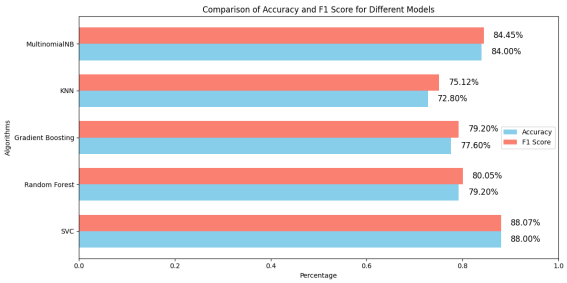

Fig 14: Comparison of Accuracy & F1 Score

Evaluation Metrics Analysis: Among the evaluated models, SVC demonstrated the best overall performance, making it the optimal choice for our project. While Random Forest and Gradient Boosting showed decent accuracy (0.79 and 0.78, respectively) and high precision, their recall and F1 scores lagged behind, indicating potential challenges in identifying true positives consistently. Multinomial Naive Bayes performed well with balanced metrics (accuracy and F1 score of 0.84), but it fell short of SVC's precision and recall. KNN, with the lowest accuracy (0.73) and recall, proved less reliable overall. In contrast, SVC achieved the highest accuracy (0.88), precision (0.92), and balanced F1 score and recall (0.88), ensuring both precision and consistency. Its ability to handle complex decision boundaries and minimize false positives makes it the most suitable model for our project requirements.

Confusion Matrix Analysis

The analysis of confusion matrices reveals distinct performance trends across the models.The analysis of the confusion matrices highlights the superiority of the Support Vector Classifier (SVC) over other models. SVC achieves the highest number of true positives (110) while maintaining the lowest false positives (15) and false negatives (15), indicating its effectiveness in correctly identifying both positive and negative cases. In comparison, Random Forest and Gradient Boosting have higher false positives and false negatives, making them less reliable. Multinomial Naive Bayes performs well with balanced results but still falls short of SVC in minimizing errors. K-Nearest Neighbors (KNN) has the highest false positives (34) and false negatives (34), making it the least suitable model. SVC's ability to minimize both types of errors ensures consistent and dependable performance, making it the ideal choice for the project.

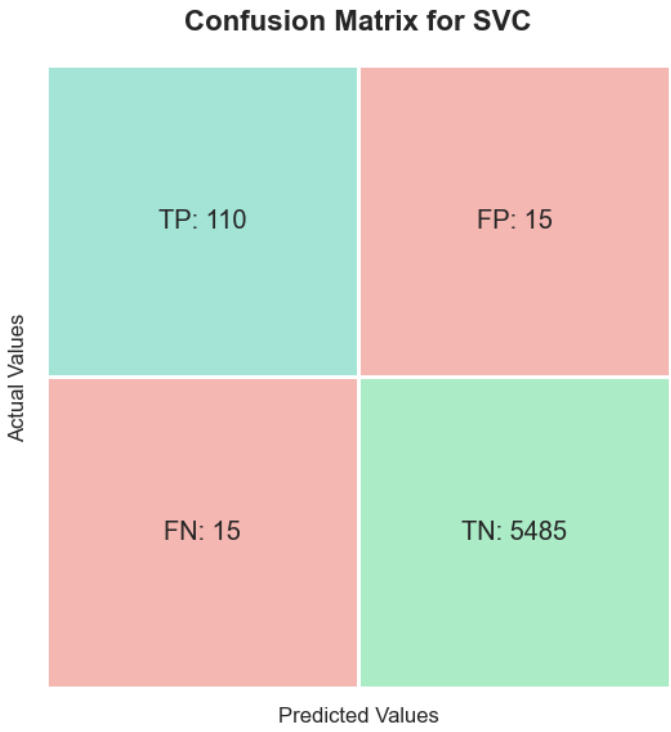

Fig 15.a: Confusion matrix for SVC

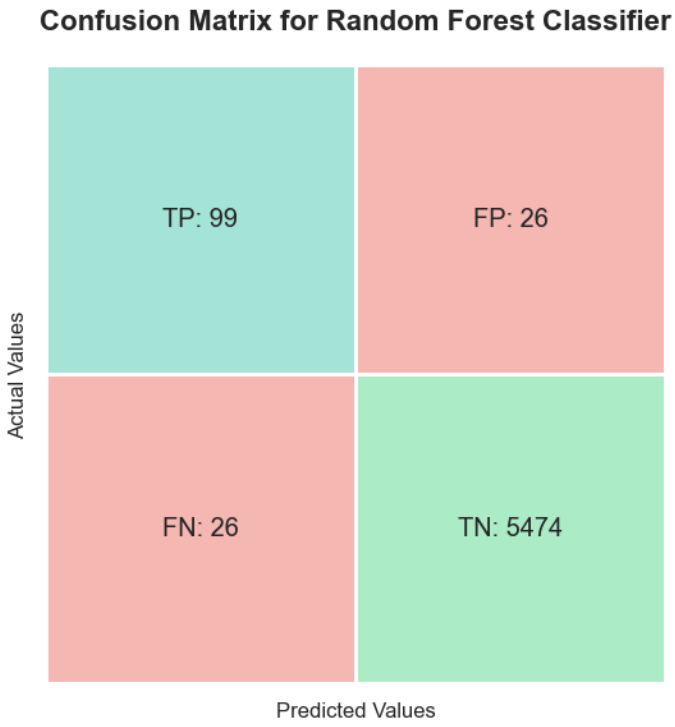

Fig 16.a: Confusion matrix for Random Forest Classifier

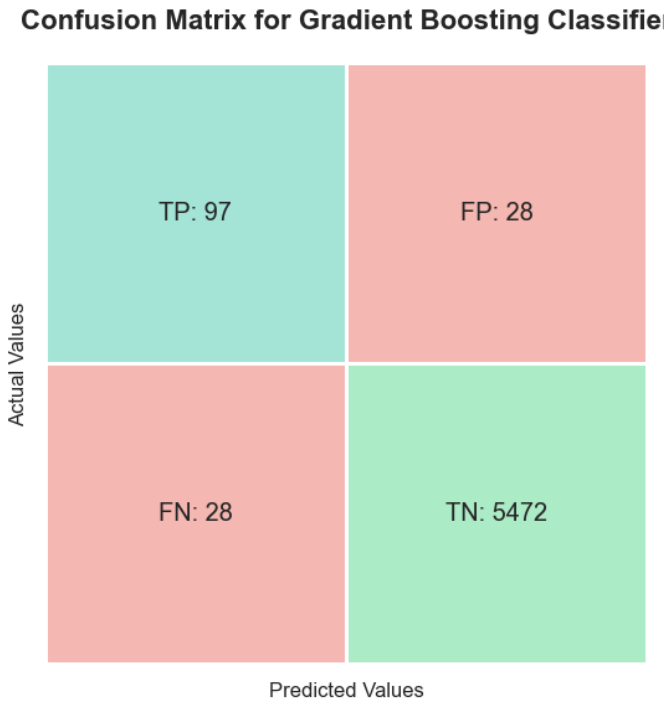

Fig 17.a: Confusion matrix for Gradient Boosting Classifier

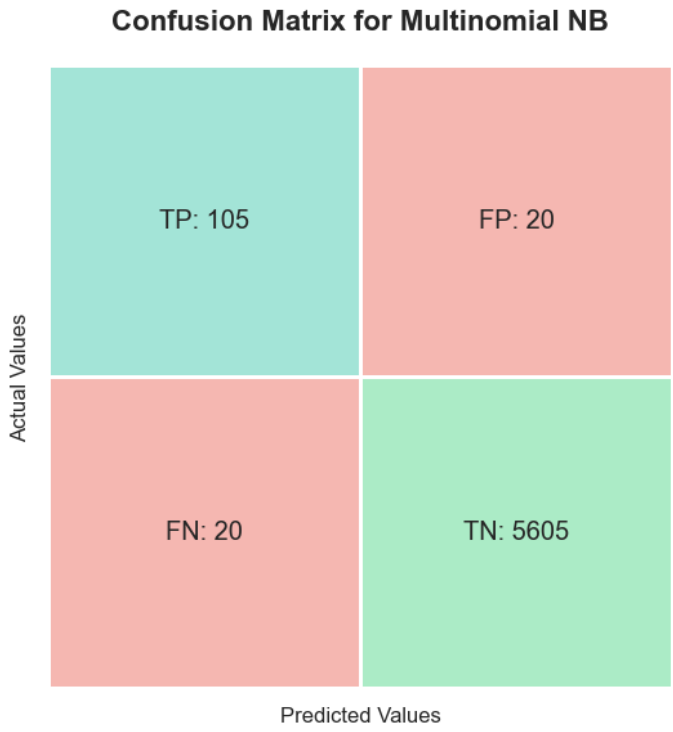

Fig 19.a: Confusion matrix for Multinomial NB

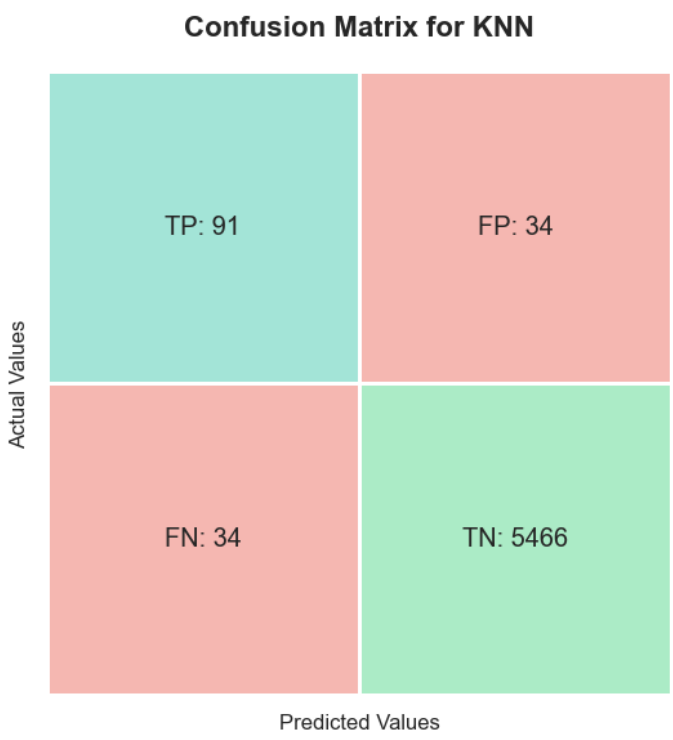

Fig 18.a: Confusion matrix for KNN

Confusion Matrix Representing With Diagonal Values

SVC demonstrates exceptional accuracy, with dominant diagonal values and minimal off-diagonal entries, showcasing its effectiveness in correctly predicting the majority of classes with minimal errors. Multinomial Naive Bayes also performs robustly, with consistently high diagonal values and minimal misclassifications, indicating strong differentiation between classes. Random Forest displays good performance, reflected in high diagonal values, but struggles with certain classes, as indicated by more frequent off-diagonal values. Gradient Boosting shows moderate confusion, with lower diagonal values for some classes and more prominent off-diagonal entries, suggesting misclassification in overlapping feature spaces. K-Nearest Neighbors (KNN) exhibit the weakest performance, with significantly lower diagonal values and frequent off-diagonal entries, reflecting substantial confusion likely caused by overlapping feature distributions or class imbalances. Overall, SVC stands out as the most reliable model, followed by Multinomial Naive Bayes, while KNN requires further optimization for improved performance.

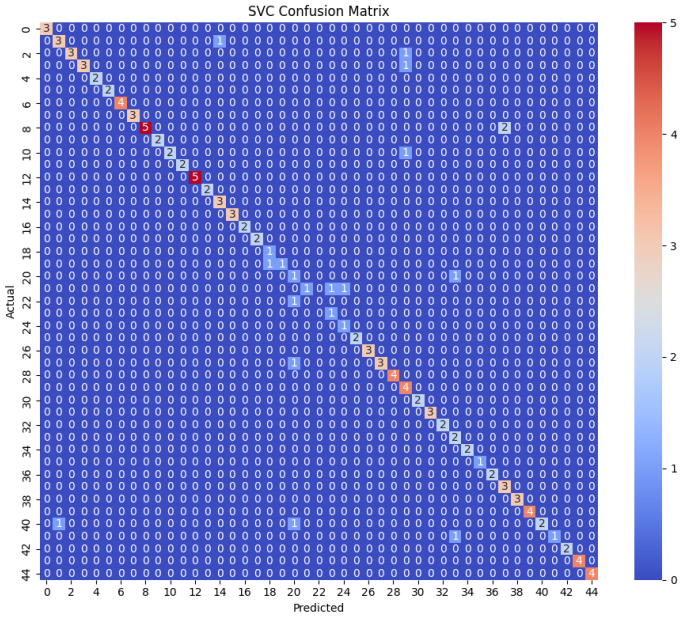

Fig 15.b: Confusion matrix for SVC

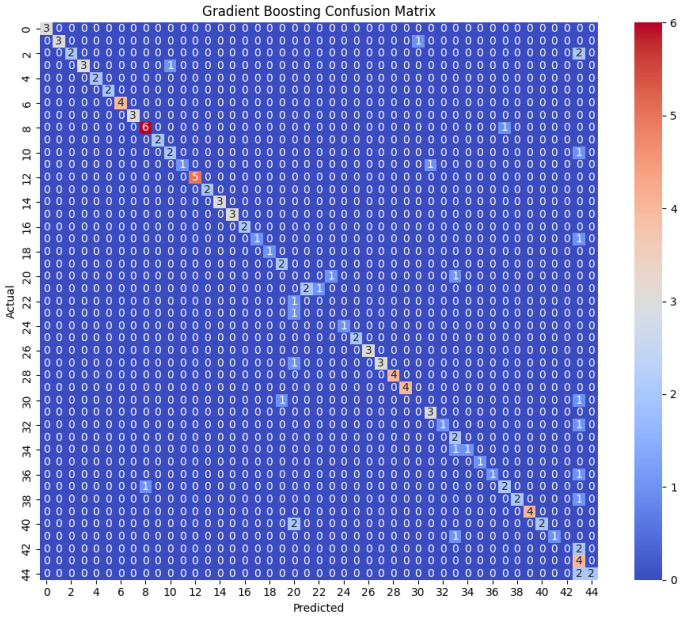

Fig 17.b: Confusion matrix for Gradient Boosting Classifier

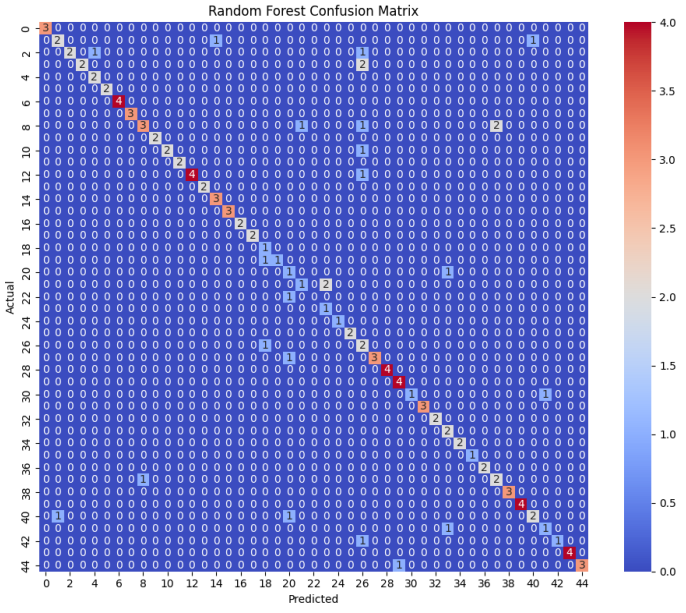

Fig 16.b: Confusion matrix for Random Forest Classifier

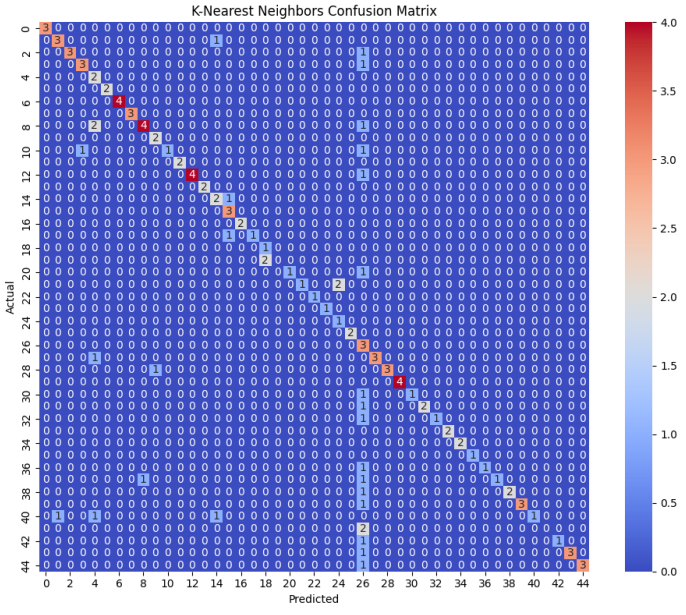

Fig 18.b: Confusion matrix for KNN

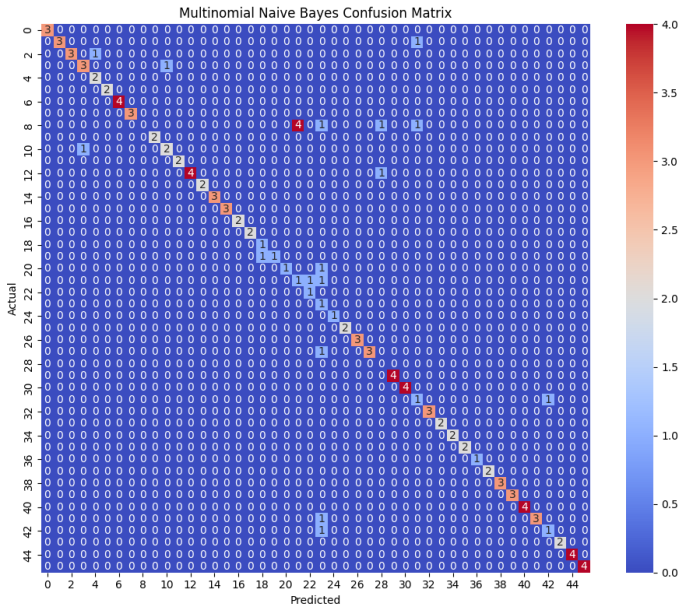

Fig 19.b: Confusion matrix for Multinomial NB

Error Analysis:

When conducting error analysis using false positives (FP) and false negatives (FN) represented on a bar graph, the results clearly highlight the differences in model performance. SVC stands out with the smallest error rates, showcasing its strong ability to minimize both types of errors effectively. Multinomial Naive Bayes follows closely, maintaining low FP and FN values, further solidifying its reliability. Random Forest exhibits moderate error rates, reflecting occasional challenges in handling certain classes. Gradient Boosting shows slightly higher FP and FN values, indicating greater confusion and misclassification in its predictions. Finally, K-Nearest Neighbors (KNN) demonstrates the highest error rates, with prominent FP and FN values, highlighting significant misclassifications, likely due to overlapping feature spaces or class imbalances. This analysis underscores the superior precision of SVC, followed by Multinomial Naive Bayes, while KNN requires optimization to enhance its performance.

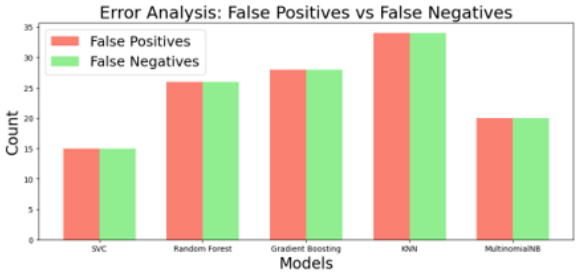

Fig 20: Error Analysis of different Models

Future Work:

- **Natural Conversation for Symptom-Based Medication**

Develop a system that recommends medication through natural, free-form conversations, eliminating rigid input formats like a real doctor-patient interactions.

- **Doctor-Patient Chatbot Environment**

Create an interactive chatbot that mimics a doctor-patient environment, enabling users to engage in human-like conversations and receive medication suggestions in a manner consistent with medical practice.

- **Scalable Dataset for Comprehensive Recommendations**

Expand the dataset to cover a broader range of symptoms, ensuring medication suggestions are comprehensive and applicable to diverse medical conditions.

- **Multilingual Support**

Implement support for multiple languages to make the system accessible to a global audience.

- **Mobile Application Development**

Develop a user-friendly mobile application for seamless access to the system.

- **Voice Message Integration**

Introduce voice messaging capabilities for easier and more convenient user interactions.

- **Specialist Recommendations**

Incorporate a feature that identifies and suggests the appropriate type of medical specialist to consult for severe symptoms.

- **Collaboration with Medical Professionals**

Collaborate with licensed medical professionals to improve the accuracy and reliability of medication suggestions.

- **Age-Specific Medication suggestions**

Include age-specific medication guidance to the user's age range for safer and more precise suggestions.

V. Conclusion

In conclusion, our webapp shows a lot of potential for predicting diseases based on symptoms. However, there's still a clear need for integrated healthcare systems that provide complete health management solutions. That's where our Symptom-Based Medication System comes in. We're looking to fill that gap by merging precise disease prediction with tailored health recommendations like medication schedules, diet plans, and exercise routines. By utilizing advanced machine learning techniques and focusing on user-friendly features, our goal is to make healthcare more accessible, especially in remote and underserved areas. Moving forward, we will work on expanding the coverage of our datasets, testing our systems in real-life situations, and creating even more intuitive interfaces. This way, we can truly enhance the impact of these technologies on public health.

REFERENCES

[1] Rajeswari, Athish Venkatachalam Parthiban, and S. S. Sree Nandha, "Disease Diagnosis Interface Using Machine Learning Technique," Advances in Web Technologies and Engineering Book Series, pp. 141–150, Jun. 2023, doi:10.4018/978-1-6684-8306-0.ch009

[2] Manikanta Sirigineedi, Matta, Rali Surya Prakash, V. Pavan, and Poojitha Tirunagari, "Symptom-Based Disease Prediction: A Machine Learning Approach," Journal of Artificial Intelligence Machine Learning and Neural Network, no. 43, pp. 8–17, Apr. 2024, doi:10.55529/jaimlnn.43.8.17

[3] S. Lucas, M. Desai, A. Khot, S. Harriet, and N. Narkar, "SmartCare: A Symptoms Based Disease Prediction Model Using Machine Learning Approach," International Journal for Research in Applied Science and Engineering Technology, vol. 10, no. 11, pp. 709–715, Nov. 2022, doi:10.22214/ijraset.2022.47434

[4] Prajakta Khairnar, Vamsi Avula, Aditya Hargane, and Pratik Baisware, "Medicine Recommend System Using Machine Learning," International Journal of Scientific Research in Science Engineering and Technology, pp. 247–250, May 2022, doi:10.32628/ijsrset2293102

[5] R. Keniya et al., "Disease Prediction From Various Symptoms Using Machine Learning," papers.ssrn.com, Jul. 27, 2020. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3661426

[6] Kedar Pingale, Sushant Surwase, Vaibhav Kulkarni, Saurabh Sarage, and Prof. Abhijeet Karve, "Disease Prediction using Machine Learning," Dec. 2019. https://www.irjet.net/archives/V6/i12/IRJET-V6I12122.pdf

[7] A. Kumar, "Disease Prediction and Doctor Recommendation System using Machine Learning Approaches," Journal for Research in Applied Science and Engineering Technology, vol. 9, no. VII, pp. 34–44, Jul. 2021, doi:10.22214/ijraset.2021.36234

[8] S. Uddin, A. Khan, M. E. Hossain, and M. A. Moni, "Comparing different supervised machine learning algorithms for disease prediction," BMC Medical Informatics and Decision Making, vol. 19, no. 1, Dec. 2019, doi:10.1186/s12911-019-1004-8

[9] D. D. Mondal, H. Shinde, Sarang Baghele, P. Kadam, and Darshan Thengal, "Machine Learning in Healthcare: Developing an AI Assistant Doctor for Symptom-Based Disease Prediction," INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT, vol. 08, no. 11, pp. 1–9, Nov. 2024, doi:10.55041/ijsrem38474

[10] T. Bhanuteja, K. V. N. Kumar, K. S. Poornachand, C. Ashish, and P. Anudeep, "Symptoms Based Multiple Disease Prediction Model using Machine Learning Approach," International Journal of Innovative Technology and Exploring Engineering, vol. 10, no. 9, pp. 67–72, Jul. 2021, doi:10.35940/ijitee.i9364.0710921

[11] V. S, S. S, V. H, and S. S, "Disease Prediction Using Machine Learning Over Big Data," SSRN Electronic Journal, 2018, doi:10.2139/ssrn.3458775

[12] Md. E. Farooqui and Dr. J. Ahmad, "DISEASE PREDICTION SYSTEM USING SUPPORT VECTOR MACHINE AND MULTILINEAR REGRESSION," International Journal of Innovative Research in Computer Science & Technology, vol. 8, no. 4, Jul. 2020, doi:10.21276/ijircst.2020.8.4.15

[13] N. Yede, R. Koul, C. Harde, K. Gaurav, and C. S. Pagar, "GENERAL DISEASE PREDICTION BASED ON SYMPTOMS PROVIDED BY PATIENT," doi:10.51397/OAIJSE06.2021.0032

[14] I. Ibrahim and A. Abdulazeez, "The Role of Machine Learning Algorithms for Diagnosing Diseases," Journal of Applied Science and Technology Trends, vol. 2, no. 01, pp. 10–19, Mar. 2021, doi:10.38094/jastt20179

[15] M. Gadekar, S. Jamadar, P. Pachpute, S. Shinde, and S. Bhosale, "SYMPTOMS BASED DISEASE PREDICTION," Fully Refereed International Journal @ International Research Journal of Modernization in Engineering, vol. 3557, pp. 2582–5208. Available: https://www.irjmets.com/uploadedfiles/paper/issue_5_may_2022/24065/final/fin_irjmets1653367944.pdf

[16] K. Gaurav, A. Kumar, P. Singh, A. Kumari, M. Kasar, and T. Suryawanshi, "Human Disease Prediction using Machine Learning Techniques and Real-life Parameters," *International Journal of Engineering*, vol. 36, no. 6, pp. 1092–1098, Jun. 2023, doi: https://doi.org/10.5829/ije.2023.36.06c.07.

[17] K. Nishitha and S. Tirumala, "SYMPTOMS BASED DISEASE PREDICTION USING MACHINE LEARNING TECHNIQUES," Audisankara College of Engineering and Technology (AUTONOMOUS), Gudur, AP, India, 2023, doi: https://jespublication.com/uploads/2023-V14I8099.pdf.

[18] M. N. Ashok, "Symptoms Based Disease Prediction System Using Machine Learning," *International Journal for Research in Applied Science and Engineering Technology*, vol. 12, no. 3, pp. 2536–2539, Mar. 2024, doi: https://doi.org/10.22214/ijraset.2024.59394.

[19] Y. Jing and Leong, "SYMPTOM-BASED DISEASE PREDICTION SYSTEM USING MACHINE LEARNING," *Journal of Theoretical and Applied Information Technology*, vol. 98, no. 19, 2020, Available: https://www.jatit.org/volumes/Vol98No19/5Vol98No19.pdf.

[20] K. Patil, S. Pawar, P. Sandhyan, and J. Kundale, "Multiple Disease Prognostication Based On Symptoms Using Machine Learning Techniques," *ITM Web of Conferences*, vol. 44, p. 03008, 2022, doi: https://doi.org/10.1051/itmconf/20224403008.