A dissertation submitted to the **University of Greenwich** in partial fulfilment of the requirements for the Degree of

Master of Science

in

Data Science

Personalized Treatment Recommendation System
Using Machine Learning

Name: Binod Pun

Student ID: 001377611

Supervisor: Alei Duan

Submission Date: 20th December 2024

Word Count: 10693

PERSONALIZED TREATMENT RECOMMENDATION SYSTEM USING MACHINE LEARNING

XXXXX

Computing & Mathematical Sciences, University of Greenwich, 30 Park Row, Greenwich, UK.

(Submitted)

Table of Contents

List of Figures

ACKNOWLEDGEMENT

I would like to sincerely thank everyone who supported me throughout my MSc project, *Personalized Treatment Recommendation System Using Machine Learning*.
First, I want to thank my supervisor, **Dr. Alei Duan**, for their guidance, advice, and encouragement. Their support helped me stay focused and improve my work at every stage.

I am also very thankful to the **faculty and staff at the University of Greenwich**, especially the Department of Computing and Mathematical Sciences, for providing me with the resources and knowledge I needed to complete this project.

I owe a big thanks to my family and friends for their patience, encouragement, and constant support. They stood by me during challenging times and motivated me to keep going.

Lastly, I would like to acknowledge the authors and creators of the research papers, datasets, and tools that helped me in this project. Their work made this project possible.

Thank you all for being part of this journey and believing in me.

Abstract

In particular, the growing incidence of diseases suggests the importance of developing effective health care models for comprehensive patient diagnosis and treatment. The idea of this project is in developing a Personalized Treatment Recommendation System based on the ML approach to forecast diseases based on certain symptoms specified by a user and to suggest medications, diet, exercises, and other precautions.

In contrast to the most existing systems approach the problem of risk assessment in the context of one disease, our solution provides disease prediction, which makes it possible to offer respective healthcare professionals and patients of health risks. Intuitive design of the system assists patient usage as well as health care workers, avoiding the complexity of the system's technical expertise. Furthermore, the system ensures interpretability through the identification of factors that affect prediction results making this system more reliable and transparent.

This project also highlights ethical concerns particularly in adherence to the legal requirements in the operation of the health sector including the HIPAA act as well as issues to do with bias in the training data that has been used in the development of models in the healthcare sector. As for the system function, the **Streamlit Framework** ensures that the system is coordinated effectively, making it scalable and highly reliable. The use of the system for indicating diseases and offering preventive measures illustrates the usefulness of the approach in the development of individualized healthcare management. This system is absolutely fundamental in the process of applying machine learning into the clinical environment, thereby increasing the diagnostic precision and, by implication, the quality of patient treatment.

Chapter 1

Introduction

There are many AI and especially Machine learning algorithms realigning the healthcare space by improving the diagnostic accuracy and effectiveness of treatment delivery. The main feature of this project is to create the Personalized Treatment Recommendation System that will include machine learning algorithms to diagnose illness according to symptom log input, suggest medications, and give individual advice concerning description, diets, medications, precautions and workout. Using various and extensive medical data, this system accelerates and enhances possibilities for better choices in the treatment by doctors and patients.

The system uses learning algorithms like Decision Trees, Random Forest Classifier, Logistic Regression, KNN to give proper disease prediction and recommendations. These algorithms consider detailed information on patient condition and risk of disease to prescribe the actions to be taken. The solution also helps to predict diseases and individual care with the indication of specific actions to maintain health or prevent illnesses. This approach solves essential issues in the modern world of healthcare, including the improvement of diagnostics, personalized treatment, and time optimization of treatment planning.

In other words, an effective and efficient user interface would mean that a user shall not come across a wall of complexities that they cannot overcome while interacting with the system or utilize the health-care related information they need from the system as a health-care professional or patient. The prescribing process comes from standardizing decision-making, using consumers' individual preferences to improve complex medical situations more effectively.

The project report logically splits over chapters of work that forms its different sections. In the Introduction, the goals and rationale for the development of the system are outlined, focusing on the imperativeness of reliable disease forecast and individual health plan. The Literature Review compared existing healthcare solutions where the System is positioned based on the shortcomings of these solutions: the use of restricted datasets and the lack of practical steps.

Problem Statement

Modern health care systems are experiencing numerous difficulties in diagnosing patients, in making correct recommendations for further treatment, and supplying the appropriate necessary treatment as soon as possible. Presently, there is a lack of trade-offs in these system architectures for handling multiple data streams and biomarkers at once, for diagnosis, and more importantly, for providing personalized recommendations based on the needs of the patient. The current procedures of diagnosing and treating diseases is mainly through some rituals of examination that can cause time wastage and inefficient results.

The lack of easy-to-use, available, and interpretable systems adds to the problem, with clinicians and patients struggling to make use of existing technologies. Moreover, the increasing variety of diseases and the need for personalized treatment strategies necessitate tools that can analyze patient-specific data and provide accurate, actionable insights for effective diagnosis and recommendations.

Besides, few current solutions meet the fundamental criteria, namely, they do not address such crucial concerns as fairness and inclusion and do not pay enough attention to ethical aspects like data privacy and the minimization of the bias effect. When such systems are not properly protected, they may make wrong predictions or recommendations that undermines the patients' trust and the overall quality of the treatment which is to be provided.

To address these challenges, this project has proposed the design of the Personalized Treatment Recommendation System with machine learning. It is designed to suggest correct diseases to be diagnosed according to symptoms shared by the user, as well as recommend medications, diets, precautions, and preventive measures a user might need. The advanced algorithms incorporated into this and every other feature, along with usability, interpretability, and ethical design make this system's goal not only to bring together these technological advancements and the necessities of healthcare but also enhance clients' health results and satisfaction.

Objectives

- Develop a machine learning-based system capable of diagnosing diseases using the information provided by users.
- Provide detailed recommendations that include suitable medications, descriptions of diseases, dietary suggestions, precautions, and workout plans tailored to the user's health conditions.
- Create an easy-to-use interface for the application that allows both healthcare professionals and patients to input data, view predictions, and understand recommendations without needing technical expertise.
- Integrate data privacy and security features compliant with the Health Insurance Portability and Accountability Act (HIPAA) to safeguard users' data against leakage and misuse, ensuring trust, legal compliance, and ethical operation.
- Ensure that the outcomes of the predicted results are easily understood, and illustrate how the factors affecting the systems decision were obtained to encourage user confidence and trust.
- Develop and deploy the system using the Streamlit framework to provide an accessible, user-friendly interface for real-time disease prediction and personalized treatment recommendations.
- The system will help healthcare professionals identify various risk factors or coexisting conditions, enabling them to choose the right treatment early and improve patient outcomes.
- Socio: Reduce tendency towards inclusiveness for gender, race and medical history issues to provide equal healthcare services for all clients based on model recommendations.
- Supervise the result of system performance, as well as consider the feedback from users of the system to modify and enhance the algorithms and assist the specialty in increasing its precision and relevance to the current condition of healthcare and standards of the world.
- Offer a complete view of a patient's health to make personalized care and treatment interventions more effective and efficient.

Significance and Motivation

Significance:

Efficient Disease Prediction: It means the system that helps to choose potential diseases with the given set of symptoms and prompt precise recommendations. This minimizes guess work and assist in speedy diagnose of possible health complications.

Actionable Recommendations: In addition to predicting diseases it gives straightforward and understandable advice, including medications, exercising, and preventive measures; in other words, the system aids the user in optimizing his or her health efficiently.

User-Friendly Interface: This approach aims to make the software easily understandable for the broad audience so that everyone could input symptoms and get predictions and suggestions without a user being a doctor or technician.

Automation of Basic Diagnostics: In this way, the system is an useful tool for getting basic diagnostic and having an access to the guide or recommendations in the shortest time with less efforts as for health concerns.

Motivation:

Simplifying Health Assessment: This is because; there are people who find it difficult distinguishing what might be ailing them from how they feel. This project could help make health assessment easy by presenting the results in an easy understandable format that gives immediate predictions.

Convenience for Users: That is the motivation why people need the system that allows them understanding their health status without interfering a healthcare professional at a first stage, even if they live in a rural area or in an appropriate developing country.

Advancements in AI: The evolution of machine learning in recent times fosters the creation of really usable applications of this technology in fulfilling ordinary daily health requirements. For a long term, machine learning capabilities fuel the creation of tools that make use of machine learning to offer easy solutions for ordinary health care needs for the population.

Addressing Common Queries: Many people want a quick idea of their health before visiting a doctor. This system helps by providing reliable predictions and easy-to-understand suggestions based on the information they provide.

Organization of Project Report

This report on the Personalized Treatment Recommendation System has several chapters, each of which discusses a particular aspect of the effort. Below is an outline of the organization of the report:

Chapter 1: Introduction

This chapter sets the foundation for the project by explaining its purpose and why it is important. It also outlines the problem being addressed and the reasoning behind creating a system that can identify possible diseases based on symptoms. Additionally, the system provides advice on medicines, diets, descriptions, precautions, and workouts.

Chapter 2: Literature Survey

This chapter reviews existing studies and solutions in the field of disease prediction and treatment recommendation systems. It highlights the strengths of current systems while identifying their gaps and limitations. These shortcomings have inspired the creation of a more efficient and user-friendly system, like the one developed in this project.

Chapter 3: System Development

This chapter gives an overview of how the system was designed and developed. The following aspects are explained: the project architecture, steps of data preparation, used algorithms (Decision Tree, Logistic Regression, Random Forest and K-nearest neighbor), and such features as identification of symptoms, offering recommendations. Critical issues encountered during the development are also examined.

Chapter 4: Testing

This chapter looks at the testing methods used to check the system's accuracy and reliability. It includes details about unit testing, end-to-end testing, and performance evaluation. It also describes the test scenarios used to ensure the system works correctly, is dependable, and meets user expectations.

Chapter 5: Results and Comparison

This chapter also discusses the system's performance, showing that it accurately predicts diseases and gathers user feedback. It includes a comparison with previous systems and analyzes the differences between those systems and the one proposed in this study.

Chapter 6: Conclusion and Future Scope

This chapter outlines the achievements of the project, showing how diseases can be predicted and recommendations can be made based on the provided data. It also discusses the challenges faced and suggests future improvements, such as covering more diseases, expanding the system, and ensuring better security.

Chapter 2

Overview of Relevant Literature

The adoption of AI and ML in the healthcare sector is a revolutionary concept to changing how diagnostic, treatment, and care processes for diseases are done. With the development of AI technologies that happened in the past decade, it's possible to create more unique tools for diagnoses, increasing the precision that can be achieved, make treatment plans more personalized, and even organize the work of hospitals in the more effective way. These technologies tap massive amounts of medical data, using Decision Tree, Logistic Regression, Random Forest and K-nearest neighbour to forecast diseases and make accurate treatment suggestions.

Interestingly, the key aspect of the contemporary healthcare systems is shifting from the mainstream treatment methods adapted for all to the individual approach when treatments rely on genetic and environmental conditions and a person's lifestyle. Machine learning based systems especially are well positioned to assist this shift because they are able to work with high dimensional data, which is a large pool of data and make sense of it by looking for patterns. For instance, modern diagnostic tools used in artificial intelligence are able to integrate both genetic and microbiome tests, while providing a noninvasive disease prediction. Likewise, decision support systems in the healthcare industry employ ML models to suggest treatments and prevention mechanisms needed in a patient.

Still, the literature shows various shortcomings that can be noticed when AI is being applied to healthcare. One concern that comes up often is the question of where to obtain high-quality labeled datasets which are needed for training ML models. Furthermore, there are questions of how algorithmic bias influences results, lack of apparent decision making by the AI, and data privacy and security issues. Another reason is the challenges arising from the practical implementation of AI solutions in the structures and processes of the working healthcare industry.

This paper aims to review the current literature that aims at understanding how AI is being applied in healthcare with special focus on disease prediction and treatment recommendation systems. It captures significant findings in this line of work, reviews the methods applied in the process, and records shortcomings and missing elements in the preceding literature. Through such approach, the Personalized Treatment Recommendation System assumes the task of improving the gap between proven high-level machine learning and real-life healthcare requirements by establishing an efficient means of accurately predicting diseases and delivering targeted treatment recommendations.

S.N.	Title	Journal & Conference	Year	Tools and Techniques	Results	Limitations
1.	Artificial Intelligence in Healthcare: Past, Present and Future	Springer Handbook of Healthcare	2018	AI, Machine Learning, Deep learning, medical data analysis	The paper provides an overview of AI's evolution in healthcare, from basic diagnostic tools to modern AI-powered decision support systems. It highlights the benefits of AI in improving diagnostic accuracy, treatment planning, and personalized care.	Ethical concerns, the need for large, high-quality datasets, and the challenge of integrating AI solutions into existing healthcare infrastructure.
2.	Artificial Intelligence in Healthcare: Review	Journal of Health care Engineering	2017	Machine Learning, deep learning, AI algorithms, medical imaging,	This paper reviews the role of AI in medical diagnostics, highlighting	AI models require large amounts of labelled data and may be limited by

				electronic health records (EHR)	applications in image recognition, predictive analytics, and personalized medicine. It emphasizes AI's potential to improve healthcare efficiency and patient outcomes.	biases present in training datasets, which could lead to disparities in healthcare outcomes.
3.	Predicting Diseases using Machine Learning: Review	Journal of Medical Systems	2019	Machine Learning, predictive modelling, classification algorithms (Logistics Regression, K-nearest neighbour, Random Forests, Decision Tress)	The study demonstrates the effectiveness of machine learning in predicting diseases from medical data and shows how different models can be applied to various types of clinical data for improved accuracy in	Limited by the availability of comprehensive and standardized datasets; challenges with model interpretability in clinical settings.

					disease diagnosis.	
4.	AI-based Diagnostic Tools for Precision Medicine	Advanced Science News	2020	AI, deep learning, genetic profiling, microbiome data, non-invasive diagnostic tools	The paper highlights how AI models integrated with patient-specific data, such as genetics and microbiomes, can facilitate non-invasive and accurate disease diagnoses. It suggests that such diagnostic tools are vital for precision medicine.	High computational requirements, scalability issues, and challenges with generalizing the findings across different populations.
5.	Ethical and Social Implications of Artificial Intelligence in Health Care	Springer	2019	Ethical analysis, machine learning in healthcare, social impact studies	This paper discusses the ethical concerns surrounding AI in healthcare, such as transparency, data privacy,	Ethical guidelines for AI in healthcare are still evolving; the paper calls for clearer policies and regulations for data

					accountability, and algorithmic bias. It highlights the importance of addressing these issues to ensure AI technologies benefit all populations equally	privacy and bias mitigation.
6.	Machine Learning in Healthcare: A Review	Journal of Healthcare Engineering	2018	Machine learning, data mining, decision trees, neural networks, regression models	This paper reviews the various machine learning techniques used in healthcare applications, such as predictive models for disease diagnosis, treatment recommendations, and outcome prediction. It concludes that	Requires high-quality, labelled datasets, and there are issues related to overfitting, model interpretability, and the integration of ML tools with existing healthcare systems.

					machine learning offers significant improvements in healthcare efficiency and decision-making	
7.	AI-Driven Personalized Medicine: An Overview	PubMed Central	2020	AI, genetic data, predictive modelling, precision medicine	his paper discusses how AI can be integrated into personalized medicine by using patient-specific genetic data to create more accurate disease predictions and tailored treatment plans. It also highlights AI's potential in improving patient outcomes.	The high complexity of patient data and difficulty in obtaining large datasets for specific diseases, especially rare conditions.
8.	Personalized Medicine in the Era of Artificial	Springer	2021	AI, genetic profiling, predictive	This article provides a detailed look into the	Data privacy concerns, the need for more research on

	Intelligence: Challenges and Opportunities			modelling, data analytics	applications of AI in personalized medicine, with an emphasis on using AI to predict and prevent diseases. It also highlights the challenges involved in collecting, integrating, and analyzing patient-specific data	the long-term effects of AI in personalized treatments, and the challenges of integrating AI tools into existing healthcare infrastructure.
9.	AI in Healthcare Decision Support Systems: A Focus on User-Centric Designs	Springer	2021	User-centric design, AI, healthcare decision support systems, machine learning models	The paper discusses how user-centric designs in AI-driven healthcare systems can improve adoption and usability among healthcare professionals. It emphasizes the	Despite the user-centric design, challenges remain with system integration into existing practices, and a lack of standardization in clinical workflows hampers

					importance of integrating AI models into clinical workflows to assist in decision-making processes.	seamless AI integration.

Key Gaps in the Literature Survey

Even though there are massive progressive CI and I2 researches studying the Application of AI and ML in healthcare, some gaps and limitations have been identified in the current literature upon which this work is based. These are good research areas of study especially in the field of customized treatment recommendation systems. Below are the key gaps identified:

Limited Access to High-Quality, Diverse Datasets:

Numerous papers emphasize the use of large, labelled datasets for training machine learning models, while, in fact, obtaining diverse, complete, and high-quality data remains a critical issue. Data sets which are used in prior studies are mostly either specific to a particular business domain or limited to a particular geographic region; therefore, it becomes challenging to extend the identified models across the global or diverse population. This limitation results in oversimplified prediction models which are not accurate when tested on real life, diverse patients.

Algorithmic Bias and Lack of Fairness

This is a common problem in most AI healthcare systems to avail them algorithmic biases. Some of the ML models are trained from datasets that are likely to have limited representation of some populations like the minorities, and the patients with rare diseases. This leads to disparities in the ability to predict diseases and to recommend the right treatment. There are currently no strong published theories which would help to avoid these biases and provide equality between different genders and ethnicities.

Limited Interpretability of Machine Learning Models

Although there are higher-order models including but not limited to deep learning, boosting, and bagging technique, their interpretation is quite challenging. The problem with AI is that it does not explain the results it gets to enables healthcare providers to trust and use the technologies in practice. This is required in order to facilitate their incorporation into action-based healthcare systems.

Integration Challenges with Existing Healthcare Systems

Most work centres around the creation of individual intelligent models and does not factor in the challenge of implementing these models into the existing healthcare processes, be it in an EHR or CDS. Informativeness and scalability are often achieved when integration is smooth and there is little that past research has done to follow this direction.

Lack of Personalization in Existing Systems

Although, there exist technologies such as predictive diagnostic systems, most systems in use today do not warrant patient-centric solutions. Modern models merely suggest general treatment plans that do not take into consideration factors like age, gender, or other chronic diseases, as well as life style. It thus emerges that there is a hiccup in typical AI-based systems; the absence of the personalization feature.

Scalability and Computational Complexity

The use of computationally intensive methods including deep learning in AI models limits their application on big data or real time data. Much emphasis has been given to the effectiveness of these models in idealistic settings based on literatures but limited analysis has been conducted in real-world contexts especially those where resources constraints are evident.

Chapter 3

Requirements and Analysis

Language Used: Python 3.11.2

Technical Requirements:

- A computer with at least 4 GB of RAM and a multi-core processor
- Internet connection

Software:

- Python 3.5 or higher
- Visual Studio Code or any other code editor

Libraries:

- Numpy (numerical computing)
- Math (mathematical operations and functions)
- Pickle
- Streamlit
- Sklearn
- Matplotlib
- Imblearn
- Pandas
- Streamlit_option_menu

Additional Requirement

- Pip

Functional Requirement

User-Friendly Interface

- The system must provide a simple and intuitive user interface for users to input their symptoms easily.
- The interface should be accessible across devices such as desktops, tablets, and smartphones.

Symptom Input

- Users must be able to enter symptoms through predefined options or free-text input.
- The system should validate the entered symptoms to ensure accurate disease prediction.

Disease Prediction

- The system should analyze the provided symptoms to predict potential diseases using machine learning models.

Prescription and Recommendations

- The system must provide a comprehensive prescription for the predicted disease(s), including:
- Medication: Suggested drugs, dosages, and any necessary precautions.
- Diet Plans: Personalized dietary recommendations tailored to the predicted disease.
- Workouts: Suggested exercises or physical activities suitable for the user's condition.
- Precautions: Specific measures to prevent complications or manage the condition effectively.

Personalized Output

- Recommendations should be tailored to individual user profiles based on their input and predicted condition.

Real-Time Processing

- The system should deliver predictions and recommendations promptly after the user inputs symptoms.

Data Privacy and Security

- User data must be handled securely and adhere to privacy laws to ensure confidentiality.

Project Design and Architecture

Performance

- The system involves making predictions and recommendations in less than 3 seconds after the submission of symptoms.

Scalability

- The system also needs to be able to integrate user growth and future additions to the user interface.
- The important thing is it should be able to handle more data as more new symptoms, diseases, or recommendations are included.

Usability

- It is important that the user interface should be easy to use, user friendly and should not take long tutorials.
- At the end of the three clicks or less all the features of the site should be within reach.

Reliability

- The system should be available round the clock and thus should have an availability of more than 99.9%.
- It needs to work with emergencies and other issues that could occur, including wrong input data or the system failure, and report this issue to the user and offer a set of possible solutions.

Maintainability

- The system, including its future changes, should be flexible, so that it can be extended with new algorithms, diseases or recommendation parameters easily.
- It must be very easy for developers to introduce change maturity onto the framework without much disturbance on other functionalities.

Security

- Personal and medical user data personnel are required to be encrypted both whiles being transmitted and when stored.
- The system should hence respect the aspect of data protection including GDPR or HIPAA depending on the requirement.

Compatibility

- The system must be integrated to work well on common operating systems (Windows, macOS, Linux) and browsers (Chrome, Firefox, Edge).

Accuracy

- Recommendations and forecast must possess an accuracy rate of not lower than 95 percent based on assessments from test data.

Accessibility

- Accessibility means the system it has to be accessible to Disabled persons that is why it has to conform to certain standards, for example WCAG.
- Other characteristics that should be added are text to speech and keyboard access.

Ethical Compliance

- Ethical requirements of healthcare similar to medical evidence must be imposed on the system, meaning that the output of the system has to be fair, transparent, and explainable. It should not recommend decisions that are influenced by biased data.

Data Preparation

The Symptom-Disease Prediction Dataset (SPDC) was sourced from Mendeley Data, a repository known for high-quality research datasets. link: SymbiPredict - Mendeley Data

Data Collection:

- Exploration of various data sources such as UCI Machine Learning Repository, Kaggle, government websites, and public health surveys.

- Consideration of tailored datasets from machine learning contests or case studies related to disease prediction systems.

- To find datasets, search in official web-sites of government health agencies, research institutions or medical repositories.

- Prioritize datasets with symptom-disease mappings, treatment plans, and preventive measures for prediction and recommendation.

Data Cleaning:

- Handle the missing values using statistical imputation techniques such as mean for numerical values and mode for categorical variables.

- Remove features that are not correlated to disease prediction to make the dataset more streamlined.

- Ensure that the columns have appropriate data types be it numerical or categorical for efficient data processing.

Feature Selection:

- Select Important and relevant features such as symptom severity, disease names, etc while irrelevant features were dropped.

- Numerically encode the symptoms to represent their presence and absence for any disease.

Data Splitting

- Split the dataset into a training and testing set with 80-20 distribution using sklearn library.

- Perform the splitting using stratified sampling to ensure that all classes were represented proportionally across the different sets.

Methodology

Personalized Treatment Recommendation System was developed using machine learning techniques to predict diseases based on symptoms and providing recommendation for medications, exercises, diet and preventive measures. The core methodology revolves around integrating a machine learning model with user-friendly interface.

The data for this system was collected from public health-care datasets, which provide information relative to different diseases, symptoms and treatments characteristics. These datasets were selected due to their variety and because of their applicability to the prediction of diseases and healthcare needs. For instance, we applied a database of features including fever, cough, and fatigue, associated with diseases including flu and pneumonia. The information also contains treatment advice, prevention strategies, and guidelines for living with these diseases. The dataset contains tons of records, with key features like:

Symptoms: Categorical and continuous data in form of symptoms type.

Treatment and Preventive Measures: The types of information include medication, diet and exercise.

In this project data was cleaned and pre-processed so that all the entries that entered had to be accurate and appropriate. Data that was incomplete was naturally dealt with in imputation which made it possible for the model to be appropriately functional despite lack of the data needed.

After the data was gathered, the collected data was pre-processed for data cleaning before feeding the model. On the cases when certain records had missing values, imputation strategies were employed such that numerical data was filled up with mean or median of the given field and other categorical data was also completed based on mode. The irrelevant or extra features like the degree of the symptoms were normalized and only important features were retained. The dataset was then split into two parts 80:20 ratio as training and testing so that the model can be trained onload of data and tested on a different data set.

After that, the machine learning models were chosen to predict diseases since the user entered the specific symptom(s). The primary models, which the system employs, include Random Forest Classifier and Logistic Regression. Random Forest Models are specifically useful to perform classification and is most appropriate in the case of disease prediction since the main objective here is to diagnose diseases coming from four different classes. Decision Trees were chosen for their ability to make the process transparent and to explain how decisions are being made to healthcare providers. The training of both modes was performed with the help of the training set and determination of hyperparameters such as grids work. All the models were tested using different ERP parameters that include accuracy, precision, recall and F1-score for the purpose of testing their reliability in the classification of diseases.

The core of the recommendation engine is to present advice produced using the disease predicted by the machine learning model. When a disease is forecasted, the system goes on and gets recommendations of medications to be administered, diet recommendations, medications precautions and workout for the forecasted disease from a knowledge base. For instance, if the system says that you have flu, you will be advised on flu treatment, on how to rest and on the kind of diet you should take to support your flu recovery. The recommendation engine ensures that the advice provided is angled at addressing specific health problems, and the advice depends on the symptom’s medical conditions.

The following step was system integration in which the implemented machine learning model and recommendation system were combined into a running web application using python streamlit framework. The platform provides easy ways for users to enter in description of their problems, and get responses and suggestions in real time. Developed with the use of streamlit, the backend takes user input, provides it to the machine learning model, and, finally, delivers disease predictions and personal treatment options. The system also interfaces the user in such a way that the interaction between the healthcare provider and the patient should be easily facilitated.

Upon development of the system, further testing and evaluation were accomplished. Testing of the system was done in terms of testing the individual entity or sub-systems of the system for example, the model, data pre-processing step and the recommendation engine. To confirm that all modules come together and didn’t fail during integration, integration testing was done. Last, user testing was conducted to evaluate the usability of the developed system and to determine if there are further enhancements to be made with the user interface. The feedback obtained from the real users allowed improving the system to the extent that numerous professionals and patients find beneficial.

Concerning the implementation several challenges were noted as outlined below. The main challenge was handling of missing or incomplete data in the symptom data which was addressed through the use of imputation. As with any machine learning model, it was also important for model performance for boosting prediction accuracy especially when dealing with syndrome diseases. Also, making the system navigation intuitive and intuitive for non-developer users to interact with that called for the interface redesigning in the final section.

Analysis of System

Legal and Social Issues

The development of the Personalized Treatment Recommendation System implementation must consider several legal and social challenges that are related to the effectiveness, compliance, and acceptance of an implemented model. From a legal perspective, the importance of the protection of the personal data should be noted, particularly the patient's data, as the system presupposes the Data Protection Act 2018 and the General Data Protection Regulation (GDPR) is aimed at the protection of all kinds of the sensitive health data. This evidence suggests that strong features like data encryption, anonymization, and controlling the access of patients' data are critical to protecting the data and adhere to the requirements. Besides, AI computing healthcare systems could also fall under the regulation of medical devices in the UK where they need to prove that the devices they provide are safe, accurate and efficient. Failure to adhere to these regulations could attract legal issues and slow down deployment. Liability is also a concern for VHA because wrong advice may be detrimental to patient welfare and there are questions about who is to blame. These concerns should be dealt with.

From a social environment point of view, access and fairness are key concerns. Currently, the United Kingdom is experiencing the digital evolution where some areas especially rural areas and other underserved areas greatly lack in the provision of new innovative health care technologies. It is crucial that the system is accessible to a range of different socioeconomic and geographical backgrounds, as to make known health conditions worse by introducing new barriers. The last yet important one is public trust because controversies surrounding the AI reliability and misuse of data may blunt patients and even the healthcare professionals trust. Combatting these legal and social issues, will simply not only make the system legal and correct but also proceed in a manner that promotes the penetration of the system in the UK health sector with people's trust.

Professional and Ethical Issues

The introduction of Personalized Treatment Recommendation System in healthcare situation raises some relevant professional and ethics that needs consideration. For job purpose, the effectiveness and credibility of this system is very important. Doctors, nurses and other care givers depend on accurate information before they can make the next move on patients, hence if the system is wrong, patient safety could be at risk. An essential factor is the need to have the system checked continually and updated professionally for professionalism stability and effectiveness owing to the constant change in medical practice. Furthermore, the level of transparency is also critical, clinicians should know why a particular AI recommendation should be implemented in the decision-making process to apply it fully. Education on how the system reaches those decisions also preserves the doctors' confidence and support so that the innovation can be implemented.

On ethical issues, we get to realize that the use of sensitive patents data involve privacy and confidentiality. A failure to follow such regulations limits the safety of the patients' data, even though much more needs to be done from an ethical perspective – principles like data minimization and data storage security, to mention but a few. The final ethical consideration relates to consent; patients must comprehend how their data is going to be used and must be allowed to revoke this consent at any one time. The other important ethical issue in relation to AI models is that of bias. There is the need to use ethical standards to guarantee fairness and equality when implementing the system.

Implementation

Creating a system for customized therapy has several distinctive difficulties. One of the challenges is to obtain large datasets that are of good quality, diverse, and representative. Nowadays, medical data can be noisy, and contain missing values, inconsistencies or learned bias, which can have an impact on prediction accuracy and bias. Sustaining the accuracy of the machine learning models irrespective of patients' characteristics and the naturalistic clinical settings is another challenge since the models may be very sensitive to the input data distribution. The Treatment Recommendation System requires using a machine learning algorithm together with a streamlit web application for disease forecasting and individualized recommendations. The system uses a trained and tuned Random Forest model serialized and deployed using pickle. The application initiates with the importation of modules, streamlit for developing the interface and Pandas for handling the datasets. On startup, the system comes with several datasets including symptoms, precautions, medications, diets, workouts, disease descriptions etc. that serve as contents from which the comprehensive suggestions are derived (Ranschaert, Morozov and Algra, 2019b).

Helper function get_disease_data in streamlit application returns data related to the predicted disease from the corresponding datasets. This function allows the system to be able to retrieve disease descriptions, precaution, medicines, diets and exercises that correspond to the disease that the model has detected. A predetermined lexicon converts all responses given by the user into a numeric vector after which random forest is used to predict the result. The predicted disease is the gateway to suggestions and makes the system very sensitive and extensible.

Several html pages have been deployed in the implementation of the application. The form enables users to enter symptoms that the system processes as inputs. In this model, when the user submits their symptoms, the application decodes the input, make the prediction and developed the result. The streamlit based GUI allows users to easily either select or type their symptoms and the app uses the random forest model fine-tuned and pickled in order to make prediction of the disease and the matching prescription containing the medication, diets, workouts, etc. back to the user.

This implementation also guarantees proper integration of machine learning with suitable user interface. As a result, through dynamic data retrieval, the important and scalable architecture, and real-time interaction with the system, it provides the healthcare industry users with up-to-date healthcare information.

Figure 1: Importing Libraries

```python
# Importing libraries
import pandas as pd
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,  accuracy_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import mutual_info_classif
from sklearn.decomposition import PCA
from sklearn.metrics import accuracy_score, classification_report
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RandomizedSearchCV
import pickle
```
Python

First part of the model building involves importing the necessary libraries used for the process. This includes libraries such as:

- Pandas: To handle and manipulate the data.
- Sklearn: Scikit Learn provides us with various machine learning tools such as:
    1. Classification Report: To generate comprehensive reports for models evaluation including f1 scores.
    2. Decision Tree Classifier, Random Forest Classifier, KNeighborsClassifier, Logistic Regression: Different models used for prediction.
    3. Train Test Split: To split the dataset into training and testing parts to make sure that we can evaluate the model effectively.
    4. Mutual Info Classif: To find the most important features for disease prediction.
    5. RandomizedSearchCV: To find the best hyperparameters for machine learning model, in this case, Random Forest Classifier model.

- Pickle: To save the trained model in a pickle file so that it can be used through the GUI without having to train the model again.
- Matplotlib: To create visualizations used in the EDA.
- Imblearn: It is imported to use various tools such as:
    1. Smote: To handle imbalanced dataset and improves the model's ability to predict minority class.

Figure 2: Loading Datasets

Loading and cleaning the dataset

```python
# Loading dataset
data = pd.read_csv('training.csv')

print("First 5 Rows of the Dataset:")
print(data.head())

print("\nDataset Info:")
print(data.info())

print(f"\nDataset Shape: {data.shape}")
```

Loading the dataset and viewing details such as the head, the first 5 rows of the dataset. Displaying the data.info for it and displaying the shape of the dataset, meaning the rows and columns of the data.

Figure 3: Checking for Missing Values

```python
# Checking for missing values
print("Missing Values Before Cleaning:")
print(data.isnull().sum())
```

Checking for missing values in the dataset as they impact the model's performance.

Figure 4: Removing Null Values

```python
# Handling missing values

#filling mean value for numerical values
for col in data.select_dtypes(include=['number']).columns:
    if data[col].isnull().sum() > 0:
        print(f"Filling missing values in numerical column: {col}")
        data[col].fillna(data[col].mean(), inplace=True)

#filling mode value for categorical values
for col in data.select_dtypes(include=['object']).columns:
    if data[col].isnull().sum() > 0:
        print(f"Filling missing values in categorical column: {col}")
        data[col].fillna(data[col].mode()[0], inplace=True)

# cheking if missing values are dealt with
print("\nMissing Values After Cleaning:")
print(data.isnull().sum())
```

Removing the null values in the dataset, For this:

If the data is numerical, then calculating the mean value for the column and filling the mean in the place of the missing numerical value.

If the data is categorical and not numerical, then finding the mode of the column and filling that in the place.

Figure 5: Summary Statistics of Cleaned Dataset

```python
# Displaying summary of the cleaned dataset
print("Summary of Cleaned Data:")
print(data.describe())
print("\nFirst 5 Rows of the Cleaned Data:")
print(data.head())
```

```
Summary of Cleaned Data:
            itching     skin_rash  nodal_skin_eruptions  continuous_sneezing  \
count  20480.000000  20480.000000          20480.000000         20480.000000
mean       0.157454      0.180496              0.046066             0.056029
std        0.388465      0.403625              0.246457             0.264255
min       -1.124399     -1.144893             -1.092302            -1.512659
25%        0.000000      0.000000              0.000000             0.000000
50%        0.000000      0.000000              0.000000             0.000000
75%        0.077260      0.162045              0.000000             0.000000
max        1.858422      1.958978              1.716941             1.915482

          shivering        chills    joint_pain  stomach_pain       acidity  \
count  20480.000000  20480.000000  20480.000000  20480.000000  20480.000000
mean       0.033510      0.167133      0.146635      0.055867      0.056390
std        0.221736      0.396188      0.378021      0.263256      0.267639
min       -1.012638     -1.023749     -1.004327     -1.048944     -1.007642
25%        0.000000      0.000000      0.000000      0.000000      0.000000
50%        0.000000      0.000000      0.000000      0.000000      0.000000
75%        0.000000      0.099153      0.036029      0.000000      0.000000
max        1.608409      2.056088      1.817375      1.743437      1.855333

       ulcers_on_tongue  ...  pus_filled_pimples    blackheads      scurring  \
count      20480.000000  ...        20480.000000  20480.000000  20480.000000
mean           0.034757  ...            0.033746      0.034811      0.032742
```

Now, displaying the summary of cleaned dataset, after the data has no null values to see if there has been improvement in the dataset for the process to continue.

Figure 6: Dataset Overview

```
# Dataset Overview
print("Dataset Shape:", data.shape)
print("Columns in the Dataset:", data.columns.tolist())

print("First 5 Rows:")
print(data.head())

print("Summary Statistics of Numerical Features:")
print(data.describe())

print("Missing Values in Each Column:")
print(data.isnull().sum())
```

```
Dataset Shape: (20480, 133)
Columns in the Dataset: ['itching', 'skin_rash', 'nodal_skin_eruptions', 'continuous
First 5 Rows:
    itching  skin_rash  nodal_skin_eruptions  continuous_sneezing  shivering  \
0  0.000000   1.000000              0.000000              0.00000   0.000000
1  0.146774   0.205156             -0.021239              0.43378   0.074531
2  0.000000   0.000000              0.000000              0.00000   0.000000
3  0.000000   0.000000              0.000000              0.00000   0.000000
4  0.000000   0.000000              0.000000              0.00000   0.000000

     chills  joint_pain  stomach_pain   acidity  ulcers_on_tongue  ...  \
0  1.000000    1.000000      0.000000  0.000000          0.000000  ...
1 -0.013983   -0.861033     -0.032538 -0.016484          0.988003  ...
2  0.000000    0.000000      0.000000  0.000000          0.000000  ...
3  0.000000    0.000000      0.000000  0.000000          0.000000  ...
4  0.000000    0.000000      0.000000  0.000000          0.000000  ...

   blackheads  scurring  skin_peeling  silver_like_dusting  \
0    0.000000   0.00000      0.000000             0.000000
```

Describing the summary statistics of the cleaned dataset and for both the numerical and categorical columns.

Figure 7: Features and Target Variables

```
# Separating features and target
features = data.columns[:-1]
target = data.columns[-1]

X = data[features]
y = data[target]

print("Features Selected:")
print(features)

print("\nTarget Selected:")
print(target)

class_distribution = data['prognosis'].value_counts()
```

```
Features Selected:
Index(['itching', 'skin_rash', 'nodal_skin_eruptions', 'continuous_sneezing',
       'shivering', 'chills', 'joint_pain', 'stomach_pain', 'acidity',
       'ulcers_on_tongue',
       ...
       'pus_filled_pimples', 'blackheads', 'scurring', 'skin_peeling',
       'silver_like_dusting', 'small_dents_in_nails', 'inflammatory_nails',
       'blister', 'red_sore_around_nose', 'yellow_crust_ooze'],
      dtype='object', length=132)

Target Selected:
prognosis
```

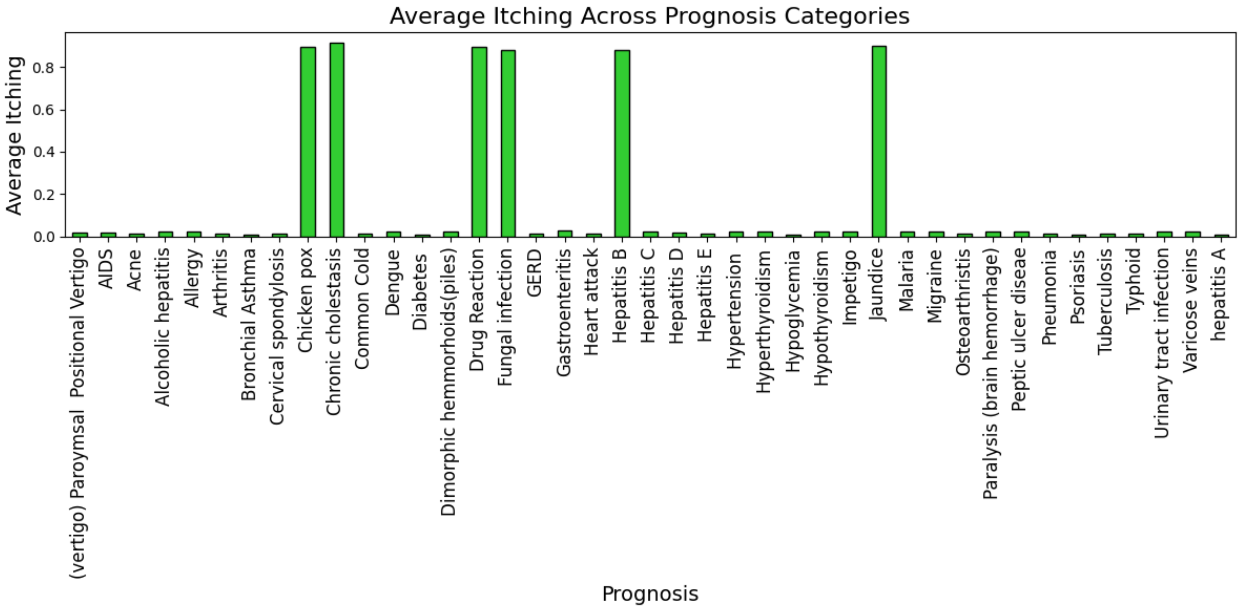
Separating the features and the target, in this model's case, the features include the symptoms and the target variable is the prognosis, meaning the diagnosis or the disease.

Figure 8: Symptom and Prognosis Code

```python
# Visualizing a relationship between a symptom and prognosis
symptom = 'itching'

plt.figure(figsize=(12, 6))
data.groupby('prognosis')[symptom].mean().plot(kind='bar', color='limegreen', edgecolor='black')
plt.title(f"Average {symptom.capitalize()} Across Prognosis Categories", fontsize=16)
plt.xlabel("Prognosis", fontsize=14)
plt.ylabel(f"Average {symptom.capitalize()}", fontsize=14)
plt.xticks(rotation=90, fontsize=12)
plt.tight_layout()
plt.show()
```

Exploratory Data Analysis (EDA)

Figure 9: Symptom and Prognosis Chart

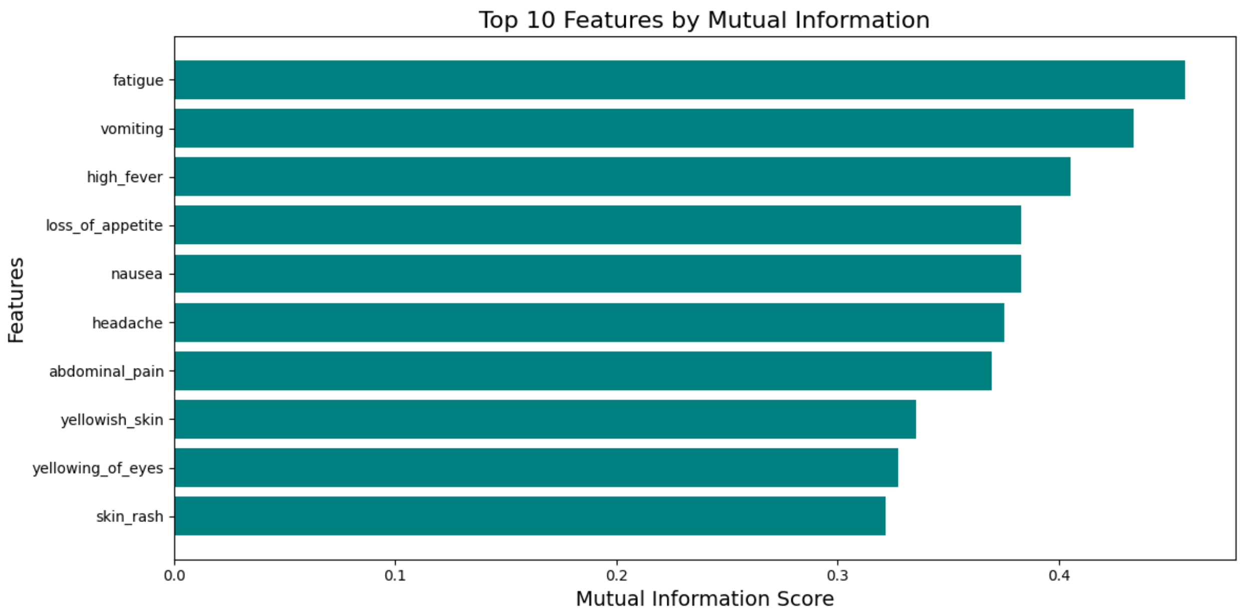


In the above diagram, we can see the relationship between a symptom (itching) and the target variable, prognosis which is the diagnosis. It indicates the mean of the scores of the level of itching for each disease. We can see that itching is very highly correlated with chicken pox and not related with diseases like hepatitis A and pneumonia. With this visualization, one can find out which diseases are most closely linked to itching.

Figure 10: Top 10 Features by Mutual Information Code

```python
# Computing mutual information
X = data.iloc[:, :-1]  # Excluding target column
y = data['prognosis'].astype('category').cat.codes
mutual_info = mutual_info_classif(X, y, random_state=42)

feature_importance = pd.Series(mutual_info, index=X.columns).sort_values(ascending=False)[:10]

# Plotting top 10 features
plt.figure(figsize=(12, 6))
plt.barh(feature_importance.index, feature_importance.values, color='teal')
plt.title("Top 10 Features by Mutual Information", fontsize=16)
plt.xlabel("Mutual Information Score", fontsize=14)
plt.ylabel("Features", fontsize=14)
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```

Figure 11: Top 10 Features by Mutual Information Bar chart

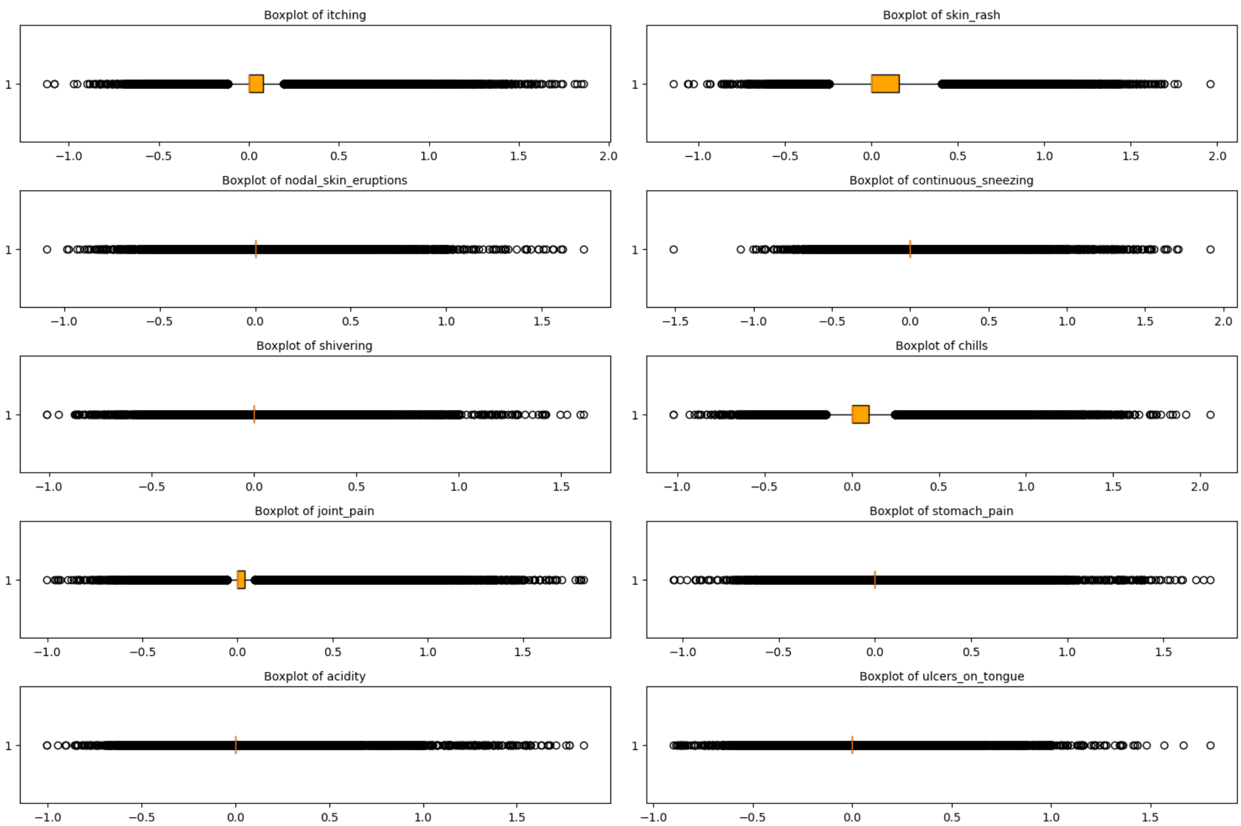


The above EDA bar chart shows the top 10 features by mutual information. Fatigue and vomiting have high scores, meaning that these symptoms are more closely associated with certain prognosis, the target variable, meaning diagnosis than the lower ranked items such as skin rash and yellowing of eyes.

Figure 12: Boxplot Code

```python
# Selecting a subset of numeric columns for demonstration
sample_columns = data.columns[:10]

plt.figure(figsize=(15, 10))
for i, col in enumerate(sample_columns):
    plt.subplot(5, 2, i + 1)
    plt.boxplot(data[col], vert=False, patch_artist=True, boxprops=dict(facecolor='orange'))
    plt.title(f'Boxplot of {col}', fontsize=10)
    plt.tight_layout()
plt.show()
```

Figure 13: Boxplot

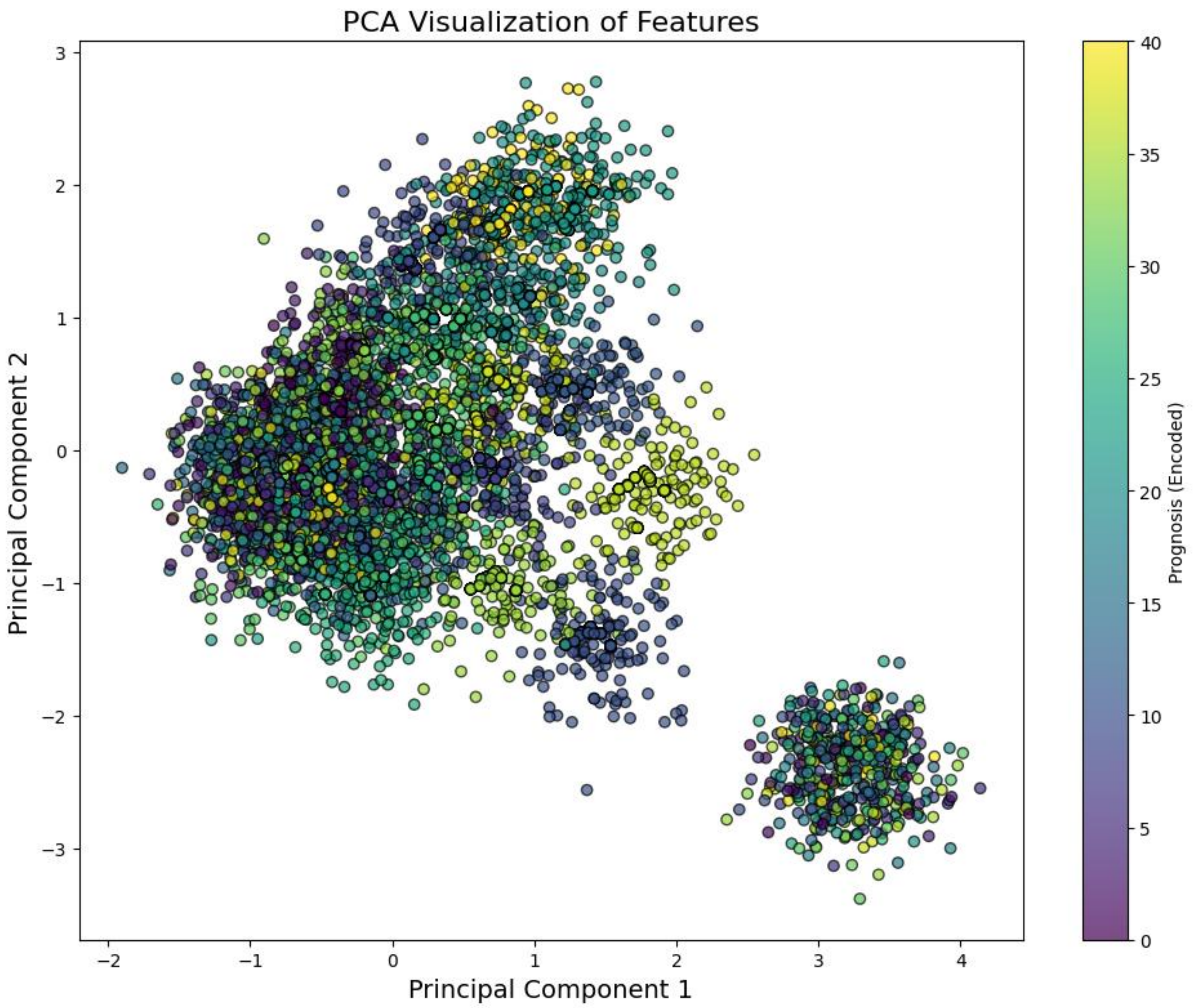


This boxplot displays key symptoms (itching, rashses, sneezing). For each symptom, a boxplot provides an overall picture of data distribution. The majority of values are closer to zero, meaning that the data is standardized, and variability is low. Orange line inside of the big rectangle characterizes the median while the rectangle itself displays the values of the 25-75 percentiles, upper and lower quartiles. The dots placed beyond the bars of the whiskers are outliers since they show rarely occurring case or extreme values.

Figure 14: PCA Code

```
# Performing PCA on the dataset
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

plt.figure(figsize=(10, 8))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y, cmap='viridis', edgecolor='k', alpha=0.7)
plt.title("PCA Visualization of Features", fontsize=16)
plt.xlabel("Principal Component 1", fontsize=14)
plt.ylabel("Principal Component 2", fontsize=14)
plt.colorbar(label="Prognosis (Encoded)")
plt.tight_layout()
plt.show()
```

Figure 15: PCA visualization



This plot is also called PCA (Principal Component Analysis) dispersion where high number of features are represented by only two features, x and y. Each dot is a data point, or a patient or a case and colour maps to disease classes encoded in the color bar. The clustering patterns indicate similarities between disease symptoms. The decision to separate clusters stems from the fact that different diseases are grouped separately and us to understand similarities as well as disparities of the diseases in the data set.

Figure 16: Standard Scaler

```
#applying standardscaler to the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Using StandardScaler to scales data by bringing the mean of the feature to 0 and the standard deviation of the feature to 1. This is important because it aligns all features to the same scale pre-processing data to prevent scenarios where some features are larger as a result of machining leaning than others.

Figure 17:SMOTE

```
#appling smote to balance the data
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_scaled, y)
class_distribution = data['prognosis'].value_counts()
```

Since the data is heavily imbalanced, using SMOTE. SMOTE synthesizes new examples for the minority class through the calculation of distance measure between the existing samples and achieving a better-balanced data set. This assist machine learning models to attain better accuracy through excluding bias towards majority class and raising the likelihood of the underrepresented one.

Figure 18:Train-Test Split

```
#performing a 80/20 train-test split
X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2, random_state=42)
print("Shapes of Train and Test sets:")
print(f"X_train: {X_train.shape}, X_test: {X_test.shape}")
print(f"y_train: {y_train.shape}, y_test: {y_test.shape}")
```

```
Shapes of Train and Test sets:
X_train: (25977, 132), X_test: (6495, 132)
y_train: (25977,), y_test: (6495,)
```

Performing an 80/20 training testing split. 80% of the data will be used for the training of the models and the other 20% will be used for testing purposes. After that, displaying the shapes of the training and testing datasets.

The accuracies of the 4 different models was compared to see the most accurate model. Decision tree model was the model with the least accuracy 0.89. The best model was found to be random forest model with 0.9424. This model was further fine-tuned with RandomizedSearchCV. This model was then tested again to find 0.9426 as the accuracy score. This was the finalized model which was then pickled to use from the GUI.

Figure 19: Logistic Regression Model

```
# logistic regression model
logistic_model = LogisticRegression(random_state=42, max_iter=1000)
logistic_model.fit(X_train, y_train)

# Making predictions
y_pred = logistic_model.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}\n")

print("Classification Report for logistic regression model:")
print(classification_report(y_test, y_pred))
```

Figure 20: Logistic Regression Report

```
Accuracy: 0.9176

Classification Report for logistic regression model:
              precision    recall  f1-score   support

           0       0.87      0.92      0.89       146
           1       0.93      0.89      0.91       159
           2       0.93      0.88      0.90       171
           3       0.92      0.92      0.92       156
           4       0.88      0.87      0.87       165
           5       0.94      0.91      0.92       165
           6       0.90      0.87      0.88       149
           7       0.88      0.91      0.90       151
           8       0.90      0.94      0.92       167
           9       0.89      0.94      0.92       163
          10       0.93      0.86      0.89       171
          11       0.91      0.93      0.92       137
          12       0.94      0.91      0.93       162
          13       0.94      0.91      0.93       164
          14       0.95      0.92      0.93       171
          15       0.91      0.90      0.91       177
          16       0.95      0.94      0.95       159
          17       0.91      0.92      0.92       147
          18       0.88      0.91      0.90       156
          19       0.89      0.93      0.91       150
...
    accuracy                           0.92      6495
   macro avg       0.92      0.92      0.92      6495
weighted avg       0.92      0.92      0.92      6495
```

Figure 21: Random Forest Model

```python
# Random Forest model
rf_model = RandomForestClassifier(random_state=42, n_estimators=100)
rf_model.fit(X_train, y_train)

# Making predictions
y_pred = rf_model.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}\n")

print("Classification Report for random forest:")
print(classification_report(y_test, y_pred))
```

Figure 22: Random Forest Report

```
Accuracy: 0.9424

Classification Report for random forest:
              precision    recall  f1-score   support

           0       0.88      0.98      0.93       146
           1       0.95      0.92      0.93       159
           2       0.96      0.92      0.94       171
           3       0.95      0.95      0.95       156
           4       0.95      0.93      0.94       165
           5       0.97      0.93      0.95       165
           6       0.94      0.88      0.91       149
           7       0.90      0.93      0.92       151
           8       0.95      0.94      0.95       167
           9       0.95      0.96      0.95       163
          10       0.96      0.91      0.94       171
          11       0.94      0.96      0.95       137
          12       0.95      0.93      0.94       162
          13       0.96      0.93      0.94       164
          14       0.95      0.95      0.95       171
          15       0.93      0.93      0.93       177
          16       0.97      0.96      0.97       159
          17       0.95      0.92      0.93       147
          18       0.95      0.96      0.95       156
          19       0.93      0.93      0.93       150
...
    accuracy                           0.94      6495
   macro avg       0.94      0.94      0.94      6495
weighted avg       0.94      0.94      0.94      6495
```

Figure 23: KNN

```
# KNN model
knn_model = KNeighborsClassifier(n_neighbors=5)
knn_model.fit(X_train, y_train)

# Making predictions
y_pred = knn_model.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}\n")

print("Classification Reportfor KNN model:")
print(classification_report(y_test, y_pred))
```

Figure 24: KNN Report

```
Accuracy: 0.9361

Classification Reportfor KNN model:
              precision    recall  f1-score   support

           0       0.86      0.97      0.91       146
           1       0.94      0.92      0.93       159
           2       0.92      0.93      0.93       171
           3       0.87      0.95      0.91       156
           4       0.93      0.93      0.93       165
           5       0.91      0.93      0.92       165
           6       0.94      0.89      0.91       149
           7       0.89      0.93      0.91       151
           8       0.92      0.94      0.93       167
           9       0.97      0.94      0.96       163
          10       0.96      0.91      0.94       171
          11       0.90      0.95      0.93       137
          12       0.94      0.92      0.93       162
          13       0.97      0.92      0.95       164
          14       0.95      0.91      0.93       171
          15       0.94      0.93      0.94       177
          16       0.97      0.96      0.96       159
          17       0.96      0.91      0.93       147
          18       0.97      0.90      0.93       156
          19       0.91      0.92      0.91       150
...
    accuracy                           0.94      6495
   macro avg       0.94      0.94      0.94      6495
weighted avg       0.94      0.94      0.94      6495
```

Figure 25: Decision Tree Classifier

```
# Decision Tree model
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)

# Making predictions
y_pred = dt_model.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}\n")

print("Classification Report for Decision Tree model:")
print(classification_report(y_test, y_pred))
```

Figure 26: Decision Tree Report

```
Accuracy: 0.8993

Classification Report for Decision Tree model:
              precision    recall  f1-score   support

           0       0.88      0.93      0.91       146
           1       0.90      0.84      0.87       159
           2       0.90      0.89      0.90       171
           3       0.88      0.92      0.90       156
           4       0.93      0.84      0.88       165
           5       0.93      0.88      0.91       165
           6       0.88      0.86      0.87       149
           7       0.91      0.86      0.88       151
           8       0.91      0.92      0.91       167
           9       0.85      0.90      0.87       163
          10       0.91      0.89      0.90       171
          11       0.93      0.91      0.92       137
          12       0.88      0.87      0.88       162
          13       0.94      0.91      0.93       164
          14       0.89      0.89      0.89       171
          15       0.90      0.90      0.90       177
          16       0.92      0.92      0.92       159
          17       0.90      0.90      0.90       147
          18       0.90      0.91      0.91       156
          19       0.88      0.92      0.90       150
...
    accuracy                           0.90      6495
   macro avg       0.90      0.90      0.90      6495
weighted avg       0.90      0.90      0.90      6495
```

Figure 27: Hyperparameter Tuning

Refining Random Forest Model by Hyperparameter Tuning Using RandomizedSearchCV

```python
param_dist = {
    'n_estimators': [50, 100, 150, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['sqrt', 'log2', None]
}

# RandomizedSearchCV
rf_random = RandomizedSearchCV(estimator=RandomForestClassifier(random_state=42),
                               param_distributions=param_dist,
                               n_iter=50,
                               cv=3,
                               verbose=2,
                               n_jobs=-1,
                               random_state=42)

# Fitting the model
rf_random.fit(X_train, y_train)

# Getting the best parameters and model
print(f"Best Parameters: {rf_random.best_params_}")
print(f"Best Score: {rf_random.best_score_}")
```

```
Fitting 3 folds for each of 50 candidates, totalling 150 fits
Best Parameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2', 'max_depth': None}
Best Score: 0.9398313893059246
```

The best performing model, i.e. random forests classifier model was then gone through hyperparameter tuning using randomized search CV.

n_estimators: The number of trees in the forest.

max_depth: The maximum of depth each tree can reach.

min_samples_split: The smallest number of instances allowed in a split a node.

min_samples_leaf: The minimum number of samples required in a leaf node.

max_features: The number of features to consider when splitting a node.

By running multiple iterations, the RandomizedSearchCV identifies the best possible parameters to identify the best combination of parameters for highest accuracy. This improves the overall accuracy and precision of the model.

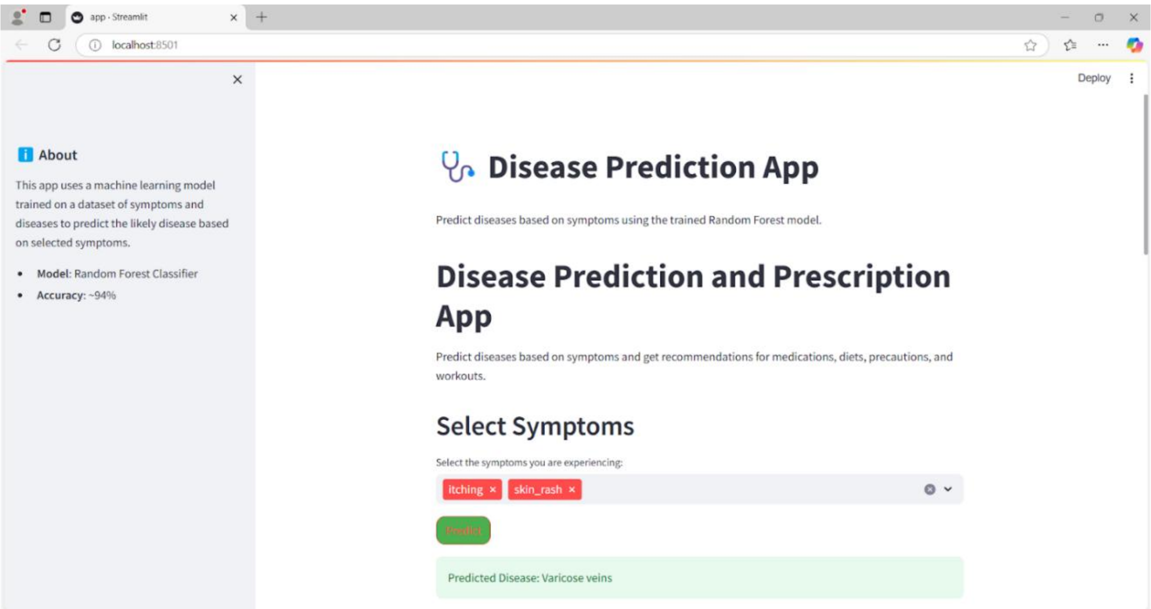
Figure 28: Pickling Model

```python
# Pickling the tuned model
model_file = "tuned_random_forest_model.pkl"

with open(model_file, 'wb') as file:
    pickle.dump(best_rf_model, file)

print('Model saved as '+model_file)
```

Pickling the fine-tuned random forest model in pickle format.

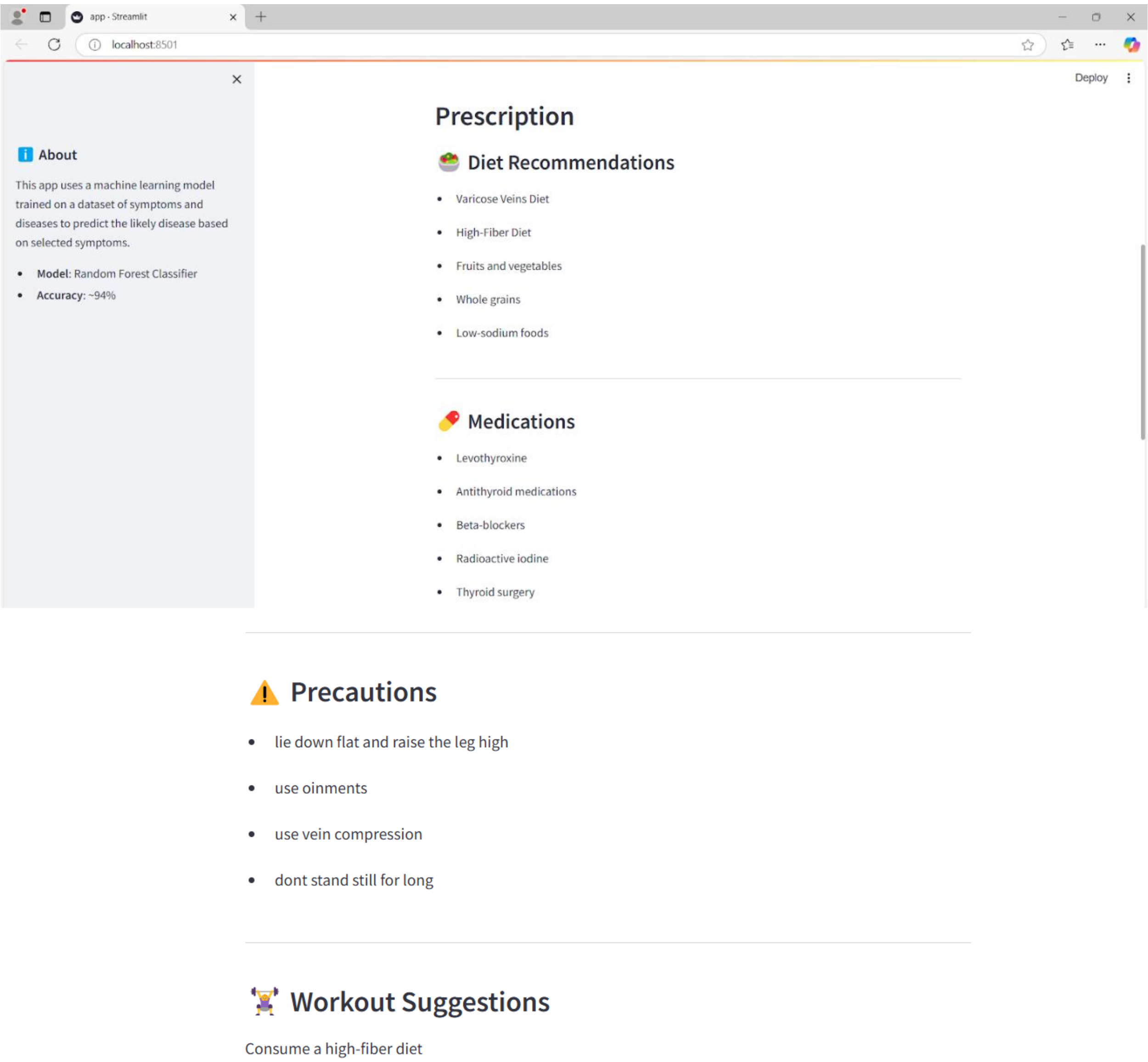
Graphical User Interface (GUI)

Figure 29: GUI Prediction



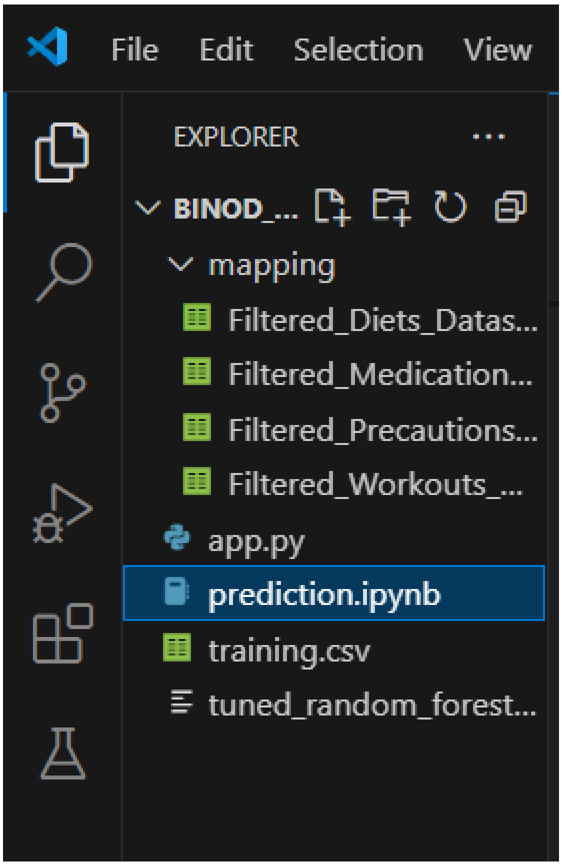
The UI is developed using python streamlit. The interface consists of an input field with a dropdown, in which users can either select symptoms or they can type their own symptoms. Based on those symptoms, the GUI uses the pickled model file which results in the disease being predicted, the corresponding medications, workout, prevention and diets are then fetched based on the predicted disease from the datasets. Then the GUI uses the fetched data to display the prescription to the user.

Figure 30: GUI Prescription



Prescription is shown in a layout inspired by the real-life doctor's prescription with predicted disease, it's description, precautions, medication, workout and diet recommendations.

Figure 31: Folder Structure



This shows the folder structure of the project with the notebook, streamlit application along with datasets used for prediction and the machine learning model saved after training.

Chapter 4

Testing Strategy

Unit testing

In unit testing, the emphasis was made on the verification of the functionality of the subcomponents of the Personalized Treatment Recommendation System. Therefore, each developed disease prediction, medication recommendation, and lifestyle guidance modules were evaluated individually to establish strike rates and product suitability. For example, the effectiveness of the disease prediction module was tested with regard to the symptoms and possible diseases with high accuracy. Likewise, on the medication recommendation module, the function was tested against different sets of inputs to verify that it gives recommendations for the right drugs in the dosage as pertinent to human body. There was an effort to verify that the efficacy of the lifestyle guidance module to provide the correct diet plan together with other procedures depending on some conditions which are displayed in the 2nd part of the Table 6. These tests meant that each module ran effectively independently to ensure its proper functioning when connected to the others.

End to End

Emulation confirmed the system’s performance since end-to-end testing was conducted in a more realistic setting. The tests performed investigated the effectiveness of the application from pre-processed inputs to the delivery of recommendations. Users offered their symptoms or conditions and the overall performance of the system was examined with regard to how it takes inputs, detects diseases, and gives individualized advice. Several conditions, normal or abnormal situations which could be as well called as exceptional cases were modelled to test its performance and stability. This broad testing phase reassured that each component. An integrated system was further tested to evaluate its effectiveness and user satisfaction in their entirety.

Test Cases and Outcomes

Test Case 1: Input Validation Test

Objective: Provide invalid inputs, such as non-numeric values or out-of-range values, for each disease prediction form.

Expected Outcome: Provide invalid inputs, such as non-numeric values or out-of-range values, for each disease prediction form.

Test Case 2: Model Prediction Test

Objective: Provide valid medical data for disease prediction.

Expected Outcome: The system should predict diseases accurately based on the input data.

Test Case 3: Edge Case Test

Objective: Test the system with extreme inputs, such as minimum and maximum allowable values.

Expected Outcome: The system should handle these inputs gracefully without errors, providing accurate predictions.

Test Case 4: Performance Evaluation Test

Objective: Evaluate accuracy using pre-labeled datasets with known outcomes.

Expected Outcome: Predictions should match the ground truth labels with high accuracy.

Test Case 5: Model Interpretability Test

Objective: Verify the system's ability to explain prediction outcomes.

Expected Outcome: The system should provide clear insights into the factors influencing its predictions, enhancing transparency.

Chapter 5

Results

Comparison with Existing Solutions

Specifically, the following should be regarded as the key strengths of our proposed Personalized Treatment Recommendation System addressing the needs of the healthcare industry. Unlike conventional systems where early disease prediction is targeted at a specific disease, our system allows for early prediction of diseases, for example diseases given certain symptoms input by the user. This kind of model has advantages over existing models in that it informs users of their possibilities for getting diseases, whereas before the models may only tell them about disease. Most existing implementations tend to allow for feature-rich APIs that are heavy to use and somewhat require a good understanding of software engineering. Observe, however, that our system does not sacrifice the friendly user interface; the goal is to make it as easy to use for patients and clinics. The use of maps and search bars means that it is not necessary to have a great technical understanding of the computer in order to effectively input symptoms and receive thorough results and advisement.

Moreover, as noted in many of the current approaches, most existing systems offer prediction capabilities while being devoid of prescriptive guidance. In doing so, our solution fills this gap by providing prescriptions such as medications, a diet plan, an exercise regimen and prevention measures in relation to the identified condition. This integrated framework guarantees people get real advice in addition to forecasts.

Chapter 6

Conclusion

Overview

The development of the Personalized Treatment Recommendation System proves that machine learning can do so much betterment for healthcare. In areas where medications, diet, preventive measures and workouts are suggested based on risks predicted by the system, there are major gaps that the healthcare system has not been able to fill. It makes the autonomy and density of the patient and the service providers in an unmistakable manner, which facilitates better and timely decision making better.

The system is also found to be efficient accurate and does not pose any complexities in operation. Testing showed that the performance of the proposed machine learning model is very high whereas the usage of the knowledge base makes the recommendations applicable and reasonable. The user interface has been developed in such a way that it can be used by people with little or no knowledge of computers.

During the evaluation, the system appears to have achieved the aimed objectives, but it can be enhanced further. The conclusion can be made that the system is beneficial for more individuals if there were more multiple conditions to include in the training datasets and an even bigger variety of conditions in the knowledge base. With that being said, the interface and further real-world testing will also be updated periodically to make sure it represents a perfect world in the real emerging world of health care.

In conclusion, while this project focuses on the opportunity for AI in targeted healthcare delivery, it opens the potential floor for developing more advanced options and ideas as well. Since filling the existing gaps of the current healthcare system and ensuring it meets high ethical and professional standards is within its ability, it has the chance to contribute positively to the modern world. The work reminds those stakeholders who uses such appliances that integrating the advanced technology into healthcare to provide personalized treatment to everyone.

Summary of the Personalized Treatment Recommendation System

The Personalized Treatment Recommendation System has achieved all the primary objectives demonstrating that machine learning in healthcare is possible. Firstly, it forecasts diseases given user-inputted symptoms and secondly, it prescribes correct medications along with correct diets, measures to be taken and workouts to be done. During the testing phase the application of the machine learning model proved well accurate while the recommendation engine worked as expected, offering realistic solutions pertinent to their clients.

The system's interface that was quite friendly, and doing away with complex templates where the users could easily input their symptoms and get results in return. Thanks to that people with all levels of IT literacy could employ the system in an efficient way. The tests were carried out intensively in order to ensure the high probability of the system function and optimal protection of the patient's valuable information. Adherence to GDPR standards provided additional ethic to the project and become a guarantee of users' trust and protection. Moreover, the system capability was tested in real life health care situations and proved to be an effective tool for practical use.

There are few weaknesses in the project, the evaluation identified a number of potential areas for development. While the training dataset was a success, it was not the most expansive than could be used. Increasing these datasets would also increase accuracy through the reduction of bias and increase confidence that the system could recommend equal quality of healthcare across different populations. Also, enlarging the system's knowledge database, which is currently vast but limited to specific diseases, treatments, or suggestions, may be useful. These improvements would augment the system's benefits and openness for use for all.

Findings and Recommendations

The Personalized Treatment Recommendation System has demonstrated that the use of machine learning is can be extremely impactful on healthcare systems. It was also able to give correct predictions of diseases from the symptoms, specific recommendations of diets, medications and precautions measures to undertake. This project has demonstrated that with the help of smart algorithms healthcare knowledge base the most relevant and truly beneficial solutions can be provided to people.

However, it also identified potential opportunities of the system to develop and be enhanced in the future. The possibilities are more diverse, then current training data sets need more diversification. Although this current dataset was satisfactory, it is could be further enriched with data from the other populations and conditions. This would help the system in providing recommendations tailored, without discrimination, for all people at a common optimum quality.

Another area for improvement is the knowledge base which provides the base for the system's recommendations. The inclusion of information concerning particular diseases as well as additional treatment methods would make the system even more useful and capable of addressing greater number of conditions in order to offer the clients fully comprehensive advice.

In sum, all these points suggest the direction on how to continue its development in order to turn the system into more open, powerful, and user-friendly tool in the sector of healthcare.

Areas for Future Work

Although Personalized Treatment Recommendation System is effective and has a promising use in medical field have some room for improvement and development in the future. There are a number of areas that need fixes, the training data of the machine learning model as a crucial part. Currently, the dataset is useful, although, we can improve its variation and add information from different populations and areas. This would help the system to be bias free and therefore come up with better recommendations to people of different nature.

The knowledge base of the system could also be expanded the integrated system could also be improved. Information concerning the presence of the exotic diseases, other forms of treatment, and different lifestyle conditions would increase its applicability more comprehensively. This would ensue that the system gives a more comprehensive and personalized guideline to both patients and clinician.

One area for future work is the development of improved user interface which will be implemented in the next release. Though the current interface is user friendly, there is always space for further improvement of the over interface design. It could include options such as voice recognition in order to input symptoms, or being able to use the system with different languages, so that the system is more accessible from users with different requirements.

Other associated considerations for future work include scalability. Additional users will mean a increase in both the amount of data that it would have to process and the number of requests in the system at any one time. Maintaining the overall efficiency of the system through enhancing the scalability factor will guarantee that this system remains fast and efficient as the user base is expanding.

Last but not the least, adding live data directly from the health monitoring devices such as wearable or health tracking gadgets could take the system a notch higher and even more tailored. This would enable user to get suggestions in line with their current health status and therefore the system would be of more usefulness in continual health care provision. These improvements will assist to develop the system and become a much more effective system in the future.

Personal Evaluation

The Personalized Treatment Recommendation System project has been a valuable opportunity to learn and experience developing machine learning solutions to solve real world problems. From this project, the most important change is that I realized the potential of machine learning in healthcare and its applications for learning the patterns. It would be inspiring witnessing how technology can assist people in their real lives and thus the goal was motivating at all times. Perhaps the greatest thing I found out was to design things simple and easy to use as much possible. It was not easy to develop the machine learning model but I came to understand that developing an interface as well was equally essential. A unique requirement of the project was that the system not just function correctly; it needed to be understandable by many stakeholders who could not be expected to have a particularly strong computer literacy. Juggling these two aspects helped me realized just how important it is to consider the end-users who will be utilizing this technology.

One additional thing that I learned was just how valuable or valuable a diverse and inclusive dataset can be. Examining the results, I realized that the system’s performance might depend on the categories when the amount of data is restricted. This really shed light to me on the issue of fairness and inclusiveness in AI and it is something I will have to consider any time I am developing new models.

It also made me comprehend the importance of readiness to learn and flexibility in that process. Sometimes the performer does not achieve his goals, and the child has to invent plus develop something or introduce changes based on the opinions. That success is not an end’s product but a continuous process of testing, learning, and experimenting.

Overall, this experience has helped me grow as a better and innovative problem solver and I remain passionate in using technology in solving people everyday issues. I take pride in what has been done and I am aware that the knowledge gotten will help in future.

Future Scope

The System is a great opportunity for growth in the future and is put a high value on the improvement the doctors. The authors also emphasized that the scope of expansion is possible in terms of the range of diseases that can be diagnosed using a systemic approach. At this moment the given system can predict only a rather small number of conditions, while future refinement could include all diseases, from rare to chronic, meaning the system would be rather more valuable as a diagnostics tool. This would greatly extend its usability in a broad range of health care circumstance.

Another interesting avenue is the extension of the collaboration with wearable health devices. Depending on the possibility of using smartwatch or a fitness tracker, the system could use fresh data on heart rate, sleep, and activity to help its users. Such integration would help to make the predictions and the subsequent recommendations faster and more accurate with the help of permanently updated material.

The interface that would be established to accommodate multiple languages would enhance its usability to affected populace around the world. Supporting multiple languages would guarantee that users of different linguistics passes would find the interface easy to use. Moreover, the system could be further developed if superior AI and machine learning algorithms arisen in future. The deep learning models and the ideas connected with the concept of the continuous learning could enable the given system to operate on more extensive datasets and improve the results step by step.

Last, the use of cloud computing can only improve the scalability and hence the efficiency of the system. A cloud-based platform would enable this system to support more users and different data sets and at the same time optimized. With such improvements, the Personalized Treatment Recommendation System could be considered as a major auxiliary tool in modern health care systems worldwide, helping users and professionals make right choices, -propose effective treatments, and eventually contribute to the enhancement of the quality of life for millions of people.

References

1. Ranschaert, E. R., Morozov, S. and Algra, P. R. (2019) Artificial intelligence in medical imaging, *Springer eBooks*, [online] Available at: https://doi.org/10.1007/978-3-319-94878-2.
2. Lyell, D., Coiera, E., Chen, J., Shah, P. and Magrabi, F. (2021b) How machine learning is embedded to support clinician decision making: an analysis of FDA-approved medical devices, *BMJ Health & Care Informatics*, 28(1), p. e100301, [online] Available at: https://informatics.bmj.com/content/28/1/e100301?
3. Bhati, D., Neha, F. and Amiruzzaman, M. (2024) A survey on Explainable Artificial intelligence (XAI) techniques for visualizing deep learning models in medical imaging, *Journal of Imaging*, 10(10), p. 239, [online] Available at: https://www.mdpi.com/2313-433X/10/10/239.
4. Küstner, T., Qin, C., Sun, C., Ning, L. and Scannell, C. M. (2024) The intelligent imaging revolution: artificial intelligence in MRI and MRS acquisition and reconstruction, *Magnetic Resonance Materials in Physics Biology and Medicine*, 37(3), pp. 329–333, [online] Available at: https://doi.org/10.1007/s10334-024-01179-2.
5. Seeram, E. (2023) Artificial intelligence in Medical Imaging: An Overview, In *Springer eBooks*, pp. 119–131, [online] Available at: https://doi.org/10.1007/978-3-031-46266-5_9.
6. Singh, D., Nagaraj, S., Mashouri, P., Drysdale, E., Fischer, J., Goldenberg, A. and Brudno, M. (2022) Assessment of Machine Learning–Based Medical Directives to Expedite care in Pediatric Emergency Medicine, *JAMA Network Open*, 5(3), p. e222599, [online] Available at: https://doi.org/10.1001/jamanetworkopen.2022.2599.
7. Filiberto, A. C., Leeds, I. L. and Loftus, T. J. (2021) Editorial: Machine Learning in Clinical Decision-Making, *Frontiers in Digital Health*, 3, [online] Available at: https://doi.org/10.3389/fdgth.2021.784495.
8. Anon (n.d.) Rawan AlSaad, [online] Available at: https://scholar.google.com/citations?hl=en&user=uIWECA0AAAAJ.
9. Filiberto, A. C., Donoho, D. A., Leeds, I. L. and Loftus, T. J. (2023) Commentary: Machine learning in clinical decision-making, *Frontiers in Digital Health*, 5, [online] Available at: https://doi.org/10.3389/fdgth.2023.1214111.
10. H2O.ai (2018) Clinical Decision Making with Machine Learning, *YouTube*, Video, [online] Available at: https://www.youtube.com/watch?v=AfEtluZIDnY.

11. C3 Digital Transformation Institute (2021) Machine Learning for Clinical Decision-Making: Panacea or Pandora's Box, *YouTube*, Video, [online] Available at: https://www.youtube.com/watch?v=QX8ZX1so--s.