**A Project Report on**

**A HEALTHCARE SYSTEM USING MACHINE LEARNING SYSTEMS FOR DISEASE PREDICTION WITH CHATBOT ASSISTANCE**

*Submitted to the*

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD**

*In partial fulfilment of the requirement for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**BY**

|  |  |
| --- | --- |
| ANAGONDI SHIVA PRASAD | (19WJ1A0519) |
| BADEMPET PAVAN | (19WJ1A0531) |
| CHEERALA PRASHANTH | (19WJ1A0561) |

Under the Esteemed Guidance of

**Mr. K. CHANDRA SHEKAR**

**Associate Professor, Department of CSE**

A red and black logo

Description automatically generated with low confidence

**Department of Computer Science & Engineering**

**GURU NANAK INSTITUTIONS TECHNICAL CAMPUS (AUTONOMOUS)**

**School of Engineering & Technology**

**Ibrahimpatnam, R.R District- 501506**

**2023**

A group of logos on a white background

Description automatically generated with low confidence

Date: 22-06-2023

**Department of Computer Science & Engineering**

**CERTIFICATE**

This is to certify that the project report titled **“A HEALTHCARE SYSTEM USING MACHINE LEARNING SYSTEMS FOR DISEASE PREDICTION WITH CHATBOT ASSISTANCE”** by **ANAGONDI SHIVA PRASAD (19WJ1A0519), BADEMPET PAVAN (19WJ1A0531) & CHEERALA PRASHANTH (19WJ1A0561)** is submitted in partial fulfilment of the requirements for the degree of **Bachelor of Technology** in **Computer Science & Engineering** of the **Jawaharlal Nehru Technological University Hyderabad** during the academic year 2022 - 23, is a Bonafede record of work carried out under our guidance and supervision.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**INTERNAL GUIDE PROJECT CO-ORDINATOR HOD CSE**

**Mr. K. Chandra Shekar Mr. B. Samirana Acharya Dr. J. Rajeshwar**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**EXTERNAL EXAMINER**

A picture containing text, screenshot, letter, document

Description automatically generated

**ACKNOWLEDGEMENT**

We wish to express our sincere thanks to **Mr. Sardar Tavinder Singh Kohli, Chairman, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We wish to express our sincere thanks to **Mr. Sardar Gagandeep Singh Kohli, Vice Chairman, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We wish to express our sincere thanks to **Dr. Harvinder Singh Saini, Managing Director, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We wish to express our sincere thanks to **Dr. Koduganti Venkat Rao, Director, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project.

We wish to express our sincere thanks to **Dr. Rishi Sayal, Associate Director, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We would like to say our sincere thanks to **Dr. J. Rajeshwar, Professor & Head, Department of CSE, GNITC** for providing seamless support and right suggestions are given in the development of the project.

We would like to say our sincere thanks to **Mr. B. Samirana Acharya, Assistant Professor, Department of CSE, Project Coordinator,** for providing seamless support and right suggestions are given in the development of the project.

We have been truly blessed to have a wonderful internal guide **Mr. K. Chandra Shekar, Associate Professor, Department of CSE,** for guiding us to explore the ramification of our work and we express our sincere gratitude towards him for leading me through the completion of Project.

Finally, we would like to thank our family members for their moral support and encouragement to achieve goals.

**ANAGONDI SHIVA PRASAD (19WJ1A0519)**

**BADEMPET PAVAN (19WJ1A0531)**

**CHEERALA PRASHANTH (19WJ1A0561)**

**TABLE OF CONTENTS**

**CONTENTS PAGE**

**LIST OF FIGURES……………………………………………..…………………………...iv**

**LIST OF SNAPSHOTS……………………………………..………………………………..v**

**LIST OF SYMBOLS…………………………………………...………………….………...vi**

**LIST OF ABBREVIATIONS…………………………….…………………………………ix**

**ABSTRACT……………………………………………...…………………………...............x**

**CHAPTER 1: INTRODUCTION……………………….…………………………………..1**

1.1 AIM OF PROJECT………….…………….….…………………………………...2

1.2 OBJECTIVE OF PROJECT…………..…………………………………………...2

1.3 SCOPE OF THE PROJECT……………..………………………………………...2

1.4 EXISTING SYSTEM………………………..…………………………………….3

1.4.1 EXISTING SYSTEM DISADVANTAGES………..……………………3

1.5 LITERATURE SURVEY…………………………………………..……………...3 1.6 PROPOSED SYSTEM………….…………………………………………………4

1.6.1 PROPOSED SYSTEM ADVANTAGES…………………….………….5

**CHAPTER 2: PROJECT DESCRIPTION…………………………………………………7**

2.1 MODULES…….….……………………………………………………………….7

2.2 TECHNIQUE USED OR ALGORITHM USED…………..……………………...9

**CHAPTER 3: REQUIREMENTS ENGINEERING……………...……………………...11**

3.1 HARDWARE REQUIREMENTS………………………………………………..11

3.2 SOFTWARE REQUIREMENTS………………………………………………...11

3.3 FUNCTIONAL REQUIREMENTS……………………………………………...12

3.4 NON-FUNCTIONAL REQUIREMENTS……………………………………….12

**CHAPTER 4: SYSTEM DESIGN…………………………………………………………13**

4.1 SYSTEM ARCHITECTURE…………………………………………………….13

4.2 LEVEL 0 DATA FLOW DIAGRAM…………………………………….………14

4.3 LEVEL 1 DATA FLOW DIAGRAM………………………………………...…..15

4.4 E-R DIAGRAM…………………………………………………………….……16

4.5 USE CASE DIAGRAM.........................................................................................17

4.6 CLASS DIAGRAM...............................................................................................18

4.7 OBJECT DIAGRAM.............................................................................................19

4.8 SEQUENCE DIAGRAM……………………..………………………………….20

4.9 COLLABORATION DIAGRAM………………………………………………..21

4.10 STATE DIAGRAM……………………………………………………………..22

4.11 ACTIVITY DIAGRAM………………………………………………………...23

4.12 COMPONENT DIAGRAM…………………………………………………….24

4.13 DEPLOYMENT DIAGRAM…………………………………………………...25

**CHAPTER 5: DEVELOPMENT TOOLS……………….………….…………………….26**

5.1 PYTHON…………………………………………………………………………26

5.2 HISTORY OF PYTHON…………………………………………………………26

5.3 IMPORTANCE OF PYTHON………………....………....………………………27

5.4 FEATURES OF PYTHON……………….………………………………………27

5.5 LIBRARIES USED………………………………………………………………28

**CHAPTER 6: IMPLEMENTATION……………………………………………………...29**

6.1 SAMPLE SOURCE CODE……………………………………………………...29

**CHAPTER 7: SNAPSHOTS…………………………………………..…………………...45**

7.1 VARIOUS SNAPSHOTS…………………………….…………………………..45

**CHAPTER 8: SOFTWARE TESTING……………………………………….…………...48**

8.1 DEVELOPING METHODOLOGIES……………………...……………………48 8.2 TYPES OF TESTING……………………………………………………………48

8.3 BUILD THE TEST PLAN……………………………………………………….50

8.3.1 TEST CASES……………………………..……………………………50

**CHAPTER 9: CONCLUSION & FUTURE ENHANCEMENT……………...…………52**

9.1 CONCLUSION…...……………………………………………………………..52 9.2 FUTURE ENHANCEMENT……………………………………………………52

**CHAPTER 10: BIBLIOGRAPHY…………………………………………………….…..53**

10.1 REFERENCE PAPERS…...……………………………………………………53 10.2 REFERENCE BOOKS…………………………………………………………54

10.3 REFERENCE WEBSITES……………………………………………………..54

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **NAME OF THE FIGURE** | **PAGE NO.** |
| 4.1 | SYSTEM ARCHITECTURE | 13 |
| 4.2 | LEVEL 0 DATA FLOW DIAGRAM | 14 |
| 4.3 | LEVEL 1 DATA FLOW DIAGRAM | 15 |
| 4.4 | E-R DIAGRAM | 16 |
| 4.5 | USE CASE DIAGRAM | 17 |
| 4.6 | CLASS DIAGRAM | 18 |
| 4.7 | OBJECT DIAGRAM | 19 |
| 4.8 | SEQUENCE DIAGRAM | 20 |
| 4.9 | COLLABORATION DIAGRAM | 21 |
| 4.10 | STATE DIAGRAM | 22 |
| 4.11 | ACTIVITY DIAGRAM | 23 |
| 4.12 | COMPONENT DIAGRAM | 24 |
| 4.13 | DEPLOYMENT DIAGRAM | 25 |

**LIST OF SNAPSHOTS**

|  |  |  |
| --- | --- | --- |
| **SNAPSHOT NO.** | **NAME OF THE SNAPSHOT** | **PAGE NO.** |
| 7.1 | INTERFACE BEFORE RECOMMENDING | 45 |
| 7.2 | MOVIE SELECTION DROPDOWN | 46 |
| 7.3 | UPON RECOMMEND ACTION, RUNNING OF  MODEL | 46 |
| 7.4 | FETCHING OF BACKDROPS | 47 |
| 7.5 | GENERATION OF SIMILAR MOVIES | 47 |

**LIST OF NOTATIONS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **NAME** | **NOTATION** | **DESCRIPTION** |
| 1. | Class | *Class Name*  *-attribute-attribute*  *+operation*  *+operation*  *+operation*  *+ public*  *- private*  *# protected* | Represents a collection of similar entities grouped together. |
| 2. | Association | NAME  CLASS A  CLASS B  CLASS B  CLASS A | Associations represents static relationships between classes. Roles represents the way the two classes see each other. |
| 3. | Actor |  | It aggregates several classes into a single class |
| 4. | Aggregation | CLASS B  CLASS A  CLASS B  CLASS A | Interaction between the system and external environment |
| 5. | Relation  (uses) | Uses | Used for additional process communication. |
| 6. | Relation  (extends) | extends | Extends relationship is used when one use case is similar to another use case but does a bit more. |
| 7. | Communication |  | Communication between various use cases. |
| 8. | State | State | State of the process. |
| 9. | Initial State |  | Initial state of the object |
| 10. | Final state |  | Final state of the object |
| 11. | Control flow |  | Represents various control flow between the states. |
| 12. | Decision box |  | Represents decision making process from a constraint |
| 13. | Use Case |  | Interact ion between the system and external environment. |
| 14. | Component |  | Represents physical modules which is a collection of components. |
| 15. | Node |  | Represents physical modules which are a collection of components. |
| 16. | Data Process/State |  | A circle in DFD represents a state or process which has been triggered due to some event or action. |
| 17. | External entity |  | Represents external entities such as keyboard, sensors, etc. |
| 18. | Transition |  | Represents communication that occurs between processes. |
| 19. | Object Lifeline |  | Represents the vertical dimensions that the object communications. |
| 20. | Message | Message | Represents the message exchanged. |

**LIST OF ABBREVATIONS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **ABBREVATION** | **EXPANSION** |
| 1**.** | ML | Machine Learning |
| 2. | GUI | Graphical User Interface |
| 3. | NLP | Natural Language Processing |
| 4. | XGBoost | eXtreme Gradient Boosting |
| 5. | AI | Artificial Intelligence |
| 6. | DL | Deep Learning |

**A HEALTHCARE SYSTEM USING MACHINE LEARNING TECHNIQUES FOR DISEASE PREDICTION WITH CHATBOT ASSISTANCE**

**ABSTRACT:** This project aims to develop a healthcare system that can predict multiple diseases, including liver disease, diabetes and heart disease using machine learning techniques. These are the most common types of Diseases that plaque the Indians today. This system will use XGBoost (eXtreme Gradient Boosting) for these diseases while maintaining accuracy, speed, resource utilization, stability and reducing complexity. The system will also include a healthcare chatbot that can provide personalized health recommendations to users.

The system will be developed using Python and its many Machine Learning Libraries & Streamlit Framework. Machine learning models will be trained on publicly available datasets to predict diseases accurately. The developed models will be evaluated using performance metric like accuracy of training and testing data. The healthcare chatbot will be developed with NLP and/or OpenAI API Integration Techniques to provide personalized health recommendations to users. The system will be deployed on a cloud-based platform to ensure scalability and availability.

The Healthcare System has the potential to improve the accuracy of disease diagnosis, care, and early intervention, thereby reducing healthcare costs and improving patient outcomes. The healthcare chatbot can provide accessible and personalized health recommendations, leading to a healthier and happier life.

**CHAPTER 1**

**INTRODUCTION**

In today's era, healthcare systems face numerous challenges in accurately diagnosing and predicting diseases, leading to increased healthcare costs and compromised patient outcomes. To address these challenges, this project aims to develop a comprehensive healthcare system that utilizes machine learning techniques to predict multiple diseases, including liver disease, diabetes, and heart disease. These diseases are prevalent in the Indian population, making them a primary focus of this endeavour.

To ensure the system's effectiveness, the project will employ XGBoost (eXtreme Gradient Boosting), a powerful machine learning algorithm known for its accuracy, speed, and resource utilization. By leveraging the capabilities of XGBoost, the system will deliver robust predictions while reducing complexity and maintaining stability.

Additionally, the developed healthcare system will incorporate a healthcare chatbot that offers personalized health recommendations to users. This chatbot will utilize Natural Language Processing (NLP) techniques and/or potentially integrate with OpenAI APIs to provide users with tailored advice and support.

The implementation of the healthcare system will be carried out using Python programming language, which offers a rich ecosystem of machine learning libraries. Furthermore, the Streamlit framework will be utilized to create a user-friendly interface for seamless interaction with the system.

To ensure the accuracy of disease predictions, the machine learning models will be trained on publicly available datasets that encompass a wide range of patient information. Performance evaluation of the models will be conducted using metrics such as accuracy on both training and testing data, ensuring their reliability and effectiveness.

For scalability and availability, the healthcare system will be deployed on a cloud-based platform. This approach ensures that the system can accommodate increasing user demands while maintaining consistent accessibility.

**1.1 AIM OF PROJECT**

The ultimate goal of this healthcare system is to significantly improve disease diagnosis accuracy, promote early intervention, and enhance patient care. By doing so, it has the potential to reduce healthcare costs and, more importantly, improve patient outcomes. Moreover, the integration of a healthcare chatbot into the system will enable users to access personalized health recommendations conveniently, fostering a healthier and happier life.

**1.2 OBJECTIVE**

The objectives of this project are to develop a healthcare system that leverages machine learning techniques to accurately predict prevalent diseases like liver disease, diabetes, and heart disease, with a specific focus on the Indian population. The system aims to utilize the XGBoost algorithm, renowned for its accuracy, speed, and efficiency, to ensure precise disease predictions while maintaining stability and reducing complexity. Additionally, the project aims to incorporate a healthcare chatbot into the system, utilizing techniques such as Natural Language Processing (NLP) and OpenAI API integration to provide personalized health recommendations to users. The implementation will be carried out using the Python programming language and various machine learning libraries, with the Streamlit framework facilitating a user-friendly interface. The machine learning models will be trained on publicly available datasets, enabling robust disease predictions, and their performance will be evaluated using metrics such as accuracy on both training and testing data. Deployment on a cloud-based platform will ensure scalability and availability of the healthcare system. Ultimately, the project seeks to enhance disease diagnosis accuracy, enable early intervention, reduce healthcare costs, and improve patient outcomes. The integration of a healthcare chatbot aims to offer accessible and tailored health recommendations, empowering users to lead healthier and happier lives.

**1.3 SCOPE OF THE PROJECT**

The scope of this project includes the development of a healthcare system that utilizes machine learning techniques to predict common diseases, such as liver disease, diabetes, and heart disease, with a focus on the Indian population. The project also involves integrating a healthcare chatbot to provide personalized health recommendations. The system will be developed using Python, machine learning libraries, and the Streamlit framework. The machine learning models will be trained on publicly available datasets, and the system will be deployed on a cloud-based platform for scalability and availability.

**1.4 EXISTING SYSTEM**

In the case of the existing system, there are very few existing systems in the market that have the potential to evolve into a comprehensive All-in-One Healthcare System for Disease Prediction. However, even those existing systems that do exist face significant limitations. These systems often require a substantial amount of computing resources, yet they struggle to achieve high levels of accuracy in disease prediction.

In terms of disease prediction algorithms, the existing system relies on Decision Trees for liver and heart disease prediction, and K-Nearest Neighbours (KNN) for diabetes prediction. However, each of these algorithms has its own drawbacks and limitations, leading to suboptimal performance in disease prediction.

Moreover, the existing systems lack a well-designed user interface (UI) and struggle with issues related to scalability and ease of use. They also fall short in providing personalized recommendations to users, which limits their effectiveness in addressing individual healthcare needs.

Given these limitations, there is a clear opportunity to develop a more advanced and comprehensive healthcare system that surpasses the existing solutions in terms of accuracy, usability, scalability, and personalized recommendations.

### **1.4.1 EXISTING SYSTEM DISADVANTAGES**

* KNN can be computationally expensive and slow when dealing with large datasets, as it requires calculating the distance between the new input point and every point in the training dataset.
* Decision Trees are prone to overfitting, especially when dealing with noisy data or complex datasets with many features. This can lead to poor generalization and low accuracy on new data.
* Both KNN and Decision Trees can suffer from the curse of dimensionality, where the performance of the model decreases as the number of features increases. This can lead to increased computation time and poor accuracy.

**1.5 LITERATURE SURVEY**

**Title:** Implementation and Use of Disease Diagnosis Systems for Electronic Medical Records Based on Machine Learning: A Complete Review

**Author:** Jahanzaib Latif, Chuangbai Xiao, Shanshan Tu, Sadaqat Ur Rehman, Azhar Imran, Anas Bilal

**Year:** 2021

**Description:** Electronic health records are used to extract patient's information instantly and remotely, which can help to keep track of patients' due dates for checkups, immunizations, and to monitor health performance. The Health Insurance Portability and Accountability Act (HIPAA) in the USA protects the patient data confidentiality, but it can be used if data is re-identified using `HIPAA Safe Harbor' technique. Usually, this re-identification is performed manually, which is very laborious and time captivating exertion. Various techniques have been proposed for automatic extraction of useful information, and accurate diagnosis of diseases. Most of these methods are based on Machine Learning and Deep Learning Methods, while the auxiliary diagnosis is performed using Rule-based methods. This review focuses on recently published papers, which are categorized into Rule-Based Methods, Machine Learning (ML) Methods, and Deep Learning (DL) Methods.

**Title:** Deployment of Containerized Deep Learning Applications in the Cloud

**Author:** Rim Doukha, Sidi Ahmed Mahmoudi, Mostapha Zbakh, Pierre Manneback

**Year:** 2021

**Description:** During the last years, the use of Cloud computing environment has increased as a result of the various services offered by Cloud providers (Amazon Web Services, Google Cloud, Microsoft Azure, etc.). Many companies are moving their data and applications to the Cloud in order to tackle the complex configuration effort, for having more flexibility, maintenance, and resource availability. However, it is important to mention the challenges that developers may face when using a Cloud solution such as the variation of applications requirements (in terms of computation, memory and energy consumption) over time, which makes the deployment and migration a hard process. In fact, the deployment will not depend only on the application, but it will also rely on the related services and hardware for the proper functioning of the application. In this paper, we propose a Cloud infrastructure for automatic deployment of applications using the services of Kubernetes, Docker, Ansible and Slurm. Our architecture includes a script to deploy the application depending of its requirement needs. Experiments are conducted with the analysis and the deployment of Deep Learning (DL) applications and more particularly images classification and object localization.

**Title:** Disease Prediction using Machine Learning Algorithms

**Author:** Sneha Grampurohit; Chetan Sagarnal

**Year:** 2020

**Description:** The development and exploitation of several prominent Data mining techniques in numerous real-world application areas (e.g. Industry, Healthcare and Bio science) has led to the utilization of such techniques in machine learning environments, in order to extract useful pieces of information of the specified data in healthcare communities, biomedical fields etc. The accurate analysis of medical database benefits in early disease prediction, patient care and community services. The techniques of machine learning have been successfully employed in assorted applications including Disease prediction. The aim of developing classifier system using machine learning algorithms is to immensely help to solve the health-related issues by assisting the physicians to predict and diagnose diseases at an early stage. A Sample data of 4920 patients' records diagnosed with 41 diseases was selected for analysis. A dependent variable was composed of 41 diseases. 95 of 132 independent variables(symptoms) closely related to diseases were selected and optimized. This research work carried out demonstrates the disease prediction system developed using Machine learning algorithms such as Decision Tree classifier, Random forest classifier, and Naïve Bayes classifier.

**Title:** Comparison of Gradient Boosting and Extreme Boosting Ensemble Methods for Webpage Classification

**Author:** J Dutta, Yong Woon Kim, Dalia Dominic

**Year:** 2020

**Description:** Web page classification is an important task in various areas like web content filtering, contextual advertising and maintaining or expanding web directories etc. Machine Learning methods have been found to perform well to classify web pages, and ensemble models have been used to improve the results obtained from single classifiers. The Gradient Boosting and Extreme Boosting ensemble models are used in this work for binary classification. The dataset containing URLs of web pages have been collected manually. The comparison between the two boosting algorithms validated the improvement in accuracy and speed obtained through Extreme boosting. Extreme boosting has been found to be around ten times faster than Gradient boosting and also shows improvement in accuracy. The effect of three preprocessing techniques; lemmatization, stop words removal and regular expressions shows that these preprocessing techniques improves the accuracy of the results but not significantly.

**Title:** An Improved XGBoost Model Based on Spark for Credit Card Fraud Prediction

**Author:** Hongwei Chen, He Ai; Zhihui Yang, Weiwei Yang, Zhiwei Ye, Dawei Dong

**Year:** 2020

**Description:** Credit card fraud causes huge economic losses for many financial institutions. Given the imbalance of dataset and the huge amount of data in the field of credit card fraud, an improved XGBoost model based on Spark is proposed. In this project, the Smote algorithm was used to to balance the training set. And the XGBoost classifier based on Spark was used as the fraud detection mechanism. Finally, the test sets were classified in parallel. In the model comparison experiment, the model proposed in this project is compared with logistic regression model, decision tree model, random forest model, and original XGBoost model. The experimental results show that in the three metrics of Recall, Fl-Score, and AUC, the model proposed in this project is the best, which is 9.1%, 1.4%, and 1.2% ahead of the model ranked second respectively. In the speedup experiment, the speedup on the dataset of 70,000, 140,000, and 280,000 samples are 2.06, 3.28, and 3.75 respectively.

## 1.6 PROPOSED SYSTEM

The proposed system aims to overcome the limitations of existing systems by introducing advanced features and robust algorithms for disease prediction. In the proposed system, XGBoost (eXtreme Gradient Boosting), an Ensemble Boosting Technique, is utilized for accurate prediction of liver disease, heart disease, and diabetes.

By incorporating XGBoost, the proposed system ensures superior performance in disease prediction compared to traditional algorithms. XGBoost is known for its ability to handle complex and noisy medical datasets, resulting in more accurate and interpretable results.

Furthermore, the proposed system includes a healthcare chatbot that provides personalized health recommendations to users. This chatbot leverages the power of the XGBoost models to offer tailored advice and support based on individual healthcare needs.

To enhance usability and convenience, the proposed system incorporates a user-friendly interface (UI) that provides easy access to all the tools and functionalities in one place. This centralized UI allows users to interact with the disease prediction models and healthcare chatbot seamlessly.

The proposed system aims to be powerful, efficient, and comprehensive, delivering accurate disease predictions and personalized health recommendations. By leveraging advanced algorithms and providing a user-friendly interface, it seeks to provide a holistic solution for disease prediction and management, ultimately improving healthcare outcomes for individuals.

### **1.6.1 PROPOSED SYSTEM ADVANTAGES: -**

* XGBoost uses optimized algorithms and parallel processing techniques to handle large datasets more efficiently.
* XGBoost addresses overfitting issue by using regularization techniques, such as L1 and L2 regularization, to prevent overfitting and improve the generalization of the model.
* XGBoost uses feature selection and feature engineering techniques to reduce the number of features and improve the performance of the model.

**CHAPTER 2**

**PROJECT DESCRIPTION**

This project involves the development of a healthcare system that utilizes XGBoost (eXtreme Gradient Boosting), an Ensemble Boosting Technique, for accurate disease prediction. The system focuses on predicting liver disease, heart disease, and diabetes. Additionally, a healthcare chatbot is integrated into the system to provide personalized health recommendations. The project aims to overcome the limitations of existing systems by employing robust algorithms, offering a user-friendly interface, and delivering accurate and interpretable results.

**2.1 METHODOLOGIES**

**2.1.1 MODULES**

* Data Collection
* Data Preprocessing
* Model Training
* Model Evaluation & Hyperparameter Tuning
* Web Application
* Containerization & Deployment on Cloud

**1. Data Collection**

The Data Collection module is a crucial component of the healthcare system for disease prediction. It involves the systematic gathering and acquisition of relevant data from various sources. In this project, the data collection process aims to obtain comprehensive datasets that encompass the necessary features for accurate disease prediction. The sources include publicly available healthcare databases, research studies, clinical records, and other reliable sources of medical information. Kaggle is a platform that hosts a variety of datasets that can be used for different projects. To ensure that the data is up to date, it is important to constantly revise the dataset and make minimal to no changes to the code. This ensures that the analysis is based on the most recent data available.

**2. Data Preprocessing**

The Data Preprocessing module plays a vital role in preparing the collected healthcare data for further analysis and model training. It involves a series of data transformation and normalization techniques to enhance the quality and usability of the data. The preprocessing steps typically include data cleaning, where missing values are handled through imputation or removal, outliers are identified and treated, and any inconsistencies or errors in the data are resolved. Feature scaling and normalization methods are applied to ensure that all features are on a similar scale, preventing certain features from dominating the analysis due to their larger magnitude. Categorical variables are encoded into numerical representations suitable for machine learning algorithms. Feature selection or dimensionality reduction techniques may also be employed to eliminate redundant or irrelevant features, reducing model complexity and improving computational efficiency. The preprocessing module aims to transform the raw data into a clean, structured, and standardized format that is suitable for training accurate and robust disease prediction models.

**3. Model Training**

The Model Training module is responsible for developing accurate disease prediction models using machine learning techniques. It involves training the models on the preprocessed healthcare data to learn patterns, relationships, and predictive features associated with different diseases. Various machine learning algorithms, such as XGBoost (eXtreme Gradient Boosting), can be employed for this purpose. The training process involves splitting the dataset into training and validation subsets to assess model performance and prevent overfitting. The models are trained using optimization techniques, such as gradient descent, to iteratively adjust model parameters and minimize the prediction errors. Hyperparameter tuning is also performed to find the optimal configuration that yields the best model performance. Model evaluation metrics, such as accuracy, precision, recall, and F1 score, are used to assess the model's performance on the training and validation datasets. The trained disease prediction models aim to provide accurate and reliable predictions, enabling early detection and intervention for improved healthcare outcomes.

**4. Model Evaluation & Hyperparameter Tuning**

The Model Evaluation & Hyperparameter Tuning module is an essential component of the healthcare system, focusing on assessing and fine-tuning the performance of the trained disease prediction models. This module involves evaluating the models using appropriate evaluation metrics such as accuracy, precision, recall, and F1 score. The models are tested on separate testing datasets to assess their generalization capability and their ability to make accurate predictions on unseen data. Additionally, hyperparameter tuning techniques are employed to optimize the models' performance by finding the best combination of hyperparameters. Hyperparameters control the behavior and complexity of the models, and their values can significantly impact the prediction accuracy. Techniques like grid search, random search, or Bayesian optimization can be used to systematically explore the hyperparameter space and find the optimal configuration.

**5. Web Application**

The Web Application module serves as a user-friendly interface for disease prediction, leveraging the Streamlit framework. It comprises multiple pages for predicting diabetes, heart disease, liver disease, and interacting with a healthcare chatbot. Users input relevant medical data, triggering pre-trained machine learning models to make predictions. The application displays the results and diagnoses, indicating whether the person has the respective disease or not. The chatbot page enables users to ask healthcare-related queries, utilizing the OpenAI API to generate responses. The Web Application module provides an accessible platform for disease prediction and healthcare interactions, enhancing user experience and facilitating informed decision-making.

**6. Containerization & Deployment on Cloud**

The Containerization & Deployment on Cloud module employs Docker and Google Cloud Platform for the containerization and cloud deployment of the web application. Docker facilitates the creation of a container image containing all necessary dependencies, libraries, and configurations, while Heroku serves as the cloud platform for hosting the containerized application. By leveraging Docker and Google Cloud Platform, the module enables seamless deployment, scalability, and secure access to the web application, empowering users to utilize the disease prediction and healthcare chatbot functionalities effortlessly.

**CHAPTER 3**

**REQUIREMENTS ENGINEERING**

Requirements engineering is a critical phase in software development where the needs, expectations, and constraints are considered, analysed, and documented to define the desired system behaviour. It involves gathering, prioritizing, and validating requirements to ensure that the final software solution meets the intended objectives and user.

**3.2 HARDWARE REQUIREMENTS**

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system does and not how it should be implemented.

* Processor : Core i3 Processor
* Ram : 4GB DDR3 RAM
* Hard Disk : 500 GB

**3.3 SOFTWARE REQUIREMENTS**

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team’s progress throughout the development activity.

* Operating System : Linux 3.0/ Windows 7
* Front End : HTML, CSS, Streamlit, Python
* Containerization : Docker
* Back End : Cloud, Python
* Version Control : GitHub

**3.3 FUNCTIONAL REQUIREMENTS:**

Functional requirements define the specific functionalities and features that a software system or application must possess to fulfill the needs of its users. These requirements describe what the system should do in terms of its behavior, operations, and interactions with users and other software components. They typically include details about user interactions, data processing, input/output operations, system responses, and any specific functional capabilities or constraints that need to be met. Functional requirements provide a clear understanding of the expected system behavior and serve as a basis for designing, implementing, and testing the software.

**3.4 NON-FUNCTIONAL REQUIREMENTS**

Non-functional requirements, also known as quality attributes or system qualities, describe the overall characteristics and qualities that a software system must possess, beyond its specific functionalities. These requirements focus on the system's performance, reliability, usability, security, scalability, maintainability, and other aspects that contribute to its overall effectiveness and user satisfaction.

**Reliability**

Ensure consistent and error-free system performance under normal and exceptional conditions, minimizing failures and downtime.

**Performance**

Deliver fast response times, handle large workloads efficiently, and scale effectively as user demand grows.

**Usability**

Provide a user-friendly interface, intuitive navigation, and support for users with varying technical expertise.

**Security**

Safeguard user data through robust authentication, encryption, and measures against unauthorized access and threats.

**Scalability**

Design the system to handle increasing workloads, data volumes, and user bases without compromising performance.

**Maintainability**

Enable easy updates, bug fixes, and enhancements through well-documented code, standards, and modularity

**CHAPTER 4**

**DESIGN ENGINEERING**

Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

**4.1 SYSTEM ARCHITECTURE:**

A screenshot of a computer

Description automatically generated with medium confidence

Fig 4.1: System Architecture

## EXPLANATION

The system architecture of this project is designed as a web-based application that utilizes machine learning models for disease prediction. The application is built using the Streamlit framework for the frontend, which allows users to input their health data. The backend consists of the loaded machine learning models for diabetes, heart disease, and liver disease, and a Healthcare Chatbot which are used to process the input data and generate predictions or answers. The application provides an intuitive and user-friendly interface for users to input their health parameters, and the system processes the data using the appropriate model to provide accurate disease predictions. This architecture ensures a seamless and efficient user experience in predicting multiple diseases based on individual health data by using the a very powerful algorithm and containerization.

## 4.2 LEVEL 0 DATA FLOW DIAGRAM

**Level 0:**

For collected data create a dataframe in .csv

Search for available dataset for Diabetes, Heart & Liver Diseases

Collect Data from UCI/Kaggle

User

**4.3 LEVEL 1 DATA FLOW DIAGRAM**

**LEVEL 1:**

Train using XGBoost(eXtreme Gradient Boosting), Validate & Tune Hyperparameters & save like Model.pkl

Data Pre-processing

Disease Datasets

User

**4.3 LEVEL 2 DATA FLOW DIAGRAM**

**LEVEL 2:**

User Health data input

Load TrainedModel.pkl & Apply converted data.

Disease Prediction

Float/int, Array conversion

**EXPLANATION**

A data flow diagram (DFD) is a graphical representation of the “flow” of the data through an information system, modelling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFD’s can also be used for the visualization of data processing (structured design). A DFD shows what kind of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.