MACHINE LEARNING BASED MULTIPLE DISEASE PREDICTION SYSTEM

Mastanbi Shaik[1], Satya Anvesh Burle[2], Dileep Kumar Chalumuri[3], Venkata Raghavendra Dalavai [4], Devaraj Burle[5]

[1]*Assistant Professor, Department of ECE, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India*

[2-5]*Undergraduate Students, Department of ECE, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India*

ABSTRACT

This study introduces a novel approach to healthcare-focused machine learning models, aiming to predict multiple diseases through a unified user interface. By addressing conditions such as diabetes, heart disease, and Parkinson's disease simultaneously, the system endeavors to safeguard human health and save numerous lives. The methodology involves implementing various classification algorithms, including Support Vector Machine Model and Logistic Regression, to accurately forecast these diseases. To ascertain the most effective predictor, the accuracy of each algorithm is rigorously compared. Furthermore, tailored datasets specific to each disease are employed to ensure precision in the predicted outcomes. Ultimately, the overarching goal is to leverage machine learning technology to develop a user-friendly web platform capable of forecasting diseases encompassing diabetes, heart disease, and Parkinson's disease.

Keyword: - *Machine Learning, Multiple disease prediction, Public Website, and User-friendly interface*

1. INTRODUCTION

In the evolving landscape of healthcare, the integration of machine learning techniques marks a pivotal advancement. Predictive models have emerged as indispensable assets, enabling early detection and management of diseases. At the forefront of this transformation is the Multiple Disease Prediction System, which harnesses sophisticated machine learning algorithms to analyze diverse datasets encompassing patient demographics, medical histories, and lifestyle factors. By accurately forecasting the likelihood of various diseases, this system empowers healthcare professionals to intervene proactively, thus enhancing patient outcomes and potentially saving lives.

Furthermore, machine learning offers a unique advantage in identifying intricate patterns within complex datasets. Through its analytical prowess, it provides insights that traditional methods may overlook, thereby augmenting our understanding of disease dynamics. As we navigate this era of technological innovation in healthcare, the integration of machine learning promises to revolutionize disease management, shaping a future where early intervention and personalized care are the cornerstones of healthcare delivery.

2. EXISTING SYSTEM

The current healthcare analysis landscape often requires separate models for individual diseases, resulting in a time-consuming process for users who must navigate between different models for conditions like diabetes or heart disease. This fragmented approach becomes particularly concerning for individuals with multiple health issues, as existing systems may fail to predict all relevant diseases in advance, potentially leading to increased mortality rates. There is a pressing need for a more integrated solution that consolidates multiple disease models into a unified platform. Such an approach would streamline the analysis process, providing users with comprehensive insights into their health status while also mitigating risks associated with undetected or untreated conditions. Ultimately, this integrated approach has the potential to significantly improve patient outcomes and enhance the quality of care.

2.1 Disadvantages in existing system

Various existing systems in healthcare exhibit different disadvantages. Among the most prevalent issues are the absence of public websites for many prediction models, rendering it difficult for users to execute the code on their systems. Additionally, a considerable number of people lack the technical know-how to run code independently. Furthermore, many prediction models focus solely on forecasting a single disease, thereby limiting users from assessing multiple conditions within the same model. Another common drawback is the use of local websites without user-friendly interfaces, making it challenging for users to navigate and understand how to utilize the model for disease prediction. Some models even lack local websites altogether, requiring users to manually train and test the data. Lastly, a notable disadvantage present in certain models is their lower accuracy in predicting outcomes.

3. PROPOSED SYSTEM

In this advanced multi-disease prediction model, users gain the remarkable ability to forecast multiple diseases simultaneously, streamlining the process and sparing them the hassle of navigating between different models. This innovative approach not only saves valuable time but also holds promise in reducing mortality rates by facilitating the early detection of multiple conditions. Moreover, the accompanying webpage serves as a comprehensive platform, not only predicting diseases but also providing insights into user-specific concerns associated with identified conditions.

Furthermore, the model's integration with GitHub ensures seamless updates, allowing for the addition of new diseases to the webpage with ease. Its intuitive and user-friendly interface enhances accessibility, empowering users to predict diseases effortlessly. What's more, the model's status as a public website means that users can access it from any location via smart devices and an internet connection, simply by navigating to the webpage's URL. This blend of convenience, functionality, and accessibility marks a significant advancement in disease prediction technology, promising to revolutionize healthcare delivery and improve patient outcomes on a global scale.

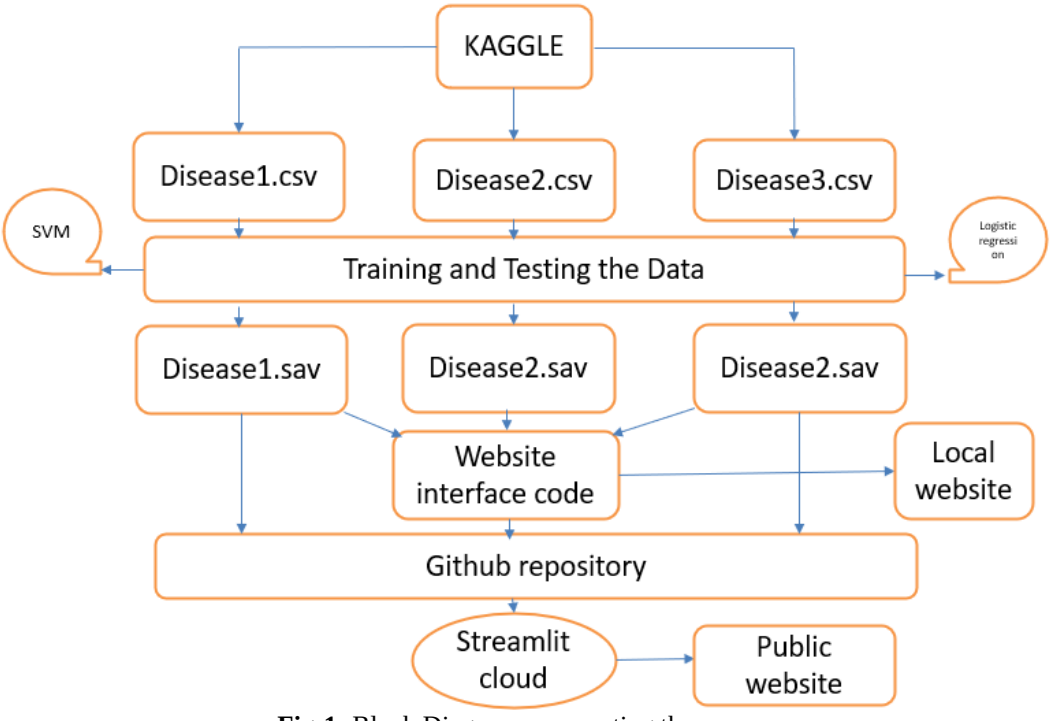

Fig-1: Block Diagram representing the process

3.1 Prerequisites:

- **Google co-lab:** Google Colab is a free cloud-based platform by Google for collaborative Python coding. It offers a Jupyter notebook interface, runs entirely in the cloud, and provides access to powerful computing resources like GPUs and TPUs. Users can seamlessly save and share their work via integration with Google Drive.
- **Anaconda:** Anaconda is a distribution of Python and R languages tailored for data science and machine learning. It comes with pre-installed packages like Jupyter notebooks and Spyder IDE, simplifying setup and management. Anaconda streamlines workflows for developers and data scientists, providing a robust platform for building data-driven applications.
- **Spyder Notebook:** Spyder Notebook is an integrated development environment (IDE) bundled with the Anaconda distribution. It provides a convenient interface for interactive Python coding and data analysis, combining the features of a Jupyter notebook with the capabilities of Spyder IDE. With Spyder Notebook, users can write and execute Python code in cells, visualize data, and document their work using markdown cells. It offers a flexible and user-friendly environment for scientific computing and data exploration within the Anaconda ecosystem.
- **Github:** GitHub is a web-based platform for version control and collaboration, primarily used for managing and sharing code repositories. It offers features such as issue tracking, pull requests, and project management tools, making it a central hub for software development teams to collaborate on projects efficiently. With GitHub, users can track changes to their code, collaborate with others, and contribute to open-source projects.
- **Streamlit-cloud:** Streamlit Cloud is a platform that allows users to deploy and share Streamlit web applications with ease. It simplifies the process of turning data scripts into shareable web apps, providing a seamless experience for data scientists and developers. With Streamlit Cloud, users can quickly deploy their apps and share them with others without worrying about infrastructure or deployment complexities.

3.2 Libraries:

- **numpy:** It is a powerful Python library for numerical computing, providing support for arrays, matrices, and mathematical functions. It offers efficient operations on large datasets, making it essential for scientific computing, data analysis, and machine learning tasks. NumPy's array-oriented computing capabilities enable faster execution of mathematical operations compared to traditional Python lists, making it a cornerstone library in the Python ecosystem for numerical computing tasks.
- **Pandas:** It is a popular Python library for data manipulation and analysis. It provides data structures and functions for efficiently working with structured data, such as tables and time series. Pandas simplifies tasks like data cleaning, transformation, and aggregation, making it a versatile tool for data wrangling in data science, machine learning, and other analytical workflows.
- **Scikit learn:** It is a widely-used Python library for machine learning, offering a simple and efficient interface for various classification, regression, clustering, and dimensionality reduction algorithms. It provides tools for data preprocessing, model selection, and evaluation, making it an essential tool for building and deploying machine learning models. With scikit-learn, users can easily implement and experiment with different machine learning techniques to solve a wide range of real-world problems.
- **Pickle:** Pickle is a Python module used for serializing and deserializing Python objects. It allows objects to be converted into a byte stream for storage or transmission, and then reconstructed back into their original form. Pickle is commonly used for saving and loading trained machine learning models, configuration settings, and other Python objects, providing a convenient way to persist data between sessions or share it with other applications.
- **Streamlit:** It is a Python library that simplifies the creation of interactive web applications directly from Python scripts. It enables data scientists and developers to quickly build and share data-driven applications using simple Python syntax, without requiring knowledge of web development languages like HTML, CSS, or JavaScript. With Streamlit, users can create custom interactive dashboards, visualizations, and user interfaces for their machine learning models, data analyses, and other Python projects with ease.
- **Streamlit-option-menu:** It is an extension of the Streamlit library in Python, providing a convenient way to create dropdown menus for selecting options within Streamlit web applications. It allows users to easily define and display dropdown menus with various options, enabling interactive selection and configuration

of parameters or settings within the application. Streamlit-option-menu simplifies the process of incorporating user input and customization into Streamlit apps, enhancing their functionality and usability.

3.3 Algorithms used:

- **Support Vector Machine (SVM):** Support Vector Machine (SVM) is a supervised learning algorithm designed for classification tasks. Its core objective is to find the optimal hyperplane that maximizes the margin between different classes in a high-dimensional feature space. By identifying support vectors, which are data points closest to the decision boundary, SVM ensures robustness and efficiency in defining the hyperplane. Through the use of kernel functions, SVM can handle non-linear decision boundaries by mapping the input features into a higher-dimensional space, allowing for effective separation of classes. Regularization parameter C is utilized to balance between minimizing classification error and maximizing margin, thereby controlling overfitting. SVM's versatility in handling both linear and non-linear classification problems, along with its ability to generalize well, makes it a widely-used algorithm in various domains of machine learning.

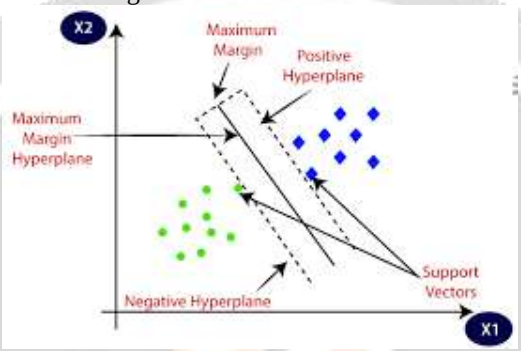

Fig-2: Support Vector Machine in machine learning

- **Logistic Regression:** Logistic Regression is a fundamental supervised learning algorithm used primarily for binary classification tasks. Despite its name, it's a method for classification, not regression. It models the probability that a given input belongs to a particular class using the logistic function, which maps input features onto a probability score between 0 and 1. Logistic Regression assumes a linear relationship between the input features and the log-odds of the outcome. During training, it optimizes parameters (coefficients) using techniques like maximum likelihood estimation or gradient descent to minimize the error between predicted probabilities and actual class labels. It's widely used due to its simplicity, interpretability, and efficiency, especially for problems where the decision boundary between classes is linear or can be well approximated by a linear function. Additionally, it can be extended to handle multi-class classification using techniques like one-vs-rest or SoftMax regression.

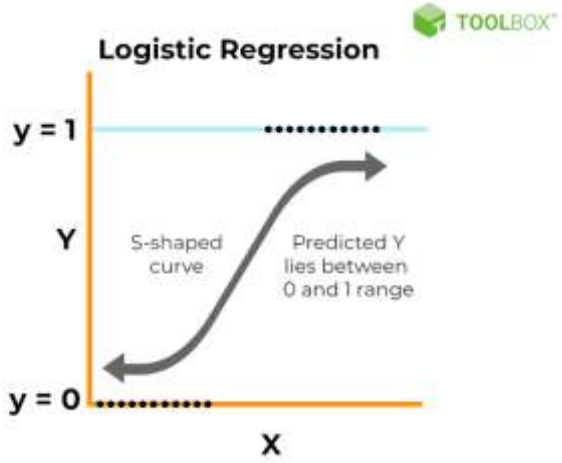

Fig-3: Logistic regression in machine learning

3.4 Methodology:

In this prediction model, users can access a public website via its URL to initiate the disease prediction process. Upon reaching the site, users navigate through a menu to select the disease of interest. Subsequently, they input vital details into designated boxes, providing crucial information for the prediction algorithm. After inputting all necessary data, users trigger the prediction by pressing the enter key.

Once the prediction is initiated, the system conducts an analysis based on the provided information. If the analysis indicates that the user has the selected disease, the system displays a confirmation along with any associated health issues identified. Conversely, if the analysis concludes that the user does not have the selected disease, the system provides a notification to that effect. However, even in the absence of the selected disease, if the analysis identifies any other health issues, the system promptly alerts the user to these concerns. This comprehensive approach ensures that users receive accurate and actionable insights into their health status, facilitating informed decision-making and proactive healthcare management.

Steps to implement are as follows:

- Install the necessary prerequisites on your system to begin the setup process.
- Create a GitHub account using your email address to facilitate version control and collaboration.
- Similarly, create an account on Streamlit Cloud with the same email address for deploying web applications.
- Utilize datasets from platforms like Kaggle to procure data relevant to the diseases being analyzed.
- Launch Google Colab, a cloud-based platform, and open a notebook to perform data analysis and model training using SVM and Logistic regression algorithms.
- Following data processing and model training in Google Colab, save the trained models as .sav files using the Pickle library.
- Open Anaconda, a Python distribution, and create a new environment to manage dependencies. Install required libraries within this environment and launch Spyder IDE for development.
- Employ the .sav files generated earlier to integrate model functionalities into a web interface using Spyder.
- Once the development is complete, deploy all files, including code and trained models, to a GitHub repository for version control and easy access.
- Access Streamlit Cloud via a web browser and use the GitHub repository to create a public website for hosting the prediction model.
- Finally, obtain the URL from Streamlit Cloud, enabling users to access the public website and utilize the prediction model for disease forecasting.

4. RESULT

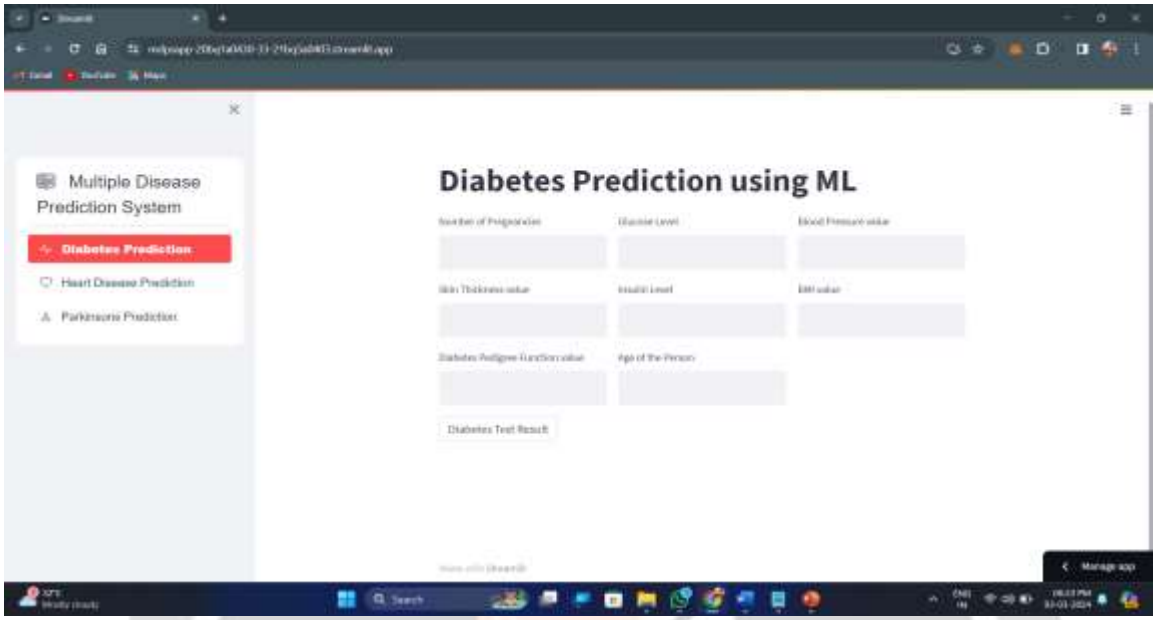

Fig-4: Public website

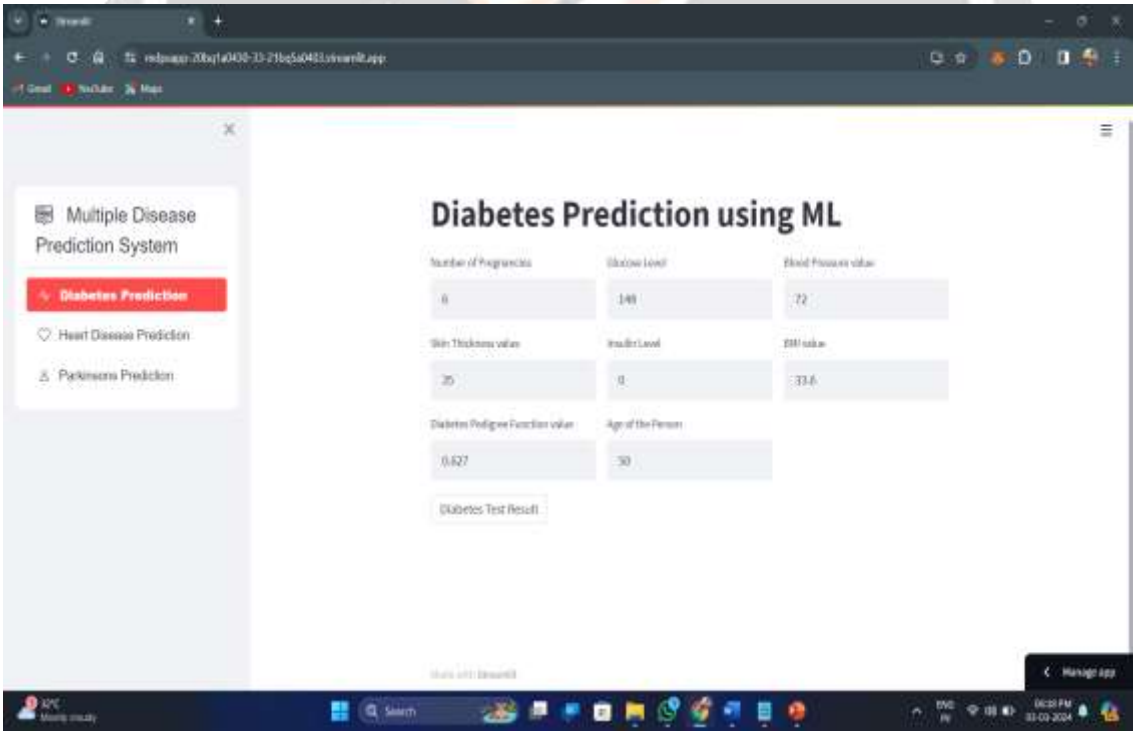

Fig-5: Select disease and give the vitals

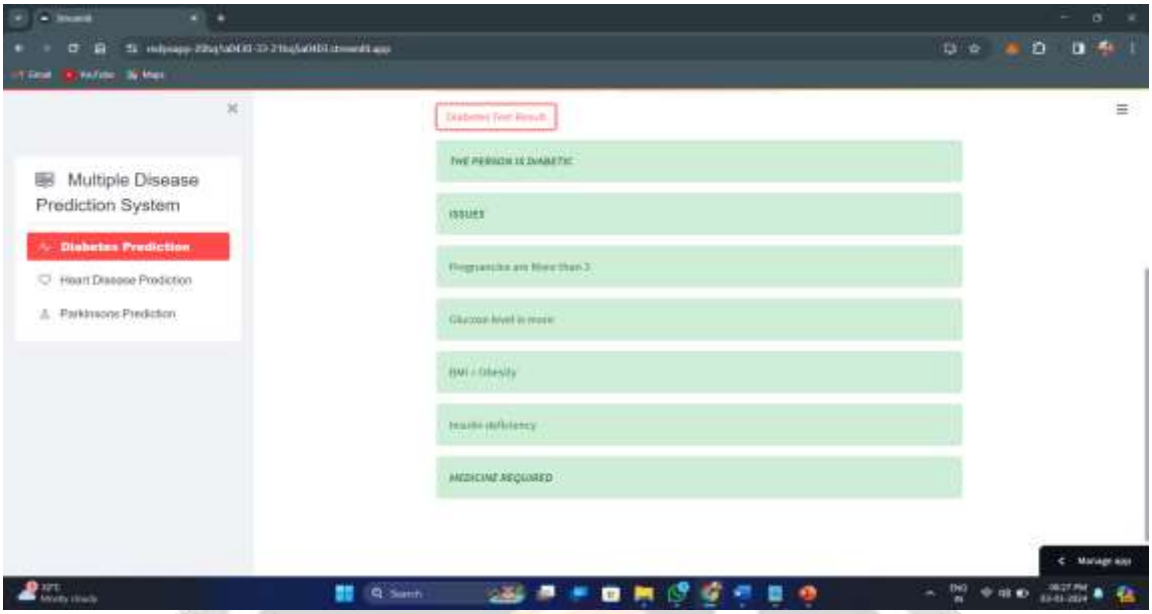

Fig-6: Prediction for the disease

5. CONCLUSION

In conclusion, this research underscores the profound impact of integrating machine learning methodologies into healthcare, particularly in disease prediction and management. Through a meticulously outlined methodology, we have developed a robust multi-disease prediction model that not only addresses the limitations of existing systems but also sets a new standard for accessibility and efficiency in healthcare technology.

By harnessing the capabilities of platforms like Google Colab, Anaconda, and Streamlit Cloud, alongside essential libraries such as NumPy, Pandas, and Scikit-learn, we have created a comprehensive ecosystem for data processing, model training, and web application deployment. This approach not only simplifies the technical complexities but also enhances collaboration and knowledge-sharing within the healthcare community.

Furthermore, the incorporation of collaboration tools like GitHub facilitates version control and enables seamless sharing of code and models, fostering a culture of transparency and openness in healthcare innovation. This collaborative spirit, combined with the user-friendly interface of the developed web application, empowers individuals worldwide to take control of their health and make informed decisions.

In essence, this research represents a significant milestone in the journey towards democratizing healthcare. By providing individuals with the tools and knowledge to predict and manage multiple diseases effectively, we aim to improve health outcomes, reduce mortality rates, and ultimately enhance the quality of life for people around the globe. As we continue to refine and expand upon this work, we remain committed to advancing the frontiers of healthcare technology for the betterment of society.

6. REFERENCES

[1]. https://docs.anaconda.com/free/anaconda/install/windows/
[2]. https://docs.spyder-ide.org/current/plugins/notebook.html
[3]. https://docs.github.com/en/get-started/onboarding/getting-started-with-your-github-account
[4]. https://colab.research.google.com/
[5]. https://share.streamlit.io/deploy
[6]. https://ieeexplore.ieee.org/document/10094382