Multi Disease Prediction Model by using Machine Learning and Flask API

# **Abstract:**

A large number of machine learning models now in use for healthcare analysis focus on just one disease at a time. For example, separate analyses could be conducted for diabetes, cancer, or skin conditions. There isn't a single system where an analysis can predict multiple diseases at once. This study proposes a system which is used to predict multiple diseases by using Flask API. Diabetes, cardiovascular disease, and heart disease were all analyzed in this study.

Later on, other illnesses such as skin conditions, fever analysis, and numerous other illnesses may be included. TensorFlow, Flask API, and machine learning techniques were employed to implement numerous illness analyses Python pickling is used to save the model behavior and python unpickling is used to load the pickle file whenever required. The significance of this investigation's analysis in the when a sickness is analyzed, all of the factors that contribute to its development are taken into account, making it feasible to determine the greatest amount of harm the illness may inflict.

Final models behavior will be saved as a python pickle file. The Flask API is created. The disease's name and parameters must be sent by the user in order to access this API. Using Flask API, the relevant model will be called and gives back the patient's status. The significance of this analysis lies in its examination of the most common diseases, which enables continuous patient monitoring and early patient education to reduce the death rate.

# **Introduction:**

One sickness at a time was taken into consideration throughout extensive research of the health care systems now in place. Most research focuses on a single illness. Anytime a company wishes to examine their patient's medical records, they will need to use a large number of models. The current system's methodology is helpful for analyzing specific diseases exclusively. These days, a greater number of people die from diseases that are difficult to diagnose. Even after being healed of one illness, the patient could still have another one.

A few parameters were employed by certain systems in use to analyze the sickness. Because of this, it might not be able to determine which diseases will result from the influence

of such illness. For instance, due to diabetes, there may be a chance of heart disease, neuropath, retinopathy, hearing loss, and dementia.

This study examined data sets related to diabetes, heart disease, and cardiovascular disease. Many more illnesses in the future, such as skin conditions, could be Infections, fever-related illnesses, among numerous others. This study is adaptable and has now been expanded to include a wide range of disorders. The developer must include the model file for the new illness analysis when adding any new disease analysis to this current API. When developing a new disease, the developer has to prepare python picking to save model behavior. When using this Flask API, the developer can load pickled files to retrieve the model behavior.

If the user wants to analyze the patient's health, they can either utilize the report's parameters to predict a certain disease or are utilized to forecast other illnesses, then this analysis will yield the greatest number of pertinent illnesses. This study goal is to stop the death ratio from rising daily by alerting people ahead of time about their health conditions. The cost of patient analysis can be minimized because several diseases can be predicted and modeled in one location.

## PROPOSED WORK AND PROCEDURE FOR MODEL DESIGN:

## **Existing system:**

Numerous analyses that are currently in use analyze certain diseases. When a user needs to utilize one analysis for diabetes and another for heart disease, they need to use the same analysis once more models must be used by the user. This procedure takes some time. Additionally, if a user has multiple diseases and the current system can only anticipate one of them, there is a potential that the death rate may rise as a result of the inability to detect the other sickness beforehand.

## **Proposed system:**

The proposed system integrates machine learning algorithms within a Streamlit-Flask web application to predict three critical health conditions: heart disease, diabetes, and pancreatic disease. This system offers a user-friendly interface where individuals can input relevant health data and receive predictions regarding their likelihood of developing these diseases.

Utilizing machine learning models trained on comprehensive datasets, the system employs predictive analytics to generate accurate assessments. Users interact with the application through an intuitive web interface, which guides them through the input process and provides clear outputs regarding their health risks.

The system serves as a preventive healthcare tool, empowering individuals to proactively manage their well-being by understanding potential health risks early on. By leveraging the power of machine learning, the system can analyze complex patterns within health data to provide personalized predictions, fostering informed decision-making and encouraging lifestyle modifications to mitigate disease risks.

Moreover, the integration of Streamlit and Flask enables seamless deployment and scalability, making the system accessible to a wide audience. With its user-centric design and robust predictive capabilities, the proposed system represents a valuable resource in promoting public health awareness and facilitating early disease detection, thereby contributing to improved healthcare outcomes and quality of life.

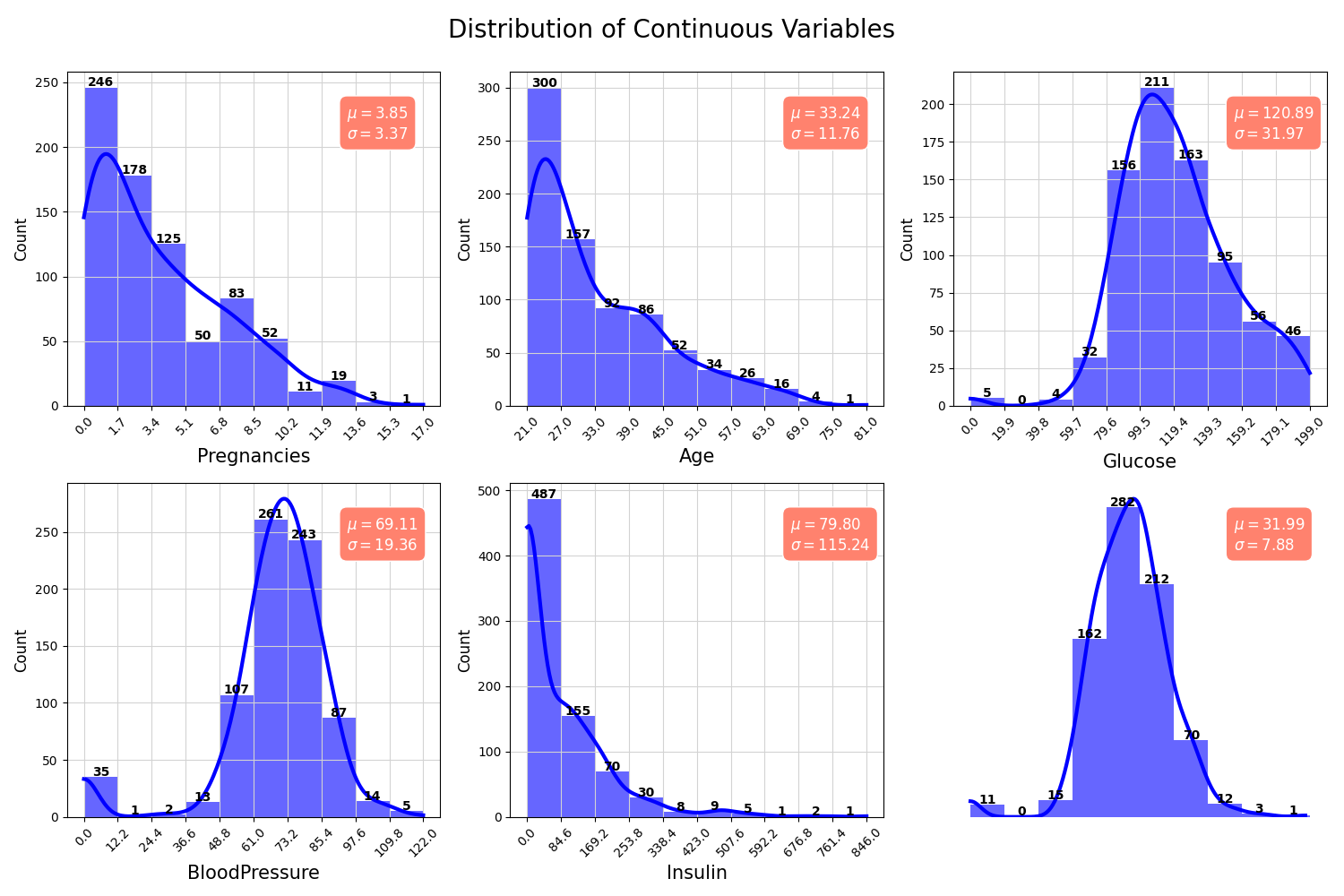
# **Methodology:**

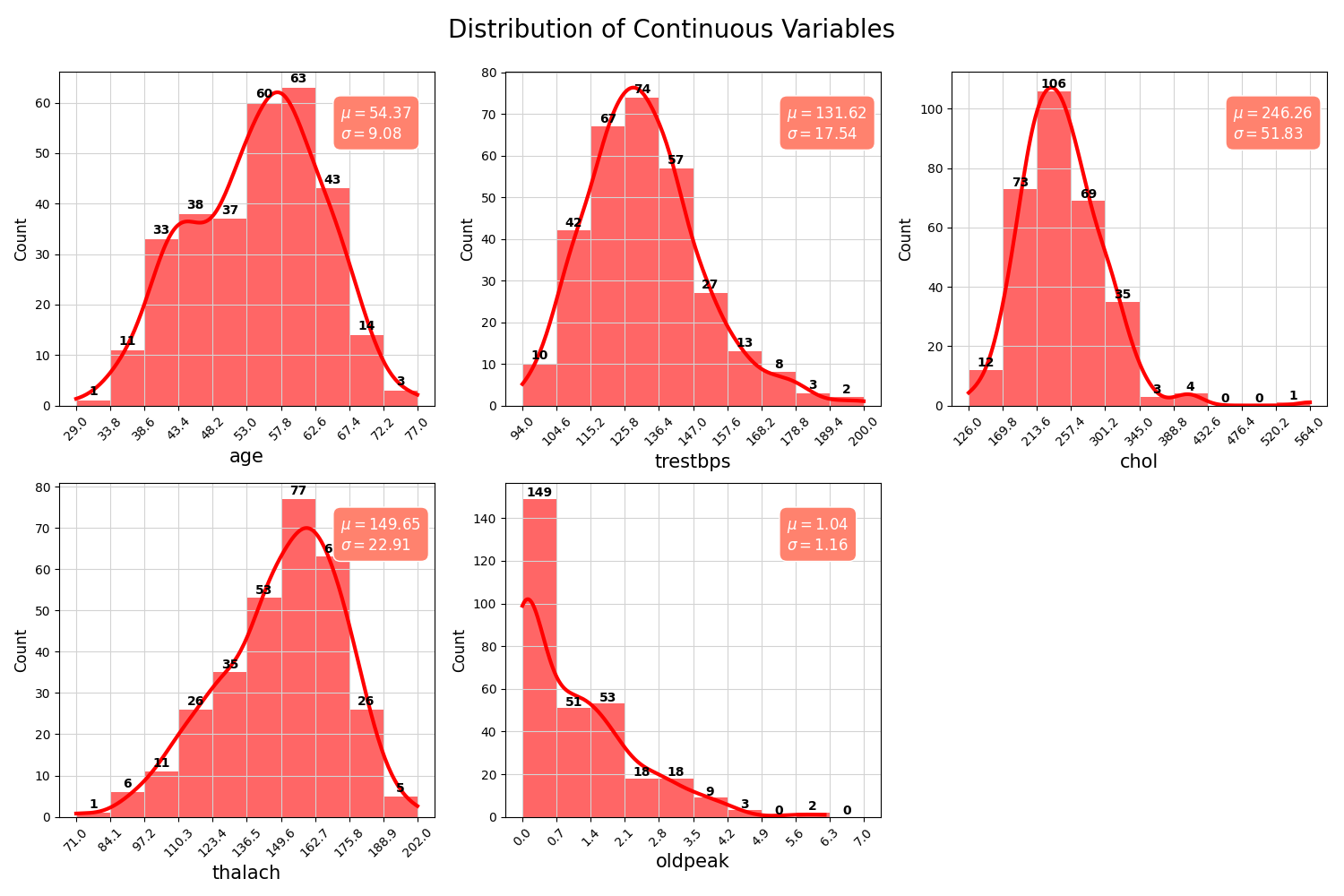
**Data Collection and Preprocessing:**

Gather datasets containing relevant health features such as demographic information, medical history, lifestyle factors, and clinical test results for heart disease, diabetes, and pancreatic disease. Ensure data quality by addressing missing values, outliers, and inconsistencies.

Exploratory Data Analysis (EDA):

Perform EDA to gain insights into the distributions, correlations, and patterns within the datasets. Visualize key relationships and characteristics to understand the underlying data structure and inform feature selection.





**Feature Engineering:**

Extract and select relevant features from the datasets to build predictive models for each disease. Use domain knowledge, statistical techniques, and data transformations to enhance the predictive power of the features.

**Model Selection and Training:**

Choose appropriate machine learning algorithms for each disease prediction task based on the nature of the data and the problem complexity. Train and validate multiple models, such as logistic regression, decision trees, random forests, support vector machines, and neural networks, to identify the most effective ones.

**Model Evaluation:**

Evaluate the performance of the trained models using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC). Use cross-validation techniques to assess model generalization and robustness.

**Hyperparameter Tuning:**

Fine-tune the hyperparameters of the selected models to optimize their performance and prevent overfitting. Utilize techniques like grid search, random search, and Bayesian optimization to find the best combination of hyperparameters.

**Integration with Streamlit-Flask Web Application:**

Develop a user-friendly web interface using Streamlit and Flask frameworks to enable users to input their health data and receive disease predictions in real-time. Implement interactive features, validation checks, and error handling to enhance user experience and ensure data integrity.

## **Implementation:**

The implementation of the proposed system involved leveraging a variety of tools and frameworks within the Python ecosystem, including Pandas, Matplotlib, scikit-learn, SVM, Logistic Regression, Random Forest, Streamlit, Pickle, Flask, and the Spyder IDE within the Anaconda environment.

Pandas facilitated data manipulation and preprocessing, enabling seamless handling of datasets containing health-related features. Matplotlib was instrumental in visualizing exploratory data analysis (EDA) insights and model evaluation metrics, enhancing interpretability and insights into the data.

The scikit-learn library provided robust implementations of machine learning algorithms such as Support Vector Machines (SVM), Logistic Regression, and Random Forest, which were trained and evaluated for disease prediction tasks. Streamlit and Flask frameworks were utilized to develop a user-friendly web interface, allowing users to input their health data and receive real-time disease predictions.

Models were serialized using Pickle for efficient storage and retrieval. The Spyder IDE facilitated development and debugging, offering a convenient environment for coding and experimentation. Overall, the implementation combined the strengths of these tools and frameworks to create a comprehensive system for disease prediction, empowering users to make informed decisions about their health and well-being.

**Reading from Glucose Sensor**

A glucose sensor is a vital device used to monitor blood glucose levels in individuals with diabetes. By continuously measuring glucose levels from interstitial fluid or blood, it provides real-time data, allowing patients to manage their condition effectively. These sensors employ various technologies, including enzyme-based electrochemical reactions or fluorescence, to detect glucose concentrations accurately. The sensor data can be transmitted wirelessly to compatible devices such as smartphones or insulin pumps, enabling timely adjustments in diet, medication, or insulin dosages. Ultimately, glucose sensors empower individuals with diabetes to maintain optimal glucose levels and minimize the risk of complications associated with hypo- or hyperglycemia.

**The result analysis**

involves examining the performance of the machine learning models in predicting heart disease, diabetes, and Parkinson's disease based on user input received through Flask. Here's how the analysis might proceed:

**Heart Disease Prediction:**

After receiving user input regarding relevant health features such as age, gender, blood pressure, cholesterol levels, and exercise habits, the Flask app feeds this data into the trained heart disease prediction model. The model then generates a prediction regarding the likelihood of the user having heart disease. The analysis involves evaluating the accuracy, precision, recall, and F1-score of the heart disease prediction model based on the user input data.

**Diabetes Prediction:**

Similar to the heart disease prediction process, the Flask app collects user input related to factors such as age, BMI, family history of diabetes, blood glucose levels, and lifestyle habits. The input data is then utilized by the diabetes prediction model to estimate the probability of the user developing diabetes. The analysis includes assessing the performance metrics of the diabetes prediction model and examining its predictive accuracy and reliability.

**Parkinson's Disease Prediction:**

For Parkinson's disease prediction, the Flask app prompts users to input relevant information such as age, gender, tremor severity, rigidity, bradykinesia, and postural instability. This data is utilized by the Parkinson's disease prediction model to predict the likelihood of the user having Parkinson's disease. The analysis involves evaluating the sensitivity, specificity, and overall performance of the Parkinson's disease prediction model based on the user input data.

**Overall Evaluation:**

The result analysis includes an overall evaluation of the predictive accuracy and performance of the machine learning models across all three disease prediction tasks. This evaluation helps assess the effectiveness of the system in providing accurate predictions and assisting users in understanding their health risks. Additionally, user feedback and interaction with the Flask app can provide insights into the usability and practicality of the system in real-world scenarios.

By conducting thorough result analysis based on user input through Flask, the system can offer valuable insights into the prediction of heart disease, diabetes, and Parkinson's disease, thereby empowering individuals to make informed decisions about their health and well-being.