A
Minor Project-I Report
On

"MULTIPLE DISEASE PREDICTION SYSTEM"

Submitted in partial fulfillment of
The requirements for the 5th Semester Sessional Examination of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

By
NAME
BELLANA SRI HARSHA
PATTA SATYA GUNA DEEP
LAKOJI HARSHIT
Registration No.
21UG01LE36
21UG010473
21UG010405
Under the able Supervision of
Ms. SRIBANI TRIPATHY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GIET UNIVERSITY, GUNUPUR

2023 - 24

GIET UNIVERSITY,GUNUPUR

Dist. - Rayagada, Odisha-765022, Contact:- +91 7735745535, 06857-250170,172, Visit us:- **www.giet.edu**

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project work entitled "Multiple Disease Prediction System" is done by Name- Bellana Sri Harsha, Patta Satya Guna Deep, Lakoji Harshit , Regd. No.- 21UG01LE36, 21UG010473, 21UG010405 in partial fulfillment of the requirements for the 5th Semester Sessional Examination of Bachelor of Technology in Computer Science and Engineering during the academic year 2023-24. This work is submitted to the department as a part of evaluation of 5th Semester Minor Project-I.

Project Supervisor *Class Teacher*

Project Coordinator, 3rd Year *HoD,CSE*

ACKNOWLEDGEMENT

We express our sincere gratitude to Prof. (Ms.) Shibani Tripathy of Computer science and engineering for giving us an opportunity to accomplish the project. Without his/her active support and guidance, this project report has not been successfully completed.

We also thank Mr. Bhavani Sankar Panda, Project Coordinator, Dr. Kakita Murali Gopal, Head of the Department of Computer Science and Engineering, Prof. (Dr.) Sanjay Kumar Kuanar, Dy. Dean, Computational Science, SOET for their consistent support, guidance and help.

Bellana Sri Harsha (21UG01LE36)
Patta Satya Guna Deep (21UG010473)
Lakoji Harshit (21UG010405)

TABLE OF CONTENTS

1. ABSTRACT

Advancements in healthcare technologies have paved the way for the development of predictive models that leverage machine learning algorithms to forecast the onset and progression of various diseases. This paper presents a comprehensive disease prediction system that integrates multiple machine learning techniques for enhanced accuracy and reliability. The system incorporates a diverse range of data sources, including patient demographics, clinical history, genetic information, and lifestyle factors, to provide a holistic approach to disease prediction.

The proposed system employs a preprocessing module to handle data cleaning, normalization, and feature extraction, ensuring the input data is well-suited for the subsequent machine learning models. Feature selection techniques are applied to identify the most relevant variables, reducing dimensionality and improving the efficiency of the prediction models. The system supports a wide array of input data formats, facilitating interoperability with electronic health records and other healthcare databases.

Machine learning algorithms such as Support Vector Machines, Random Forests, and Neural Networks are implemented to build predictive models for various diseases, including cardiovascular diseases, diabetes, and cancer. The ensemble learning approach combines the strengths of multiple algorithms, enhancing the overall predictive performance and robustness of the system. Model training is performed on a large and diverse dataset, ensuring the generalizability of the predictive models across different populations.

A key strength of the proposed system lies in its interpretability, as it provides insights into the significant features contributing to the predictions. Explainable AI techniques are incorporated to enhance the transparency of the models, aiding healthcare professionals in understanding and trusting the predictions. The system also supports continuous learning, allowing the models to adapt and improve over time as new data becomes available.

To validate the performance of the disease prediction system, extensive experiments are conducted on real-world datasets. The results demonstrate high accuracy, sensitivity, and specificity across multiple diseases, showcasing the efficacy of the integrated approach. The system's user-friendly interface allows healthcare practitioners to input patient data easily, receive predictions, and interpret results with minimal effort.

The process of determining a condition based on a person’s symptoms and indicators is known as medical diagnosis. In the diagnostic process, one or more diagnostic procedures, such as diagnostic tests, are performed. Diagnosis of chronic illnesses is a vital issue in the medical industry since it is based on many symptoms. It is a complex procedure that frequently leads to incorrect assumptions. When diagnosing illnesses, the clinical judgment is based mostly on the patient’s symptoms as well as the physicians' knowledge and experience. Furthermore, when medical systems evolve and new treatments become available, it becomes more difficult for physicians and doctors to stay up with the current innovations in clinical practice. For effective therapy, medical practitioners and doctors must be well-versed in all pertinent diagnostic criteria, patient history, and a mix of medication therapy. However, mistakes are possible since

they make judgments instinctively based on information and experience gained from past experience with patients. Because of factors such as multitasking, restricted analysis, and memory capacity, their cognitive capacities are restricted. As a result, it is difficult for a physician to make the right judgment on a consistent basis if he is not supported by clinical tests and patient history information. Even experienced physicians can benefit from a computer-aided diagnostic system in making sound medical judgments. Thus, medical professionals are very interested in automating the diagnosis process by integrating machine learning techniques with physician expertise. Data mining and machine learning approaches are making significant efforts to intelligently translate accessible data into valuable information in order to improve the diagnostic process's efficiency. Several studies have been conducted to explore the use of machine learning in terms of diagnostic abilities. It was discovered that, when compared to the most experienced physician, who can diagnose with 79.97% accuracy, machine learning algorithms could identify with 91.1% correctness. Machine learning techniques are explicitly used to illness datasets to extract features for optimal illness diagnosis, prediction, prevention, and therapy.

In conclusion, the proposed disease prediction system represents a significant advancement in the field of healthcare analytics. By integrating multiple machine learning algorithms, preprocessing techniques, and interpretable models, the system offers a comprehensive solution for accurate and transparent disease predictions. The versatility of the system makes it applicable to a wide range of diseases, contributing to early diagnosis and personalized healthcare interventions.

2. INTRODUCTION

According to the World Health Organization, every year more than 12 million deaths occur worldwide due to various specific Disease. Heart, Kidney and diabetics diseases are one of the biggest causes of morbidity and mortality among the population of the world. Prediction of these diseases is regarded as one of the most important subjects in the section of data analysis. The load of such diseases are rapidly increasing all over the world from the past few years. Many researches have been conducted in attempt to pinpoint the most influential factors of the diseases as well as accurately predict the overall risk. It is even highlighted as a silent killer which leads to the death of the person without obvious symptoms. The early diagnosis of the diseases plays a vital role in making decisions on lifestyle changes in high-risk patients and in turn reduces the complications. Machine learning proves to be effective in assisting in making decisions and predictions from the large quantity of data produced by the health care industry. This project aims to predict future diseases by analyzing data of patients which classifies whether they have the particular disease or not using machine-learning algorithm. Machine Learning techniques can be a boon in this regard. Even though heart, kidney and diabetic diseases can occur in different forms, there is a common set of core risk factors that influence whether someone will ultimately be at risk for these diseases or not. By collecting the data from various sources, classifying them under suitable headings & finally analyzing to extract the desired data we can say that this technique can be very well adapted to do the prediction of the disease.

2.1 PURPOSE

The purpose of a multiple disease prediction system lies in its ability to address various healthcare challenges by leveraging advanced technologies and predictive analytics. Here are key points highlighting the purpose of such a system:

- **Early Disease Detection:**
  Identify potential health risks and diseases at an early stage, enabling timely intervention and treatment.
- **Holistic Health Assessment:**
  Consider a diverse range of factors, including demographics, clinical history, genetics, and lifestyle, for a comprehensive understanding of an individual's health status.
- **Preventive Healthcare:**
  Facilitate proactive healthcare measures by predicting disease risks, allowing individuals to adopt preventive lifestyle changes and reduce the likelihood of developing certain conditions.
- **Customized Treatment Plans:**
  Provide healthcare professionals with valuable insights to tailor treatment plans based on individualized risk profiles, optimizing patient care and outcomes.

- **Resource Allocation:**

Assist healthcare providers in efficiently allocating resources by prioritizing patients at higher risk, thereby optimizing healthcare delivery and resource utilization.

- **Reduced Healthcare Costs:**
  Contribute to cost savings by preventing the onset or progression of diseases, reducing the need for extensive and expensive medical interventions.
- **Interoperability with Healthcare Systems:**
  Seamlessly integrate with electronic health records and other healthcare databases, enhancing interoperability and facilitating the exchange of critical patient information.
- **Ensemble Learning for Robust Predictions:**
  Combine the strengths of multiple machine learning algorithms through ensemble learning, improving the robustness and accuracy of disease predictions.
- **Explainable AI for Trust and Understanding:**
  Enhance transparency and trust in predictive models by incorporating explainable AI techniques, allowing healthcare professionals to understand and interpret the factors influencing predictions.
- **Continuous Learning and Adaptability:**
  Support continuous learning by updating predictive models with new data, ensuring that the system remains current and adapts to evolving healthcare trends.

2.2 PROJECT SCOPE

The scope of a multiple disease prediction system is broad and encompasses various aspects of healthcare, technology, and public health. Here are key dimensions that define the scope of such a system:

- **Disease Spectrum:**
  The system can address a wide range of diseases, including but not limited to cardiovascular diseases, diabetes, cancer, infectious diseases, and chronic conditions, offering a versatile tool for health risk assessment.
- **Demographic Diversity:**
  The scope includes the ability to cater to diverse populations with different demographic characteristics, considering factors such as age, gender, ethnicity, and geographical location in disease prediction models.
- **Data Integration:**
  The system can integrate and analyze data from various sources, including electronic health records, genetic databases, lifestyle data, and environmental factors, providing a holistic view of an individual's health.
- **Machine Learning Algorithms:**

The use of diverse machine learning algorithms, such as Support Vector Machines, Random Forests, Neural Networks, and ensemble learning techniques, expands the scope by ensuring adaptability to different types of data and diseases.

- **Interoperability with Healthcare Systems:**
  The system can seamlessly integrate with existing healthcare systems, allowing for easy adoption by healthcare providers and ensuring compatibility with electronic health records for streamlined information flow.
- **Explainable AI:**
  Incorporating explainable AI techniques enhances the interpretability of the system, making it accessible to healthcare professionals and contributing to its scope in gaining trust and acceptance in clinical settings.
- **Preventive Healthcare Focus:**
  The system's primary focus is on preventive healthcare, extending its scope to support proactive health management, early detection, and intervention strategies to reduce the burden of diseases.
- **Continuous Learning and Improvement:**
  The system has the capability for continuous learning, adapting to new data and evolving healthcare knowledge, ensuring that it remains relevant and effective over time.
- **Patient-Centric Approach:**
  The scope includes a patient-centric approach, empowering individuals to actively participate in their healthcare by providing personalized risk assessments, encouraging lifestyle modifications, and fostering a sense of responsibility for health.
- **Public Health Impact:**
  The system's insights contribute to public health planning by providing data on disease prevalence, risk factors, and trends, supporting policymakers in developing targeted interventions and strategies.

2.3 PRODUCT FEATURES

A multiple disease prediction system typically incorporates a variety of features to effectively analyze diverse datasets and provide accurate predictions for various health conditions. Here are key features commonly found in such systems:

- **Data Integration:**
  Ability to integrate data from multiple sources, including electronic health records, genetic information, lifestyle data, and environmental factors, to provide a comprehensive view of an individual's health.

- **Preprocessing Module:**
  A preprocessing module for data cleaning, normalization, and feature extraction to ensure the quality and relevance of input data for subsequent analysis.

- **Feature Selection Techniques:**
  Implementation of feature selection methods to identify and prioritize the most relevant variables, reducing dimensionality and enhancing the efficiency of predictive models.
- **Machine Learning Algorithms:**
  Integration of various machine learning algorithms such as Support Vector Machines, Random Forests, Neural Networks, and ensemble learning techniques to build predictive models for different diseases.
- **Ensemble Learning:**
  Use of ensemble learning methods to combine the predictions of multiple models, improving overall accuracy and robustness.
- **Interoperability:**
  Seamless integration with existing healthcare systems, electronic health records, and other databases to facilitate easy data exchange and interoperability.
- **Explainable AI Techniques:**
  Incorporation of explainable AI techniques to enhance transparency and interpretability, providing insights into the factors influencing predictions and fostering trust among healthcare professionals.
- **User-Friendly Interface:**
  A user-friendly interface for healthcare professionals to input patient data easily, visualize predictions, and interpret results with minimal effort.
- **Customization and Adaptability:**
  Customization options to tailor the system to specific healthcare settings and the adaptability to incorporate new data for continuous learning and model improvement.
- **Disease-Specific Models:**
  Development of disease-specific predictive models to address the unique characteristics and risk factors associated with different health conditions.
- **Real-time Predictions:**
  Capability to provide real-time predictions, enabling timely interventions and proactive healthcare management.
- **Cross-Validation Techniques:**
  Utilization of cross-validation techniques to assess the generalizability and performance of predictive models on different datasets, ensuring robustness.
- **Scalability:**
  Scalability to handle large datasets and accommodate the growing volume of healthcare information generated over time.
- **Security and Privacy Measures:**
  Implementation of robust security and privacy measures to protect sensitive patient information and ensure compliance with healthcare regulations.

- **Feedback Mechanism:**
  Incorporation of a feedback mechanism to allow healthcare professionals to provide insights, corrections, or additional information for model refinement.
- **Educational Resources:**
  Provision of educational resources and documentation to aid healthcare professionals in understanding the system, its predictions, and the underlying algorithms.
- **Public Health Analytics:**
  Features for public health analytics, allowing the system to contribute to epidemiological studies, disease surveillance, and the development of public health policies.

The combination of these features in a multiple disease prediction system enhances its overall effectiveness in providing accurate, interpretable, and actionable predictions for a wide range of health conditions.

3. WORKS DONE IN RELATED AREA

Research and work in the area of multiple disease prediction systems have gained significant attention due to the potential impact on healthcare outcomes and resource optimization. Here are some key aspects of the work done in related areas:

- **Integration of Multi-Omics Data:**
  Researchers have focused on integrating multi-omics data, including genomics, transcriptomics, and proteomics, to enhance the accuracy of disease prediction models. This approach provides a more comprehensive understanding of the molecular basis of diseases.
- **Deep Learning Approaches:**
  Deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have been applied to multiple disease prediction. These models can automatically learn hierarchical representations from complex data, contributing to improved predictive performance.
- **Explainable AI in Healthcare:**
  Efforts have been made to enhance the interpretability of predictive models through the incorporation of explainable AI techniques. This addresses the challenge of trust and understanding among healthcare professionals and patients, making the predictions more actionable.
- **Ensemble Learning Strategies:**
  Ensemble learning, which combines the predictions of multiple models, has been explored to improve the robustness and generalizability of disease prediction systems. This approach helps mitigate the limitations of individual algorithms and enhances overall performance.
- **Incorporation of Environmental and Social Factors:**
  Recognizing the impact of environmental and social determinants on health, researchers have worked on incorporating these factors into disease prediction models. This holistic approach provides a more accurate representation of an individual's risk profile.
- **Real-time Predictive Analytics:**
  The development of real-time predictive analytics tools has been a focus, allowing for timely interventions and personalized healthcare strategies. These systems enable healthcare professionals to make decisions based on the most current patient data.
- **Population Health Management:**
  Work has been done in the application of predictive analytics for population health management. This involves identifying at-risk populations, developing targeted interventions, and optimizing resource allocation to improve overall public health.

- **Cross-Disease Predictions:**
  Some studies have aimed at developing models capable of making predictions across multiple diseases simultaneously. This approach considers shared risk factors and commonalities in disease pathways to provide a more holistic view of an individual's health risks.
- **Continuous Learning Systems:**
  The development of continuous learning systems allows predictive models to adapt and improve over time as new data becomes available. This ensures that the models remain relevant in dynamic healthcare environments.
- **Clinical Validation and Implementation:**
  Research efforts have extended to the clinical validation and real-world implementation of disease prediction systems. Studies have assessed the impact of these systems on patient outcomes, healthcare workflows, and the overall quality of care.
- **Ethical and Privacy Considerations:**
  Work has been done to address ethical and privacy concerns associated with the use of sensitive health data. Researchers are exploring ways to balance the benefits of predictive analytics with the need to protect patient privacy and maintain data security.

The cumulative work in these areas contributes to the ongoing evolution and refinement of multiple disease prediction systems, bringing us closer to personalized, data-driven healthcare solutions.

4. SYSTEM ANALYSIS

4.1 USER REQUIREMENTS

Creating a multiple disease prediction system requires careful consideration of user requirements to ensure the system is effective, user-friendly, and meets the needs of both healthcare professionals and patients. Here are some key user requirements for a multiple disease prediction system:

1. **Accuracy and Reliability:**
   Users expect the system to provide accurate and reliable predictions based on relevant data. The predictive models should be well-validated and regularly updated to reflect the latest medical knowledge and research.
2. **User-Friendly Interface:**
   The system should have an intuitive and user-friendly interface that is easy to navigate for both healthcare professionals and patients. Clear and concise visualizations of data and predictions can enhance the usability of the system.
3. **Data Security and Privacy:**
   Users, especially healthcare professionals and patients, are concerned about the security and privacy of their medical data. The system must adhere to strict data protection standards and comply with relevant regulations, such as HIPAA (Health Insurance Portability and Accountability Act).
4. **Interoperability:**
   The system should be able to integrate with existing healthcare information systems and electronic health records (EHRs). This ensures seamless communication and data exchange between different components of the healthcare infrastructure.
5. **Customization and Personalization:**
   Healthcare professionals may have specific needs based on their specialties or the patient population they serve. The system should allow customization to accommodate different medical specialties and be able to provide personalized predictions based on individual patient data.
6. **Real-Time Monitoring:**
   For chronic diseases or conditions that require ongoing management, users may require real-time monitoring capabilities. The system should be able to track changes in health status and provide timely alerts or recommendations.
7. **Education and Explanation:**
   Users, especially patients, may not be familiar with medical terminology or predictive models. The system should include educational components to explain predictions in a way that is easily understandable, empowering users to make informed decisions about their health.

8. **Scalability:**

The system should be scalable to accommodate an increasing volume of users and data. As the user base grows, the infrastructure should be able to handle the additional load without compromising performance.

9. **Feedback Mechanism:**
   Users should have a mechanism to provide feedback on the accuracy of predictions and the overall performance of the system. This feedback loop is essential for continuous improvement.
10. **Mobile Accessibility:**
    Given the increasing use of mobile devices, the system should be accessible on mobile platforms, ensuring that users can access predictions and health information on the go.
11. **Training and Support:**
    Healthcare professionals and users may require training to effectively use the system. Adequate support mechanisms, such as documentation, training materials, and helpdesk support, should be in place.
12. **Cost-Effectiveness:**
    Consideration should be given to the cost-effectiveness of the system, taking into account the financial constraints of healthcare organizations and the affordability for individual users.

By addressing these user requirements, a multiple disease prediction system can be designed to better meet the diverse needs of both healthcare providers and patients.

4.2 HARDWARE REQUIREMENTS

- **Processor brand:** Intel or Ryzen
- **Processor type:** i5(intel) or 5000(Ryzen)
- **Ram size:** 8 GB
- **Internet Connectivity:** 4G/5G Internet connection

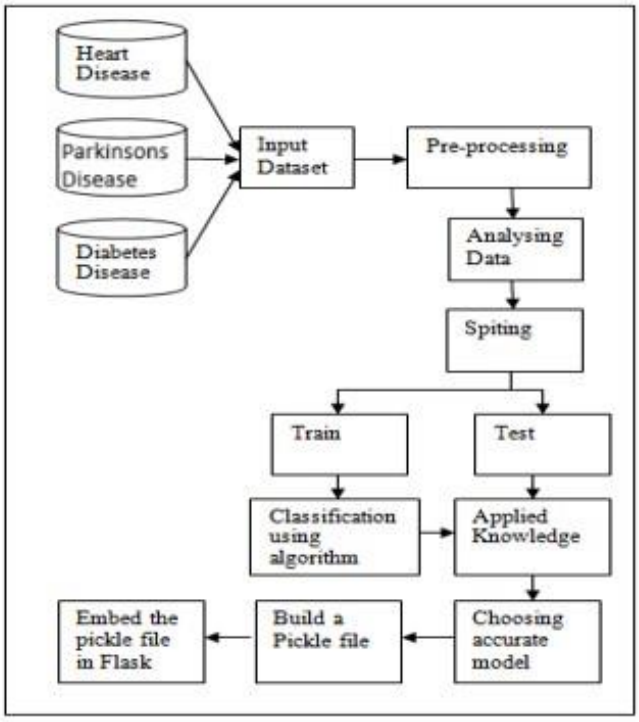
4.3 SOFTWARE REQUIREMENTS

- **Programming Language:** Python
- **Libraries and Frameworks:** Pandas, NumPy, Scikit-learn, etc.
- **Web Browser:** Compatible web browsers (Chrome, Firefox, Edge etc.)
- **Integrated Development Environment (IDE):** Spyder, Google Colabatory
- **Operating System:** Windows

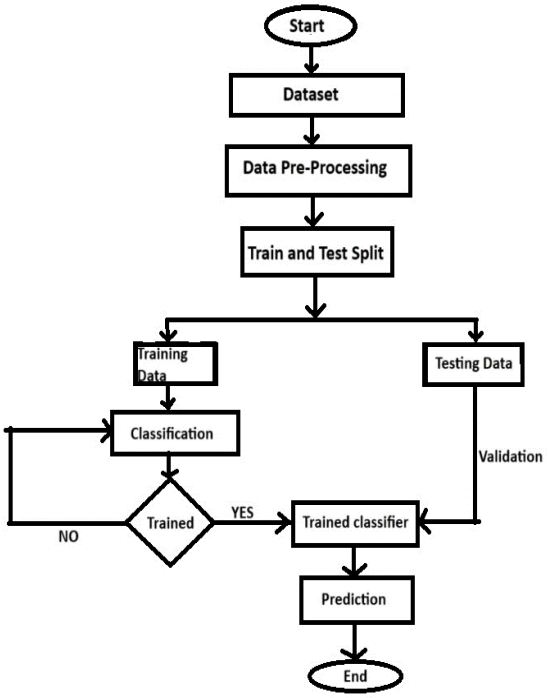
5. SYSTEM DESIGN AND ANALYSIS

5.1 HIGH LEVEL DESIGN (HLD)

5.1.1 Project Model

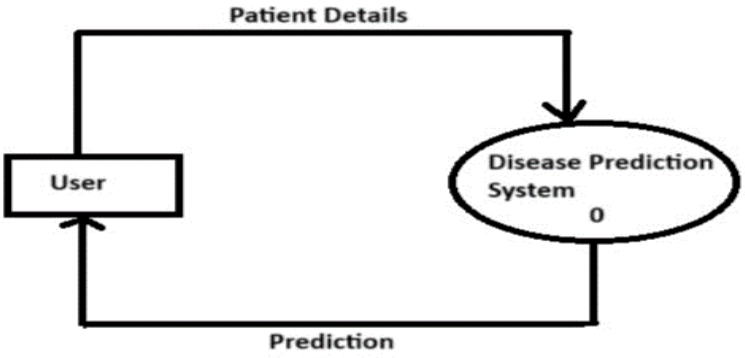


5.1.2 Structure Chart

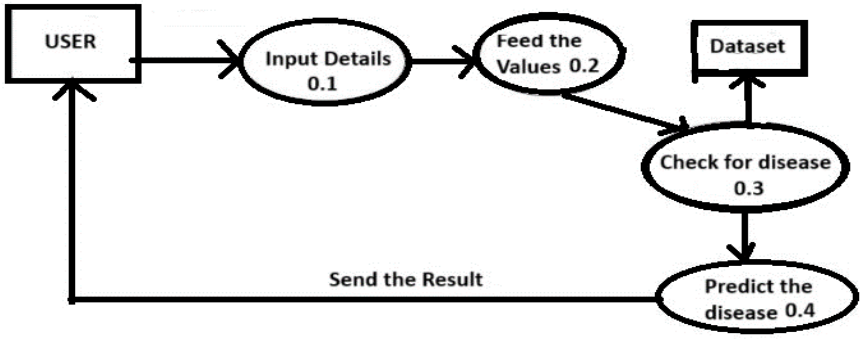


5.1.3 Data Flow Diagram

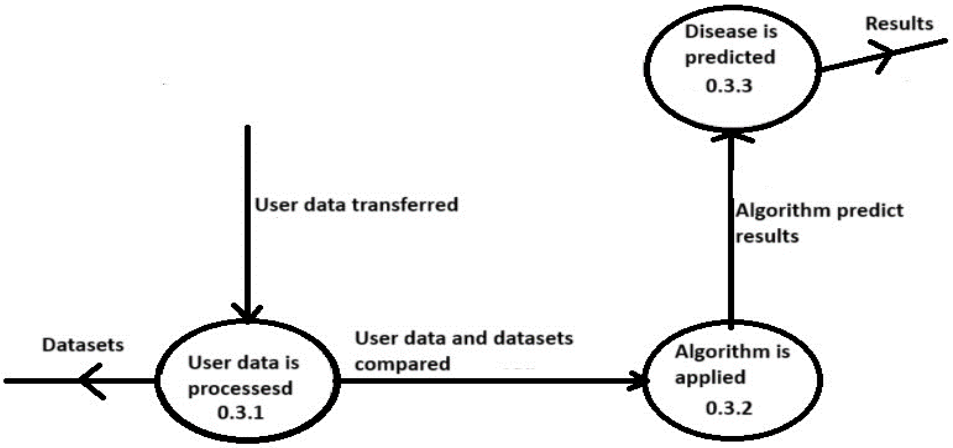
- Level 0 DFD



- Level 1 DFD

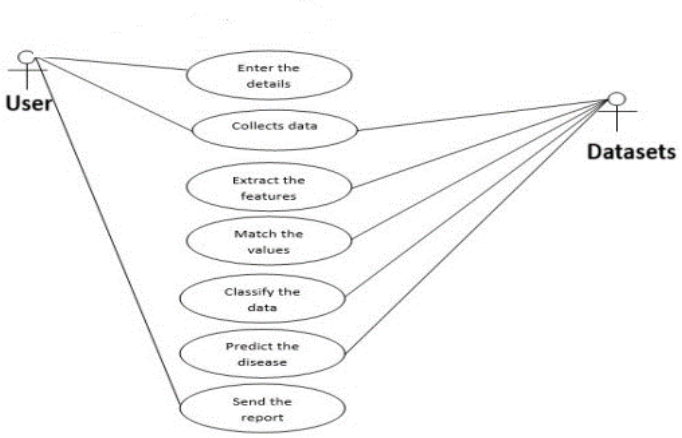


- Level 2 DFD

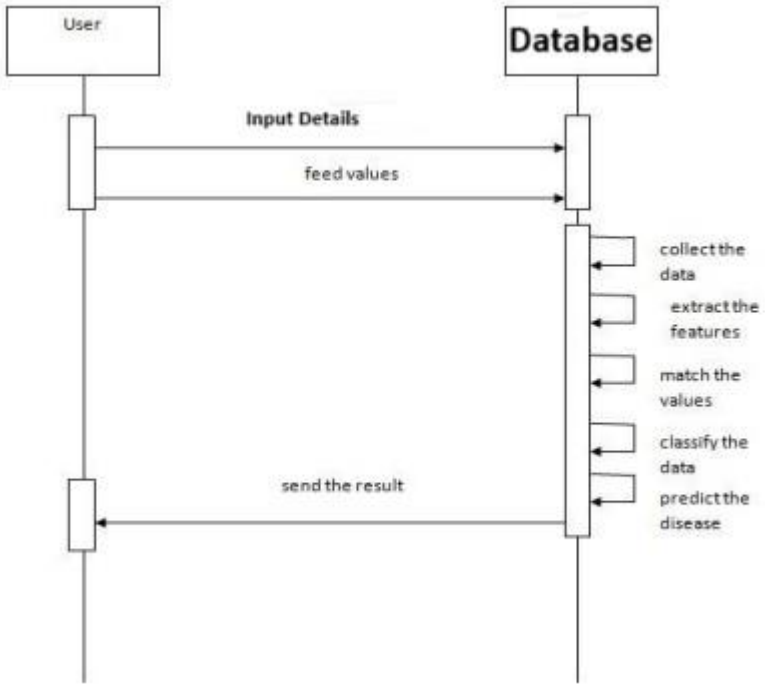


5.1.4 UML DIAGRAM

- Use Case Diagram



- Sequence Diagram



5.2 LOW LEVEL DESIGN (LLD)

5.2.1 Process Specification

1. **Collection of Dataset**: We collected three datasets namely for heart disease, parkinsons disease and diabetic disease prediction. After the collection of the dataset, we split the dataset into training data and testing data. The training dataset is used for prediction model learning and testing data is used for evaluating the prediction model. For this project, 80% of training data is used and 20% of data is used for testing as per each dataset.
2. **Selection of attributes**: Attribute or Feature selection includes the selection of appropriate attributes for the prediction system. This is used to increase the efficiency of the system. Various attributes of the patient like gender, chest pain type, fasting blood pressure, serum cholesterol, etc. are selected for the prediction.

3. **Data Pre-Processing:** Data pre-processing is an important step for the creation of a machine learning model. Initially, data may not be clean or in the required format for the model which can cause misleading outcomes. In pre-processing of data, we transform data into our required format. It is used to deal with noises, duplicates, and missing values of the dataset. Data pre-processing has the activities like importing datasets, splitting datasets, attribute scaling, etc. Preprocessing of data is required for improving the accuracy of the model.
4. **Balancing of Data:** Imbalanced datasets can be balanced in two ways. They are Under Sampling and Over Sampling

   (a) Under Sampling: In Under Sampling, dataset balance is done by the reduction of the size of the ample class. This process is considered when the amount of data is adequate.

   (b) Over Sampling: In Over Sampling, dataset balance is done by increasing the size of the scarce samples. This process is considered when the amount of data is inadequate.

5. Disease Prediction: Various machine learning algorithms like Decision Tree, Random Tree, Logistic Regression and KNN are used for classification. Comparative analysis is performed among algorithms and the algorithm that gives the highest accuracy is used for disease prediction.

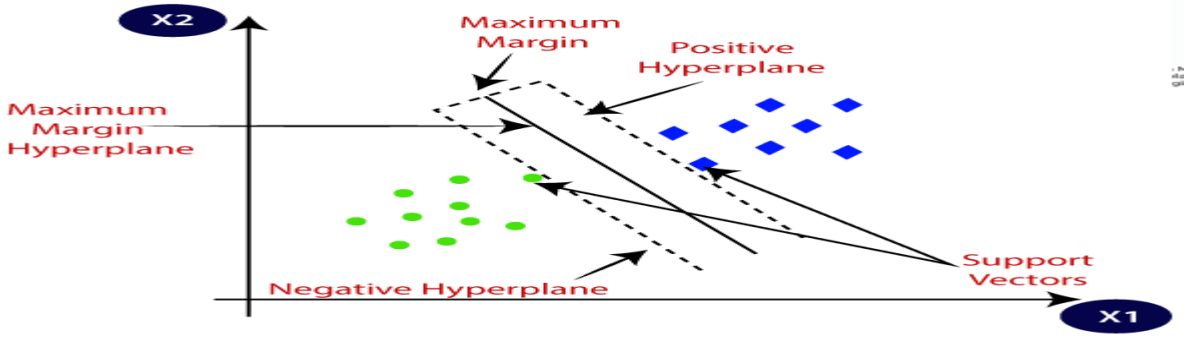
6. Developing the user interface: We have used Streamlit library to develop user interface.

5.2.2 Algorithms Used:

1. Support Vector Machine (SVM)

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

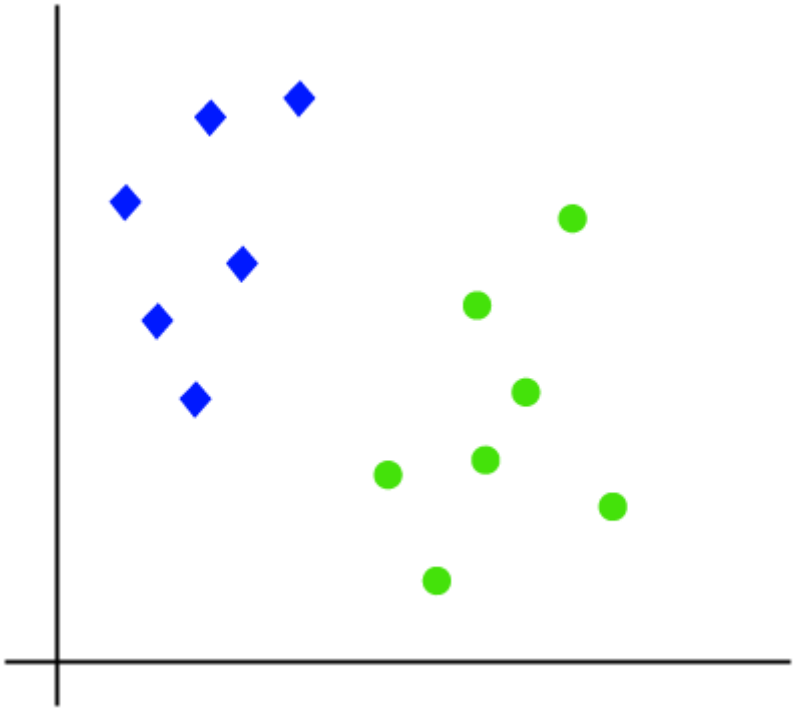


SVM can be of two types:

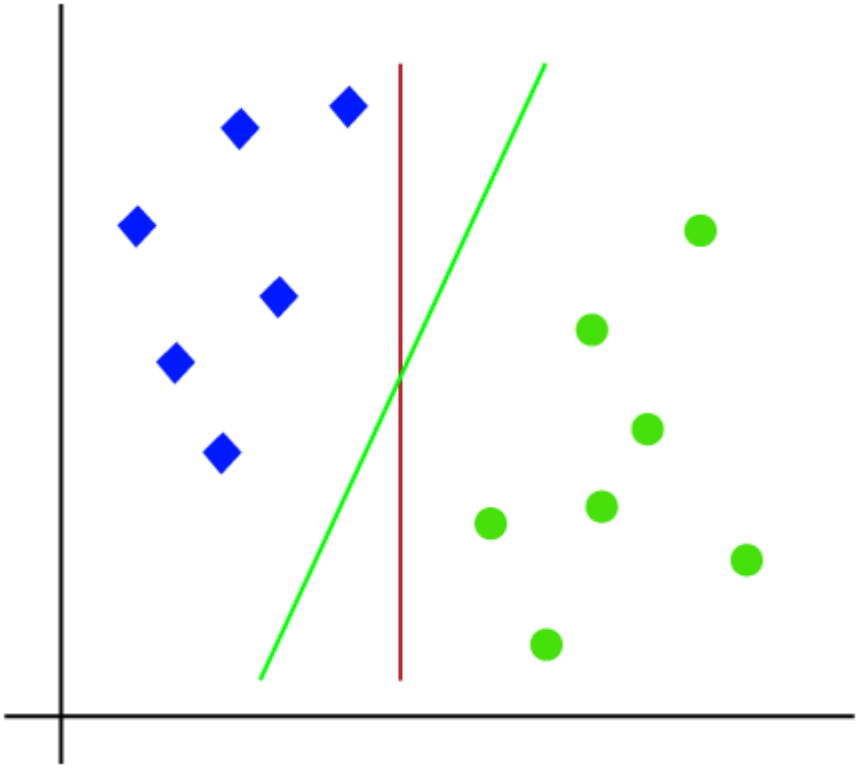
- Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- Non-linear SVM: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Linear SVM:

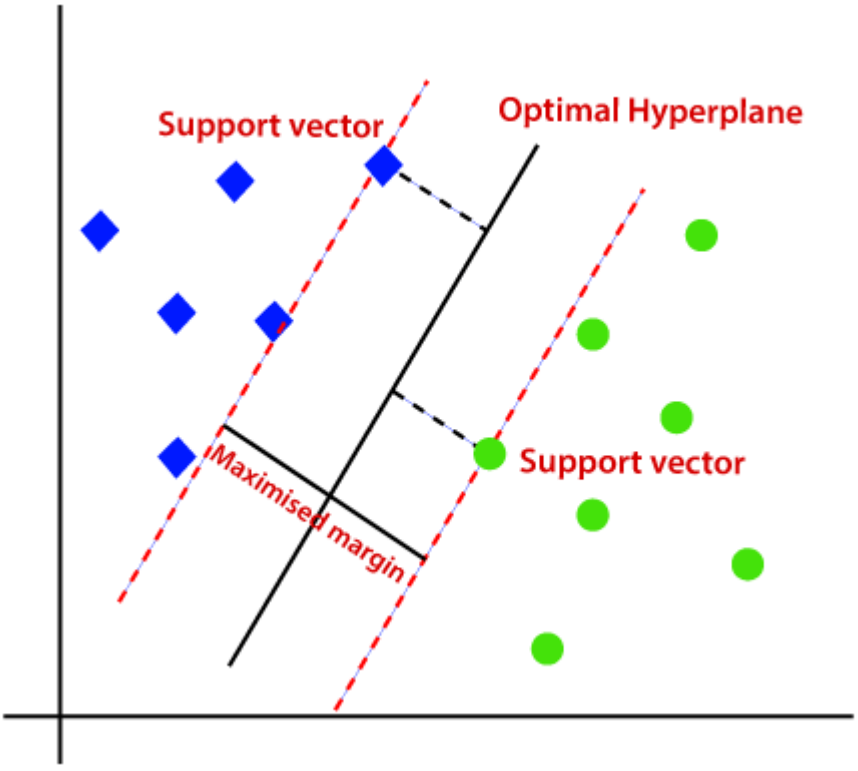
- The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:



- So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:

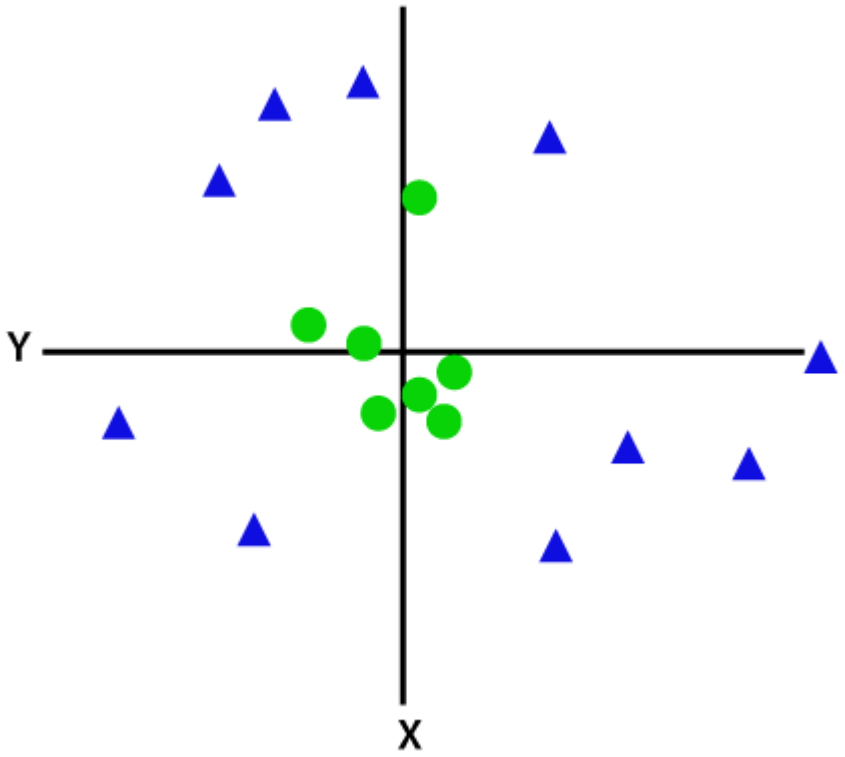


- Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a hyperplane. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as margin. And the goal of SVM is to maximize this margin. The hyperplane with maximum margin is called the optimal hyperplane.



Non-Linear SVM:

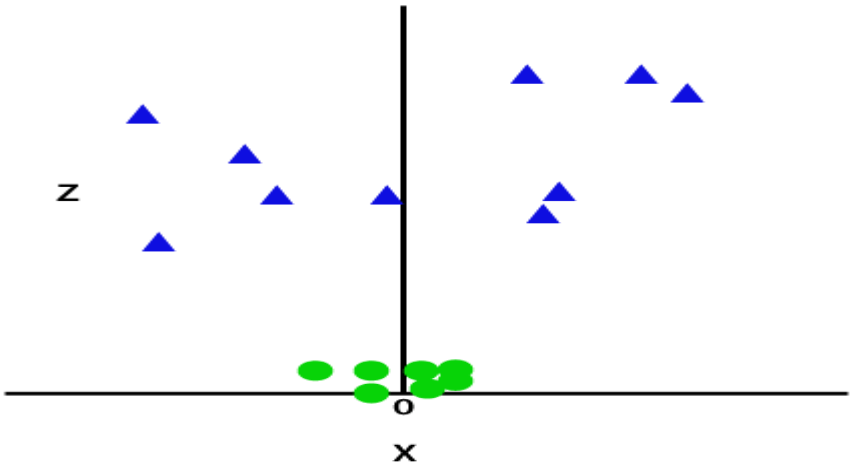
- If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



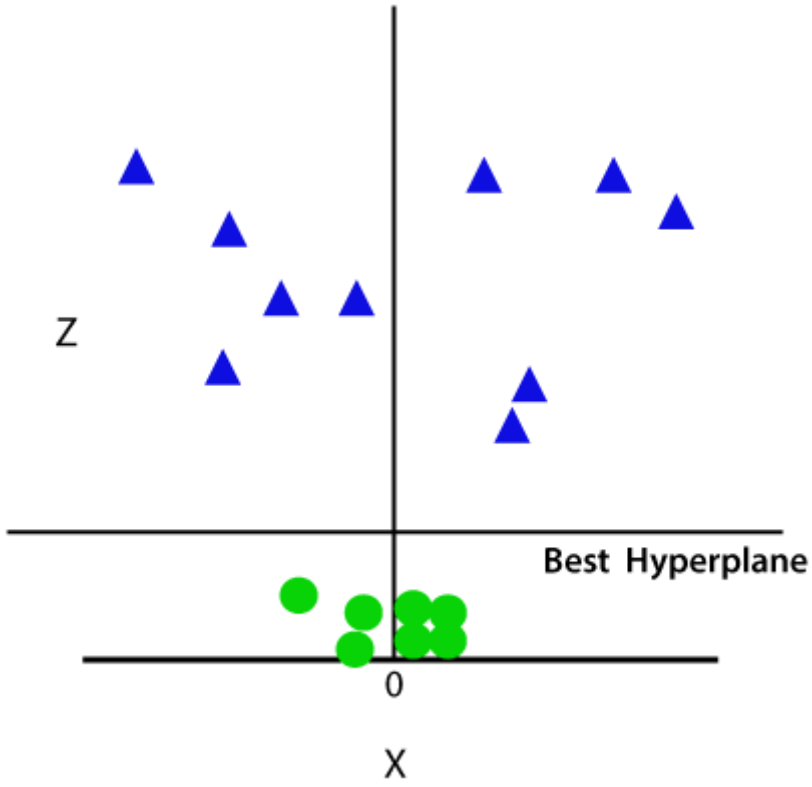
- So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as:

$$z=x^2 +y^2$$

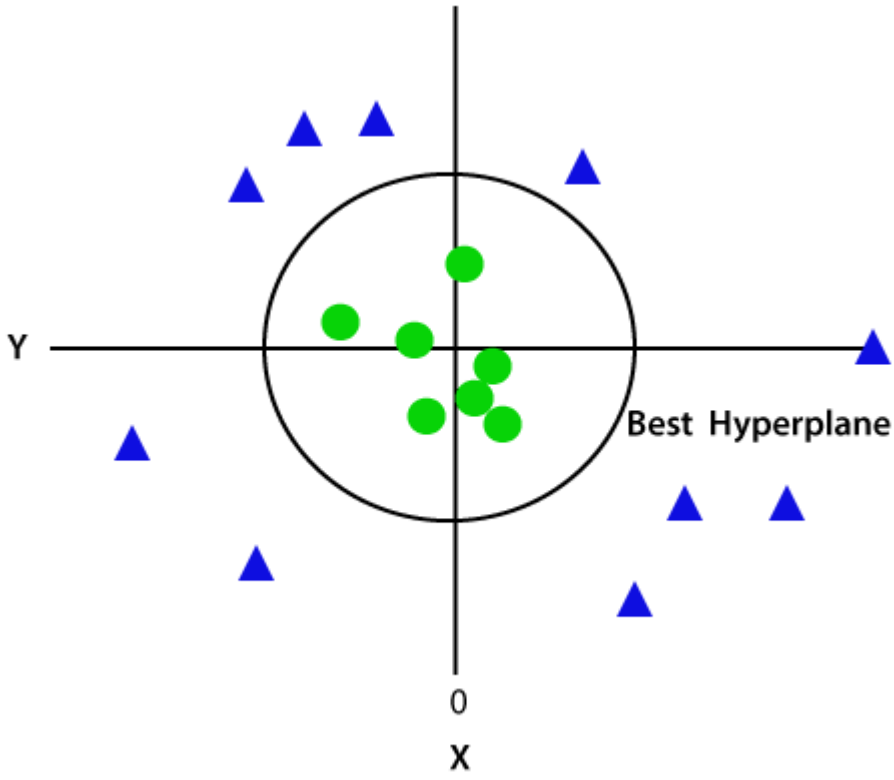
- By adding the third dimension, the sample space will become as below image:



- So now, SVM will divide the datasets into classes in the following way. Consider the below image:



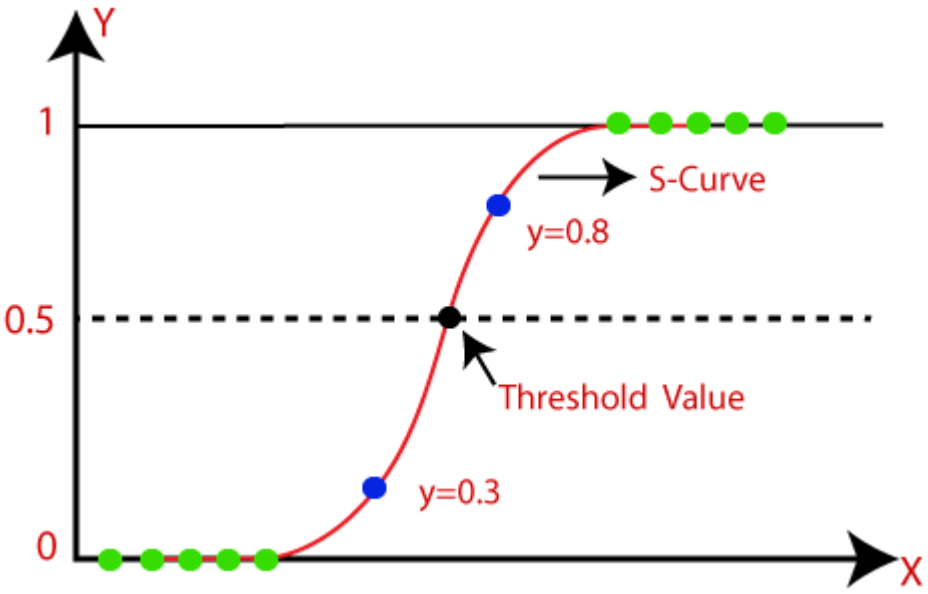
- Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become as:



- Hence we get a circumference of radius 1 in case of non-linear data.

2. Logistic Regression Algorithm

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression:

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation.

The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y= b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}\text{; 0 for y= 0, and infinity for y=1}$$

- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

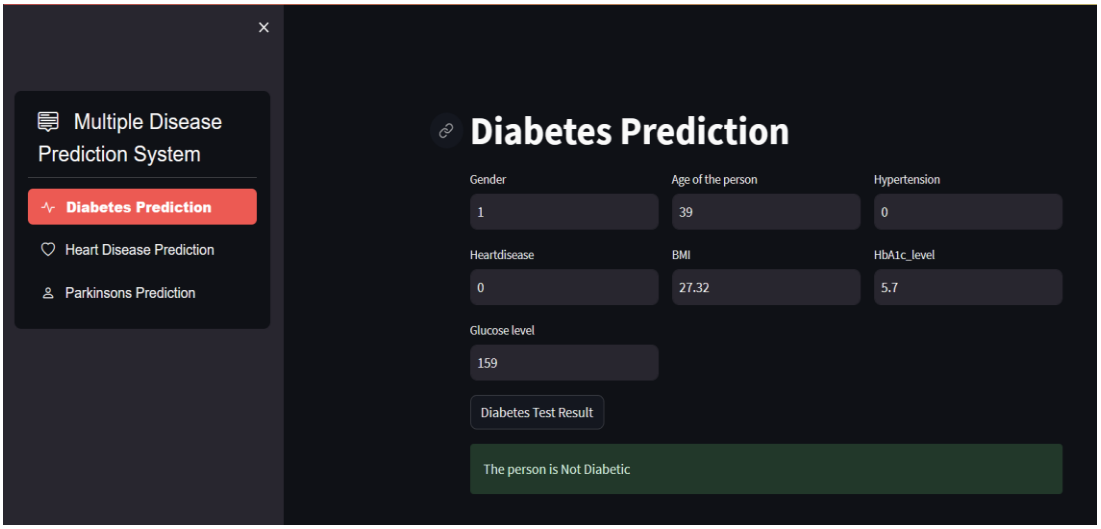
- Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Terminologies involved in Logistic Regression:

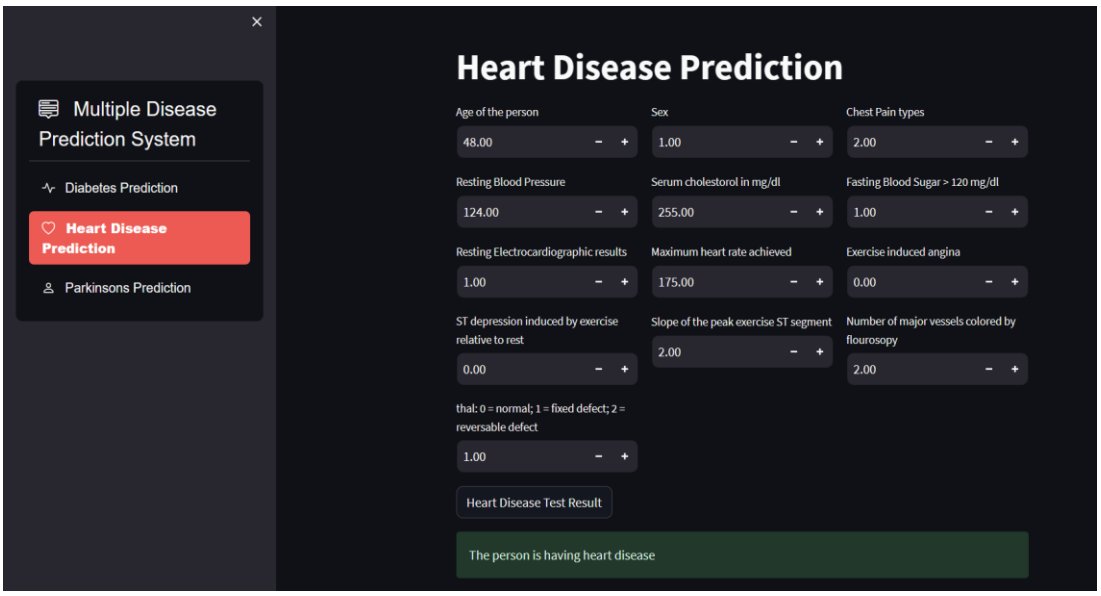
Here are some common terms involved in logistic regression:

- Independent variables: The input characteristics or predictor factors applied to the dependent variable's predictions.
- Dependent variable: The target variable in a logistic regression model, which we are trying to predict.
- Logistic function: The formula used to represent how the independent and dependent variables relate to one another. The logistic function transforms the input variables into a probability value between 0 and 1, which represents the likelihood of the dependent variable being 1 or 0.
- Odds: It is the ratio of something occurring to something not occurring. it is different from probability as the probability is the ratio of something occurring to everything that could possibly occur.
- Log-odds: The log-odds, also known as the logit function, is the natural logarithm of the odds. In logistic regression, the log odds of the dependent variable are modeled as a linear combination of the independent variables and the intercept.
- Coefficient: The logistic regression model's estimated parameters, show how the independent and dependent variables relate to one another.
- Intercept: A constant term in the logistic regression model, which represents the log odds when all independent variables are equal to zero.
- Maximum likelihood estimation: The method used to estimate the coefficients of the logistic regression model, which maximizes the likelihood of observing the data given the model.

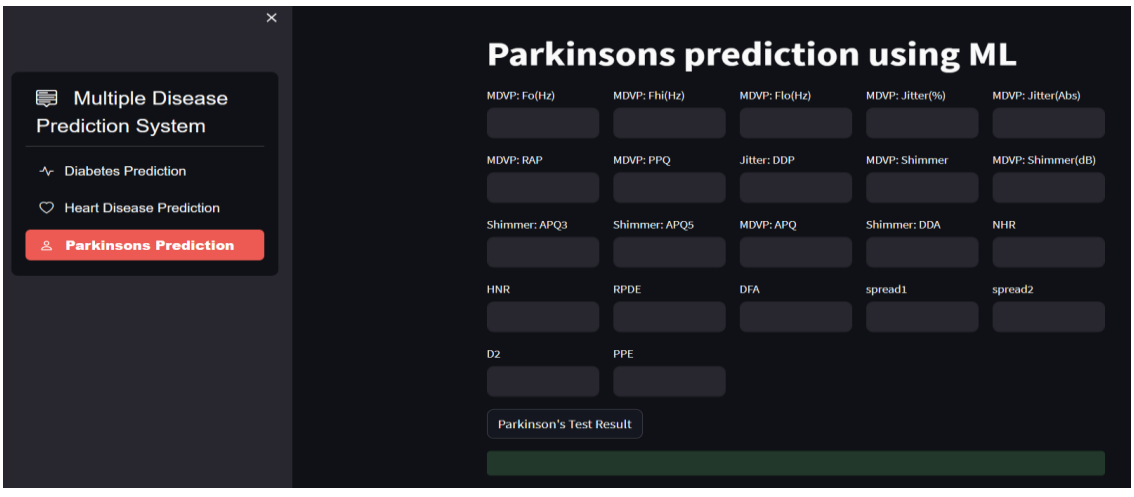
5.2.3 Screen-Shot Diagram



Diabetes Prediction



Heart Disease Prediction



9. CODING

9.1 DIABETES MODEL

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
# loading the diabetes dataset to a pandas DataFrame
diabetes_dataset = pd.read_csv('/content/diabetes_prediction_dataset.csv')
# number of rows and Columns in this dataset
diabetes_dataset.shape
for i in range(0,len(diabetes_dataset)):
        if(diabetes_dataset["gender"][i]=="Female"):
          diabetes_dataset["gender"][i]=0
        else:
          diabetes_dataset["gender"][i]=1
df = pd.DataFrame(diabetes_dataset)
# saving the DataFrame as a CSV file
gfg_csv_data = df.to_csv('GfG.csv', index = True)
diabetes_dataset.head()
diabetes_dataset.tail()
# getting the statistical measures of the data
diabetes_dataset.describe()
diabetes_dataset['diabetes'].value_counts()
diabetes_dataset.groupby('diabetes').mean()
# separating the data and labels
X = diabetes_dataset.drop(columns = 'diabetes', axis=1)
Y = diabetes_dataset['diabetes']
print(X)
print(Y)
```

```
scaler = StandardScaler()

scaler.fit(X)

standardized_data = scaler.transform(X)

print(standardized_data)

X = standardized_data

Y = diabetes_dataset['diabetes']

X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, stratify=Y, random_state=2)

print(X.shape, X_train.shape, X_test.shape)

classifier = svm.SVC(kernel='linear')

#training the support vector Machine Classifier

classifier.fit(X_train, Y_train)

# accuracy score on the training data

X_train_prediction = classifier.predict(X_train)

training_data_accuracy = accuracy_score(X_train_prediction, Y_train)

print('Accuracy score of the training data : ', training_data_accuracy)

# accuracy score on the test data

X_test_prediction = classifier.predict(X_test)

test_data_accuracy = accuracy_score(X_test_prediction, Y_test)

print('Accuracy score of the test data : ', test_data_accuracy)

input_data = (0,42,0,0,24.81,9,159)


# changing the input_data to numpy array

input_data_as_numpy_array = np.asarray(input_data)


# reshape the array as we are predicting for one instance

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)


# standardize the input data

std_data = scaler.transform(input_data_reshaped)
```

```
print(std_data)

prediction = classifier.predict(std_data)
print(prediction)

if (prediction[0] == 0):
  print('The person is not diabetic')
else:
  print('The person is diabetic')
import pickle
filename='diabetes_model.sav'
pickle.dump(classifier,open(filename,'wb'))
# loading the saved model
loaded_model = pickle.load(open('diabetes_model.sav', 'rb'))
input_data = (0,42,0,0,24.81,9,159)

# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
std_data = scaler.transform(input_data_reshaped)

prediction = loaded_model.predict(std_data)
print(prediction)

if (prediction[0] == 0):
  print('The person is not diabetic')
else:
```

```
print('The person is diabetic')
```

9.2 HEART DISEASE MODEL

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
# loading the csv data to a Pandas DataFrame
heart_data = pd.read_csv('/content/heart.csv')
# print first 5 rows of the dataset
heart_data.head()
# print last 5 rows of the dataset
heart_data.tail()
# number of rows and columns in the dataset
heart_data.shape
# getting some info about the data
heart_data.info()
# checking for missing values
heart_data.isnull().sum()
# statistical measures about the data
heart_data.describe()
# checking the distribution of Target Variable
heart_data['target'].value_counts()
X = heart_data.drop(columns='target', axis=1)
Y = heart_data['target']
Print(X)
Print(Y)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_state=42)
```

```
print(X.shape, X_train.shape, X_test.shape)

print(X_train)

model = LogisticRegression(random_state=42)

# training the LogisticRegression model with Training data

model.fit(X_train, Y_train)

# accuracy on training data

X_train_prediction = model.predict(X_train)

training_data_accuracy = accuracy_score(X_train_prediction, Y_train)

print('Accuracy on Training data : ', training_data_accuracy)

# accuracy on test data

X_test_prediction = model.predict(X_test)

test_data_accuracy = accuracy_score(X_test_prediction, Y_test)

print('Accuracy on Test data : ', test_data_accuracy) input_data =
(66,1,0,112,212,0,0,132,1,0.1,2,1,2)

# change the input data to a numpy array

input_data_as_numpy_array= np.asarray(input_data)


# reshape the numpy array as we are predicting for only on instance

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)

print(prediction)


if (prediction[0]== 0):

  print('The Person does not have Heart Disease')

else:

  print('The Person has Heart Disease')

import pickle

filename='heart_disease_model.sav'

pickle.dump(model,open(filename,'wb'))
```

```
# loading the saved model
loaded_model = pickle.load(open('heart_disease_model.sav', 'rb'))
input_data = (66,1,0,112,212,0,0,132,1,0.1,2,1,2)

# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
prediction = loaded_model.predict(input_data_reshaped)
print(prediction)

if (prediction[0] == 0):
  print('The person does not have heart disease')
else:
  print('The person has heart disease')
```

9.3 PARKINSONS DISEASE MODEL

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from sklearn.metrics import accuracy_score
# loading the data from csv file to a Pandas DataFrame
parkinsons_data = pd.read_csv('/content/parkinsons.csv')
# printing the first 5 rows of the dataframe
parkinsons_data.head()
# number of rows and columns in the dataframe
parkinsons_data.shape
```

```
# getting more information about the dataset

parkinsons_data.info()

# checking for missing values in each column

parkinsons_data.isnull().sum()

# getting some statistical measures about the data

parkinsons_data.describe()

# distribution of target Variable

parkinsons_data['status'].value_counts()

# grouping the data bas3ed on the target variable

parkinsons_data.groupby('status').mean()

X = parkinsons_data.drop(columns=['name','status'], axis=1)

Y = parkinsons_data['status']
print(X)
print(Y)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
print(X.shape, X_train.shape, X_test.shape)
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)

X_test = scaler.transform(X_test)
print(X_train)
model = svm.SVC(kernel='linear')
# training the SVM model with training data

model.fit(X_train, Y_train)
# accuracy score on training data

X_train_prediction = model.predict(X_train)

training_data_accuracy = accuracy_score(Y_train, X_train_prediction)
print('Accuracy score of training data : ', training_data_accuracy)
# accuracy score on training data

X_test_prediction = model.predict(X_test)

test_data_accuracy = accuracy_score(Y_test, X_test_prediction)
print('Accuracy score of test data : ', test_data_accuracy)
```

```
input_data =
(197.07600,206.89600,192.05500,0.00289,0.00001,0.00166,0.00168,0.00498,0.01098,0.09700,0.005
63,0.00680,0.00802,0.01689,0.00339,26.77500,0.422229,0.741367,-
7.348300,0.177551,1.743867,0.085569)

# changing input data to a numpy array

input_data_as_numpy_array = np.asarray(input_data)

# reshape the numpy array

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

# standardize the data

std_data = scaler.transform(input_data_reshaped)

prediction = model.predict(std_data)

print(prediction)

if (prediction[0] == 0):

        print("The Person does not have Parkinsons Disease")

else:

        print("The Person has Parkinsons")

import pickle
filename='parkinsons_model.sav'

pickle.dump(model,open(filename,'wb'))
        # loading the saved model
loaded_model = pickle.load(open('parkinsons_model.sav', 'rb'))
input_data =
(197.07600,206.89600,192.05500,0.00289,0.00001,0.00166,0.00168,0.00498,0.01098,0.09700,0.005
63,0.00680,0.00802,0.01689,0.00339,26.77500,0.422229,0.741367,-
7.348300,0.177551,1.743867,0.085569)


# changing the input_data to numpy array

input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

std_data = scaler.transform(input_data_reshaped)
```

```
prediction = loaded_model.predict(std_data)

print(prediction)

if (prediction[0] == 0):

      print('The person is not having Parkinsons disease')

else:

      print('The person is having parkinsons disease')
```

9.4 STREAMLIT INTERFACE

```
import pickle

from sklearn.preprocessing import StandardScaler

import pandas as pd

import numpy as np

import streamlit as st

from streamlit_option_menu import option_menu


scaler1=StandardScaler()

diabetes=pd.read_csv('C:/Users/harsh/OneDrive/Desktop/Multiple disease prediction system/dia.csv',
sep=',',header=0)

X = diabetes.drop(columns = 'diabetes', axis=1)

scaler1.fit(X)

scaler2=StandardScaler()

parkinsons=pd.read_csv('C:/Users/harsh/OneDrive/Desktop/Multiple disease prediction
system/dataset/parkinsons.csv',sep=',',header=0)

Y=parkinsons.drop(columns='status',axis=1)

scaler2.fit(Y)

#loading the saved models

diabetes_model = pickle.load(open('C:/Users/harsh/OneDrive/Desktop/Multiple disease prediction
system/saved models/diabetes_model.sav','rb'))

heart_disease_model = pickle.load(open('C:/Users/harsh/OneDrive/Desktop/Multiple disease
prediction system/saved models/heart_disease_model.sav','rb'))
```

```
parkinsons_model = pickle.load(open('C:/Users/harsh/OneDrive/Desktop/Multiple disease prediction system/saved models/parkinsons_model.sav','rb'))

#sidebar for navigation

with st.sidebar:

    selected  = option_menu('Multiple Disease Prediction System',

                        ['Diabetes Prediction','Heart Disease Prediction','Parkinsons Prediction'],

                        icons = ['activity','heart','person'],

                        default_index = 0)

# Diabetes prediction page

if (selected == 'Diabetes Prediction'):

    #page title

    st.title('Diabetes Prediction ')

    #getting the input data from the user

    #columns for input fields

    col1, col2, col3 = st.columns(3)

    with col1:

        Gender = st.text_input('Gender')

    with col2:

        Age = st.text_input('Age of the person')

    with col3:

        Hypertension = st.text_input('Hypertension')

 with col1:

        Heartdisease = st.text_input('Heartdisease')



    with col2:

        BMI = st.text_input('BMI')

 with col3:

        HbA1c_level = st.text_input('HbA1c_level')
```

```
 with col1:
        bloodglucose = st.text_input('Glucose level')
    #code for prediction
    diab_dignosis = ''
#creating a button for prediction
if st.button('Diabetes Test Result'):
        # changing the input_data to numpy array
        input_data=[Gender,Age,Hypertension,Heartdisease, BMI,HbA1c_level,bloodglucose]
        input_data_as_numpy_array = np.asarray(input_data)
        # reshape the array as we are predicting for one instance
        input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
        std_data = scaler1.transform(input_data_reshaped)
        diab_prediction = diabetes_model.predict(std_data)
        if (diab_prediction[0]==1):
            diab_dignosis = 'The person is Diabetic'
        else:
            diab_dignosis = 'The person is Not Diabetic'
    st.success(diab_dignosis)
 #Heart disease prediction
if (selected == 'Heart Disease Prediction'):
      #page title
     st.title('Heart Disease Prediction')
      #getting the input data from the user
      #columns for input fields
     col1, col2, col3 = st.columns(3)
     with col1:
         age = st.number_input('Age of the person')
 with col2:
         sex = st.number_input('Sex')
```

```
with col3:
    cp = st.number_input('Chest Pain types')
with col1:
    trestbps = st.number_input('Resting Blood Pressure')
with col2:
    chol = st.number_input('Serum cholestorol in mg/dl')
with col3:
    fbs = st.number_input('Fasting Blood Sugar > 120 mg/dl')
with col1:
    restecg = st.number_input('Resting Electrocardiographic results')
with col2:
    thalach = st.number_input('Maximum heart rate achieved')
with col3:
    exang = st.number_input('Exercise induced angina')
with col1:
    oldpeak = st.number_input('ST depression induced by exercise relative to rest')
with col2:
    slope = st.number_input('Slope of the peak exercise ST segment')
with col3:
    ca = st.number_input('Number of major vessels colored by flourosopy')
with col1:
    thal = st.number_input('thal: 0 = normal; 1 = fixed defect; 2 = reversable defect')
#code for prediction
heart_dignosis = ''
#creating a button for prediction
if st.button('Heart Disease Test Result'):
    heart_prediction = heart_disease_model.predict([[age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal  ]])
 (heart_prediction[0]==1):
```

```
            heart_dignosis = 'The person is having heart disease'
        else:
            heart_dignosis = 'The person does not have any heart disease'
    st.success(heart_dignosis)
```

Parkinsons Disease

```
if (selected == 'Parkinsons Prediction'):
 #page title
st.title('Parkinsons prediction using ML')
 col1, col2, col3, col4, col5 = st.columns(5)
with col1:
        fo = st.text_input('MDVP: Fo(Hz)')
  with col2:
        fhi = st.text_input('MDVP: Fhi(Hz)')
 with col3:
        flo = st.text_input('MDVP: Flo(Hz)')
 with col4:
        Jitter_percent = st.text_input('MDVP: Jitter(%)')
 with col5:
        Jitter_Abs = st.text_input('MDVP: Jitter(Abs)')
 with col1:
        RAP = st.text_input('MDVP: RAP')
 with col2:
        PPQ = st.text_input('MDVP: PPQ')
 with col3:
        DDP = st.text_input('Jitter: DDP')
 with col4:
        Shimmer = st.text_input('MDVP: Shimmer')
 with col5:
        Shimmer_dB = st.text_input('MDVP: Shimmer(dB)')
```

```
with col1:
    APQ3 = st.text_input('Shimmer: APQ3')
with col2:
    APQ5 = st.text_input('Shimmer: APQ5')
with col3:
    APQ = st.text_input('MDVP: APQ')
with col4:
    DDA = st.text_input('Shimmer: DDA')
with col5:
    NHR = st.text_input('NHR')
with col1:
    HNR = st.text_input('HNR')
with col2:
    RPDE = st.text_input('RPDE')
with col3:
    DFA = st.text_input('DFA')
with col4:
    spread1 = st.text_input('spread1')
with col5:
    spread2 = st.text_input('spread2')

  with col1:
    D2 = st.text_input('D2')
with col2:
    PPE = st.text_input('PPE')
# code for Prediction
  parkinsons_diagnosis = ''
 # creating a button for Prediction
  if st.button("Parkinson's Test Result"):
```

```
    input_data2=[[fo, fhi, flo, Jitter_percent, Jitter_Abs, RAP,
PPQ,DDP,Shimmer,Shimmer_dB,APQ3,APQ5,APQ,DDA,NHR,HNR,RPDE,DFA,spread1,spread2,
D2,PPE]]

    input_data_as_numpy_array2= np.asarray(input_data2)

    # reshape the array as we are predicting for one instance

    input_data_reshaped2 = input_data_as_numpy_array2.reshape(1,-1)

    std_data = scaler2.transform(input_data_reshaped2)

    if not all([fo, fhi, flo, Jitter_percent, Jitter_Abs, RAP, PPQ, DDP, Shimmer, Shimmer_dB,
APQ3, APQ5, APQ, DDA, NHR, HNR, RPDE, DFA, spread1, spread2, D2, PPE]):

        st.warning("Please fill in all the fields.")

    else:

        parkinsons_prediction = parkinsons_model.predict(std_data)


        if (parkinsons_prediction[0] == 1):

          parkinsons_diagnosis = "The person has Parkinson's disease"

        else:

          parkinsons_diagnosis = "The person does not have Parkinson's disease"


  st.success(parkinsons_diagnosis)
```

10. TESTING

10.1 Heart Disease

- Input data= 66,1,0,112,212,0,0,132,1,0.1,2,1,2
- Expected output= “The person has heart disease”
- Actual output= “The person has heart disease”

```
input_data = (66,1,0,112,212,0,0,132,1,0.1,2,1,2)
# change the input data to a numpy array
input_data_as_numpy_array= np.asarray(input_data)

# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
prediction = model.predict(input_data_reshaped)
print(prediction)

if (prediction[0]== 0):
  print('The Person does not have Heart Disease')
else:
  print('The Person has Heart Disease')
```

```
[0]
The Person does not have Heart Disease
```

10.2 Diabetes Disease

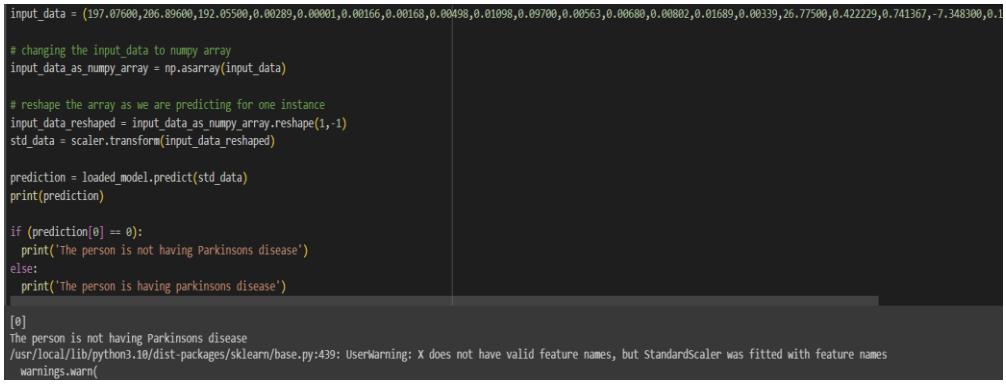
- Input data=0,42,0,0,24.81,9,159
- Expected output= “The person is diabetic”
- Actual output= “The person is diabetic”

```
input_data = (0,42,0,0,24.81,9,159)

# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
std_data = scaler.transform(input_data_reshaped)

prediction = loaded_model.predict(std_data)
print(prediction)

if (prediction[0] == 0):
  print('The person is not diabetic')
else:
  print('The person is diabetic')
```

```
[1]
The person is diabetic
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: U
  warnings.warn(
```

10.3Parkinsons Disease

- Input Data= 197.07600,206.89600,192.05500,0.00289,0.00001,0.00166,0.00168,0.00498,0.01098,0.09700,0.00563,0.00680,0.00802,0.01689,0.00339,26.77500,0.422229,0.741367,-7.348300,0.177551,1.743867,0.085569
- Expected output= “The person does not have parkinsons disease”
- Actual output= “The person does not have parkinsons disease”



11. CONCLUSION AND LIMITATIONS

11.1 CONCLUSION

The above diseases are a major killer in India and throughout the world, application of promising technology like machine learning to the initial prediction of such diseases will have a profound impact on society. The early prognosis of diseases can aid in making decisions on lifestyle changes in high-risk patients and in turn reduce the complications, which can be a great milestone in the field of medicine. The number of people facing diseases is on a raise each year. This prompts for its early diagnosis and treatment. The utilization of suitable technology support in this regard can prove to be highly beneficial to the medical fraternity and patients. This project successfully helps to predict the heart disease with 80%accuracy, kidney disease with 83% accuracy and diabetic disease with 78% accuracy. Further for its enhancement we can train or add on models and predict the types of cardiovascular disease and can even use more enhanced models.

The expected attributes leading to diseases in patients are available in the dataset that are useful to evaluate the system are selected among them. If all the features taken into the consideration then the efficiency of the system the author gets is less. To increase efficiency, attribute selection is done. In this n features have to be selected for evaluating the model which gives more accuracy. The correlation of some features in the dataset is almost equal and so they are removed. If all the attributes present in the dataset are taken into account then the efficiency decreases considerably.

Machine learning has been driving substantial impact on several aspects of health care. This fact can be clearly seen / measured based on the number of new health care start-ups that have machine learning at their core. Four main areas where machine learning has made / is making meaningful contributions are hospital operations, medical imaging, drug discovery and development, and disease prediction and treatment.

Machine learning, like several other things in healthcare that have been viewed with skepticism of late, such as value-based care, could be transformational for healthcare “at large”. Its transformative power is largely rooted in the fact that it has the potential to impact all health care stakeholders, most importantly to patients. Notwithstanding the prevailing skepticism surrounding machine learning, the fact that it has overwhelming support from key stakeholders such as insurance companies, drug manufacturers, researchers, etc. potentially means that it is here to stay.

11.2 LIMITATIONS

While disease prediction systems using machine learning (ML) have shown great promise in improving healthcare, they also come with certain limitations. It's important to be aware of these limitations to ensure the responsible and effective deployment of such systems. Here are some common limitations:

1. **Data Quality and Bias:**
    - Incomplete or Biased Data: ML models heavily depend on the quality and representativeness of the data they are trained on. If the data used for training is incomplete or biased, the model may not generalize well to diverse populations.
    - Selection Bias: If the training data is not representative of the target population, the model may make inaccurate predictions for specific demographic groups.
2. **Generalization Challenges:**
    - Contextual Changes: ML models may struggle to adapt to new environments or contexts that differ significantly from the training data. This can be problematic when dealing with diseases that evolve or manifest differently over time.
    - Transferability Issues: Models trained on data from one healthcare system or geographic region may not perform well when applied to a different system or region.
3. **Interpretability and Explainability:**
    - Black Box Nature: Many ML models, especially complex ones like deep learning models, are often considered "black boxes" because their decision-making processes are not easily interpretable. Lack of interpretability can be a barrier to gaining trust from healthcare professionals and patients.
    - Explainability: It can be challenging to provide clear explanations for why a particular prediction was made, which is crucial for gaining acceptance in the medical community.
4. **Dynamic Nature of Diseases:**
    - Evolution of Diseases: Diseases can evolve, and new strains may emerge. If the model is not continuously updated with relevant data, it may become obsolete and less accurate over time.
    - Changing Patient Profiles: Patient profiles and health conditions can change, and ML models may not adapt well to these changes without regular updates.

5. **Privacy and Ethical Concerns:**
    - Patient Privacy: Access to sensitive health data raises concerns about patient privacy. Implementing robust privacy measures is essential to prevent unauthorized access and misuse of personal health information.
    - Ethical Considerations: Decision-making in healthcare has ethical dimensions, and ML models may inadvertently perpetuate or amplify existing biases present in the data.

6. **Resource Constraints:**
    - Infrastructure and Expertise: Implementing and maintaining a disease prediction system requires significant computational resources and expertise. Many healthcare facilities may not have the necessary infrastructure or personnel.
7. **Regulatory Challenges:**
    - Regulatory Approval: Obtaining regulatory approval for deploying ML-based disease prediction systems can be a lengthy and complex process. Compliance with healthcare regulations is crucial but may pose challenges.

It's essential to address these limitations through careful model development, ongoing monitoring, and collaboration between data scientists, healthcare professionals, and regulatory bodies. Responsible implementation involves considering the broader ethical, social, and legal implications of these systems in healthcare settings.

12. REFERENCE/BIBLIOGRAPHY

- www.geeksforgeeks.org
- www.python.org
- www.flask.palletsprojects.com
- www.djangoproject.com