AI-BASED DIABETES PREDICTION SYSTEM

TEAM MEMBER: yogeshwari s

810021106705

PHASE 5: PROJECT DOCUMENTATION AND SUBMISSION

PROJECT DEFINITION

AI in Healthcare isan industry thatalways makes itnecessary to make a precisedecision,whether it isa

Treatment,test,ordischarge.Diabetes is common dueto modernfoodintake,andit isnecessary to keep trackof the body .AI in

Diabeteshelps to predictor Detect Diabetes. Anyneglect inhealthcanhave a highcost for the patientsandthe medical practitioner. It

Becomes challenging for thepatient totrust that thisdecisionis taken by the machinethatdoesnotexplainhow it reachesaparticular conclusion. Sothe useof theexplainable AI is mandatory inpredictingthe diabetesdisease that will helpgainthe confidence ofan AI system

Result.Explainable AIhelps to get the fairand correctoutput without errors.

PROJECT THINKING

DATA

It explains the data usedfor theprediction, their Correlation,and EDA(Exploratory Data Analysis) to understandthe hidden data patterns.It tellshow thedata is to be usedfor the AI system.

ALGORITHM

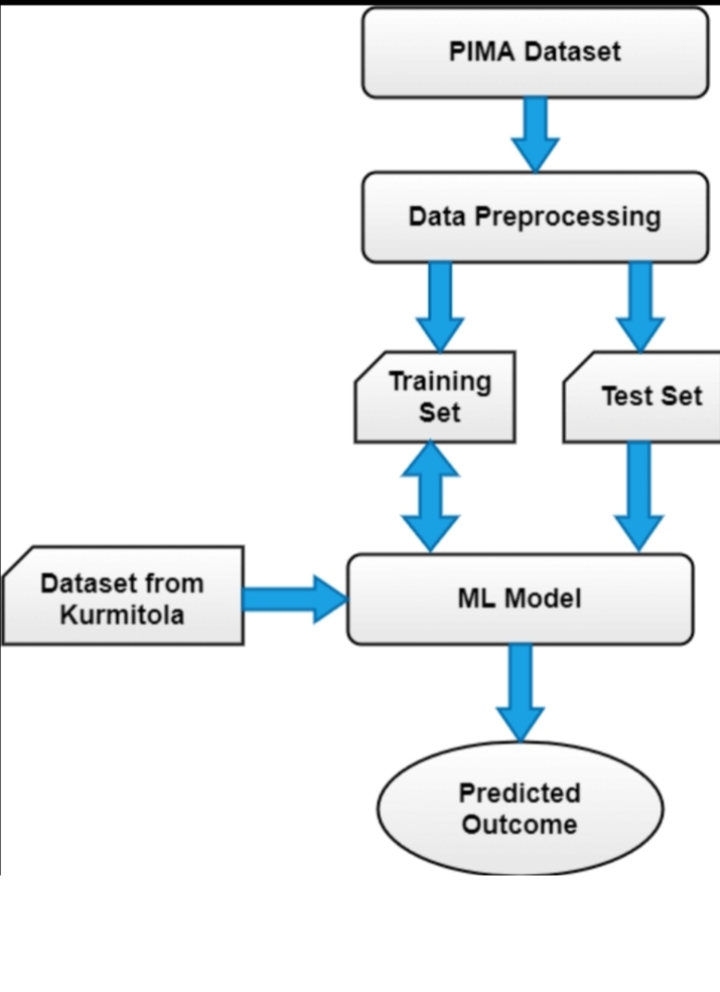
A complete transparencyof the system’salgorithm isgiven with the reason why the system chooses itandhow it canbe

Beneficial for the prediction.

MODEL

Akira AIgivesadetailed explanationof modelperformance and working ina user friendly manner

DATASET

There are several datasets commonly used for diabetes prediction systems. One popular dataset is the Pima Indian Diabetes dataset, which contains various health-related features for individuals, along with information about whether they have diabetes. Another widely used dataset is the Diabetes dataset from the UCI Machine Learning Repository, which includes patient characteristics and diabetes outcome. These datasets are often used for training machine learning models to predict diabetes or assess the risk of diabetes in individuals.

DATA PREPROCESSING

1. \*\*Data Cleaning:\*\*

- Handle missing values: You can impute missing values using techniques like mean, median, or using predictive models.

- Remove duplicates: Check for and remove duplicate records if they exist in your dataset.

2. \*\*Feature Selection:\*\*

- Identify and select relevant features that are most likely to impact the prediction. Feature selection techniques like correlation analysis or feature importance from machine learning models can help.

3. \*\*Feature Scaling:\*\*

- Normalize or standardize numerical features to ensure that they are on a similar scale. Common methods include Min-Max scaling or Z-score normalization.

4. \*\*Categorical Encoding:\*\*

- If your dataset contains categorical variables, you may need to encode them into numerical values. Common techniques include one-hot encoding or label encoding.

5. \*\*Data Splitting:\*\*

- Split your dataset into training and testing sets. This helps evaluate your model’s performance on unseen data.

6. \*\*Handling Class Imbalance:\*\*

- If your dataset has imbalanced classes (e.g., more non-diabetic cases than diabetic cases), consider techniques like oversampling, undersampling, or using synthetic data generation methods to balance the classes.

7. \*\*Outlier Detection and Handling:\*\*

- Identify and address outliers in your data. Outliers can negatively impact the performance of predictive models.

8. \*\*Normalization:\*\*

- Depending on the specific algorithms you use, you may need to normalize your data to ensure that all features have the same influence on the model.

9. \*\*Feature Engineering:\*\*

- Create new features or transform existing ones to provide more meaningful information for the prediction task. For example, you can calculate the body mass index (BMI) from height and weight features.

10. \*\*Data Validation:\*\*

- Ensure that your data is consistent and valid. Check for any anomalies or errors in the dataset.

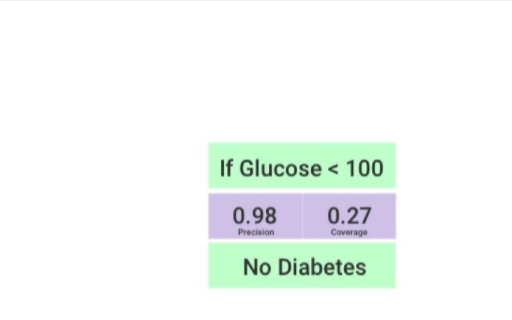
FEATURE INFLUENCES

The glucose value of a person influences the Result more while predicting whether a person

Can have diabetes or not.

WHAT IS THE VALUE OF THE GLUCOSE NEED TO MAINTAIN IN BODY?

The person’s glucose value plays a Significant part in predicting whether a person Can have diabetes. Anchors can answer what Glucose’s value that an individual needs to Maintain to have diabetes. So this figure depicts That if that person maintains the Glucose value Less than 100, it cannot have diabetes.



WHAT IS THE GENERAL RULE THE SYSTEM USED TO GENERATE AN OUTPUT?

This is an essential question. A satisfying answer To that question can clear many doubts about

The customers and bring their confidence in the System. If the system can provide the general

Rule that the system obeys to give the output, it Can clear all the doubts regarding the

Algorithm’s working.

We can use various methodologies to answer The question according to the algorithm we

Used to predict the output as we are using the Random Forest Classifiers Algorithm. Hence, we

Have some methods such as inTRess, defrag trees,etc,that we can use to extract the

Random Forest Algorithm rule.Here we used the defragTrees method that

Generates rules that the system used to predict An output, as shown in Figure below. It also

Provides the performance attributes from Which the user can analyze and track the

Algorithm’s performance and rule to predict an Output. It follows the model’s behavior on a

Global scale, so they can inspect it and find out Whether the model has picked up any

Undesired functioning.

The figure describes two rules on which the System is working. It provides the value of the

Parameters that help to reach the system at the Output. As given in Rule 1, if the value of the

Glucose <= 100 and BMI <= 30 and Age <= 30 And Pregnancies <= 4 and Skin thickness <= 35

And Insulin <= 35, then the system will say that A lady doesn’t have diabetes. Whereas opposite

Of these values as given in Rule 2 if the values

Of the parameters are Glucose > 100 and BMI > 30 and Age > 30 and Pregnancies > 4 and Skin

Thickness > 35 and Insulin> 35, then it will say That the lady can have diabetes.

It also provides the Test error, Test coverage, And the Overlap values to understand the

System’s performance.

Diabetes Prediction using Machine Learning

Diabetes, is a group of metabolic disorders in which there are high blood sugar levels over a prolonged

Period. Symptoms of high blood sugar include frequent urination, increased thirst, and increased hunger.

If left untreated, diabetes can cause many complications. Acute complications can include diabetic

Ketoacidosis, hyperosmolar hyperglycemic state, or death. Serious long-term complications include

Cardiovascular disease, stroke, chronic kidney disease, foot ulcers, and damage to the eyes.

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The

Objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on

Certain diagnostic measurements included in the dataset. Several constraints were placed on the

Selection of these instances from a larger database. In particular, all patients here are females at least 21

Years old of Pima Indian heritage.

Details about the dataset:

The datasets consists of several medical predictor variables and one target variable, Outcome. Predictor

Variables includes the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

Pregnancies: Number of times pregnant

Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test

BloodPressure: Diastolic blood pressure (mm Hg)

SkinThickness: Triceps skin fold thickness (mm)

Insulin: 2-Hour serum insulin (mu U/ml)

BMI: Body mass index (weight in kg/(height in m)^2)

DiabetesPedigreeFunction: Diabetes pedigree function

Age: Age (years)

Outcome: Class variable (0 or 1)

1. Exploratory Data Analysis

#Installation of required libraries

Import numpy as np

Import pandas as pd

Import statsmodels.api as sm

Import seaborn as sns

Import matplotlib.pyplot as plt

From sklearn.preprocessing import scale, StandardScaler

From sklearn.model\_selection import train\_test\_split, GridSearchCV, cross\_val\_score

From sklearn.metrics import confusion\_matrix, accuracy\_score, mean\_squared\_error, r2\_score,

Roc\_auc\_score, roc\_curve, classification\_report

From sklearn.linear\_model import LogisticRegression

From sklearn.neighbors import KNeighborsClassifier

From sklearn.svm import SVC

From sklearn.neural\_network import MLPClassifier

From sklearn.tree import DecisionTreeClassifier

From sklearn.ensemble import RandomForestClassifier

From sklearn.ensemble import GradientBoostingClassifier

From lightgbm import LGBMClassifier

From sklearn.model\_selection import KFold

Import warnings

Warnings.simplefilter(action = “ignore”)

#Reading the dataset

Df = pd.read\_csv(“../input/pima-indians-diabetes-database/diabetes.csv”)

# The first 5 observation units of the data set were accessed.

Df.head()

SOME INNOVATIVE IDEAS FOR DIABETICS PREDICTION SYSTEM

1. \*\*Machine Learning Algorithms:\*\* Utilize advanced machine learning algorithms like deep neural networks to analyze a person’s historical health data, lifestyle factors, and genetics to predict their risk of developing diabetes.
2. \*\*Wearable Health Tech:\*\* Develop a wearable device, such as a smartwatch or a continuous glucose monitoring system, that tracks real-time health data and uses AI to provide early warnings of potential diabetes risks.
3. \*\*Nutrition Recommender:\*\* Create a mobile app that uses AI to analyze users’ dietary habits, recommends personalized nutrition plans, and tracks their progress to prevent or manage diabetes.
4. \*\*Voice Assistant Integration:\*\* Incorporate voice assistants like Amazon Alexa or Google Assistant into diabetes prediction systems, allowing users to ask questions about their health and receive real-time feedback and advice.
5. \*\*Image Analysis:\*\* Develop a smartphone app that allows users to take pictures of their meals, and then uses image recognition and AI to estimate the nutritional content, helping users make healthier food choices.
6. \*\*Social Network Integration:\*\* Build a social platform where users with diabetes or at risk can connect, share experiences, and receive support and advice from others in similar situations.
7. \*\*Predictive Genetic Testing:\*\* Offer genetic testing services that analyze an individual’s DNA for diabetes-related risk factors and provide personalized recommendations for prevention or management.
8. \*\*Telemedicine Integration:\*\* Combine diabetes prediction with telemedicine services, allowing users to consult with healthcare professionals remotely, receive regular check-ups, and monitor their health effectively.
9. \*\*Utilising\*\*Organize community-based health screening events equipped with AI-powered diagnostic tools to identify individuals at risk of diabetes in underserved areas.
10. \*\*Data Privacy and Security:\*\* Ensure robust data privacy and security measures to protect users’ sensitive health information in compliance with healthcare regulations.
11. Data Preprocessing

2.1) Missing Observation Analysis

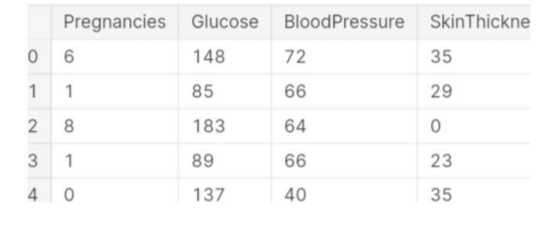
We saw on df.head() that some features contain 0, it doesn’t make sense here and this indicates missing

Value Below we replace 0 value by NaN:

Df[[‘Glucose’,’BloodPressure’,’SkinThickness’,’Insulin’,’BMI’]] =

Df[[‘Glucose’,’BloodPressure’,’SkinThickness’,’Insulin’,’BMI’]].replace(0,np.NaN)

Df.head()



# Now, we can look at where are missing values

Df.isnull().sum()

Pregnancies 0

Glucose 5

BloodPressure 35

SkinThickness 227

Insulin 374

BMI 11

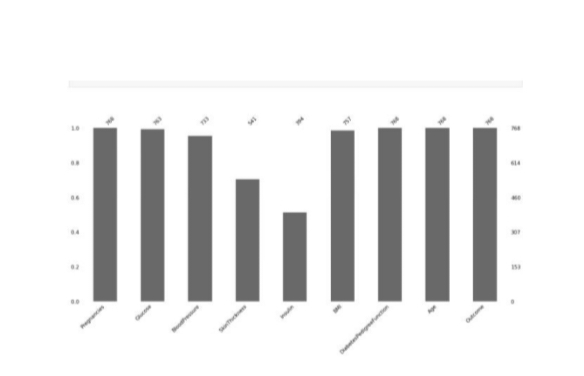
DiabetesPedigreeFunction 0

Age 0

Outcome 0

Dtype: int64

# Have been visualized using the missingno library for the visualization of missing observations.



# The missing values will be filled with the median values of each variable.

Def median\_target(var):

Temp = df[df[var].notnull()]

Temp = temp[[var, ‘Outcome’]].groupby([‘Outcome’])[[var]].median().reset\_index()

Return temp

# The values to be given for incomplete observations are given the median value of people who are not

Sick and the median values of people who are sick.

Columns = df.columns

Columns = columns.drop(“Outcome”)

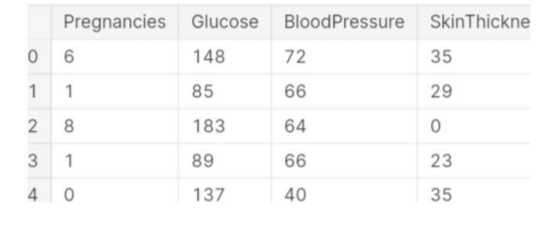
For I in columns:

Median\_target(i)

Df.loc[(df[‘Outcome’] == 0 ) & (df[i].isnull()), i] = median\_target(i)[i][0]

Df.loc[(df[‘Outcome’] == 1 ) & (df[i].isnull()), i] = median\_target(i)[i][1]

Def.head()



# Missing values were filled.

Df.isnull().sum()

Pregnancies 0

Glucose 0

BloodPressure 0

SkinThickness 0

Insulin 0

BMI 0

DiabetesPedigreeFunction 0

Age 0

Outcome 0

Dtype: int64

2.2) Outlier Observation Analysis

# In the data set, there were asked whether there were any outlier observations compared to the 25%

And 75% quarters.

# It was found to be an outlier observation.

For feature in df:

Q1 = df[feature].quantile(0.25)

Q3 = df[feature].quantile(0.75)

IQR = Q3-Q1

Lower = Q1- 1.5\*IQR

Upper = Q3 + 1.5\*IQR

If df[(df[feature] > upper)].any(axis=None):

Print(feature,”yes”)

Else:

Print(feature, “no”)

Pregnancies yes

Glucose no

BloodPressure yes

SkinThickness yes

Insulin yes

BMI yes

DiabetesPedigreeFunction yes

Age yes

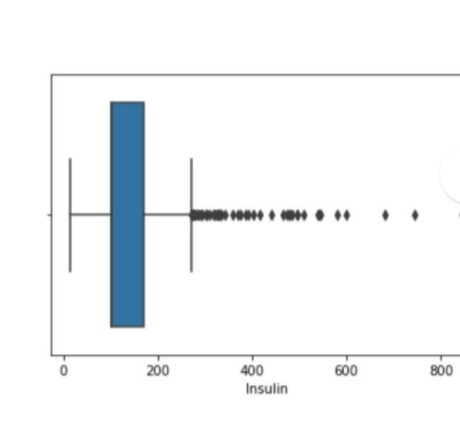
Outcome no

# The process of visualizing the Insulin variable with boxplot method was done. We find the outlier

Observations on the chart.

Import seaborn as sns

Sns.boxplot(x = df[“Insulin”]);



#We conduct a stand alone observation review for the Insulin variable

#We suppress contradictory values

Q1 = df.Insulin.quantile(0.25)

Q3 = df.Insulin.quantile(0.75)

IQR = Q3-Q1

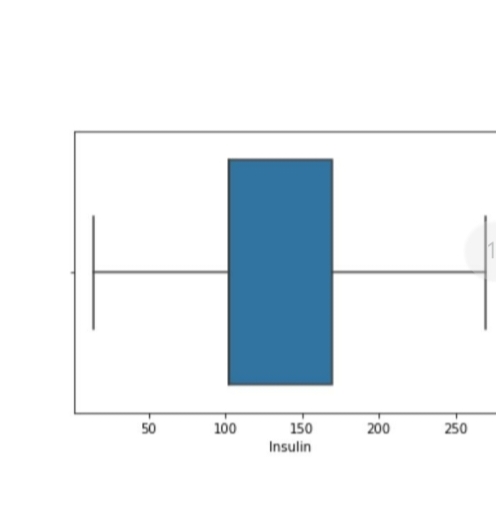
Lower = Q1 – 1.5\*IQR

Upper = Q3 + 1.5\*IQR

Df.loc[df[“Insulin”] > upper,”Insulin”] = upper

Import seaborn as sns

Sns.boxplot(x = df[“Insulin”]);



Df\_scores = lof.negative\_outlier\_factor\_

Np.sort(df\_scores)[0:30]

Array([-3.05893469, -2.37289269, -2.15297995, -2.09708735, -2.0772561 ,

-1.95255968, -1.86384019, -1.74003158, -1.72703492, -1.71674689,

-1.70343883, -1.6688722 , -1.64296768, -1.64190437, -1.61620872,

-1.61369917, -1.60057603, -1.5988774 , -1.59608032, -1.57027568,

-1.55876022, -1.55674614, -1.51852389, -1.50843907, -1.50280943,

-1.50160698, -1.48391514, -1.4752983 , -1.4713427 , -1.47006248])

#We choose the threshold value according to lof scores

Threshold = np.sort(df\_scores)[7]

Threshold

-1.740031580305444

#We delete those that are higher than the threshold

Outlier = df\_scores > threshold

Df = df[outlier]

# The size of the data set was examined.

Df.shape

(760, 9)

MODEL TRAINING

1. \*\*Data Collection\*\*: Gather a comprehensive dataset that includes relevant features such as age, weight, family history, physical activity, and, most importantly, blood glucose levels. Ensure the data is diverse and representative.
2. \*\*Data Preprocessing\*\*: Clean the data by handling missing values, outliers, and normalizing or standardizing features. This is essential for accurate predictions.
3. \*\*Feature Selection/Engineering\*\*: Identify the most relevant features and consider creating new features if necessary. Feature engineering can significantly improve model performance.
4. \*\*Model Selection\*\*: Choose an appropriate machine learning or deep learning model for prediction. Algorithms like logistic regression, decision trees, random forests, or neural networks are commonly used.
5. \*\*Training the Model\*\*: Split the dataset into training and testing sets to train and evaluate the model. Use techniques like cross-validation to optimize hyperparameters.
6. \*\*Evaluation\*\*: Assess the model’s performance using metrics like accuracy, precision, recall, F1-score, and ROC-AUC. Make sure to evaluate its ability to predict diabetes accurately.
7. \*\*Deployment\*\*: Integrate the trained model into an application or system, which could be a mobile app or a web-based tool. Ensure it’s user-friendly and accessible to the target audience.
8. \*\*Continuous Monitoring\*\*: Regularly update the model with new data and fine-tune it to maintain accuracy over time.
9. \*\*Compliance and Privacy\*\*: Ensure that the system complies with privacy regulations and ethical considerations, especially when handling healthcare data.
10. \*\*User Education\*\*: Educate users about the system’s limitations, and the importance of consulting a healthcare professional for a definitive diagnosis.

MODEL EVALUATION STEPS

Data Splitting:

Split your dataset into three subsets: training, validation, and test sets. The training set is used to train the model, the validation set is used for hyperparameter tuning, and the test set is reserved for final evaluation.

Performance Metrics:

Choose appropriate evaluation metrics based on the nature of your diabetes prediction task. Common metrics include:

Accuracy: The proportion of correctly predicted cases.

Precision: The ratio of true positive predictions to all positive predictions.

Recall: The ratio of true positive predictions to all actual positives.

F1-score: The harmonic mean of precision and recall.

AUC-ROC: The area under the receiver operating characteristic curve, useful for binary classification.

Model Evaluation:

Assess the performance of your diabetes prediction model using the selected evaluation metrics on the validation set. This step helps you fine-tune the model’s hyperparameters.

Cross-Validation (Optional):

Perform k-fold cross-validation to assess the model’s generalization ability and reduce the risk of overfitting.

Final Model Selection:

Based on the performance on the validation set and cross-validation results (if used), select the best-performing model.

Model Testing:

Evaluate the final model on the test dataset. This dataset should be representative of real-world data the model will encounter. The test results give you an estimate of the model’s performance in practice.

Interpretability (Optional):

If your model is complex, consider using techniques to make its predictions more interpretable. Explainable AI methods can help healthcare professionals and patients understand why a particular prediction was made.

Bias and Fairness Assessment:

Assess the model for bias to ensure that it makes predictions fairly across different demographic groups. Mitigate any identified biases.

Ethical and Regulatory Compliance:

Ensure that your diabetes prediction system complies with ethical guidelines and healthcare regulations, especially regarding patient data privacy and informed consent.

Feedback Loop:

Establish a feedback mechanism to continuously collect data and user feedback from the real-world use of the system. Use this information to make necessary updates and improvements.

Monitoring and Maintenance:

Continuously monitor the model’s performance in a production environment and update it as needed to maintain accuracy and relevance.

Documentation:

Thoroughly document the evaluation process, results, and any corrective actions taken. This documentation is essential for compliance and transparency.

WELL STRUCTURED README FILE

Table of Contents

Installation

Usage

Data

Model

Evaluation

Contributing

License

Installation

Explain how to install the system. Include any dependencies, installation steps, and requirements. You can use code blocks to illustrate installation commands:

Bash

Copy code

# Example installation command

Pip install diabetes-prediction

Usage

Provide instructions on how to use the diabetes prediction system. Include code snippets or examples to demonstrate how to make predictions with the system:

Python

Copy code

# Example code for making predictions

From diabetes\_prediction import DiabetesPredictor

Predictor = DiabetesPredictor()

Result = predictor.predict(data)

Print(result)

Data

Explain where the data for training and testing the model can be obtained. If applicable, provide links or instructions for downloading or accessing the dataset. Include any data preprocessing steps if necessary.

Model

Describe the machine learning model used for diabetes prediction. Include information about the model architecture, hyperparameters, and any special considerations:

Model architecture: (e.g., logistic regression, neural network)

Hyperparameters: (e.g., learning rate, number of hidden layers)

Training process: (e.g., data splitting, cross-validation)

Evaluation

Detail how the system’s performance can be assessed. Include information on the evaluation metrics used and how to interpret the results. If applicable, discuss fairness and bias considerations.

Contributing

Explain how others can contribute to the project. Include guidelines for submitting bug reports, feature requests, or code contributions. Provide contact information for the project maintainers.

License

Specify the project’s license and any usage restrictions. Include a link to the full license text.

Acknowledgments

If you used third-party libraries, tools, or datasets, acknowledge them and provide links or references.

Contact

Provide contact information for project maintainers or a link to the project’s issue tracker or discussion forum.

Changelog

Include a version history and a summary of changes in each version of the system.

FAQ

Add frequently asked questions and answers if applicable.

Support

Include information on where users can get support or assistance if they encounter issues or have questions.

This README template can be customized to fit the specific details of your diabetes prediction system. It’s important to keep the documentation clear, concise, and up to date to help users understand and use your system effectively.