Semester project

BS in Artificial Intelligence



**Department of Software Engineering**

**Faculty of Computer Science & Information Technology**

**The Superior University, Lahore**

**FALL 2024**

**Project title:Diabates prediction model**

# Project Details

1. Course: artificial intelligence
2. Instructor: (Sir rasikh Ali)
3. Semester: (third)
4. Section: (3B) 5. Submission Date: (5/12/2024)

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Roll No** | **Email** | **Contact** |
| Sara hanif | 082 | su92-bsaim-f23-082@superior.edu.pk | 03425029619 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Abstract

The **Diabetes Prediction System** is an intelligent healthcare application that utilizes machine learning to predict the likelihood of diabetes in individuals based on key health metrics. This project integrates data preprocessing, model training, and interactive deployment to provide a user-friendly, efficient, and accurate prediction tool.

The application leverages a **Random Forest Classifier**, a robust ensemble learning algorithm, to classify individuals as diabetic or non-diabetic based on attributes such as glucose level, blood pressure, BMI, and more. The dataset is preprocessed by filling missing values with mean imputation and scaling features using **StandardScaler**, ensuring data consistency and model reliability.

An interactive user interface, built using **Streamlit**, allows users to input their health parameters and receive real-time predictions. The system not only achieves high prediction accuracy but also demonstrates the potential of artificial intelligence in aiding healthcare diagnostics.

This project showcases the seamless integration of machine learning with a web-based platform, making it a valuable tool for preliminary diabetes risk assessment. It highlights the significance of AI in healthcare, providing an accessible and innovative solution for individuals and healthcare professionals alike.

# Table of Contents

1. Introduction
2. Objectives
3. System Requirements
4. Methodology
5. Implementation
6. Challenges and Solutions
7. Conclusion

### **Introduction**

In today's world, where healthcare is becoming increasingly data-driven, predictive models are transforming the way medical conditions are diagnosed and managed. The **Diabetes Prediction System** is an innovative application that harnesses the power of **machine learning** to predict the likelihood of diabetes in individuals based on their health metrics.

This project is designed to assist both healthcare professionals and individuals by offering a quick, reliable, and user-friendly tool for diabetes risk assessment. It focuses on analyzing essential health attributes such as glucose levels, blood pressure, BMI, and more to determine whether a person is at risk of developing diabetes.

#### **Key Features:**

1. **Advanced Machine Learning:**  
   The system uses a **Random Forest Classifier**, a highly accurate and reliable ensemble learning algorithm known for handling classification tasks effectively.
2. **Data Preprocessing for Precision:**  
   The application preprocesses the dataset by handling missing values and scaling features to ensure uniformity and accuracy.
3. **Interactive and Intuitive User Interface:**  
   Built with **Streamlit**, the web-based interface allows users to input their health data conveniently and receive real-time predictions in an accessible and visually appealing manner.
4. **Seamless Integration:**  
   Combines backend machine learning algorithms with a frontend interface for an end-to-end diabetes prediction solution.

**OBJECTIVE:**

The primary objectives of the **Diabetes Prediction System** are as follows:

1. **Accurate Prediction of Diabetes Risk:**  
   To develop a machine learning-based model that accurately predicts whether an individual is at risk of diabetes using health metrics such as glucose levels, BMI, blood pressure, and more.
2. **Data-Driven Insights:**  
   To preprocess and analyze health data effectively, ensuring data quality and consistency for improved model performance.
3. **User-Friendly Interface:**  
   To design an intuitive and interactive web application using **Streamlit** that allows users to input their health parameters easily and receive real-time predictions.
4. **Accessibility and Convenience:**  
   To provide a quick, reliable, and accessible tool that assists both healthcare professionals and individuals in assessing diabetes risks without the need for complex testing.
5. **Demonstrate AI in Healthcare:**  
   To showcase how artificial intelligence and machine learning can be applied in healthcare for predictive analysis and early diagnosis.
6. **Encourage Preventive Healthcare:**  
   To promote awareness and encourage users to take preventive measures by identifying potential health risks early.

.

### **System Requirements**

To run the **Diabetes Prediction System**, the following system requirements must be met:

#### **1. Hardware Requirements:**

* **Processor:** Dual-core processor (minimum), Quad-core or higher (recommended).
* **RAM:** At least 4 GB (8 GB or more recommended for better performance).
* **Storage:** At least 500 MB of free disk space for the application files, dataset, and dependencies.
* **Graphics:** Basic graphics capabilities to support the Streamlit interface.

#### **2. Software Requirements:**

* **Operating System:** Windows 10/11, macOS, or Linux.
* **Python Version:** Python 3.8 or higher.

#### **3. Python Libraries:**

The following Python libraries are required:

* **pandas:** For data manipulation and handling.
* **scikit-learn:** For machine learning model development and evaluation.
* **Streamlit:** For building the web-based user interface.
* **numpy:** For numerical computations (automatically installed with pandas or scikit-learn).

### **4. Methodology**

The **Diabetes Prediction System** follows these key steps:

1. **Data Preparation:**
   * Load the diabetes dataset (CSV file).
   * Fill missing values using column means for data completeness.
   * Split data into features (X) and target (y).
2. **Data Preprocessing:**
   * Scale features using **StandardScaler** for uniformity.
   * Split data into training (80%) and testing (20%) sets.
3. **Model Development:**
   * Train a **Random Forest Classifier** on the training data.
   * Evaluate model performance using accuracy on the test set.
4. **Streamlit Interface:**
   * Build an interactive web app for user inputs (e.g., glucose, BMI, age).
   * Preprocess user inputs and predict outcomes with the trained model.
   * Display predictions as "Diabetes Positive" or "Diabetes Negative."
5. **Validation & Error Handling:**
   * Validate the model with test data.
   * Handle errors gracefully, ensuring a smooth user experience.

### **5. Implementation**

The **Diabetes Prediction System** is implemented as follows:

1. **Data Loading and Preparation:**
   * The dataset is loaded using **pandas**.
   * Missing values are filled with column-wise mean.
   * Features (X) are separated from the target variable (y).
2. **Data Preprocessing:**
   * Features are scaled using **StandardScaler**.
   * The data is split into training (80%) and testing (20%) sets.
3. **Model Training:**
   * A **Random Forest Classifier** is trained on the training data.
   * The model is evaluated using accuracy on the test set.
4. **Streamlit Interface:**
   * An interactive web interface is created to collect user inputs (e.g., glucose, BMI, age).
   * The trained model predicts diabetes likelihood based on the user inputs.
5. **Error Handling:**
   * Error handling is implemented to manage invalid inputs and ensure smooth user interaction.

This combination of data handling, machine learning, and a user interface makes the system an effective tool for diabetes prediction.

### **6. Challenges and Solutions**

#### **1. Handling Missing Data**

* **Challenge:** The dataset may contain missing or incomplete values, which can affect the model's performance.
* **Solution:** Missing values are handled by replacing them with the mean of the respective columns using df.fillna(df.mean(), inplace=True), ensuring data consistency without losing valuable information.

#### **2. Feature Scaling**

* **Challenge:** Features with varying scales (e.g., glucose level vs. BMI) could lead to biased model performance.
* **Solution:** All features are standardized using **StandardScaler** to bring them to the same scale, ensuring the model treats all features equally.

#### **3. Model Overfitting**

* **Challenge:** A model may overfit the training data and perform poorly on unseen test data.
* **Solution:** The **Random Forest Classifier** is used, which is less prone to overfitting due to its ensemble nature. Additionally, hyperparameters like the number of estimators are set to default values that balance bias and variance.

#### **4. Handling User Inputs**

* **Challenge:** Users may input invalid or extreme values that could cause errors or unpredictable behavior.
* **Solution:** **Streamlit**'s input validation, such as setting appropriate ranges for each input field (e.g., glucose level from 0 to 200), ensures that the inputs remain within valid bounds. Error handling is also implemented to manage any exceptions.

#### **5. Model Interpretation**

* **Challenge:** Understanding the model’s decision-making process can be difficult, especially with black-box models like Random Forest.
* **Solution:** While the system doesn't provide a detailed explanation for each prediction, future improvements can include adding model interpretability features (e.g., using **SHAP** or **LIME**) to provide insights into the model's decision process.

#### **6. User Experience**

* **Challenge:** Ensuring the interface is intuitive and responsive for users, particularly those without technical knowledge.
* **Solution:** The **Streamlit** interface is designed to be simple and easy to navigate, with clear input labels and a straightforward prediction output, enhancing the overall user experience.

### **7. Conclusion**

The **Diabetes Prediction System** successfully combines machine learning and interactive web technologies to provide an efficient tool for predicting the likelihood of diabetes based on key health metrics. By leveraging the **Random Forest Classifier**, the system achieves reliable predictions with good accuracy.

Key challenges, such as handling missing data, feature scaling, overfitting, and ensuring a smooth user experience, were addressed through effective solutions, including data imputation, standardization, model selection, and input validation. The use of **Streamlit** enhances the user experience by providing a simple and accessible interface for input and output.

This project demonstrates the power of combining machine learning with real-time applications and paves the way for more advanced health prediction systems. Further enhancements, like model interpretability and additional health features, could improve the system's capabilities and user engagement.

Code: import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score

import streamlit as st

df = pd.read\_csv(r'C:\Users\Sara Hanif\Desktop\project AI\diabetes.csv')

df.fillna(df.mean(), inplace=True)

X = df.drop(columns=['Outcome'])

y = df['Outcome']

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

rf\_model = RandomForestClassifier(n\_estimators=100, random\_state=42)

rf\_model.fit(X\_train, y\_train)

y\_pred\_rf = rf\_model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred\_rf)

st.write(f"Random Forest Accuracy: {accuracy \* 100:.2f}%")

st.title("Diabetes Prediction")

pregnancies = st.number\_input("Pregnancies", min\_value=0, max\_value=20, value=0)

glucose = st.number\_input("Glucose Level", min\_value=0, max\_value=200, value=90)

blood\_pressure = st.number\_input("Blood Pressure", min\_value=0, max\_value=200, value=70)

skin\_thickness = st.number\_input("Skin Thickness", min\_value=0, max\_value=100, value=20)

insulin = st.number\_input("Insulin", min\_value=0, max\_value=900, value=80)

bmi = st.number\_input("BMI", min\_value=0.0, max\_value=100.0, value=25.0)

diabetes\_pedigree = st.number\_input("Diabetes Pedigree Function", min\_value=0.0, max\_value=2.5, value=0.5)

age = st.number\_input("Age", min\_value=0, max\_value=120, value=25)

if st.button("Predict"):

    try:

        inputs = [pregnancies, glucose, blood\_pressure, skin\_thickness, insulin, bmi, diabetes\_pedigree, age]

        inputs\_scaled = scaler.transform([inputs])

        prediction = rf\_model.predict(inputs\_scaled)

        result = "Diabetes Positive" if prediction == 1 else "Diabetes Negative"

        st.success(f"Prediction: {result}")

    except Exception as e:

        st.error(f"Error: {str(e)}")