KCB AI & Data Science

# Predicting Diabetes Risk Using Patient Health Indicators and AI Models

Submitted By

Name: GILBERT WALIUBA

Email: gilbert@haofinder.com

Phone: +254 715 560 734

Institution

ADNIAN ABS

Date of Submission

15th SEPT 2025

Dataset Source

Pima Indians Diabetes Database (Kaggle)

# 1. Abstract

This project applies data science and machine learning techniques to predict diabetes using the Pima Indians Diabetes Database. The workflow includes data preprocessing, exploratory data analysis, feature engineering, model training, evaluation, and optimization.

# 2. Problem Statement

Diabetes is a chronic disease with serious health implications. Early detection is crucial. This project builds predictive models to determine the likelihood of diabetes in patients.

# 3. Dataset Description

Pima Indians Diabetes Database with 768 records and 9 attributes.

# 4. Data Collection and Exploration

The dataset was loaded and inspected for structure, data types, and summary statistics.

Code:

import pandas as pd  
  
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"  
columns = ["Pregnancies","Glucose","BloodPressure","SkinThickness","Insulin","BMI","DiabetesPedigreeFunction","Age","Outcome"]  
df = pd.read\_csv(url, names=columns)  
  
print(df.shape)  
print(df.head())

Output:

(768, 9)  
 Pregnancies Glucose BloodPressure SkinThickness Insulin BMI DiabetesPedigreeFunction Age Outcome  
0 6 148 72 35 0 33.6 0.627 50 1  
1 1 85 66 29 0 26.6 0.351 31 0  
2 8 183 64 0 0 23.3 0.672 32 1  
3 1 89 66 23 94 28.1 0.167 21 0  
4 0 137 40 35 168 43.1 2.288 33 1

# 5. Data Preprocessing

Zero values in critical columns were replaced with NaN, imputed with median values, and standardized.

Code:

from sklearn.impute import SimpleImputer  
from sklearn.preprocessing import StandardScaler  
  
cols\_to\_fix = ["Glucose","BloodPressure","SkinThickness","Insulin","BMI"]  
df[cols\_to\_fix] = df[cols\_to\_fix].replace(0, pd.NA)  
  
imputer = SimpleImputer(strategy="median")  
df[cols\_to\_fix] = imputer.fit\_transform(df[cols\_to\_fix])  
  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(df.drop("Outcome", axis=1))

Output: Missing values handled, dataset normalized.

# 6. Exploratory Data Analysis (EDA)

Histograms, heatmaps, and boxplots were generated.

Code:

import seaborn as sns  
import matplotlib.pyplot as plt  
  
sns.countplot(x=df['Outcome'])  
plt.title("Diabetes Outcome Distribution")  
plt.show()

Output: Class imbalance detected (~65% non-diabetic, ~35% diabetic).

# 9. Model Training and Evaluation

Logistic Regression model example:

from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LogisticRegression  
from sklearn.metrics import classification\_report  
  
X = df.drop("Outcome", axis=1)  
y = df["Outcome"]  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)  
  
model = LogisticRegression(max\_iter=1000)  
model.fit(X\_train, y\_train)  
y\_pred = model.predict(X\_test)  
print(classification\_report(y\_test, y\_pred))

Output:

precision recall f1-score support  
 0 0.82 0.86 0.84 99  
 1 0.70 0.63 0.66 55  
 accuracy 0.78 154  
 macro avg 0.76 0.75 0.75 154  
weighted avg 0.78 0.78 0.78 154

Loading dataset...

Loaded: (768, 9)

Pregnancies Glucose BloodPressure SkinThickness Insulin BMI \

0 6 148 72 35 0 33.6

1 1 85 66 29 0 26.6

2 8 183 64 0 0 23.3

3 1 89 66 23 94 28.1

4 0 137 40 35 168 43.1

DiabetesPedigreeFunction Age Outcome

0 0.627 50 1

1 0.351 31 0

2 0.672 32 1

3 0.167 21 0

4 2.288 33 1

Data info:

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 768 entries, 0 to 767

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Pregnancies 768 non-null int64

1 Glucose 768 non-null int64

2 BloodPressure 768 non-null int64

3 SkinThickness 768 non-null int64

4 Insulin 768 non-null int64

5 BMI 768 non-null float64

6 DiabetesPedigreeFunction 768 non-null float64

7 Age 768 non-null int64

8 Outcome 768 non-null int64

dtypes: float64(2), int64(7)

memory usage: 54.1 KB

None

Statistics:

count mean std min 25% \

Pregnancies 768.0 3.845052 3.369578 0.000 1.00000

Glucose 768.0 120.894531 31.972618 0.000 99.00000

BloodPressure 768.0 69.105469 19.355807 0.000 62.00000

SkinThickness 768.0 20.536458 15.952218 0.000 0.00000

Insulin 768.0 79.799479 115.244002 0.000 0.00000

BMI 768.0 31.992578 7.884160 0.000 27.30000

DiabetesPedigreeFunction 768.0 0.471876 0.331329 0.078 0.24375

Age 768.0 33.240885 11.760232 21.000 24.00000

Outcome 768.0 0.348958 0.476951 0.000 0.00000

50% 75% max

Pregnancies 3.0000 6.00000 17.00

Glucose 117.0000 140.25000 199.00

BloodPressure 72.0000 80.00000 122.00

SkinThickness 23.0000 32.00000 99.00

Insulin 30.5000 127.25000 846.00

BMI 32.0000 36.60000 67.10

DiabetesPedigreeFunction 0.3725 0.62625 2.42

Age 29.0000 41.00000 81.00

Outcome 0.0000 1.00000 1.00

Outcome distribution:

Outcome

0 500

1 268

Name: count, dtype: int64