AI BASED DIABETES PREDICTION SYSTEM

PROBLEM DEFINITION AND DESIGN THINKING

TEAM 3

COLLEGE CODE: 3128

èREXLIN FELIX S

èG GOWTHAM

èRAMYA DEVI

èM NETHAJI

èSIVA RANJINI

Creating an AI-based diabetes prediction system involves several steps, from data collection and preprocessing to model development and evaluation. Here's a general guide on how you can approach building such a system:

1. Data Collection:

Gather relevant datasets containing information about individuals, including features like age, BMI, family history of diabetes, physical activity, diet, and other relevant health indicators.

Ensure the data is diverse, representative, and of high quality to enhance the model's accuracy.

2. Data Preprocessing:

Handle missing data by imputing or removing it.

Normalize or standardize numerical features.

Encode categorical variables.

Split the dataset into training and testing sets.

3. Feature Selection:

Identify the most relevant features for predicting diabetes.

Use techniques like correlation analysis or feature importance from tree-based models.

4. Model Selection:

Choose an appropriate machine learning model. Common choices for classification tasks like this include:

Logistic Regression

Decision Trees

Random Forest

Support Vector Machines (SVM)

Neural Networks (Deep Learning)

5. Model Training:

Train the selected model on the training dataset.

Tune hyperparameters to optimize performance.

Consider techniques like cross-validation to assess generalization.

6. Evaluation:

Evaluate the model using the testing dataset.

Metrics such as accuracy, precision, recall, F1 score, and area under the ROC curve are commonly used for classification tasks.

Understand the trade-offs between different metrics based on the specific goals of your prediction system.

7. Deployment:

Once satisfied with the model's performance, deploy it as part of a web application, mobile app, or another interface.

Implement the necessary security measures, especially if dealing with sensitive health data.

8. Continuous Monitoring and Improvement:

Regularly update the model with new data to improve its accuracy and relevance.

Monitor the model's performance in a real-world setting and update it as needed.

9. Ethical Considerations:

Be mindful of potential biases in the data and the model's predictions.

Ensure that your system complies with privacy and ethical standards, especially when dealing with health-related data.

10. User Interface (UI) Design:

Design a user-friendly interface for interacting with the prediction system.

Clearly communicate the predictions and any associated risks.

11. Education and Explanation:

Provide information to users about how the model works and the limitations of its predictions.

Include educational resources about diabetes prevention and management.

12. Regulatory Compliance:

Ensure that your system complies with relevant data protection and healthcare regulations.

Remember that the success of an AI-based system also depends on collaboration with healthcare professionals, domain experts, and end-users to ensure it meets real-world needs and adheres to ethical standards.

Creating an AI-based diabetes prediction system involves several steps, including data preparation, model training, and deployment. Below is a simple implementation in Python using popular libraries such as scikit-learn and Flask for creating a web-based interface. Keep in mind that this is a basic example, and in a real-world scenario, you would need a more extensive dataset and potentially a more sophisticated model.

Step 1: Install Dependencies

bash

pip install flask scikit-learn pandas

Step 2: Create a Python Script

Create a Python script (e.g., diabetes\_prediction.py) with the following code:

python

# Import necessary libraries

from flask import Flask, request, render\_template

import pickle

import numpy as np

# Create a Flask web application

app = Flask(\_\_name\_\_)

# Load the trained machine learning model

with open('diabetes\_model.pkl', 'rb') as model\_file:

model = pickle.load(model\_file)

# Define a route for the home page

@app.route('/')

def home():

return render\_template('index.html')

# Define a route for predicting diabetes

@app.route('/predict', methods=['POST'])

def predict():

# Get input values from the form

features = [float(x) for x in request.form.values()]

# Convert input values to a NumPy array

input\_data = np.array(features).reshape(1, -1)

# Make a prediction using the trained model

prediction = model.predict(input\_data)

# Display the prediction on a new page

return render\_template('result.html', prediction=prediction[0])

# Run the Flask application

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

Step 3: Create HTML Templates

Create two HTML templates in a folder named templates:

1. index.html:

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Diabetes Prediction</title>

</head>

<body>

<h2>Diabetes Prediction System</h2>

<form action="/predict" method="post">

<label>Pregnancies:</label>

<input type="text" name="Pregnancies" required><br>

<label>Glucose:</label>

<input type="text" name="Glucose" required><br>

<label>BloodPressure:</label>

<input type="text" name="BloodPressure" required><br>

<label>SkinThickness:</label>

<input type="text" name="SkinThickness" required><br>

<label>Insulin:</label>

<input type="text" name="Insulin" required><br>

<label>BMI:</label>

<input type="text" name="BMI" required><br>

<label>DiabetesPedigreeFunction:</label>

<input type="text" name="DiabetesPedigreeFunction" required><br>

<label>Age:</label>

<input type="text" name="Age" required><br>

<button type="submit">Predict</button>

</form>

</body>

</html>

2. result.html:

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Diabetes Prediction Result</title>

</head>

<body>

<h2>Diabetes Prediction Result</h2>

<p>The predicted outcome is: {{ prediction }}</p>

</body>

</html>

Step 4: Train and Save the Model

Train a machine learning model using a suitable dataset and save it using the pickle library. Replace this with your actual machine learning model and dataset.

Step 5: Run the Application

Run your Flask application:

bash

python diabetes\_prediction.py

Remember, this is a basic example, and in a real-world scenario, you'd need to address security, model performance, and other considerations. Additionally, the quality of predictions heavily depends on the quality and representativeness of the training data and the chosen machine learning model.

Creating an AI-based diabetes prediction system using data mining involves several steps. Here's an overview of the process

INTRODUCTION:

**6 fundamentals used:**

Data collection

Data exploration

Data visualization

Data cleansing

Advanced analysis

Data preprocessing

2. PROPOSED SYSTEM

This section describes the working procedures and implementation of various machine learning techniques to design the proposed automatic diabetes prediction system. Figure [1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0001/) shows the different stages of this research work. First, the dataset was collected and preprocessed to remove the necessary discrepancies from the dataset, for example, replacing null instances with mean values, dealing with imbalanced class issues etc. Then the dataset was separated into the training set and test set using the holdout validation technique. Next, different classification algorithms were applied to find the best classification algorithm for this dataset. Finally, the best‐performed prediction model is deployed into the proposed website and smartphone application framework.

A diagram of a machine learning process

Description automatically generated

[FIGURE 1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0001/)

Working sequences of the proposed diabetes prediction system

2.1. Dataset

The Pima Indian dataset is an open‐source dataset [[6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0006)] that is publicly available for machine learning classification, which has been used in this work along with a private dataset. It contains 768 patients’ data, and 268 of them have developed diabetes.

Figure [2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0002/) shows the ratio of people having diabetes in the Pima Indian dataset. Table [1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0001/) demonstrates the eight features of the open‐source Piman Indian dataset.

A comparison of a pie chart

Description automatically generated

[FIGURE 2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0002/)

Percentage of people having diabetes in the Pima Indian dataset

TABLE 1

Features of the Pima Indian Dataset

|  |  |  |
| --- | --- | --- |
| **Pregnancies** | **Skin thickness** | **Diabetes pedigree function** |
| Glucose | Insulin | Age |
| Blood pressure | BMI |  |

RTML private dataset: A significant contribution of this work is to present a private dataset from Rownak Textile Mills Ltd, Dhaka, Bangladesh, referred to as RTML, to the scientific community. Following a brief explanation of the study to the female volunteers, they voluntarily agreed to participate in the study. This dataset comprises six features, that is, pregnancy, glucose, blood pressure, skin thickness, BMI, age, and outcome of diabetes from 203 female individuals aged between 18 and 77. In this work, blood glucose was measured by the GlucoLeader Enhance blood sugar meter. The blood pressure and skin thickness of the participants were obtained by OMRON HEM‐7156T and digital LCD body fat caliper machines, respectively. Table [2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0002/) illustrates distinct features of the private RTML dataset with their minimum, maximum, and average values.

TABLE 2

Features of the RTML private dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Minimum** | **Maximum** | **Average** |
| Pregnancies | 0 | 8 | 1.61 |
| Glucose (mg/dL) | 52.2 | 274 | 109.39 |
| Blood pressure (mm Hg) | 5.9 | 115 | 71.09 |
| Skin thickness (mm) | 2.9 | 23.3 | 10.78 |
| BMI (kg/m2) | 2.61 | 41.62 | 22.69 |
| Age (years) | 17 | 77 | 27.02 |

2.2. Dataset preprocessing

In the merged dataset, we discovered a few exceptional zero values. For example, skin thickness and Body Mass Index (BMI) cannot be zero. The zero value has been replaced by its corresponding mean value. The training and test dataset has been separated using the holdout validation technique, where 80% is the training data and 20% is the test data.

Mutual Information: Mutual information attempts to measure the interdependence of variables. It produces information gain, and its higher values indicate greater dependency [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)].

Figure [3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0003/) shows the mutual information of various features, that is, the importance of each attribute of this dataset. For example, according to this figure, the diabetes pedigree function seems less important according to this mutual information technique.

A graph of a number of people with diabetes

Description automatically generated with medium confidence

[FIGURE 3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0003/)

Feature importance hierarchy

Semi‐supervised learning: A combined dataset has been used in this work by incorporating the open‐source Pima Indian and private RTML datasets. According to Table [2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0002/), the RTML dataset does not contain the insulin feature, which is predicted using a semi‐supervised approach. Before merging the collected dataset with the Pima Indian dataset, a model was created using the extreme gradient boosting technique (XGB regressor). Various regression and ensemble learning techniques have been successfully used in many works to predict missing values [[25](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0025), [26](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0026)]. An extensive investigation has been performed while choosing the best‐performed regressor technique to predict the insulin feature of the RTML dataset from the Pima Indian dataset. As the actual value of the insulin was not available in the RTML dataset, the Pima Indian dataset was initially used to select the best regression model. First, the Pima Indian dataset was divided into an 8:2 ratio and three supervised regression models, extreme gradient boosting technique (XGB), support vector regression (SVR), and Gaussian process regression (GPR), have been employed to predict the selected outcome, that is, insulin of the validation samples of the Pima Indian dataset. Next, we computed the root mean square error (RMSE) of various regression frameworks as

RMSE=∑Ni=1(Predictedi−Actuali)2N−−−−−−−−−−−−−−−−−−−−−−−−√

(1)

where *N* denotes the total number of validation samples of the Pima Indian dataset.

According to Table [3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0003/), the XGB technique exhibits the lowest RMSE of insulin on the Pima Indian dataset. Therefore, this model has been used to predict the missing insulin column of the collected RTML dataset from the Pima Indian dataset. The working steps of predicting insulin in the RTML dataset have been illustrated in Figure [4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0004/).

TABLE 3

RMSE of various regression models on the Pima Indian dataset

|  |  |
| --- | --- |
| **Regression model** | **RMSE** |
| XGB | 0.36 |
| SVR | 0.45 |
| GPR | 0.43 |

A diagram of a model

Description automatically generated

[FIGURE 4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0004/)

Working steps of predicting insulin of the RTML dataset

Merged dataset: After the semi‐supervised approach, we predicted the insulin feature and merged the RTML dataset with the Pima Indian dataset. The merged dataset contained 877 data with all the features, excluding the diabetes pedigree function, as it was the least important feature according to mutual information.

SMOTE and ADASYN for class imbalance: The merged dataset used in this work comprises the imbalance problem with 302 and 669 diabetes and non‐diabetes samples, respectively. To take care of this problem, the SMOTE and ADASYN techniques have been applied to the training dataset, leaving the testing data unaffected. Adaptive Synthetic Sampling, known as ADASYN, is a synthetic data generation technique with the characteristics of not duplicating minority samples and generating more data for ‘harder to learn’ examples [[13](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0013)]. As a result, the minority class will be sampled to the same extent as the majority class.

Min–Max normalization: In this research, we used the min–max normalization technique. The data has been scaled to the same range using the following equation:

Xscaled=X−XminXmax−Xmin

(2)

where *X* max and *X* min denote maximum and minimum values in the individual feature column, respectively.

2.3. Machine learning classifiers

In this work, various machine learning and ensemble techniques have been employed to implement the automatic diabetes prediction system, briefly discussed below. GridSearchCV framework has been employed in this research to find the optimal values of different hyperparameters for all the machine learning models to prevent overfitting.

Decision tree: A decision tree represents the learning function provided by a set of rules. The decision tree learning technique performs a method for approximating discrete‐valued target functions. Gini or entropy [[7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0007)] are used to determine information gain, and each node is chosen based on these coefficients, which are expressed as

Ginii=1−∑k=1n(pi,k)2

(3)

Entropy=∑i=1n−pilog2pi

(4)

In ([3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-disp-0003)) and ([4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-disp-0004)), *n* represents the number of distinct class values. We observed that max depth = 2, minimum samples leaf = 50, and ‘Gini’ impurity metrics work well in the employed dataset in this work using the GridSearchCV hyperparameter tuning.

KNN classifier: A discrete‐valued function can be approximated by *K* number of nearest classifiers [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)]. To categorize, it creates a plane with the available training points and calculates the distance between the query and trained points. It determines the *K* number of neighbours (depending on the dataset) and classifies them using majority voting. In our research, we used *K* = 5 for the binary classification.

Random forest: Random forest is a machine learning system that averages the predictions of several decision trees. As a result, the random forest can be considered an ensemble learning model [[7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0007)]. In this research, we have applied random forest with estimators = 400, minimum samples leaf = 5, and ‘Gini’ impurity metrics utilizing hyperparameter tuning.

Support vector machine: SVM performs supervised classification by choosing the best hyperplane [[11](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0011)]. In this study, we experimented with various SVM kernels in the training set. Finally, we discovered the SVM with a linear kernel, parameters *C* = 10 and gamma = 1, produces the best results in this dataset.

Logistic regression: Logistic regression can be used to predict a binary class. To predict the outcome, it fits an ‘S’ shaped function [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)]. The hyperparameter optimization technique obtained the maximum number of iterations for the convergence of the logistic regression model to be 150.

AdaBoost: AdaBoost is an ensemble technique. This classifier initially works on the original dataset, then fits repeated copies of the classifier to the same dataset. This framework adjusts the weights of improperly classified instances so that successive classifiers focus more on difficult circumstances. We have applied AdaBoost with estimator = 50 and learning rate = 0.10 in this work.

XGBoost: XGBoost is an ensemble machine learning technique based on decision trees that employ a gradient boosting approach [[20](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0020)]. The parameters used for the proposed XGBoost classifier are as follows: estimators’ maximum depth = 4 and ‘binary logistic’ objective function.

Voting classifier: It is an ensemble technique to improve the classification by voting [[7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0007)]. This paper implements a voting classifier that selects the majority class predicted by each classifier with a ‘soft’ voting hyperparameter.

Bagging: Bagging classifiers are ensemble classifiers that fit base classifiers to random subsets of the original dataset and then aggregate their individual predictions voting to generate a final classification [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)]. In the implemented bagging classifier, base estimators = 500, maximum number of samples = 100, and out‐of‐bag score = ‘True’ are used as various hyperparameters.

2.4. Deployment of the prediction system

The proposed machine learning‐based diabetes prediction system has been deployed into a website and smartphone application framework to work instantaneously on real data.

Web application: We have used HTML and CSS for the frontend part of the proposed website. After that, we finalized the machine learning model XGBoost with ADASYN, as it provided the best performance. The model deployment has been done with Spyder, a Python environment platform that works with Anaconda. Figure [5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0005/) shows the illustration of the website application development process.

A diagram of a computer program

Description automatically generated

[FIGURE 5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0005/)

Development of the web application

Android smartphone application: To demonstrate the automatic diabetes forecasting system in real time, we also designed an Android smartphone application to test its performance. Android Studio is used for the frontend part of this application. We employed Java as the necessary coding language. After that, the model has been implemented in Android Studio using the pickle package. While developing the API, we used Heroku to host our model on the corresponding hosting server. Figure [6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0006/) demonstrates the necessary steps in developing the proposed Android application.

A diagram of a software development process

Description automatically generated

[FIGURE 6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0006/)

Working sequences of the proposed android application development

3. RESULTS AND DISCUSSION

This section presents the results and discussion of the proposed automatic diabetes prediction system. First, the performance of various machine learning techniques is discussed. Next, the implemented website framework and Android smartphone application are demonstrated. We used precision, recall, F1 score, AUC, and classification accuracy to evaluate various ML models. Equations of these metrics are expressed as

Precision=TPTP+FP

(5)

Recall=TPTP+FN

(6)

F1score=2×Recall×PrecisionRecall+Precision

(7)

where TP denotes the model is predicting positive, and the result is also positive. FP indicates the positive prediction of the model, but the result is negative. TN expresses the model is predicting negative, and the result is also negative. FN indicates the model predicts negative, but the result is positive. In this work, the holdout validation approach with a stratified 8:2 train‐test split has been used for all the machine learning models.

Table [4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0004/) compares different performance metrics of various classifiers for the merged dataset with SMOTE synthetic oversampling technique. According to this table, the bagging classifier achieved the best overall performance with 79% accuracy and 0.79 and 0.87 F1 score and AUC, respectively.

TABLE 4

Performance metrics of various classifiers with SMOTE technique in the merged dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **AUC** |
| Logistic regression | 0.78 | 0.77 | 0.77 | 77% | 0.88 |
| KNN | 0.78 | 0.76 | 0.76 | 76% | 0.85 |
| Random forest | 0.78 | 0.78 | 0.78 | 78% | 0.87 |
| Decision tree | 0.75 | 0.73 | 0.73 | 73% | 0.75 |
| **Bagging** | **0.80** | **0.79** | **0.79** | **79%** | **0.87** |
| Adaboost | 0.79 | 0.78 | 0.78 | 78% | 0.85 |
| XGboost | 0.78 | 0.78 | 0.78 | 78% | 0.84 |
| Voting | 0.79 | 0.79 | 0.79 | 79% | 0.86 |
| SVM | 0.78 | 0.75 | 0.76 | 75% | 0.87 |

Table [5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0005/) shows various performance metrics of all the classifiers using the ADASYN approach in the merged datasets. According to Table [4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0004/), the XGBoost framework performed better than other classifiers with 81% accuracy and 0.84 AUC. Conversely, the decision tree approach achieved the lowest accuracy and F1 score.

TABLE 5

Performance metrics of various classifiers using adasyn in the merged dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Auc** |
| Logistic regression | 0.76 | 0.75 | 0.75 | 75% | 0.84 |
| KNN | 0.76 | 0.73 | 0.73 | 73% | 0.82 |
| Random forest | 0.76 | 0.76 | 0.76 | 76% | 0.84 |
| Decision tree | 0.81 | 0.72 | 0.72 | 72% | 0.78 |
| Bagging | 0.80 | 0.79 | 0.79 | 79% | 0.84 |
| AdaBoost | 0.75 | 0.76 | 0.76 | 76% | 0.84 |
| **XGBoost** | **0.81** | **0.81** | **0.81** | **81%** | **0.84** |
| Voting | 0.77 | 0.77 | 0.77 | 77% | 0.84 |
| SVM | 0.78 | 0.78 | 0.77 | 78% | 0.83 |

Next, the domain adaptation approach has been applied where the machine learning model is trained and evaluated on different samples, that is, source and target datasets, respectively. In this work, initially, the automatic diabetes prediction model is trained on the open‐source Pima Indian dataset with a larger size. Finally, the model is evaluated on the private RTML dataset with a much smaller dimension. Table [6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0006/) demonstrates the performance metrics for the private dataset. It is interesting to note that the XGBoost with ADASYN framework has been applied in the training dataset in this case.

TABLE 6

Performance metrics for the private dataset (domain adaptation technique)

|  |  |  |  |
| --- | --- | --- | --- |
| **Precision** | **Recall** | **F1 score** | **Accuracy** |
| 0.95 | 0.96 | 0.95 | 96% |

Figure [7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0007/) depicts the confusion matrix for XGBoost with ADASYN. According to this figure, the XGBoost technique correctly classified 141 instances with TP = 43 and TN = 98.

A chart with different colored squares

Description automatically generated

[FIGURE 7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0007/)

Confusion matrix for XGBoost with ADASYN technique

The ROC curve of the XGBoost with the ADASYN approach has been illustrated in Figure [8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0008/). This figure shows the AUC value of XGBoost is 0.84.

A blue and orange line graph

Description automatically generated

[FIGURE 8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0008/)

ROC curve and AUC value for the XGBoost with ADASYN

Next, explainable AI techniques with SHAP and LIME frameworks are implemented to understand how the model predicts the decision. Figure [9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0009/) shows the XGBoost with ADASYN feature importance with the help of explainable AI, SHAP library.

A graph showing a number of yellow bars

Description automatically generated with medium confidence

[FIGURE 9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0009/)

Explainable AI interpretation of feature importance of XGBoost with ADASYN

Figure [10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0010/) illustrates an interpretation of the XGBoost model implemented by the LIME explainable AI method. According to this figure, the model predicts diabetes correctly for this specific person with 80% confidence. The ML model predicts this class as the person has a glucose level of more than 140.25 and involves pregnancies of more than 6.

A graph with numbers and text

Description automatically generated with medium confidence

[FIGURE 10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0010/)

LIME explainable AI prediction interpretation

Finally, the proposed automatic diabetes prediction system has been deployed into a website and Android smartphone application employing the XGBoost machine learning framework with ADASYN. Figure [11](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0011/) shows an instantaneous diabetes prediction by the designed web application with real data.

A screenshot of a medical form

Description automatically generated

[FIGURE 11](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0011/)

Instantaneous diabetes prediction by the designed web application

Figure [12](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0012/) displays the home screen of the proposed Android mobile application created using the best classification algorithm XGBoost. Finally, a survey was conducted in which users rated the application's various features. Figure [13](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0013/) illustrates the review details of the implemented Android application's survey results. Sixteen volunteers reviewed the application in total, and all of them were female. The participants rated each feature on a scale of 1 to 10, and their average was calculated. According to this figure, the diabetes prediction and daily diet chart features of the application achieved the highest ratings of 8.40 and 8, respectively.

A screenshot of a cell phone

Description automatically generated

[FIGURE 12](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0012/)

Home screen of the proposed android application

A graph of blue bars

Description automatically generated with medium confidence

[FIGURE 13](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0013/)

Android application review ratings

It is worth mentioning that the RTML dataset's insulin feature has been predicted from the Pima Indian dataset by applying the XGB regression technique for all of the results discussed above. However, alternative investigations have been conducted to obtain the insulin feature of the RTML dataset, that is, the mean and median imputation of various patients’ insulin of the Pima Indian dataset. Tables [7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0007/) and  [8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0008/) demonstrate various performance metrics of the machine learning models with the ADASYN technique when the RTML dataset's missing insulin features are obtained from the mean and median values of the Pima Indian dataset.

TABLE 7

Performance metrics of classifiers in the merged dataset (RTML insulin obtained from Pima Indian mean)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** |
| AdaBoost | 0.77 | 0.77 | 0.77 | 77% |
| Random Forest | 0.77 | 0.76 | 0.76 | 76% |
| XGBoost | 0.78 | 0.78 | 0.78 | 78% |

TABLE 8

Performance metrics of classifiers in the merged dataset (RTML insulin obtained from Pima Indian median)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** |
| AdaBoost | 0.78 | 0.78 | 0.78 | 78% |
| Random Forest | 0.76 | 0.76 | 0.76 | 76% |
| XGBoost | 0.77 | 0.76 | 0.76 | 76% |

Finally, another scenario has been considered where the insulin feature of the Pima Indian dataset has been removed to maintain consistency with the RTML dataset. Table [9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0009/) depicts various performance metrics of the merged dataset after removing the insulin feature. According to this table, the performance of all the prediction models degraded.

TABLE 9

Performance metrics of classifiers in the merged dataset (insulin removed from Pima Indian)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** |
| AdaBoost | 0.73 | 0.71 | 0.72 | 72% |
| Random Forest | 0.72 | 0.70 | 0.71 | 71% |
| XGBoost | 0.74 | 0.73 | 0.73 | 74% |

Table [10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0010/) illustrates the performance comparison of the proposed automatic diabetes prediction system with similar works to the Pima Indian dataset. According to this table, the proposed XGBoost technique with ADASYN outperformed most of the existing works concerning accuracy and F1 score.

TABLE 10

Comparison of the proposed system with similar diabetes prediction works

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Reference** | **Classifier** | **F1 score** | **Accuracy** | **Other metrics** |
| [[3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0003)] | Deep belief network model | 0.81 | N/A | Precision: 0.68  Recall: 1.0 |
| [[5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0005)] | SVM with RBF kernel |  | 82% |  |
| [[9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0009)] | SVM | 0.73 | 75% | Precision: 0.72  Recall: 0.75 |
| [[10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0010)] | Ensemble (XGBoost) | 0.81 | 88.8% | Precision: 0.84  Recall: 0.79 |
| [[21](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0021)] | Soft voting | 0.72 | 79.1% | Precision: 0.73  Recall: 0.72 |
| This work | XGBoost with ADASYN | 0.81 | 88.5% | Precision: 0.82  Recall: 0.80 |

[Open in a separate window](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0010/?report=objectonly)

This study aims to predict diabetes mellitus automatically by employing machine learning techniques. Pima Indian dataset and a new RTML dataset comprising physical examination data from the local female patients of Bangladesh have been used. The missing insulin feature values of the RTML dataset have been predicted from the Pima Indian dataset. Our research found that the XGB regression technique accomplished the lowest RMS error in predicting insulin. The mutual information‐based feature selection algorithm indicates the glucose level, BMI, age, and insulin to be the most salient features in predicting diabetes. SMOTE and ADASYN synthetic data oversampling and hyperparameters optimization techniques have been applied. The XGBoost technique with ADASYN achieved the best performance. The LIME and SHAP explainable AI frameworks interpret the prediction provided by the ML approaches. A limitation of this study is the nonavailability of the insulin feature of the used RTML dataset. The prediction of insulin obtained from the XGB regressor and produced from the mean and median values of the Pima India dataset comprises an average deviation for classification accuracy of approximately 1.33% and 2.33%, respectively.

Creating an AI-based diabetes prediction system using data mining involves several steps. Here's an overview of the process

INTRODUCTION:

**6 fundamentals used:**

Data collection

Data exploration

Data visualization

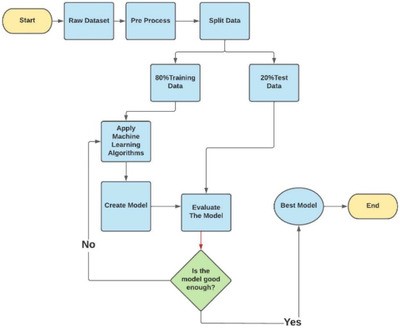
Data cleansing

Advanced analysis

Data preprocessing

2. PROPOSED SYSTEM

This section describes the working procedures and implementation of various machine learning techniques to design the proposed automatic diabetes prediction system. Figure [1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0001/) shows the different stages of this research work. First, the dataset was collected and preprocessed to remove the necessary discrepancies from the dataset, for example, replacing null instances with mean values, dealing with imbalanced class issues etc. Then the dataset was separated into the training set and test set using the holdout validation technique. Next, different classification algorithms were applied to find the best classification algorithm for this dataset. Finally, the best‐performed prediction model is deployed into the proposed website and smartphone application framework.



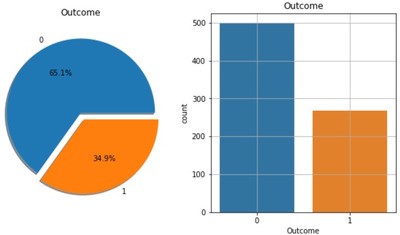
[FIGURE 1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0001/)

Working sequences of the proposed diabetes prediction system

2.1. Dataset

The Pima Indian dataset is an open‐source dataset [[6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0006)] that is publicly available for machine learning classification, which has been used in this work along with a private dataset. It contains 768 patients’ data, and 268 of them have developed diabetes.

Figure [2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0002/) shows the ratio of people having diabetes in the Pima Indian dataset. Table [1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0001/) demonstrates the eight features of the open‐source Piman Indian dataset.



[FIGURE 2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0002/)

Percentage of people having diabetes in the Pima Indian dataset

TABLE 1

Features of the Pima Indian Dataset

|  |  |  |
| --- | --- | --- |
| **Pregnancies** | **Skin thickness** | **Diabetes pedigree function** |
| Glucose | Insulin | Age |
| Blood pressure | BMI |  |

RTML private dataset: A significant contribution of this work is to present a private dataset from Rownak Textile Mills Ltd, Dhaka, Bangladesh, referred to as RTML, to the scientific community. Following a brief explanation of the study to the female volunteers, they voluntarily agreed to participate in the study. This dataset comprises six features, that is, pregnancy, glucose, blood pressure, skin thickness, BMI, age, and outcome of diabetes from 203 female individuals aged between 18 and 77. In this work, blood glucose was measured by the GlucoLeader Enhance blood sugar meter. The blood pressure and skin thickness of the participants were obtained by OMRON HEM‐7156T and digital LCD body fat caliper machines, respectively. Table [2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0002/) illustrates distinct features of the private RTML dataset with their minimum, maximum, and average values.

TABLE 2

Features of the RTML private dataset

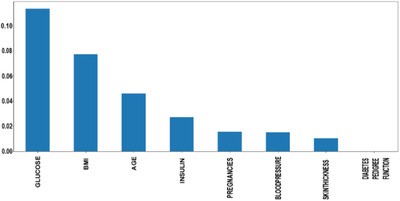
|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Minimum** | **Maximum** | **Average** |
| Pregnancies | 0 | 8 | 1.61 |
| Glucose (mg/dL) | 52.2 | 274 | 109.39 |
| Blood pressure (mm Hg) | 5.9 | 115 | 71.09 |
| Skin thickness (mm) | 2.9 | 23.3 | 10.78 |
| BMI (kg/m2) | 2.61 | 41.62 | 22.69 |
| Age (years) | 17 | 77 | 27.02 |

2.2. Dataset preprocessing

In the merged dataset, we discovered a few exceptional zero values. For example, skin thickness and Body Mass Index (BMI) cannot be zero. The zero value has been replaced by its corresponding mean value. The training and test dataset has been separated using the holdout validation technique, where 80% is the training data and 20% is the test data.

Mutual Information: Mutual information attempts to measure the interdependence of variables. It produces information gain, and its higher values indicate greater dependency [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)].

Figure [3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0003/) shows the mutual information of various features, that is, the importance of each attribute of this dataset. For example, according to this figure, the diabetes pedigree function seems less important according to this mutual information technique.



[FIGURE 3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0003/)

Feature importance hierarchy

Semi‐supervised learning: A combined dataset has been used in this work by incorporating the open‐source Pima Indian and private RTML datasets. According to Table [2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0002/), the RTML dataset does not contain the insulin feature, which is predicted using a semi‐supervised approach. Before merging the collected dataset with the Pima Indian dataset, a model was created using the extreme gradient boosting technique (XGB regressor). Various regression and ensemble learning techniques have been successfully used in many works to predict missing values [[25](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0025), [26](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0026)]. An extensive investigation has been performed while choosing the best‐performed regressor technique to predict the insulin feature of the RTML dataset from the Pima Indian dataset. As the actual value of the insulin was not available in the RTML dataset, the Pima Indian dataset was initially used to select the best regression model. First, the Pima Indian dataset was divided into an 8:2 ratio and three supervised regression models, extreme gradient boosting technique (XGB), support vector regression (SVR), and Gaussian process regression (GPR), have been employed to predict the selected outcome, that is, insulin of the validation samples of the Pima Indian dataset. Next, we computed the root mean square error (RMSE) of various regression frameworks as

RMSE=∑Ni=1(Predictedi−Actuali)2N−−−−−−−−−−−−−−−−−−−−−−−−√

(1)

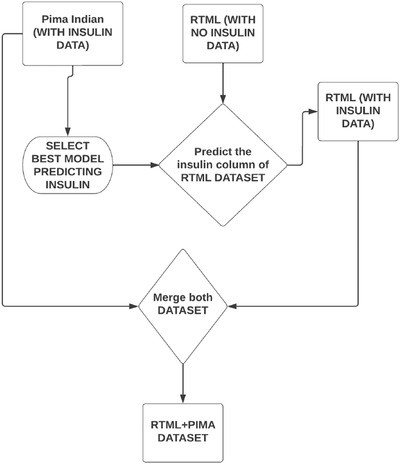
where *N* denotes the total number of validation samples of the Pima Indian dataset.

According to Table [3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0003/), the XGB technique exhibits the lowest RMSE of insulin on the Pima Indian dataset. Therefore, this model has been used to predict the missing insulin column of the collected RTML dataset from the Pima Indian dataset. The working steps of predicting insulin in the RTML dataset have been illustrated in Figure [4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0004/).

TABLE 3

RMSE of various regression models on the Pima Indian dataset

|  |  |
| --- | --- |
| **Regression model** | **RMSE** |
| XGB | 0.36 |
| SVR | 0.45 |
| GPR | 0.43 |



[FIGURE 4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0004/)

Working steps of predicting insulin of the RTML dataset

Merged dataset: After the semi‐supervised approach, we predicted the insulin feature and merged the RTML dataset with the Pima Indian dataset. The merged dataset contained 877 data with all the features, excluding the diabetes pedigree function, as it was the least important feature according to mutual information.

SMOTE and ADASYN for class imbalance: The merged dataset used in this work comprises the imbalance problem with 302 and 669 diabetes and non‐diabetes samples, respectively. To take care of this problem, the SMOTE and ADASYN techniques have been applied to the training dataset, leaving the testing data unaffected. Adaptive Synthetic Sampling, known as ADASYN, is a synthetic data generation technique with the characteristics of not duplicating minority samples and generating more data for ‘harder to learn’ examples [[13](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0013)]. As a result, the minority class will be sampled to the same extent as the majority class.

Min–Max normalization: In this research, we used the min–max normalization technique. The data has been scaled to the same range using the following equation:

Xscaled=X−XminXmax−Xmin

(2)

where *X* max and *X* min denote maximum and minimum values in the individual feature column, respectively.

2.3. Machine learning classifiers

In this work, various machine learning and ensemble techniques have been employed to implement the automatic diabetes prediction system, briefly discussed below. GridSearchCV framework has been employed in this research to find the optimal values of different hyperparameters for all the machine learning models to prevent overfitting.

Decision tree: A decision tree represents the learning function provided by a set of rules. The decision tree learning technique performs a method for approximating discrete‐valued target functions. Gini or entropy [[7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0007)] are used to determine information gain, and each node is chosen based on these coefficients, which are expressed as

Ginii=1−∑k=1n(pi,k)2

(3)

Entropy=∑i=1n−pilog2pi

(4)

In ([3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-disp-0003)) and ([4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-disp-0004)), *n* represents the number of distinct class values. We observed that max depth = 2, minimum samples leaf = 50, and ‘Gini’ impurity metrics work well in the employed dataset in this work using the GridSearchCV hyperparameter tuning.

KNN classifier: A discrete‐valued function can be approximated by *K* number of nearest classifiers [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)]. To categorize, it creates a plane with the available training points and calculates the distance between the query and trained points. It determines the *K* number of neighbours (depending on the dataset) and classifies them using majority voting. In our research, we used *K* = 5 for the binary classification.

Random forest: Random forest is a machine learning system that averages the predictions of several decision trees. As a result, the random forest can be considered an ensemble learning model [[7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0007)]. In this research, we have applied random forest with estimators = 400, minimum samples leaf = 5, and ‘Gini’ impurity metrics utilizing hyperparameter tuning.

Support vector machine: SVM performs supervised classification by choosing the best hyperplane [[11](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0011)]. In this study, we experimented with various SVM kernels in the training set. Finally, we discovered the SVM with a linear kernel, parameters *C* = 10 and gamma = 1, produces the best results in this dataset.

Logistic regression: Logistic regression can be used to predict a binary class. To predict the outcome, it fits an ‘S’ shaped function [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)]. The hyperparameter optimization technique obtained the maximum number of iterations for the convergence of the logistic regression model to be 150.

AdaBoost: AdaBoost is an ensemble technique. This classifier initially works on the original dataset, then fits repeated copies of the classifier to the same dataset. This framework adjusts the weights of improperly classified instances so that successive classifiers focus more on difficult circumstances. We have applied AdaBoost with estimator = 50 and learning rate = 0.10 in this work.

XGBoost: XGBoost is an ensemble machine learning technique based on decision trees that employ a gradient boosting approach [[20](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0020)]. The parameters used for the proposed XGBoost classifier are as follows: estimators’ maximum depth = 4 and ‘binary logistic’ objective function.

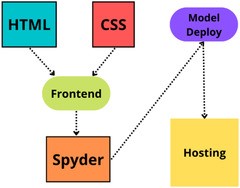
Voting classifier: It is an ensemble technique to improve the classification by voting [[7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0007)]. This paper implements a voting classifier that selects the majority class predicted by each classifier with a ‘soft’ voting hyperparameter.

Bagging: Bagging classifiers are ensemble classifiers that fit base classifiers to random subsets of the original dataset and then aggregate their individual predictions voting to generate a final classification [[8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0008)]. In the implemented bagging classifier, base estimators = 500, maximum number of samples = 100, and out‐of‐bag score = ‘True’ are used as various hyperparameters.

2.4. Deployment of the prediction system

The proposed machine learning‐based diabetes prediction system has been deployed into a website and smartphone application framework to work instantaneously on real data.

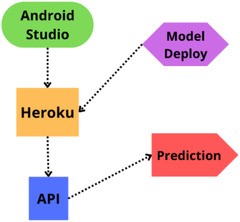
Web application: We have used HTML and CSS for the frontend part of the proposed website. After that, we finalized the machine learning model XGBoost with ADASYN, as it provided the best performance. The model deployment has been done with Spyder, a Python environment platform that works with Anaconda. Figure [5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0005/) shows the illustration of the website application development process.



[FIGURE 5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0005/)

Development of the web application

Android smartphone application: To demonstrate the automatic diabetes forecasting system in real time, we also designed an Android smartphone application to test its performance. Android Studio is used for the frontend part of this application. We employed Java as the necessary coding language. After that, the model has been implemented in Android Studio using the pickle package. While developing the API, we used Heroku to host our model on the corresponding hosting server. Figure [6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0006/) demonstrates the necessary steps in developing the proposed Android application.



[FIGURE 6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0006/)

Working sequences of the proposed android application development

3. RESULTS AND DISCUSSION

This section presents the results and discussion of the proposed automatic diabetes prediction system. First, the performance of various machine learning techniques is discussed. Next, the implemented website framework and Android smartphone application are demonstrated. We used precision, recall, F1 score, AUC, and classification accuracy to evaluate various ML models. Equations of these metrics are expressed as

Precision=TPTP+FP

(5)

Recall=TPTP+FN

(6)

F1score=2×Recall×PrecisionRecall+Precision

(7)

where TP denotes the model is predicting positive, and the result is also positive. FP indicates the positive prediction of the model, but the result is negative. TN expresses the model is predicting negative, and the result is also negative. FN indicates the model predicts negative, but the result is positive. In this work, the holdout validation approach with a stratified 8:2 train‐test split has been used for all the machine learning models.

Table [4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0004/) compares different performance metrics of various classifiers for the merged dataset with SMOTE synthetic oversampling technique. According to this table, the bagging classifier achieved the best overall performance with 79% accuracy and 0.79 and 0.87 F1 score and AUC, respectively.

TABLE 4

Performance metrics of various classifiers with SMOTE technique in the merged dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **AUC** |
| Logistic regression | 0.78 | 0.77 | 0.77 | 77% | 0.88 |
| KNN | 0.78 | 0.76 | 0.76 | 76% | 0.85 |
| Random forest | 0.78 | 0.78 | 0.78 | 78% | 0.87 |
| Decision tree | 0.75 | 0.73 | 0.73 | 73% | 0.75 |
| **Bagging** | **0.80** | **0.79** | **0.79** | **79%** | **0.87** |
| Adaboost | 0.79 | 0.78 | 0.78 | 78% | 0.85 |
| XGboost | 0.78 | 0.78 | 0.78 | 78% | 0.84 |
| Voting | 0.79 | 0.79 | 0.79 | 79% | 0.86 |
| SVM | 0.78 | 0.75 | 0.76 | 75% | 0.87 |

Table [5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0005/) shows various performance metrics of all the classifiers using the ADASYN approach in the merged datasets. According to Table [4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0004/), the XGBoost framework performed better than other classifiers with 81% accuracy and 0.84 AUC. Conversely, the decision tree approach achieved the lowest accuracy and F1 score.

TABLE 5

Performance metrics of various classifiers using adasyn in the merged dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Auc** |
| Logistic regression | 0.76 | 0.75 | 0.75 | 75% | 0.84 |
| KNN | 0.76 | 0.73 | 0.73 | 73% | 0.82 |
| Random forest | 0.76 | 0.76 | 0.76 | 76% | 0.84 |
| Decision tree | 0.81 | 0.72 | 0.72 | 72% | 0.78 |
| Bagging | 0.80 | 0.79 | 0.79 | 79% | 0.84 |
| AdaBoost | 0.75 | 0.76 | 0.76 | 76% | 0.84 |
| **XGBoost** | **0.81** | **0.81** | **0.81** | **81%** | **0.84** |
| Voting | 0.77 | 0.77 | 0.77 | 77% | 0.84 |
| SVM | 0.78 | 0.78 | 0.77 | 78% | 0.83 |

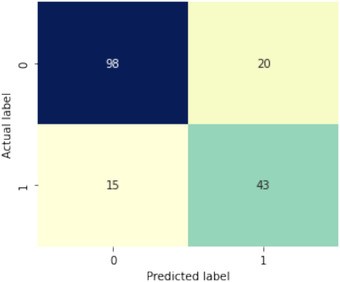
Next, the domain adaptation approach has been applied where the machine learning model is trained and evaluated on different samples, that is, source and target datasets, respectively. In this work, initially, the automatic diabetes prediction model is trained on the open‐source Pima Indian dataset with a larger size. Finally, the model is evaluated on the private RTML dataset with a much smaller dimension. Table [6](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0006/) demonstrates the performance metrics for the private dataset. It is interesting to note that the XGBoost with ADASYN framework has been applied in the training dataset in this case.

TABLE 6

Performance metrics for the private dataset (domain adaptation technique)

|  |  |  |  |
| --- | --- | --- | --- |
| **Precision** | **Recall** | **F1 score** | **Accuracy** |
| 0.95 | 0.96 | 0.95 | 96% |

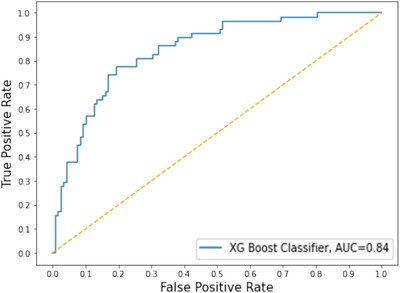
Figure [7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0007/) depicts the confusion matrix for XGBoost with ADASYN. According to this figure, the XGBoost technique correctly classified 141 instances with TP = 43 and TN = 98.



[FIGURE 7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0007/)

Confusion matrix for XGBoost with ADASYN technique

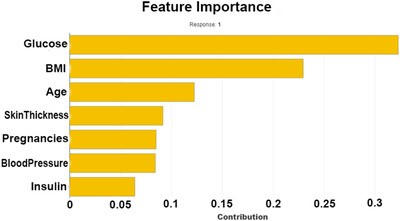
The ROC curve of the XGBoost with the ADASYN approach has been illustrated in Figure [8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0008/). This figure shows the AUC value of XGBoost is 0.84.



[FIGURE 8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0008/)

ROC curve and AUC value for the XGBoost with ADASYN

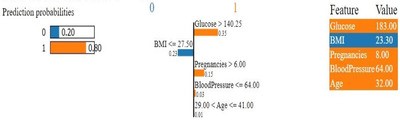
Next, explainable AI techniques with SHAP and LIME frameworks are implemented to understand how the model predicts the decision. Figure [9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0009/) shows the XGBoost with ADASYN feature importance with the help of explainable AI, SHAP library.



[FIGURE 9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0009/)

Explainable AI interpretation of feature importance of XGBoost with ADASYN

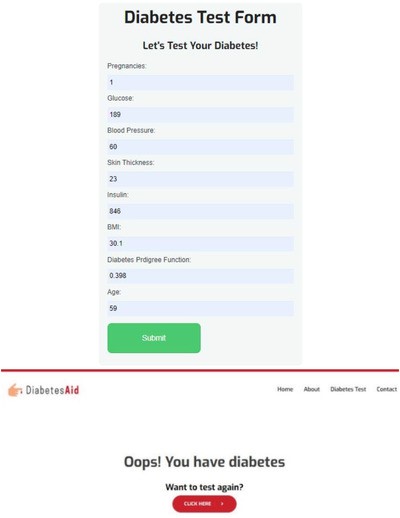
Figure [10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0010/) illustrates an interpretation of the XGBoost model implemented by the LIME explainable AI method. According to this figure, the model predicts diabetes correctly for this specific person with 80% confidence. The ML model predicts this class as the person has a glucose level of more than 140.25 and involves pregnancies of more than 6.



[FIGURE 10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0010/)

LIME explainable AI prediction interpretation

Finally, the proposed automatic diabetes prediction system has been deployed into a website and Android smartphone application employing the XGBoost machine learning framework with ADASYN. Figure [11](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0011/) shows an instantaneous diabetes prediction by the designed web application with real data.



[FIGURE 11](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0011/)

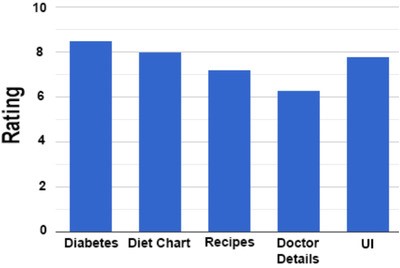
Instantaneous diabetes prediction by the designed web application

Figure [12](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0012/) displays the home screen of the proposed Android mobile application created using the best classification algorithm XGBoost. Finally, a survey was conducted in which users rated the application's various features. Figure [13](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0013/) illustrates the review details of the implemented Android application's survey results. Sixteen volunteers reviewed the application in total, and all of them were female. The participants rated each feature on a scale of 1 to 10, and their average was calculated. According to this figure, the diabetes prediction and daily diet chart features of the application achieved the highest ratings of 8.40 and 8, respectively.



[FIGURE 12](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0012/)

Home screen of the proposed android application



[FIGURE 13](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/figure/htl212039-fig-0013/)

Android application review ratings

It is worth mentioning that the RTML dataset's insulin feature has been predicted from the Pima Indian dataset by applying the XGB regression technique for all of the results discussed above. However, alternative investigations have been conducted to obtain the insulin feature of the RTML dataset, that is, the mean and median imputation of various patients’ insulin of the Pima Indian dataset. Tables [7](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0007/) and  [8](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0008/) demonstrate various performance metrics of the machine learning models with the ADASYN technique when the RTML dataset's missing insulin features are obtained from the mean and median values of the Pima Indian dataset.

TABLE 7

Performance metrics of classifiers in the merged dataset (RTML insulin obtained from Pima Indian mean)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** |
| AdaBoost | 0.77 | 0.77 | 0.77 | 77% |
| Random Forest | 0.77 | 0.76 | 0.76 | 76% |
| XGBoost | 0.78 | 0.78 | 0.78 | 78% |

TABLE 8

Performance metrics of classifiers in the merged dataset (RTML insulin obtained from Pima Indian median)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** |
| AdaBoost | 0.78 | 0.78 | 0.78 | 78% |
| Random Forest | 0.76 | 0.76 | 0.76 | 76% |
| XGBoost | 0.77 | 0.76 | 0.76 | 76% |

Finally, another scenario has been considered where the insulin feature of the Pima Indian dataset has been removed to maintain consistency with the RTML dataset. Table [9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0009/) depicts various performance metrics of the merged dataset after removing the insulin feature. According to this table, the performance of all the prediction models degraded.

TABLE 9

Performance metrics of classifiers in the merged dataset (insulin removed from Pima Indian)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Precision** | **Recall** | **F1 Score** | **Accuracy** |
| AdaBoost | 0.73 | 0.71 | 0.72 | 72% |
| Random Forest | 0.72 | 0.70 | 0.71 | 71% |
| XGBoost | 0.74 | 0.73 | 0.73 | 74% |

Table [10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0010/) illustrates the performance comparison of the proposed automatic diabetes prediction system with similar works to the Pima Indian dataset. According to this table, the proposed XGBoost technique with ADASYN outperformed most of the existing works concerning accuracy and F1 score.

TABLE 10

Comparison of the proposed system with similar diabetes prediction works

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Reference** | **Classifier** | **F1 score** | **Accuracy** | **Other metrics** |
| [[3](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0003)] | Deep belief network model | 0.81 | N/A | Precision: 0.68  Recall: 1.0 |
| [[5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0005)] | SVM with RBF kernel |  | 82% |  |
| [[9](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0009)] | SVM | 0.73 | 75% | Precision: 0.72  Recall: 0.75 |
| [[10](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0010)] | Ensemble (XGBoost) | 0.81 | 88.8% | Precision: 0.84  Recall: 0.79 |
| [[21](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/#htl212039-bib-0021)] | Soft voting | 0.72 | 79.1% | Precision: 0.73  Recall: 0.72 |
| This work | XGBoost with ADASYN | 0.81 | 88.5% | Precision: 0.82  Recall: 0.80 |

[Open in a separate window](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/table/htl212039-tbl-0010/?report=objectonly)

This study aims to predict diabetes mellitus automatically by employing machine learning techniques. Pima Indian dataset and a new RTML dataset comprising physical examination data from the local female patients of Bangladesh have been used. The missing insulin feature values of the RTML dataset have been predicted from the Pima Indian dataset. Our research found that the XGB regression technique accomplished the lowest RMS error in predicting insulin. The mutual information‐based feature selection algorithm indicates the glucose level, BMI, age, and insulin to be the most salient features in predicting diabetes. SMOTE and ADASYN synthetic data oversampling and hyperparameters optimization techniques have been applied. The XGBoost technique with ADASYN achieved the best performance. The LIME and SHAP explainable AI frameworks interpret the prediction provided by the ML approaches. A limitation of this study is the nonavailability of the insulin feature of the used RTML dataset. The prediction of insulin obtained from the XGB regressor and produced from the mean and median values of the Pima India dataset comprises an average deviation for classification accuracy of approximately 1.33% and 2.33%, respectively.

THANK YOU..........................