Final Project Report

A Binary Classification Model for Diabetes Detection

Team 4

Team Members:

Rishi Mandyam, CBE, Undergraduate

D M Zahin Sajid, ECE, Undergraduate

Date of Submission: 5/6/23

Abstract

This project focuses on building, training, and evaluating multiple machine learning models, including Artificial Neural Networks (ANN), Decision Trees, Logistic Regression, k-Nearest Neighbors (kNN), and Random Forest, for predicting diabetes based on patient data. Using various evaluation metrics such as accuracy, precision, recall, F1-score, sensitivity, and specificity, we compared the performance of these models to determine their effectiveness in identifying diabetes cases. Our findings provide insights into the potential application of machine learning techniques for early diabetes prediction and inform the development of more efficient healthcare solutions.

Introduction

Using the diabetes dataset found on Kaggle, we set out to develop a binary classification model to determine whether a patient has diabetes given their biological characteristics and health factors.

Diabetes affects around 37.3 million people in the US alone (11.3% of its population), with 8.5 million adults having undiagnosed diabetes. Many more people have Prediabetes, a condition characterized by high blood sugar, but not high enough to be considered Type II Diabetes (4). As of 2019, Diabetes is the seventh leading cause of death in the US and as of 2018, the cost of *diagnosed* diabetes was estimated to be $327 billion (3). Worldwide, Diabetes is estimated to affect 537 million adults as of 2021 and is responsible for nearly 6.7 million deaths; on average roughly one death every five seconds (5).

While rising insulin costs has been an ever-present concern, a very recent breakthrough and commitment by Civica Rx, a non-profit drug maker, to make insulin cost just $30 a vial, a monumental decrease compared to current costs (6). With insulin on the verge of being so much more affordable, it's more critical to make use of this opportunity to help as many people as possible. We hope that utilizing machine learning and improving upon existing models will help diagnose or at the very least, warrant further tests in order to diagnose, and assess strong risk factors.

According to a 2021 study reviewing current machine learning models used to predict Type II Diabetes, certain models have been able to perform very well in regards to validation parameters. In particular, one such model was able to achieve an accuracy, AUC, and sensitivity of 1 (granted, the sample size was comparatively small at n=149). In general, the "average" accuracy and AUC of the models assessed appears to be around the 0.850 and 0.850-0.900 range respectively. (7)

Data:

Link: https://www.kaggle.com/datasets/mathchi/diabetes-data-set (1)

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases . The objective is to predict based on diagnostic measurements whether a patient has diabetes. All patients here are females at least 21 years old of Pima Indian heritage. The dataset, formulated in the late 1980s, consists of a total of 9 attributes (eight features and a binary target variable) with the following labels:

- 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)

The dataset contains 786 instances, all of which are numeric values. Most of the values are whole numbers, except for the BMI and Diabetes Pedigree attributes, which are reported to 1 and 3 decimal places, respectively. Previous use of the dataset was in the study: Using the ADAP Learning Algorithm to Forecast the Onset of Diabetes Mellitus (2).

Results of the previous study: Their ADAP algorithm makes a real-valued prediction between 0 and 1. This was transformed into a binary decision using a cutoff of 0.448. Using 576 training instances, the sensitivity and specificity of their algorithm was 76% on the remaining 192 instances (2).

The diagnostic, binary-valued variable investigated is whether the patient shows signs of diabetes according to World Health Organization criteria (i.e., if the 2 hour post-load plasma glucose was at least 200 mg/dl at any survey examination or if found during routine medical care). The population lives near Phoenix, Arizona, USA.

There are missing values in the dataset, particularly in features like blood pressure, triceps skinfold thickness, and 2-hour serum insulin. These missing values are often represented as zeros in the dataset, which can be misleading if not addressed.

In our project, feature value normalization is performed using the StandardScaler from scikit-learn, which standardizes the features by removing the mean and scaling to unit variance. The dataset is split into training and testing sets using an 80-20 split, with 80% of the data used for training and 20% used for testing.

Pregnancies	Glucose	Blood Pressure	Skin Thickness	Insulin	BMI	Diabetes Pedigree function	Age	Outcome
1	89	66	23	94	28.1	0.167	21	0

Table 1: Sample of the dataset (1)

Tasks Performed:

In this project, all the code in the **Project.ipynb** was written by us. We have accomplished the following tasks:

1. Data preprocessing: We loaded and preprocessed the diabetes dataset, which involved normalizing the input features and splitting it into training and testing sets.
2. Model building: A function called 'create_model' was defined to build the ANN model. This function takes two arguments, 'optimizer' and 'neurons', which control the optimization algorithm and the number of neurons in the hidden layers, respectively. The model consisted of three layers: an input layer with a specified number of neurons and a ReLU activation function, a hidden layer with half the number of neurons of the input layer and a ReLU activation function, and an output layer with a single neuron and a sigmoid activation function. The model was compiled using binary_crossentropy loss and the given optimizer, with 'accuracy' as the evaluation metric.

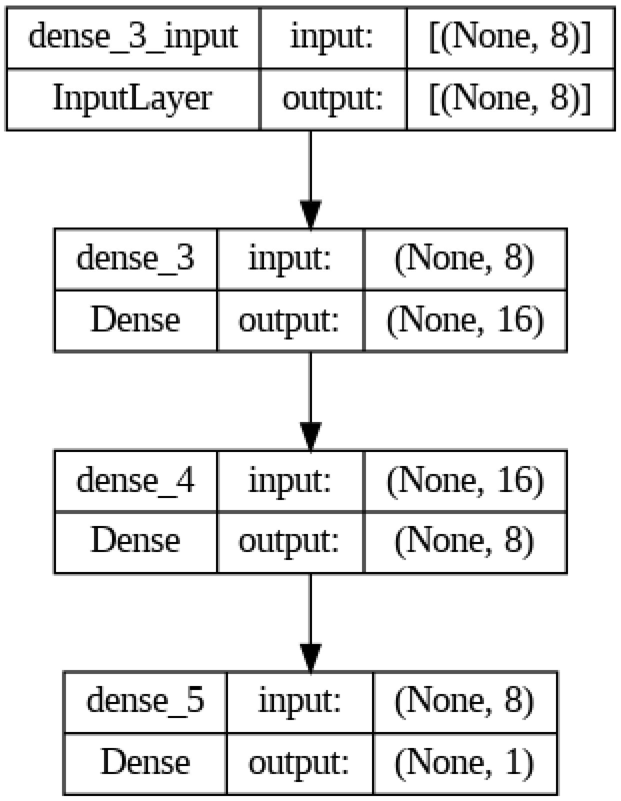

Fig 2: Our ANN Model Structure

3. Hyperparameter Optimization: To find the best combination of hyperparameters for the model, a RandomizedSearchCV object was created, wrapping the KerasClassifier with the 'create_model' function. A parameter grid was defined, containing different options for the optimizer, the number of neurons, the batch size, and the number of training epochs. RandomizedSearchCV was then used to perform a randomized search over the hyperparameter space, using 5-fold cross-validation and evaluating 10 random combinations of hyperparameters. RandomizedSearchCV was used in this project because it provides an efficient and effective approach to hyperparameter tuning. Instead of performing an exhaustive search of all possible combinations of hyperparameters like GridSearchCV, RandomizedSearchCV randomly samples a subset of the hyperparameter

space, allowing it to explore a larger range of values with fewer iterations, potentially leading to improved performance compared to the baseline models.

4. Model training and evaluation: Once the best combination of hyperparameters was found using RandomizedSearchCV, the best model was created using the 'create_model' function with these optimal hyperparameters. The model was trained on the training data using the specified number of epochs and batch size. The performance of the model was then evaluated on the testing data by calculating various evaluation metrics such as accuracy, precision, recall, F1-score, confusion matrix, sensitivity, and specificity.
5. Model comparison: To compare the performance of the optimized ANN model with other machine learning algorithms, additional models were trained and evaluated, including Decision Tree, Logistic Regression, k-Nearest Neighbors (kNN), and Random Forest. The same evaluation metrics were calculated for each model, allowing for a fair comparison of their performance on the given diabetes prediction task.
6. Visualization and analysis: To gain further insights into the performance of the optimized ANN model, additional visualizations were generated, such as the model architecture diagram and the Receiver Operating Characteristic (ROC) curve. These visualizations helped to better understand the trade-offs between true positive and false positive rates and provided a comprehensive view of the model's predictive capabilities.

Github Link of Project: https://github.com/dmj0013/CS539_Group4.git

Results

Shown in Table 2 below is the confusion matrix that was found for the ANN that we generated. The top left value is the true negative rate which is the number of times the model correctly predicted that a patient did not have diabetes. The bottom right is the true positive rate which is the number of times the model predicted that a patient did have diabetes. The bottom left is the false negative rate which is the number of times the model predicted the patient did not have diabetes when they actually did. The top right is the false positive rate which is the number of times the model predicted the patient did have diabetes when they actually did not.

Table 2: Confusion Matrix for the artificial neural network model that was generated.

Confusion Matrix	Predicted 1	Predicted 0
Actual 1	89	18
Actual 0	14	33

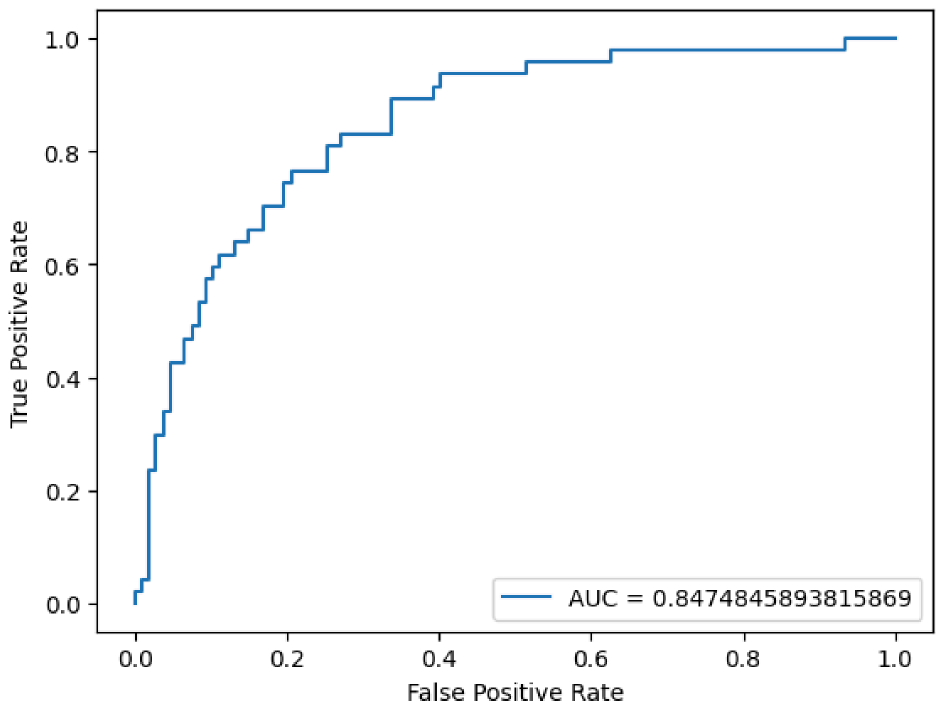

Figure 2: ROC plot of ANN model with AUC value generated

Figure 2 shows the ROC curve along with the AUC value that was generated from the ANN model that we created. It shows the true positive rate plotted against the false positive rate. The AUC value is the area under the ROC curve.

Table 3: Evaluations metrics of the artificial neural network model used in this dataset

Model Evaluation	
Metric	**Score**
Accuracy	0.7922
Precision	0.6470
Recall	0.7021
F-1 Score	0.6734
Specificity	0.8318
Sensitivity	0.7021

Table 3 shows how our model performed when classifying the data. This evaluation was performed on the testing set of our data. Using the model evaluation functions, we were able to find the accuracy, precision, recall, F-1, specificity, and sensitivity scores of our model.

Table 4: Comparison of the ROC-AUC value computed from the model we generated versus the ROC-AUC values from baseline methods and the state of the art method.

Comparison to Other Methods found in Related Works	
Model	**ROC-AUC**
Logistic Regression	0.8409
KNN	0.7910
SVC	0.8355
XGBoost	0.8015
State of the art ANN	0.9579
Our ANN	0.8475

Table 4 shows the ROC-AUC values that were computed from the baseline methods as well as the state of the art method that was found in related works. The baseline methods computed ROC-AUC values for logistic regression, K-nearest-neighbors, support-vector-classifier, and extreme-gradient boosting models. The ANN we generated has an ROC-AUC value greater than all of the other baseline models we compared it to. Our model also performed significantly worse than the state-of-the-art ANN model that was used. Our ANN's ROC-AUC score was .1105 points below the state-of-the-art model.

Table 5: Comparison of model evaluation metrics between our ANN classifier and other classifiers we generated

Model Evaluation Comparison to Models We Generated					
Metric	**Decision Tree**	**Logistic Regression**	**K-nearest neighbors**	**Random Forest**	**ANN**
Accuracy	0.7597	0.8247	0.8052	0.7857	0.7922
Precision	0.5962	0.7632	0.6977	0.6591	0.6470
Recall	0.6596	0.617	0.6383	0.617	0.7021
F-1 Score	0.6263	0.6824	0.6667	0.6374	0.6734
Specificity	0.8037	0.9159	0.8785	0.8598	0.8318
Sensitivity	0.6596	0.617	0.6383	0.617	0.7021
Sum	4.1051	4.4202	4.3247	4.176	4.25117

Table 5 shows the comparison between the evaluation of ANN model that we generated versus the other classifier models that we used. We tested the performance of our ANN against the decision tree logistic regression, KNN, and random forest classifiers. To make the model performance easier to compare, the sum of each metric was totaled and tabulated in the last row. Comparisons of the sum can be made to easily compare model performances. Evidently, the ANN we generated performed better than the decision tree and random forest models but worse than the logistic regression and KNN models.

Discussion

In this report we generated an artificial neural network model to classify patients with diabetes. We evaluated the performance of this model and compared it to related projects that have been done with this data set. We compared the performance of this model with other classification methods that we generated to see how well our model performed.

When it comes to improving the performance of artificial neural network models, the number of layers and nodes in the network should be the main consideration. The ANN model that we generated has one input layer with a specified number of neurons and a ReLU activation function, a hidden layer with half the number of neurons of the input layer and a ReLU activation function, and an output layer with a single neuron and a sigmoid activation function. In order to improve this model we would need to either increase the number of hidden layers in the model or increase the number of neurons in the layers. This would increase the accuracy and other evaluation parameters of the model. However, there is the risk of overfitting when it comes to increasing the neural network's layers and neurons. This causes the model to fit the training data too closely so it wont generalize as well on the testing data. One solution to this moving forward would be to utilize regularization to combat overfitting of our model. L1 and L2 regularization are two techniques that can be used to reduce overfitting. This is a solution to improve our model after adding more layers and neurons.

References

[1] Mathchi, "Diabetes Data Set," Kaggle, n.d. [Online]. Available: https://www.kaggle.com/datasets/mathchi/diabetes-data-set. [Accessed March 3, 2023].

[2] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S. Johannes, "Using the ADAP Learning Algorithm to Forecast the Onset of Diabetes Mellitus," in Proceedings of the Annual Symposium on Computer Application in Medical Care, 1988, pp. 261-265.

[3] American Diabetes Association, "About Diabetes," [Online]. Available: https://diabetes.org/about-us/statistics/about-diabetes. [Accessed March 3, 2023].

[4] Centers for Disease Control and Prevention, "Statistics Report | Data & Statistics | Diabetes | CDC," [Online]. Available: https://www.cdc.gov/diabetes/data/statistics-report/index.html. [Accessed March 3, 2023].

[5] International Diabetes Federation, "Diabetes around the world in 2021," [Online]. Available: https://diabetesatlas.org. [Accessed March 3, 2023].

[6] K. Collins, "Civica Rx aims to sell insulin for $30 a vial," CNN, March 3, 2022. [Online]. Available: https://www.cnn.com/2022/03/03/politics/civica-insulin-affordable-drug/index.html. [Accessed March 3, 2023].

[7] L. Fregoso-Aparicio, J. Noguez, L. Montesinos, et al., "Machine learning and deep learning predictive models for type 2 diabetes: a systematic review," Diabetology & Metabolic Syndrome, vol. 13, no. 148, 2021. doi: 10.1186/s13098-021-00767-9.

[8] T. Sharma and M. Shah, "A comprehensive review of machine learning techniques on diabetes detection," Visual Computing for Industry, Biomedicine, and Art, vol. 4, no. 30, 2021. doi: 10.1186/s42492-021-00097-7.