Early Stage Diabetes Risk Prediction

Mansi Goel, Kartikey Garg

ABSTRACT Diabetes is a chronic, metabolic disease characterized by elevated levels of blood glucose (or blood sugar), which leads over time to serious damage to the heart, blood vessels, eyes, kidneys and nerves. About 422 million people worldwide have diabetes, the majority living in low-and middle-income countries, and 1.6 million deaths are directly attributed to diabetes each year. Both the number of cases and the prevalence of diabetes have been steadily increasing over the past few decades. Classification techniques have been well accepted by researchers for risk prediction model of the disease. To predict the likelihood of having diabetes requires a dataset, which contains the data of newly diabetic or would be diabetic patient. In this work, we have used such a dataset of 520 instances, which has been collected using direct questionnaires from the patients of Sylhet Diabetes Hospital in Sylhet, Bangladesh. We have analysed the dataset with Logistic Regression, Random Forest, ANN and PNN. Random forest has been found having best accuracy on this dataset.

I. INTRODUCTION

A. DATASET DESCRIPTION

This dataset contains reports of diabetes-related symptoms of 520 persons. It includes data about peoples including symptoms that may cause diabetes. The variable 'Class' is the response variable; 'positive' corresponds to a person being at risk for diabetes, 'negative' corresponds to a person not being at risk for diabetes.

Table 1 Description of dataset

	Number of attributes	Number of instances
Diabetes symptom dataset	16	520

Table 2 Description of attribute

Attributes	Values
Age	1.20–35, 2.36–45, 3.46–55,4.56–65, 6.above 65
Sex	1.Male, 2.Female
Polyuria	1.Yes, 2.No.
Polydipsia	1.Yes, 2.No.
Sudden weight loss	1.Yes, 2.No.
Weakness	1.Yes, 2.No.
Polyphagia	1.Yes, 2.No.
Genital thrush	1.Yes, 2.No.
Visual blurring	1.Yes, 2.No.
Itching	1.Yes, 2.No.
Irritability	1.Yes, 2.No.
Delayed healing	1.Yes, 2.No.
Partial paresis	1.Yes, 2.No.
Muscle stiffness	1.Yes, 2.No.
Alopecia	1.Yes, 2.No.
Obesity	1.Yes, 2.No.
Class	1.Positive, 2.Negative.

B. MATHEMATICS BEHIND MODELS USED

1. RANDOM FOREST

Random forest is an improvement to the process of bagging.(Bagging: Taking repeated samples from the (single) training data set of the same size. In this approach we generate B different training data sets. We then train our method on the bth training set in order to get a single prediction, and finally average all the predictions). As in bagging, we build a number of decision trees on training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. A fresh sample of m predictors is taken at each split, and typically we choose $m \approx \sqrt{p}$ i.e., the number of predictors considered at each split is approximately equal to the square root of the total number of predictors.

$$h(x) = \frac{1}{B}\sum_{j=1}^{B} h_j(x)$$

2. SUPPORT VECTOR MACHINE

Developing a classifier that almost separates classes is done by the ***soft margin formula.*** We allow some observations to be on the incorrect side of the margin, or even on the incorrect side of the hyperplane. The classification rule here is to classify a test observation depending on which side of a hyperplane it lies. The hyperplane is chosen so that it correctly classifies most of the training observations, but may misclassify a few observations. SVM is further an extension of the support vector classifier that results from enlarging the feature space in a specific way, using *“**kernels**”*. When the support vector classifier is combined with a non-linear kernel, the resulting classifier is known as a support vector machine.

3. ARTIFICIAL NEURAL NETWORK

Considering we have data and would like to apply binary classification to get the desired output. Take a sample having features as X1, X2, and these features will be operated over a set of processes to predict the outcome. Each feature is associated with a weight, where X1, X2 as features and W1, W2 as weights. These are served as input to a neuron. *A neuron performs both functions. a) Summation b)Activation.* In the summation, all features are multiplied by their weights and bias are summed up. (Y=W1X1+W2X2+b).This summed function is applied over an Activation function(there are various activation functions such as tanh, sigmoidal, RELU etc).
The output from this neuron is multiplied with the weight W3 and supplied as input to the output layer. The same process happens in each neuron, but we vary the activation functions in hidden layer neurons, not in the output layer. The activation function for output layer is either sigmoid or SoftMax. The weights and biases are updated using backpropagation, wherein the differential of error function is calculated with respect to weights and biases to find out the corresponding changes bringing the error closer to zero. After iterating through a series of forward propagation and back propagation we converge to the approximated solution with some pre-decided margin of error.

4. PROBABALISTIC NEURAL NETWORK

A **probabilistic neural network (PNN)** is a neural network, which is widely used in classification. Consider the independent variables $\mathbf{X_1,\dots,X_d}$ and Y = 1,2,…,k is the response variable having k classes. Here, we devise a method by means of which we can compute;

$$p(j) = P(Y = j|X = x)$$

j is chosen for which p(j) is the highest. The predicted label for observation X will be the output of;

$$argmax_{1\le j\le k}p(j)$$

PNNs have a structure comprising of 4 layers namely Input Layer, Pattern Layer, Summation Layer and the Output Layer. Each neuron in the input layer represents a predictor variable. The pattern layer is designed to contain one neuron for each training case available and the neurons are split into the two classes. The PNN executes a training case by first presenting it to all pattern layer neurons. Each neuron in the pattern layer computes a distance measure between the presented input vector and the training example represented by that pattern neuron. The summation layer contains one neuron for each class j. These neurons compute p(j) and lastly the output layer has a single neuron which finally computes $argmax_{1\le j\le k}p(j)$.

II. EXPERIMENTAL ANALYSIS

After loading the dataset into the environment, we performed some exploratory data analysis.

On printing the first few rows of the dataset we observed that the variable values are non-numerical. The target variable ‘class’ was also a variable with two classes- ‘Positive’ and ‘Negative’.

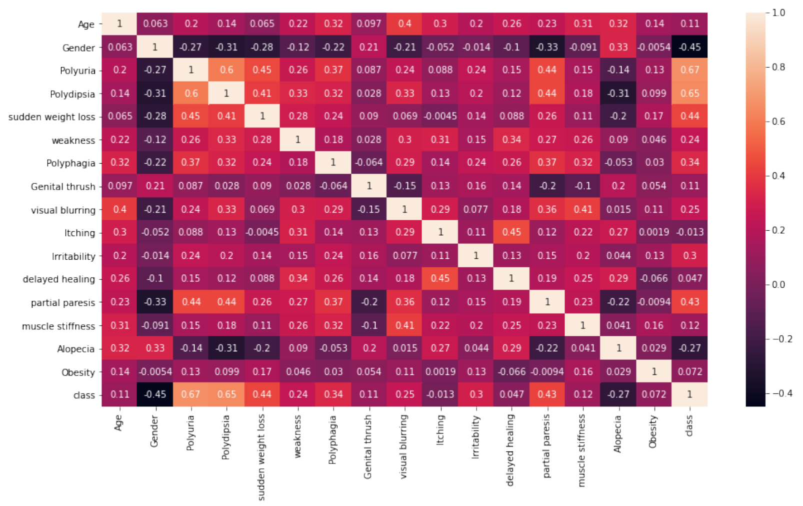
On running the following command on all variables, we were able to convert the values of all the variables to numerical ones.

```
Data['Gender'] = Data['Gender'].map({'Male':1,'Female':0})
```

Now, our dataset is in a form which can be fed into a Machine Learning or Deep Learning model.

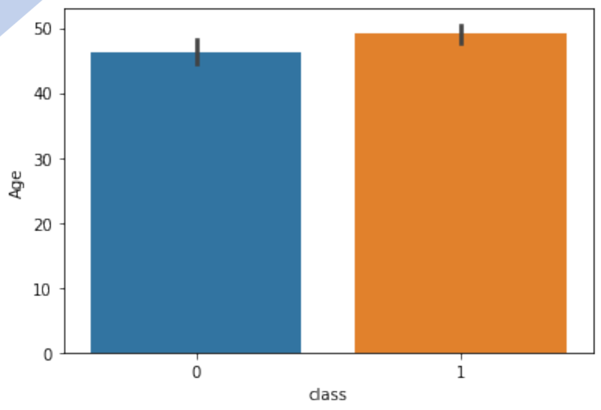
	Age	Gender	Polyuria	Polydipsia	sudden weight loss	weakness	Polyphagia	Genital thrush	visual blurring	Itching	Irritability	delayed healing	partial paresis	muscle stiffness
0	40	1	0	1	0	1	0	0	0	1	0	1	0	1
1	58	1	0	0	0	1	0	0	1	0	0	0	1	0
2	41	1	1	0	0	1	1	0	0	1	0	1	0	1
3	45	1	0	0	1	1	1	1	0	1	0	1	0	0
4	60	1	1	1	1	1	1	0	1	1	1	1	1	1

Before moving ahead with building our models, we looked at some plots of the variables involved to get a better idea of how the dataset is distributed and how various variables are related to each other.

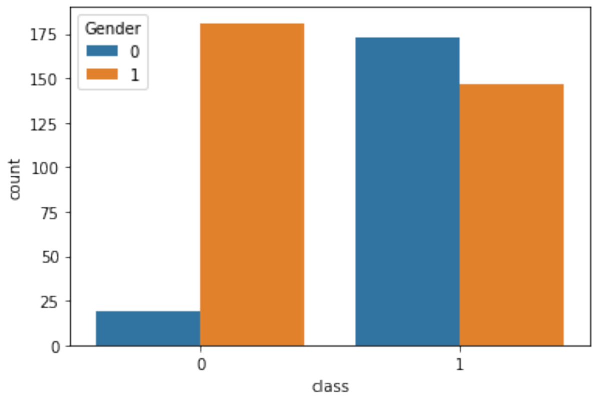


	Age	Gender	Polyuria	Polydipsia	sudden weight loss	weakness	Polyphagia	Genital thrush	visual blurring	Itching	Irritability	delayed healing	partial paresis	muscle stiffness	Alopecia	Obesity	class
Age	1	0.063	0.2	0.14	0.065	0.22	0.32	0.097	0.4	0.3	0.2	0.26	0.23	0.31	0.32	0.14	0.11
Gender	0.063	1	-0.27	-0.31	-0.28	-0.12	-0.22	0.21	-0.21	-0.052	-0.014	-0.1	-0.33	-0.091	0.33	-0.0054	-0.45
Polyuria	0.2	-0.27	1	0.6	0.45	0.26	0.37	0.087	0.24	0.088	0.24	0.15	0.44	0.15	-0.14	0.13	0.67
Polydipsia	0.14	-0.31	0.6	1	0.41	0.33	0.32	0.028	0.33	0.13	0.2	0.12	0.44	0.18	-0.31	0.099	0.65
sudden weight loss	0.065	-0.28	0.45	0.41	1	0.28	0.24	0.09	0.069	-0.0045	0.14	0.088	0.26	0.11	-0.2	0.17	0.44
weakness	0.22	-0.12	0.26	0.33	0.28	1	0.18	0.028	0.3	0.31	0.15	0.34	0.27	0.26	0.09	0.046	0.24
Polyphagia	0.32	-0.22	0.37	0.32	0.24	0.18	1	-0.064	0.29	0.14	0.24	0.26	0.37	0.32	-0.053	0.03	0.34
Genital thrush	0.097	0.21	0.087	0.028	0.09	0.028	-0.064	1	-0.15	0.13	0.16	0.14	-0.2	-0.1	0.2	0.054	0.11
visual blurring	0.4	-0.21	0.24	0.33	0.069	0.3	0.29	-0.15	1	0.29	0.077	0.18	0.36	0.41	0.015	0.11	0.25
Itching	0.3	-0.052	0.088	0.13	-0.0045	0.31	0.14	0.13	0.29	1	0.11	0.45	0.12	0.22	0.27	0.0019	-0.013
Irritability	0.2	-0.014	0.24	0.2	0.14	0.15	0.24	0.16	0.077	0.11	1	0.13	0.15	0.2	0.044	0.13	0.3
delayed healing	0.26	-0.1	0.15	0.12	0.088	0.34	0.26	0.14	0.18	0.45	0.13	1	0.19	0.25	0.29	-0.066	0.047
partial paresis	0.23	-0.33	0.44	0.44	0.26	0.27	0.37	-0.2	0.36	0.12	0.15	0.19	1	0.23	-0.22	-0.0094	0.43
muscle stiffness	0.31	-0.091	0.15	0.18	0.11	0.26	0.32	-0.1	0.41	0.22	0.2	0.25	0.23	1	0.041	0.16	0.12
Alopecia	0.32	0.33	-0.14	-0.31	-0.2	0.09	-0.053	0.2	0.015	0.27	0.044	0.29	-0.22	0.041	1	0.029	-0.27
Obesity	0.14	-0.0054	0.13	0.099	0.17	0.046	0.03	0.054	0.11	0.0019	0.13	-0.066	-0.0094	0.16	0.029	1	0.072
class	0.11	-0.45	0.67	0.65	0.44	0.24	0.34	0.11	0.25	-0.013	0.3	0.047	0.43	0.12	-0.27	0.072	1

Interesting point to note here: Gender is negatively correlated with Class i.e. It can argued that males are less likely to have diabetes as compared to females.

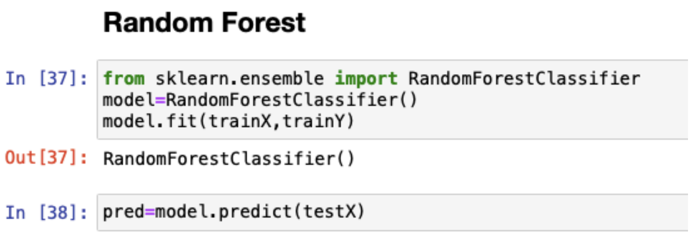


It is evident from the above plot that there's no significant relationship between age and diabetes.



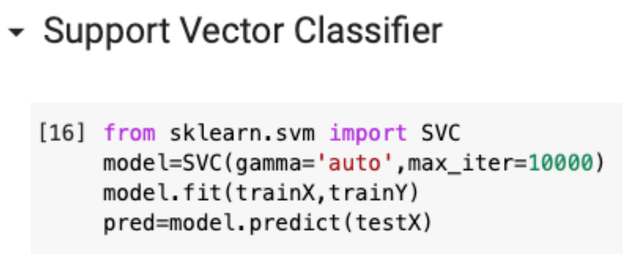
It further strengthens the argument that females are more vulnerable to diabetes as compared to males

Now, moving on to building models and making predictions, we first split the dataset into training and testing data keeping a 80-20 split and then fitted ML models – Random Forest and Support Vector Machine and then moved on to DL models – Artificial Neural Network and Probabilistic Neural Network.



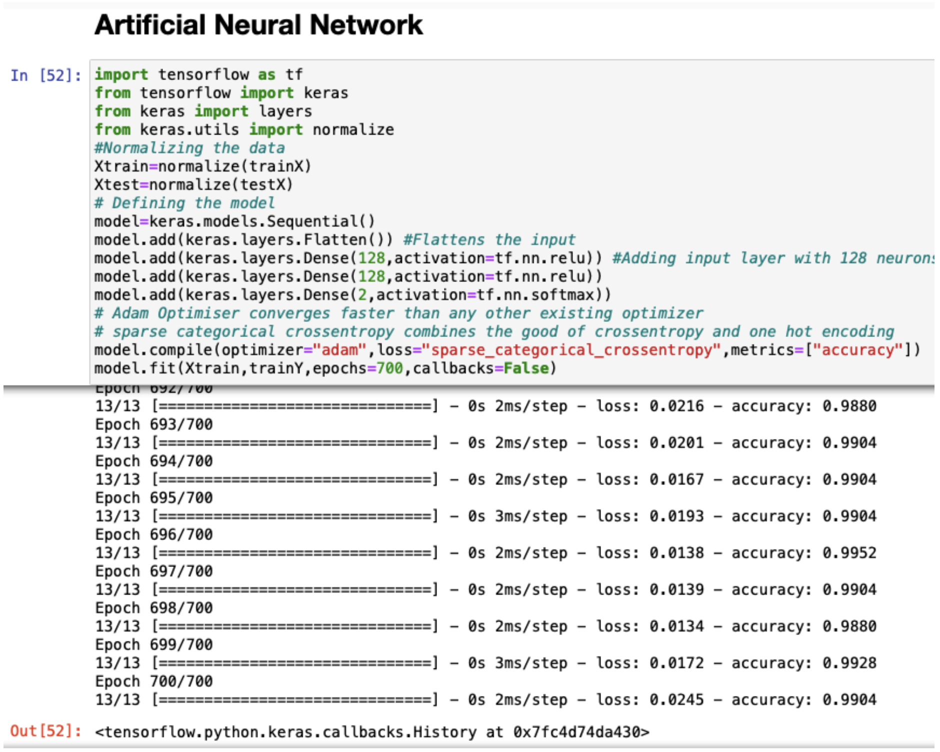
Random Forest

```
In [37]: from sklearn.ensemble import RandomForestClassifier
         model=RandomForestClassifier()
         model.fit(trainX,trainY)
Out[37]: RandomForestClassifier()

In [38]: pred=model.predict(testX)
```



Support Vector Classifier

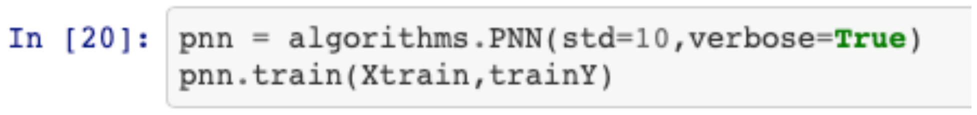
```
[16] from sklearn.svm import SVC
     model=SVC(gamma='auto',max_iter=10000)
     model.fit(trainX,trainY)
     pred=model.predict(testX)
```



Artificial Neural Network

```
In [52]: import tensorflow as tf
         from tensorflow import keras
         from keras import layers
         from keras.utils import normalize
         #Normalizing the data
         Xtrain=normalize(trainX)
         Xtest=normalize(testX)
         # Defining the model
         model=keras.models.Sequential()
         model.add(keras.layers.Flatten()) #Flattens the input
         model.add(keras.layers.Dense(128,activation=tf.nn.relu)) #Adding input layer with 128 neuron:
         model.add(keras.layers.Dense(128,activation=tf.nn.relu))
         model.add(keras.layers.Dense(2,activation=tf.nn.softmax))
         # Adam Optimiser converges faster than any other existing optimizer
         # sparse categorical crossentropy combines the good of crossentropy and one hot encoding
         model.compile(optimizer="adam",loss="sparse_categorical_crossentropy",metrics=["accuracy"])
         model.fit(Xtrain,trainY,epochs=700,callbacks=False)
         Epoch 692/700
         13/13 [==============================] - 0s 2ms/step - loss: 0.0216 - accuracy: 0.9880
         Epoch 693/700
         13/13 [==============================] - 0s 2ms/step - loss: 0.0201 - accuracy: 0.9904
         Epoch 694/700
         13/13 [==============================] - 0s 2ms/step - loss: 0.0167 - accuracy: 0.9904
         Epoch 695/700
         13/13 [==============================] - 0s 3ms/step - loss: 0.0193 - accuracy: 0.9904
         Epoch 696/700
         13/13 [==============================] - 0s 2ms/step - loss: 0.0138 - accuracy: 0.9952
         Epoch 697/700
         13/13 [==============================] - 0s 2ms/step - loss: 0.0139 - accuracy: 0.9904
         Epoch 698/700
         13/13 [==============================] - 0s 2ms/step - loss: 0.0134 - accuracy: 0.9880
         Epoch 699/700
         13/13 [==============================] - 0s 3ms/step - loss: 0.0172 - accuracy: 0.9928
         Epoch 700/700
         13/13 [==============================] - 0s 2ms/step - loss: 0.0245 - accuracy: 0.9904
Out[52]: <tensorflow.python.keras.callbacks.History at 0x7fc4d74da430>
```

Probabilistic Neural Network

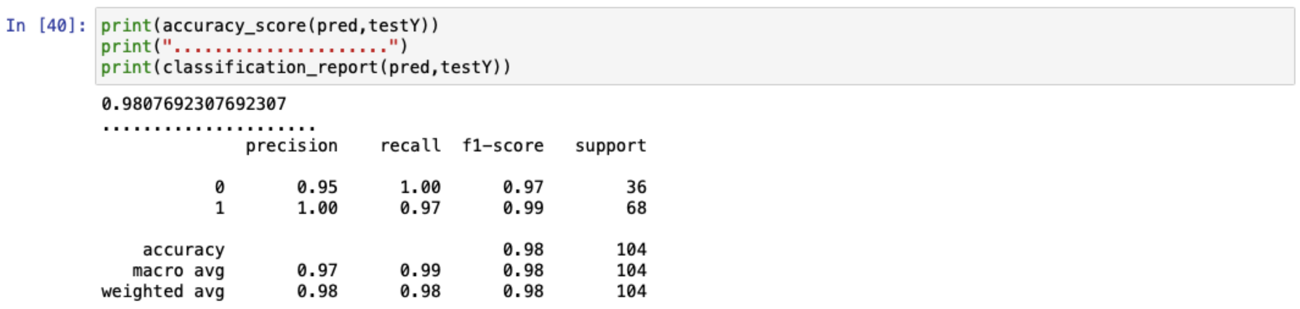


```
In [20]: pnn = algorithms.PNN(std=10,verbose=True)
         pnn.train(Xtrain,trainY)
```

III. RESULT ANALYSIS

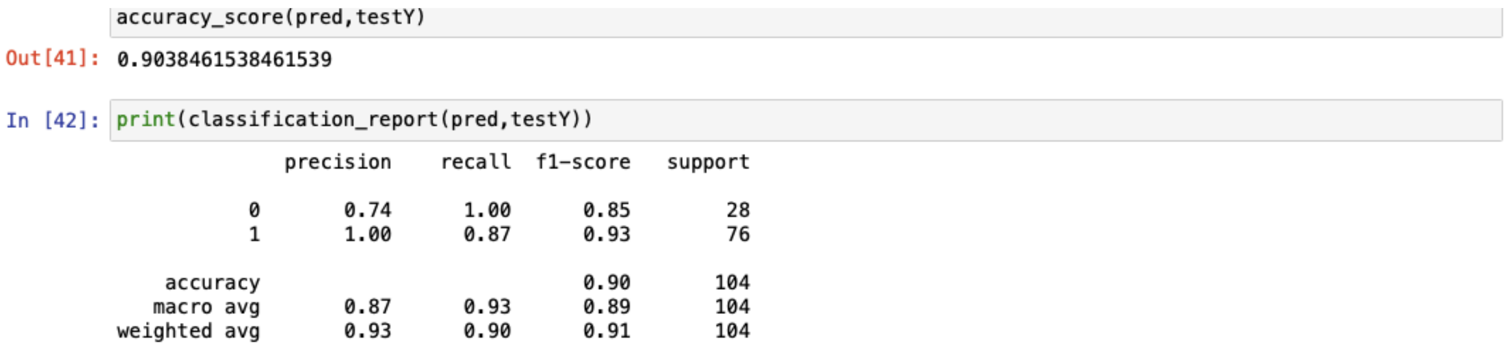
Choosing a consistent metric, the accuracy score, we made predictions on our test set. Out of the 4 models, Random Forest turned out to be the best performing model with an accuracy score of 0.981.

Random Forest



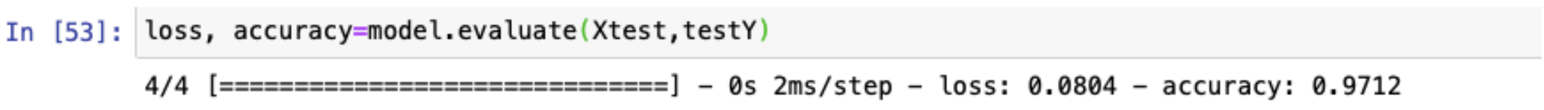
```
In [40]: print(accuracy_score(pred,testY))
         print(".....................")
         print(classification_report(pred,testY))

0.9807692307692307
.....................
              precision    recall  f1-score   support

           0       0.95      1.00      0.97        36
           1       1.00      0.97      0.99        68

    accuracy                           0.98       104
   macro avg       0.97      0.99      0.98       104
weighted avg       0.98      0.98      0.98       104
```

Support Vector Classification



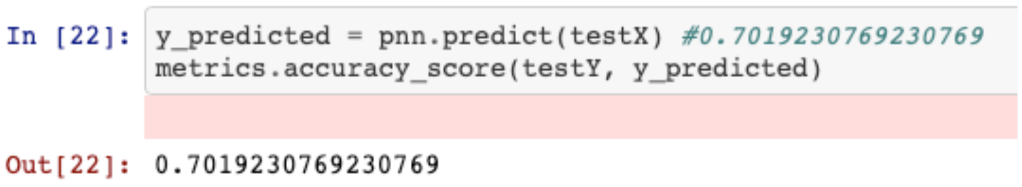
```
         accuracy_score(pred,testY)
Out[41]: 0.9038461538461539

In [42]: print(classification_report(pred,testY))

              precision    recall  f1-score   support

           0       0.74      1.00      0.85        28
           1       1.00      0.87      0.93        76

    accuracy                           0.90       104
   macro avg       0.87      0.93      0.89       104
weighted avg       0.93      0.90      0.91       104
```

ANN



```
In [53]: loss, accuracy=model.evaluate(Xtest,testY)

4/4 [==============================] - 0s 2ms/step - loss: 0.0804 - accuracy: 0.9712
```

PNN



```
In [22]: y_predicted = pnn.predict(testX) #0.7019230769230769
         metrics.accuracy_score(testY, y_predicted)

Out[22]: 0.7019230769230769
```

IV. CONCLUSION

The potentiality of diabetes is increasing among people of all age. The present study says that detection of diabetes at its early stage can play a pivotal role in treatment. Simple awareness measures such as low sugar diet, regular physical activity, and healthy lifestyle can avoid obesity. As machine learning techniques and tools are becoming more promising to predict diabetes and eventually number of patients reduce the treatment cost, its role in this medical health care is undeniable. The main contribution is to find out the best algorithm for the prediction of newly created datasets made for diabetic risk prediction. We found that the Random Forest algorithm had performed with the best accuracy on this dataset.

References

1. https://archive.ics.uci.edu/ml/datasets/Early+stage+diabetes+risk+prediction+dataset
2. https://www.kaggle.com/ishandutta/early-stage-diabetes-risk-prediction-dataset
3. https://medium.com/dev-genius/early-stage-diabetes-risk-prediction-bd42f113a20b
4. Computer Vision and Machine Intelligence in Medical Image Analysis - Mousumi Gupta, Debanjan Konar, Siddhartha Bhattacharyya, Sambhunath Biswas
5. An Introduction to Statistical Learning: With Applications in R

GitHub project link : https://github.com/kartikeypro/Early-Stage-Risk-Diabetes-Using-Deep-Learning.git