Predict diabetes using Perceptron

Ngoc Anh Do
University of Adelaide
Student ID: 1895352

Abstract

Diabetes mellitus is a common disease today that causes the death of nearly 1.5 million people each year. It is the cause of many other diseases such as blindness and heart disease. It ranks in the top 9 dangerous diseases in the world, placing a burden on the global economy and healthcare system. The research focuses on finding solutions for diabetes diagnosis using machine learning algorithms. Early diagnosis can prevent the disease from getting worse. By using Python support tools and libraries to implement such as Pytorch, Numpy, or Pandas. The model used is Multilayer Perceptron with optimization to achieve an accuracy rate of about 81%. The K-fold cross-validation technique is used to select the best hyperparameters for the model. The model contains 1 hidden layer and is the best lightweight MLP model.

I. Introduction and Background

Diabetes is a dangerous disease at the top of the world health care organization with an increase in global population. The World Health Organization (WHO) has ranked diabetes in the top 9 diseases causing death in the world's population [11]. According to data from a US study, diabetes diagnosis costs about $327 billion [1]. The number of people with diabetes has increased rapidly over the past 2 decades. WHO has shown that it increased from 108 million people in 1980 to 422 million people in 2019 and caused the deaths of about 1.5 million people [11]. Furthermore, it is the cause of diseases such as blindness, stroke, or cardiovascular disease. It leaves long-term consequences for destroying other parts of the human body. However, a large number of people go to health centers to have their blood tested for disease diagnosis. It would be time-consuming, expensive, and overload the health system, instead of using resources to treat more serious patients. To reduce these burdens, many solutions have been proposed in a variety of different fields, including combining information technology such as machine learning, deep learning, and artificial intelligence. A fairly popular technique in machine learning is the neutral network, which is based on the principle of biological neutrons. Models are formed from a network of neutrons to process input information. It can be supervised or unsupervised learning. Perceptron is a fairly popular machine learning algorithm used for binary classification problems. It is the most basic algorithm of neural network models. The research uses Multilayer Perceptron (MLP) with a hidden layer quickly and with high accuracy. The purpose of the study is to use the provided features to predict diabetes patients. The model will be applied to future patients and determine whether that person has diabetes or not. It will be an effective tool to help diagnose diseases better. Testing will be performed to optimize the accuracy of the model.

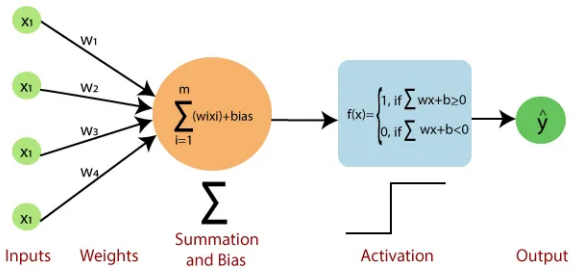

Figure 1. Perceptron algorithm

The Perceptron was introduced in 1957 by Frank Rosenblatt [10]. It is an algorithm for supervised learning of binary classification. A single-layer Perceptron uses the Activation function as a threshold to predict the class. It consists of a future vector x multiplied by weight w and adding a bias b. Activation function can ensure output is mapped with (0,1) or (1, -1)

$$y = \varphi\left(\sum_{i=1}^{n} w_i x_i + b\right) = \varphi(w^T x + b)$$

Figure 2. Perceptron algorithm formula

Multilayer perceptrons are trained on a set of input-output pairs and learn based on the correlations between those input-output pairs. Customize model parameters, weights, or biases to minimize errors. Use the training backpropagation algorithm to create weight and bias to adjust correlation with error. There are two main actions: forward and backward passes. In the forward pass, move from the input layer through the hidden layers to the output layers and make a decision on the output layer with true labels. In the backward pass use backpropagation and continuous calculation rules. Adjust parameters to bring MLP closer to error minimum (using gradient-based optimization algorithm).

The history of optimization algorithms and Gradient Descent does not begin with modern Machine Learning but rather with 19th-century France [9]. Gradient descent is one of the most popular methods for selecting the model that best fits the

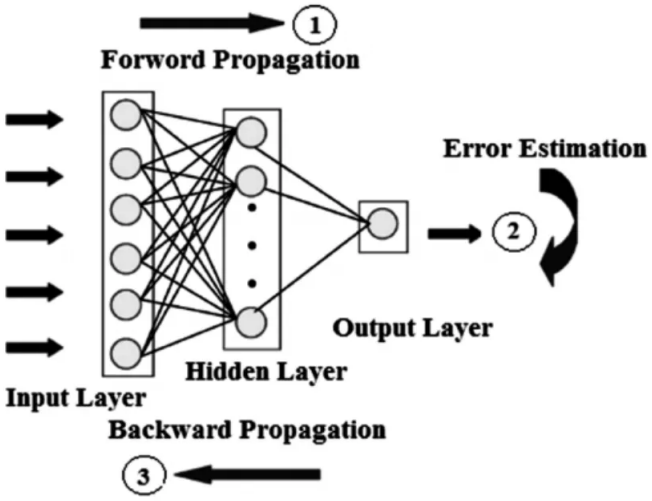

Figure 3. Multilayer Perceptron with the forward and backward passes.

$$\Delta_w(t) = -\varepsilon \frac{dE}{dw_{(t)}} + \alpha \Delta_{w(t-1)}$$

Bias, Error, Learning Rate, Gradient Current Iteration, Weight vector, Gradient Previous Iteration

Figure 4. Mathematical representation of Gradient Descent

training data. Usually, it is the model that minimizes the loss function.

It is possible to store all hidden layers of each layer for all training samples as an H matrix. The calculations are written as follows [7]:

$$\mathbf{H}^{(1)} = \phi^{(1)}\left(\mathbf{X}\mathbf{W}^{(1)\top} + \mathbf{1}\mathbf{b}^{(1)\top}\right)$$
$$\mathbf{H}^{(2)} = \phi^{(2)}\left(\mathbf{H}^{(1)}\mathbf{W}^{(2)\top} + \mathbf{1}\mathbf{b}^{(2)\top}\right)$$
$$\mathbf{Y} = \phi^{(3)}\left(\mathbf{H}^{(2)}\mathbf{W}^{(3)\top} + \mathbf{1}\mathbf{b}^{(3)\top}\right)$$

II. Related Work

Swapna G et al. [4] has researched machine learning for diabetes diagnosis with a convolutional neural network (CNN) and a long-term-short-term memory (LSTM). There is also Ram D. et al. performed logistic regression with 0.78 accuracy. Abu Naser [8] achieved 0.76 accuracy using the Naïve Bayes algorithm. Use cross-validation with 10 folds to find effective hyperparameters for models. Quan Z. et al[6]. with 0.77 when using a Decision Tree. There are also quite a few articles done using diabetes datasets and using machine learning to make diagnoses such as k-nearest Neighbor, Random Forest, AdaBoost, Naive Bayes, XGBoost. Deep learning is also used with an accuracy of about 0.82. There are many articles based on the Pima Indian Diabetes dataset to make diagnoses, they are done with many different models, it is important for the community and the world health system to avoid waste. and overload.

III. Method Description

The steps to do it are Data collection, Preprocessing data, implementing multi-layer perceptron, and optimizing the model using Grid search.

3.1. Data collection

The publicly available Pima Indians Diabetes dataset is available from the National Institute of Diabetes [5]. It includes 768 samples and contains 8 features and a class label to confirm whether the patient is diabetic (label = 1) or not (label = 0). Each feature has the following type and description:

Feature	Description
Pregnancies	Number of times pregnant
Glucose	2 hours glucose concentration test
BloodPressure	Diastolic blood pressure (mm Hg)
SkinThickness	Triceps skin-fold thickness (mm)
Insulin	Two hours of serum insulin (mu U/ml)
BMI	Body Mass Index
Diabetes PedigreeFunction	Diabetes pedigree function
Age	Age in years

Table I
Features Description

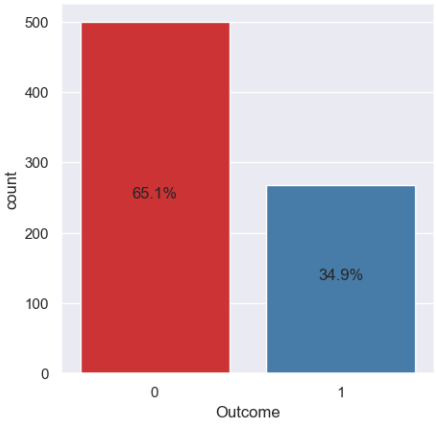

Figure 5. The distribution of labels

The picture below is the data distribution of the dataset, each feature of the two classes is displayed together in one plot, clear separation between class distributions are stronger, red color means non-diabetes and blue color diabetes:

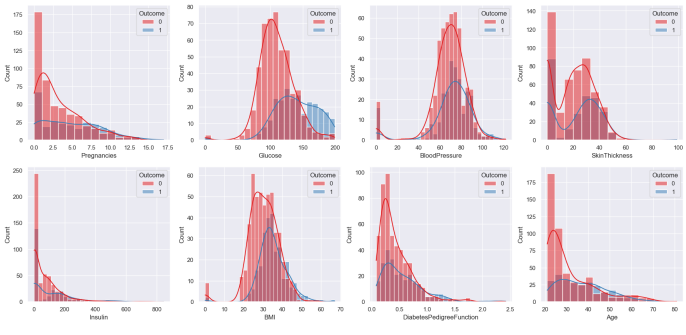

Figure 6. The distribution of features

3.2. Data pre-processing

MLP is quite sensitive to missing data. Preprocessing will have steps such as Outline rejection (O) and Missing values imputation (M). Outlier Rejection can be described as the following formula: x is the set of feature vectors in n-dimensional, Q1, Q2 and IQR are the first quartile, third quartile, and interquartile range of the feature vector respectively [9].

$$O(x) = \begin{cases} x, if Q_1 - 1.5 \times IQR \leq x \leq Q_3 + 1.5 \times IQR \\ reject, otherwise \end{cases}.$$

Missing value will be processed with mean value like the formula below [3]:

$$M(x) = \begin{cases} mean(x), if x = missed \\ x, otherwise \end{cases}.$$

The data will be separated into 3 sets: training, validation and testing with the test set being 25% of the entire data and the validation set being 25% of the training set. Perform hyper tuning parameters using Grid Search with 3 folds on both training and validation sets. After choosing the best parameters, it will be applied to the test set.

IV. Method Implementation

To implement the model we determine that the first layer has the correct number of input features, as in this dataset it will be eight input dimensions as a vector. Need an activation function after the layer, in this study, we will use ReLU, in the first layer and sigmoid function in the output layer. Sigmoid to ensure the output will be in 0 and 1. Do not select sigmoid for the hidden layer because it can lead to vanishing gradient problems in deep learning. ReLU provides better performance in both accuracy and speed. Can be divided into layers as follows:

- The first argument in the first layer will be 8 corresponding to 8 features
- The hidden layer has 8 neutrons, and the ReLU activation function will be used afterwards
- In the output layer there is a neutron, and then apply the sigmoid function.

It is shown in the code below using Pytorch:

```
def __init__(self, n_neurons=8):
    super().__init__()
    self.layer = nn.Linear(8, n_neurons)
    self.act = nn.ReLU()
    self.output = nn.Linear(n_neurons, 1)
    self.prob = nn.Sigmoid()
```

The purpose of training the model is to generate output with high accuracy, so defining a loss function to measure prediction. Here it is binary cross entropy because the problem is solving the problem of binary classification. Additionally, select optimizer an algorithm to customize model weights to produce better output. Using Adam optimizer comes with a learning rate of 0.001. Training deep learning models always has epochs and batches. Epoch runs through the entire dataset once, batch is one or more samples passed to the model, from which the gradient descent algorithm will be executed in one iteration. When all batches are exhausted, an epoch will end. This is repeated until the end of the epoch. More iterations can produce a better model. The number of errors will decrease with subsequent epochs, until convergence.

V. Experiments and Analysis

First model the baseline with a hidden neutron layer of 8, epoch of 100, and batch size of 10. Train on the train set and then apply to the validation set. We can see that the accuracy and loss of the two sets are quite close to each other. With the best validation accuracy of around 84%.

Figure 7. Model baseline 100 epoch, batch size 10 and neutrons 8

Then we perform hyper tune using Grid Search with k-fold equals 3 implemented in both the training set and validation set to change the parameters of the epoch, batch size, and neutrons in the hidden layer, to select the parameters that give the highest accuracy.

```
batch_size: [60, 80, 100],
epochs: [100, 200],
neurons: [8, 16, 32, 64, 128]
```

After doing this, we can choose the best model with the following parameters: epoch is 100, batch size is 100, and neutrons in the hidden layer are 16. Applying the model with those parameters to the test set achieves an accuracy of about 81%. The confusion matrix demonstrated the performance of the multi-layer perceptron algorithm.

- True Positives (TP)
- True negatives (TN)
- False positives (FP)
- False negatives (FN)

We can see that the accuracy of the model is calculated according to the formula below:

$$ACC = \frac{TP + TN}{TP + TN + FN + FP} = \frac{TP + TN}{P + N}$$

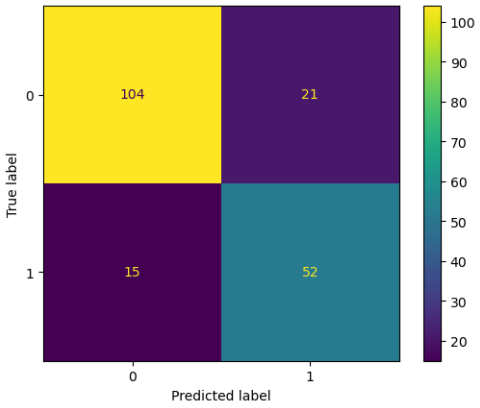

Figure 8. Confusion matrix

VI. Reflection on the Project

The study presents a model that can predict diabetes in patients. The algorithm used is a multi-layer perceptron with an accuracy of 81%. Future research will be done on more hidden layers such as 2 or 3 layers to provide more accuracy. In addition, other deep learning algorithms can be used to achieve higher accuracy. Furthermore, it is possible to use a number of methods to clean data to make the data set more accurate such as using PCA (Principal component analysis). Accurate data sets can help make training faster and more accurate when using deep learning algorithms.

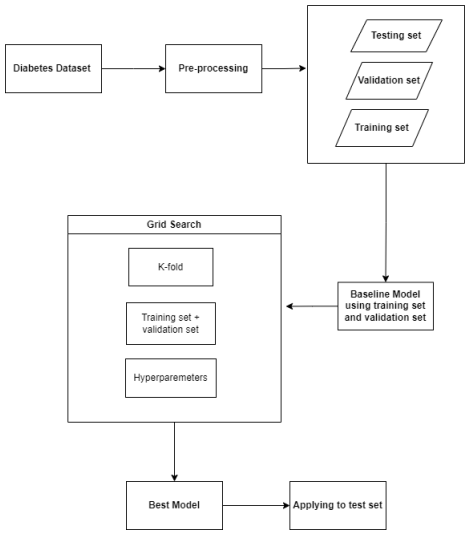

Figure 9. Diagram of a diabetes prediction.

VII. Source Code

Diabetes prediction has been achieved using perceptron model.The following link is python code using some python libraries such as Numpy, Pandas or Pytorch to complete and it show all the work in this study: *https://github.com/anhdo1896/Deep_learning_fundementals_Ass1.*

References

[1] American Diabetes Association. Economic Costs of Diabetes in the U.S. in 2017. Diabetes Care, 2018.
[2] Dewey Lonzo Whaley III. The interquartile range: Theory and estimation. PhD thesis, East Tennessee State University, 2005.
[3] Denis Cousineau and Sylvain Chartier. Outliers detection and treatment: a review. 2010.
[4] G Swapna, R Vinayakumar, and KP Soman. Diabetes detection using deep learning algorithms. 2018.
[5] PeterH Bennett, ThomasA Burch, and Max Miller. Diabetes mellitus in american (pima) indians. 1971.
[6] Quan Zou. Predicting diabetes mellitus with machine learning techniques. 2018
[7] Roger Grosse. Lecture 3: Multilayer Perceptrons. 2010.
[8] Samy S Abu-Naser. Diabetes prediction using artificial neural network. 2018.
[9] Towards Data Science. Stochastic Gradient Descent explained in real life. Carolina Bento, Jun 2, 2021.
[10] Towards Data Science. Rosenblatt's perceptron, the first modern neural network. Jean-Christophe B. Loiseau. Mar 11, 2019.
[11] World Health Organization. Diabetes, 2021.