Optimizing Diabetes Diagnosis with Single-Layer Perceptron: A Performance Evaluation on Binary Classification

Hansi Cooray
The University of Adelaide
Adelaide, South Australia, Australia
hansi.cooraywijayawarnasooriya@student.adelaide.edu.au

Abstract

Diabetes is a global health condition that affects a large number of individuals. Addiction to artificial foods can lead to diabetes even at an early age. The goal of this effort is to use neural networks to classify diabetes. In this binary classification task, a single-layer perceptron is utilized to predict the outcome. The dataset, which includes the personal data of 768 women, was acquired via the Kaggle website. This paper discusses the model's development process, findings, and potential improvements to improve the model's final performance.

1 Introduction

The World Health Organization defines diabetes as a chronic metabolic disorder that results from either insufficient or excessive insulin production by the body[2]. Although a precise treatment has not yet been discovered, early detection of the illness can greatly aid in its control and eventual cure[1]. The development of neural network models to detect diabetes in its early stages has been the subject of numerous academic articles. A paper detailing the creation of an AI diagnostic model based on a back propagation neural network in the MATLAB software platform was published by Yue Liu in 2020[3]. In addition, Hasan Temurtas and colleagues employed a multilayer neural network trained by the Levenberg-Marquardt (LM) algorithm and probabilistic neural network to diagnose Pima-diabetes in 2009. Their research revealed that neural network structures can be effectively applied to aid in the diagnosis of diabetes[4]. In this research work, a single-layer perceptron model was created by observing such essential elements from these investigations. Moreover, the report highlights the development process, training and validation of the outcomes.

2 Single Layer Perceptron

The perceptron concept was introduced to the world by Frank Rosenblatt in 1957. This concept is based on the actual neuron architecture seen in human brains. It is the most fundamental model in neural networks which also can be defined as the dense layer of a model. There are two types of perceptions: Single-layer perceptrons and multi-layer perceptrons. This study focuses on a single-layer perceptron which is capable of learning linearly separable patterns [5]. The following subsections discuss the model architecture and detail the training process of a perceptron.

2.1 Perception Architecture

A perceptron consists of four components. The input layer is the layer that consists of all the inputs/features of the particular data set. Each input neuron is assigned a weight, which quantifies the strength of the link between the input and output neurons. Also, there is a bias which is introduced to the input layer to increase the perceptron's flexibility in modelling complicated patterns in the input data. The activation function determines the perceptron's output using the weighted sum of the inputs and the bias

term [5].

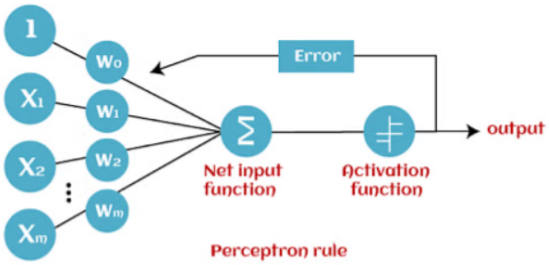

Figure 1: Perceptron Architechture

2.2 Training a Perception

Generally, a supervised learning algorithm or backpropagation is used to train a perceptron in neural networks. The weights and bias are adjusted accordingly to get a higher accuracy while training the model. Choosing the correct optimizer and activation function is crucial as they play a huge role during the training process of a perceptron. By using optimizers the learning rates and weights can be adjusted to level up the performance of a model [6]. Nevertheless, training a perceptron should be done after the data set is completely prepossessed.

3 Model Pipeline

The model pipeline steps are as follows :

1. Data Acquisition and Analysis
2. Data Preparation
3. Define a single-layer perception model
4. Model Training and Fine Tuning
5. Model evaluation and Validation

3.1 Data Acquisition and Analysis

The first stage of the model pipeline is to select a data set that best fits the purpose. The main goal of this research paper is to develop a single-layer perceptron model to diagnose diabetes. Hence, the "Pima-Indians-diabetes" data set was obtained from the Kaggle website (https://www.kaggle.com/uciml/pima-indians-diabetes-database). This data collection contains 768 women's data, containing eight attributes and one class, as shown in Table 1.

Attributes	Data Type	Feature/Labels
Pregnancies	Numerical	Feature
Glucose	Numerical	Feature
Blood Pressure	Numerical	Feature
Skin Thickness	Numerical	Feature
Insulin	Numerical	Feature
BMI	Numerical	Feature
Diabetes pedigree Function	Numerical	Feature
Age	Numerical	Feature
Outcome	Categorical	Class

Table 1: Data Structure of the Pima-Diabetes dataset

3.2 Data Preparation

The second step of the ML pipeline is to prepare the dataset for the model. During this stage, the standardized data was split into features and labels (X being the features and y being the labels). Then the features were scaled (X-Scaled) to improve gradient decent convergence [7]. The "features" and "labels" were divided into "training" and "testing" sets as the final stage of data preparation, where the training set is used to train the model while the testing test is used to test the model.

3.3 Define a single-layer perception model

The sequential() model was utilized because of the problem's binary classification nature, which aims to predict an individual's diabetes status. sequential() is the most basic model, consisting of a linear arrangement of layers that enables layer-by-layer configuration for the majority of issues.[8]. A shape of eight is inputted into the model due to the presence of eight features in total. Finally, a dense layer is added to develop a single-layer perception. Moreover to calculate the loss, the "focal-loss function" is introduced. Stochastic Gradient Descent is used to optimize the model and 'sigmoid' is used as the activation

function.

3.3.1 Focal Loss Function

An imbalance was observed in the Pima diabetes data set where one class has majority records over the other (Class 0 - 500, class 1 - 268). The Focal Loss is a modified version of cross entropy intended to address the one-stage object detection scenario in which there is a severe imbalance between foreground and background classes during training. Focal loss adds a modulating factor $(1 - p_t)^\gamma$ to the standard cross-entropy loss, where p_t is the model's estimated probability for the correct class, and γ is a focusing parameter that controls the strength of the modulation[9].

$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t) \tag{1}$$

- p_t is the predicted probability for the true class.
- α_t is a weighting factor to balance the importance of the positive and negative classes.
- $\gamma \geq 0$ is the focusing parameter. A higher value of γ reduces the loss contribution from well-classified examples, focusing more on hard-to-classified examples.

  The alpha(0.35) for the Pima-Dataset is calculated as follows :

$$\alpha = \frac{\text{Number of minority samples}}{\text{Total number of samples}} = \frac{268}{768} \tag{2}$$

The γ value is set to a low value of 1.5 to moderate down-weighting of easy examples while still letting the model learn from them.

3.4 Model Training and Fine Tuning

The model was trained by setting the epochs to 50 and the batch size to 32. However, the parameters were tuned and tested to attain better accuracy. Different learning rates of 0.0005-0.1) under two different optimizers (SGD and ADAM) were applied during the tuning process.

3.5 Model evaluation and Validation

Various parameter modifications were made when training the model, and the accuracy, F1 score, recall, and precision were recorded. The below tables summarize the findings as follows:

Table 2: Variances of Accuracies with Epochs and Batch Sizes

Epochs	Batch Size	Accuracy
100	25	61.0390
50	32	65.5844
50	25	65.5844
50	40	64.9351
30	32	65.5844

The results show that the model increases accuracy when the batch size and epochs are at a moderate value.

Table 3: Performance Metrics percentages for SGD Optimizer

Learning Rate	Acc.	F1	Precision	Recall
0.0005	65.58	62.41	51.16	80.00
0.001	65.58	64.90	51.04	89.09
0.01	43.51	55.84	38.73	100.00
0.1	35.71	52.63	35.71	100.00

The results of the SGD optimizer show that the model response is better for 0.0005 and 0.001 learning rates. Thus, results were checked for those learning rates using the ADAM optimizer to select the best out of the two optimizers.

Table 4: Performance Metrics percentages for ADAM Optimizer

Learning Rate	Acc.	F1	Precision	Recall
0.0005	58.44	61.90	46.02	94.55
0.001	58.44	61.90	46.0176	94.55
0.01	35.71	53.14	36.18	100.00

The Adam optimizer performed less accurately than the SGD optimizer. Thus, SGD was selected as the optimizer for the single-layer perception model. A better accuracy of 65.58 with a higher F1 score(64.90) was obtained with 50 epochs and 32 batch sizes along with SGD optimizer and sigmoid activation function. The loss and accuracy

curves were drawn with respect to the epochs, as illustrated in the corresponding figures.

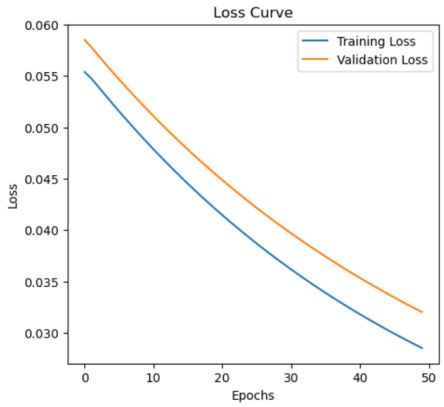

Figure 2: Loss Curve

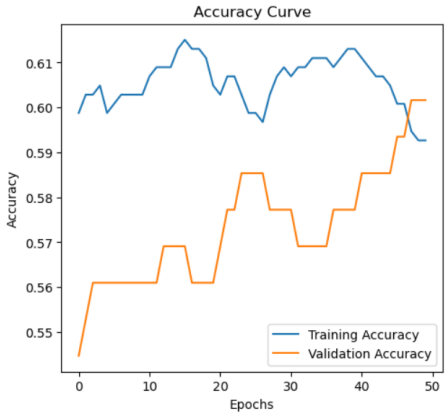

Figure 3: Accuracy Curve

4 Conclusion

In this study, the diabetes dataset was analyzed to predict the likelihood of diabetes in individuals based on various clinical and demographic features using a single-layer perceptron model. Data imbalance was handled using the Focal Loss function which gave a model accuracy of 65.58 percent with 0.001 learning rate. The model performs with a moderate accuracy and this can be elevated by implementing a multi-layer perceptron model. Moreover, effectively handling the weights of the minority class is crucial. Nevertheless training the perceptron model with a larger data set with balanced outcomes would enhance the performance of the model further.

Data Availability

The code used for the analysis is available at the following repository: https://github.com/hansi959/Single-Layer-Perceptron-Model.

References

[1] C.D. Saudek, ”Can diabetes be cured? Potential biological and mechanical approaches,” JAMA, vol. 301, no. 15, pp. 1588–1590, 2009. doi:10.1001/jama.2009.508.

[2] World Health Organization (WHO), ”Diabetes,” [Online]. Available: https://www.who.int/health-topics/diabetes.

[3] Y. Liu, ”Artificial intelligence-based neural network for the diagnosis of diabetes: Model development,” JMIR Medical Informatics, vol. 8, no. 5, p. e18682, 2020. [Online]. Available: https://medinform.jmir.org/2020/5/e18682. doi:10.2196/18682.

[4] H. Temurtas, N. Yumusak, and F. Temurtas, ”A comparative study on diabetes disease diagnosis using neural networks,” Expert Systems with Applications, vol. 36, no. 4, pp. 8610–8615, 2009. doi:10.1016/j.eswa.2008.10.032.

[5] J. Singh and R. Banerjee, ”A study on single and multi-layer perceptron neural network,” in Proc. 2019 3rd Int. Conf. Comput. Methodologies Commun. (ICCMC), Erode, India, 2019, pp. 35–40. doi:10.1109/ICCMC.2019.8819775.

[6] M. Banoula, ”What the hell is perceptron: A beginner’s guide for perceptron,” Towards Data Science, 2024. [Online]. Available:

https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53.

[7] Punyakeerthi, B.L., 2024. Understanding Feature Scaling in Machine Learning. Towards Data Science. Available at: https://medium.com/@punya8147-26846/understanding-feature-scaling-in-machine-learning-fe2ea8933b6

[8] Databricks, "Keras model," [Online]. Available: https://www.databricks.com/glossary/keras-model.

[9] Lin, T., Goyal, P., Girshick, R., He, K. and Dollár, P., 2017. Focal Loss for Dense Object Detection. arXiv preprint arXiv:1708.02002.