Predicting Diabetes from Health metrics using Perceptron model

Thi Thu Tuyen, Tran
University of Adelaide
Adelaide, SA 5005 Australia.
thutuyen1410@gmail.com

Abstract

This paper is to implement Perceptron (commonly known as dense layer neural network) for predicting diabetes, using the pre-processing Pima Indians diabetes dataset. Several experiments were implemented on the given dataset to get the insight how the model is impacted and then select the best hyperparameter for the model on diabetes dataset. Hyperparameters tuning is necessary for achieving optimal model performance in deep learning applications. Finally, the multilayers Perceptron model of 3 hidden layers, learning rate of 0.0001, with Adam optimizer was obtained. The model discriminates well between diabetic and non-diabetic cases. However, it tends to be overfitting and the given data is also small and specific, which may lead to bias on the outcome. Multilayers Perceptron model can serve as a valuable tool for diabetes diagnosis, with the potential for further optimization and adaptation to other healthcare applications. However, extending research is vital for mitigate some limitations.

1. Introduction

Machine learning has emerged as a powerful tool for medical diagnosis. This paper implements deep learning model, specifically Perceptron to classify if a patient has diabetes or not.

The paper aims to develop a Perceptron model that can predict the presence or absence of diabetes based on a set of input features (including the number of pregnancies the patient has, BMI, insulin level, age, etc). The dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. It is available on Kaggle webpage. However, the pre-processing data from CSIE webpage was used.

The key algorithm is a feedforward neural network (Perceptron model). It comprises multiple layers of interconnected artificial neurons, each performing weighted computations and applying activation functions. Various hyperparameters were considered to find the best model. Besides, visualizations were also provided to support for analysing the training, optimisation process.

2. Method

The feedforward neural network was used in this paper, consists of several layers of interconnected artificial neurons. Each neuron processes input data, applies a weighted sum, adds a bias term, and passes the result through an activation function. Specifically:

Input layer: the input layer has neurons equal to the number of features in the dataset, which are eight features representing patient health metrics.

Hidden layers: layers with numbers of neurons of 16, 32, 64, 128 were included in a list. A small experiment will be implemented to find the best number of neurons for the model. Rectified Linear Unit (ReLU) activation functions are applied to these layers.

Output layer: The output layer consists of a single neuron with a sigmoid activation function. This sigmoid activation function is used for binary classification problems like the case of this paper - diabetes prediction. Its output is a probability, representing the likelihood of a positive outcome.

The model is compiled with binary cross-entropy loss, which is suitable for binary classification problems.

At first, Adam optimizer with learning rate of 0.001 was used for training and experiment process. After that, the final model will be built up based on the best hyperparameter from the experiments.

The neural network model was trained through several steps:

Pre-processing: the input data was pre-processed, which means it was cleaned, scaled. Response variable with labels "-1" were changed into 0. After that, the dataset was split into training, validation and testing sets. The training set is used to train the model, the validation set is used to monitor its performance during training, and the testing set is used to evaluate the final model's performance.

Training: the model performs forward passes to make prediction on the training data and computes the loss. Then, it performs backward passes (backpropagation) to update the weights and biases in a way that minimizes the loss. For training and experiment processes, an epoch (iteration) of 50 was applied. In each epoch, the entire training dataset is passed through the network, and the model's parameters are updated. The validation set is used to monitor the model's performance and prevent overfitting.

3. Experimental Analysis

A series of experiments have been conducted to evaluate the performance of the model. These experiments aim to find out the hyperparameters that affect the model's accuracy and behaviour. Then the final model was built on the best hyperparameters that were obtained from these experiments.

Experiment 1: Varying the number of hidden layers.

The goal here is to investigate how the number of hidden layers impacts its performance. Generally, as number of hidden layers increases, model becomes more expressive, but there is a risk of overfitting. With this experiment, the output shows that 3 hidden layers gave the best validation accuracy. In contrast, 1, 2 and 4 layers were not that good. The result suggests that the model with 3 hidden layers achieves competitive accuracy without the risk of overfitting, make it a suitable choice for this paper's problem.

Experiment 2: Varying learning rates.

Learning rate is a critical hyperparameter that determines the step size during weight updates. This experiment helps to identify the optimal rate for convergence. A list of learning rate (including 1, 0.1, 0.01, 0.001, 0.0001) was tested. Validation accuracy of the model using these learning rates was recorded and compared to each other. The learning rate of 0.0001 stands out as the most effective, striking a balance convergence speed and stability.

Experiment 3: Testing different optimizers.

Different optimizers have distinct convergence characteristics. This experiment explores how such SGD, Adam, RMSprop, and Nadam affects the model. Similarly, the performance of models with these optimizers was recorded and compared to each other. RMSprop optimizer performed well across these optimizers. It offered stable convergence and high accuracy.

Experiment 4: Final model evaluation.

This experiment tests the performance of final model with selected hyperparameters. The final model was built with 3 hidden layers, learning rate of 0.001, RMSprop optimizer on training set, with iteration of 100. Then, it was applied on testing set to test its accuracy metric.

Confusion mat shows that true positive was quite high, which means the model can effectively recognize positive cases. However, true negative and false negative rate were not much different, indicating that the model may fail to recognize non-diabetes cases.

ROC curve and ROC_AUC score suggest that the model has a reasonably good ability to discriminate between positive and negative cases.

4. Conclusion

The experiments reveal that careful selection of hyperparameters, and model architecture is vital for achieving accurate classification model, specifically for this diabetes prediction problem. Hyperparameters tuning is crucial for achieving optimal model performance in deep learning applications. The multilayers Perceptron model in this paper can serve as a valuable tool for medical diagnosis, with the potential for further optimization and adaptation to other healthcare applications.

5. Limitations

While the multilayers Perceptron model in this paper support the claim that Deep Learning model can achieve good performance on health metrics, it still has some limitations. Particularly, in this given diabetes dataset, accuracy metric on training set is quite higher compared to testing set. The model classifies positive cases better than negative cases. These may suggest that the model tends to be overfitting.

The given data set is also quite small and specific to a group of people (the Pima Indian population near Phoenix, Arizona, USA), which may lead to bias on the outcome.

Therefore, the model should be trained and tested on larger dataset with variety of group of people. And further research is needed to apply more techniques (such as dropout, regularization) to mitigate overfitting problem.

References

Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. *In Proceedings of the Symposium on Computer Applications and Medical Care* (pp. 261--265). IEEE Computer Society Press.

Computer Science and Information Engineering n.d,, *LIBSVM Data: Classification*, CSIE, viewed 25 Sep 2023, <https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html>.

Kaggle n.d., Pima Indians Diabetes Database, Kaggle, viewed 25 Sep 2023, <https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database>.