Predict Diabetes using a Single-Layer Perceptron (SLP)

Nagadevi Nimeesha Nidadavolu
University of Adelaide
Adelaide, South Australia
a1772276@adelaide.edu.au

Abstract

A Single-Layer Perceptron (SLP) Model was used on the Pima Indian Diabetes Dataset. The aim of the model was to learn from the data and apply the pattern to predict diabetes in new patients. This report explores all the different tests and experiments done on a baseline Perceptron to improve its performance metrics. The best version was found to be the one with a sigmoid activation function and a learning rate of 0.001 with ~7 epochs. Further improvements can be made through hyperparameter optimization, providing the model with a more balanced dataset, and adding hidden layers that can help with understanding the data better.*

** https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database*

1. Introduction:

1.1. The Problem

The Indian population consists of the second largest number of Diabetic patients in the world. The recorded number of diagnosed diabetic patients was 74.9 million in 2021. This number is projected to 124.9 million by 2045. Moreover, the number of diabetic young adults has increased from 5.5% to 7.5% in the last twenty-years [1]. A study on diabetes among middle-aged women in India found that 254 of 640 districts have very high rates (over 10%), and 130 have moderately high levels (8.7–10.6%). These rates are highest in southern and eastern India and lowest in the central region [2].

The increase in the onset of this disease in extremely young adults is concerning. Diabetes significantly impacts young women and adults, with effects ranging from cardiovascular problems to increased infection risks. For women, it can disrupt menstrual cycles, affect fertility, and complicate pregnancies. In all adults, diabetes may lead to nerve damage, kidney issues, vision problems, and cognitive decline. Proper management through medication, lifestyle changes, and regular medical check-ups is crucial to mitigate these risks and maintain overall health [3][4].

This makes it crucial to find ways to predict the onset of diabetes and prevent it if possible. This paper focuses on the use of the Perceptron Machine Learning (ML) algorithm to explore the trends in the Pima Indians Diabetes Data and use it to learn and predict the likelihood of diabetes for unseen data. This ML Model, if optimized well can be useful to detect the onset of diabetes and prevent it early.

1.2. Potential Solution

The Perceptron is a single-layer Neural Network that acts as a linear binary classifier to distinguish between classes based on the learned data [5]. It is a simple ML model that can be used to prevent overfitting due to the lack of its complexity. However, due to the same reason, it may fail in capturing spatiality in dimensions. However, as our dataset is quite simple and only consists of 8 parameters and 1 binary target label, the perceptron will be a good starting point to develop a prediction algorithm for the Pima dataset. It comprises of concepts such as bias, weights and an activation function to process the given input. This algorithm will be further explained later in the paper and how it can be optimized to give ideal results.

This paper will be an exploration of the use of the Perceptron Model on the Pima Diabetes Dataset and the experiments undertaken to optimize its performance in predicting diabetes in patients.

2. Method

2.1. Dataset

The Pima Indian Diabetes dataset consists of 8 features and 1 target label, which describes a positive or negative outcome of the patient. The parameters are shown in Figure 1.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    float64
 1   Glucose                   768 non-null    float64
 2   BloodPressure             768 non-null    float64
 3   SkinThickness             768 non-null    float64
 4   Insulin                   768 non-null    float64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    float64
 8   Outcome                   768 non-null    float64
```

Figure 1: Parameters in the Pima Indian Diabetes Dataset

As this dataset was retrieved from Kaggle, it was pre-processed and had no null or infinitive negative/positive values. Therefore, no imputation was required. However, the data-distribution for each feature was analyzed through the *.describe()* pandas function[Appendix 1] and through histogram plots, shown in Figure 2.

The target label was plotted as a pie-chart and clearly shows that the dataset included more negative cases (65.1%) of diabetes over

positive (34.9%). This skewed nature of the samples will likely affect the model training and it is likely to generalize towards negative cases. This can be dangerous as a False Negative will mean that a diabetes-positive patient will not be given the appropriate treatment and may be impacted adversely.

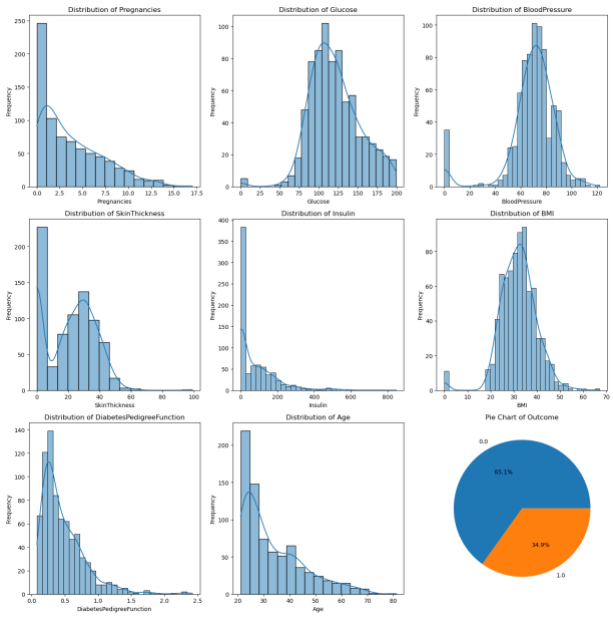

Figure 2: Plot distribution of parameters in the dataset

The distribution chart of the BMI and BloodPressure features also demonstrate potential outliers towards the left of the graph which can be another potential source, in terms of the data, to reduce the model's efficiency.

2.2. Perceptron

2.2.1 Perceptron – the algorithm

The Perceptron algorithm is the simplest form a Neural Network that mimics the basic structure of the human neuron. It consists of the input values, the weight of feature importance, a bias term, a weighted sum, and an activation function which leads to the output, demonstrated in Figure 3. A Perceptron is especially useful to linearly separate, i.e., classify the given dataset into two distinct sets [6].

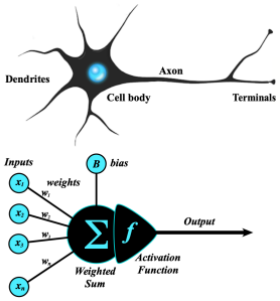

Figure 3: Structure of an SLP. [source: https://s.mriquestions.com/what-is-a-neural-network.html]

- Input Values: For each sample in the dataset, an input value from each feature is passed into the Perceptron network.

- Weights and Bias: Each feature is then multiplied by a "weight" or its importance. These weights are iteratively modified to get the best possible output.

- Weighted Sum: The features, multiplied by the weights are summed together along with a *bias* term before being passed on to the activation function. The bias term is a static term, in machine learning models helps improve flexibility and accuracy. Essentially, the bias acts as an offset, helping the model learn complex patterns more effectively. The bias term also impacts the activation function as it can be used to shift its curve [6][7].

- Activation Function: The purpose of an activation function is to determine the output of a neural network [8]. In simpler terms, it determined if a neuron should be activated or not. It is a type of threshold determined by mathematical calculations on the input data. It is synonymous to the human brain processing input signals and activating a neuron based on the input and the pathways [9].

  Relevant to the perceptron, it uses the binary step function as the threshold to decide whether to produce an output or not, shown in Figure 4 [10].

  When the input into the activation function does not reach the relevant threshold, it reflects on the quality of the input which aids the algorithm in adjusting the feature weights with a goal to activate the function and produce an output.

  Activation functions can be changed based on the data being used and the output required. Some functions include, ReLU, Sigmoid, LeakyReLU etc.

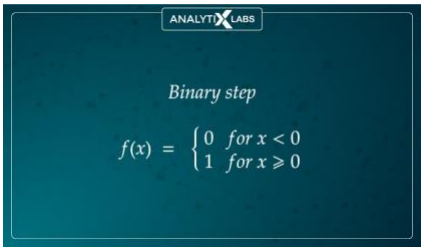

Figure 4: Traditional activation Function for perceptron [10].

This entire algorithm can be represented mathematically in the following sense:

$$f(X) = \begin{cases} 1, & if\ W.X + b > 0 \\ 0, & otherwise \end{cases}$$

Where X and W, respectively are:

$$X = \{x_1,\ x_2,\ \ldots,\ x_n\} \qquad W = \{w_0,\ w_1,\ \ldots,\ w_n\}$$

and b is the bias term. It is very similar to the general linear equation $y=mx+c$ [11].

2.2.2 Hyper-parameters:

Some hype-parameters that may impact the performance of a Perceptron Model and can be used to optimize its performance are the learning rate and the number of epochs used.

- Learning Rate(eta):
  The loss function in machine learning quantifies the difference between a model's predictions and actual outcomes, with the goal of minimizing this error by adjusting model parameters. The learning rate determines the size of these parameter updates, influencing both the speed and stability of the change process [12].
  - High learning rate: faster convergence but may cause overshooting.
  - Low learning rate: may result in slow convergence or getting stuck in suboptimal solutions.
- Epochs:
  In machine learning, an epoch refers to a full cycle where the entire training dataset is processed by the model. This process enables the algorithm to iteratively adjust its internal parameters, enhancing its capacity to identify patterns, update feature weights and generate accurate predictions. Determining the optimal number of epochs is critical, as it directly impacts the model's performance and helps strike a balance between underfitting learning and overfitting to the training data[13].

2.2.3 Chosen performance metric: F10 Score

The key performance metrics in Machine Learning are accuracy, recall, precision and F1 score. These are calculated using the formulas shown in Figure 5.

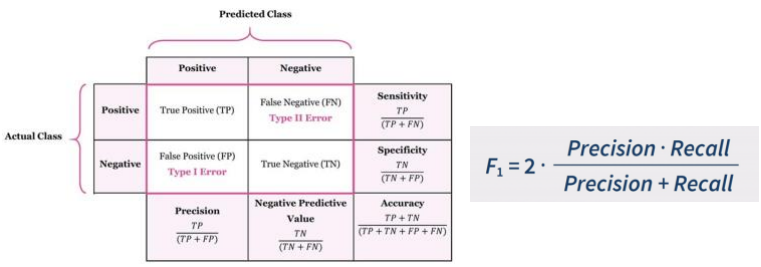

Figure 5: Performance Metric Formulas

The F1 score being a balance of precision and recall which is a measure of the overall correct positive predictions plays a key-role especially when measuring health care data.

For this model, fbeta score, F10 was the chosen performance metric as an F10 gives recall more weight over precision. Recall is the measure of the fraction of true positive predictions out of all actual positive cases. This implies that the F10 score measures recall more strongly and tries to minimize false negatives as they can be expensive in a disease detection scenario.

3. CODE:

[https://github.com/nnidadavolu4/DeepLearningFundamentals_a1772276]

4. Experimental Analysis

The dataset was split into train, validation, and test datasets. The test dataset was only used at the end with the most optimal model found through all the experiments. The final optimal model's performance metrics were compared to the Baseline Model's performance on the test dataset.

4.1. Baseline Metrics

The Baseline Metrics for the Perceptron I implemented were as follows. It was made using the Unit-Step Activation Function with a learning rate of 0.1 and 100 epochs.

```
Baseline Metrics:
Metrics for custom Perceptron:
Accuracy: 0.6104
Precision: 0.4286
Recall: 0.3333
F1 Score: 0.3750
F10 Score: 0.3341
```

Figure 6: Baseline Perceptron Performance Metrics on Validation Set

4.2. Experiment 1 – Hyperparameter Optimization: Learning Rate

For the Baseline Perceptron Model, the model was trained with learning rates – 0.1, 0.01, and 0.001, with 500 epochs and the results were as follows. They all returned the same results. However, this method was used in the following experiments where every other version of the Perceptron was optimized using 0.1, 0.01, and 0.001.

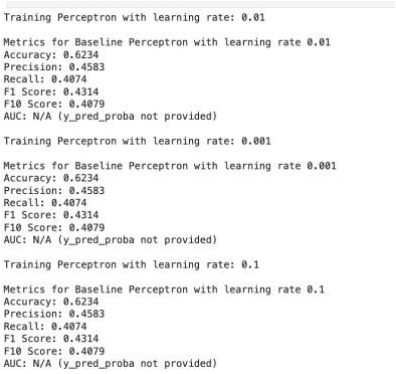

Figure 7: Baseline Hyper-parameter optimization

4.3. Experiment 2 – Change in Activation Function

A new perceptron model was created with the unit-step activation function being replaced with the sigmoid activation function. It was then optimized using 0.1, 0.01, and 0.001 learning rates and the best performance metrics were returned when the model had a learning rate of 0.01.

The sigmoid function allows for back-propagation, allowing the feature weights to be updated where necessary if the activation function is not triggered. Unlike the unit step's abrupt change, the sigmoid provides a gradual transition between 0 and 1, allowing for more nuanced outputs that can express uncertainty or partial activation.

This smooth transition more closely resembles the behavior of biological neurons, making it a more realistic model for artificial neural networks. Additionally, the sigmoid's differentiability and gradual change make it more numerically stable and easier to work with in various mathematical analyses compared to the non-smooth unit step function.

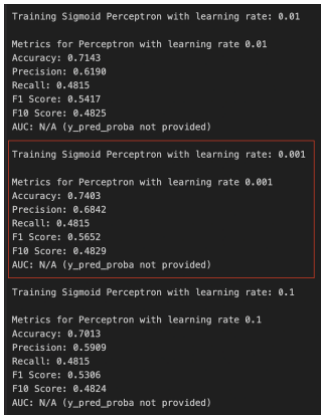

Figure 8: Performance Metrics of Perceptron Sigmoid activation function

4.4. Experiment 3 – Early Stopping: Change in loss function

As the sigmoid function produced better results, it was used for all further experiments. The next experiment used early stopping. If the model ran through 3 epochs with no improvement in validation loss, the model would stop running and return the metrics for last epoch. The results of this experiment are available in Appendix 2.

4.5. Experiment 4 – Retaining the best epoch, accuracy, and F10 Score

The last experiment is an extension of Experiment 3 as the training run will stop early at no change in validation loss for 3 consecutive epochs. However, when this happens it will also restore the epoch at which the best accuracy and F10 score were produced. This helps in understanding the relevant parameters to get the best results. This Perceptron was also run with multiple learning rates and it return the best F10 score out of all experiments, with a learning rate of 0.001. The detail of this experimental run is available in Appendix 3.

4.6. Final results

The best F10 score and accuracy thus far on the validation set were 69.23% and 79.22% respectively, found in Experiment 4. The training and validation loss graphs are presented as follows. Although there is a drop in the loss rates through further epochs, the accuracy and F10 scores started dropping around epoch ~7 and did not improve in 100 epochs, which can be seen in the graph in Appendix 3, hence the early stopping method was used to retain these scores.

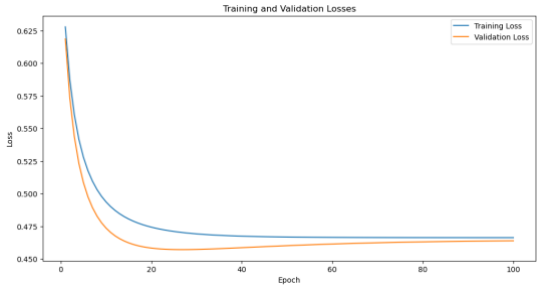

Figure 9: Training and Validation Loss trends for Best Model

This perceptron model was used on the test set and returned the following performance metrics. Figure 10 shows the performance metrics produced by both the optimized and the Baseline Model on the test set.

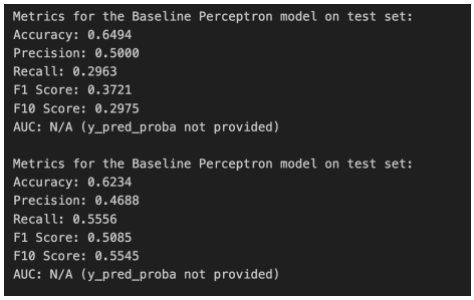

Figure 10: Performance metrics of Baseline and Optimized Perceptron Model

Although the performance metrics of the optimized perceptron model are not the highest and different from the validation metrics, the F10 score, which is or key chosen metric has a great improvement from the F10 score of the baseline model. This suggests that the Perceptron Model can be further improved, however it is important to consider that the lack of efficiency may also be coming from the skewed dataset.

A perceptron model is also quite simple and may lack the ability to capture the complex relationships between the features. Hence, another way to optimize this model could be to add hidden layer(s) to learn the data and its spatiality better.

5. Conclusion

Through experimentation and hyper-parameter optimization, the best Perceptron Model to predict diabetes in patients was found to be a Perceptron Model that uses the Sigmoid Activation function, with a slow learning rate of 0.001 and ~6 epochs. It produced an F10 score of 69.23% and 55.45% on the validation and test sets, respectively. Further improvements are for the model to perform better. However, the model also needs to be fed better training data as the current dataset has an overpowering ~70% of negative samples, which may have caused it to generalize to 0 over 1 as an output.

References

1. Maiti, S. *et al.* (2023) *Socioeconomic inequality in awareness, treatment and control of diabetes among adults in India: Evidence from National Family Health Survey of India (NFHS), 2019–2021, Nature News.* Available at: https://www.nature.com/articles/s41598-023-29978-y#:~:text=According%20to%20a%20much%20recent,to%2010.4%25%20by%2020304. (Accessed: 03 October 2024).

2. *Mapping diabetes prevalence among women in India* (2020) *News*. Available at: https://www.hsph.harvard.edu/news/hsph-in-the-news/diabetes-women-india/ (Accessed: 03 October 2024).

3. *Early signs of diabetes in women* (2023) *New Jersey's Top-Ranked Hospital Network.* Available at: https://www.hackensackmeridianhealth.org/en/healthu/2023/02/21/early-signs-of-diabetes-in-women (Accessed: 03 October 2024).

4. Department of Health & Human Services (2004) *Diabetes - long-term effects*, *Better Health Channel*. Available at: https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/diabetes-long-term-effects (Accessed: 03 October 2024).

5. Sharma, S. (2019) *What the hell is Perceptron?*, *Medium*. Available at: https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53 (Accessed: 02 October 2024).

6. Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep learning*. Cambridge, MA: The MIT Press.

7. Sharma, S. (2022) *Activation functions in neural networks*, *Medium*. Available at: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 (Accessed: 04 October 2024).

8. Baheti, P. (2021) *Activation functions in neural networks [12 types & use cases]*, *V7.* Available at: https://www.v7labs.com/blog/neural-networks-activation-functions#:~:text=datasets%2010x%20faster-,What%20is%20a%20Neural%20Network%20Activation%20Function%3F,prediction%20using%20simpler%20mathematical%20operations. (Accessed: 01 October 2024).

9. Potrimba, P. (2024) *What is an activation function? A complete guide.*, *Roboflow Blog*. Available at: https://blog.roboflow.com/activation-function-computer-vision/ (Accessed: 02 October 2024).

10. AnalytixLabs (2024) *Activation functions in neural networks - its components, uses & types*, *Medium.* Available at: https://medium.com/@byanalytixlabs/activation-functions-in-neural-networks-its-components-uses-types-23cfc9a7a6d7#:~:text=Such%20a%20function%20checks%20whether,the%20input%20crosses%20the%20threshold. (Accessed: 02 October 2024).

11. Kashyap, A. (2019) *Math behind Perceptrons*, *Medium*. Available at: https://medium.com/@iamask09/math-behind-perceptrons-7241d5dadbfc (Accessed: 01 October 2024).

12. *What is learning rate in machine learning?* (no date) *Pure Storage*. Available at: https://www.purestorage.com/knowledge/what-is-learning-rate.html#:~:text=The%20learning%20rate%20controls%20the,oscillate%20around%20the%20optimal%20solution. (Accessed: 02 October 2024).

13. Géron, A. (2020) *Hands-on machine learning with scikit-learn, keras and tensorflow: Concepts, tools and techniques to build Intelligent Systems*. Beijing: O'Reilly.

14. Topper, N. (2023) *Sigmoid activation function: An introduction*, *Built In*. Available at: https://builtin.com/machine-learning/sigmoid-activation-function (Accessed: 05 October 2024).

Appendix

1. . *describe()* function on the dataset:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

2. Metrics for Sigmoid Perceptron with early stopping

Training Perceptron with learning rate: 0.01

Early stopping at epoch 6 due to no improvement in validation loss for 3 consecutive epochs.

Metrics for Perceptron with learning rate 0.01

Accuracy: 0.7143

Precision: 0.6190

Recall: 0.4815

F1 Score: 0.5417

F10 Score: 0.4825

AUC: N/A (y_pred_proba not provided)

Training Perceptron with learning rate: 0.001

Early stopping at epoch 30 due to no improvement in validation loss for 3 consecutive epochs.

Metrics for Perceptron with learning rate 0.001

Accuracy: 0.7403

Precision: 0.6667

Recall: 0.5185

F1 Score: 0.5833

F10 Score: 0.5197

AUC: N/A (y_pred_proba not provided)

Training Perceptron with learning rate: 0.1

Early stopping at epoch 4 due to no improvement in validation loss for 3 consecutive epochs.

Metrics for Perceptron with learning rate 0.1

Accuracy: 0.7013

Precision: 0.5909

Recall: 0.4815

F1 Score: 0.5306

F10 Score: 0.4824

AUC: N/A (y_pred_proba not provided)

3. Accuracy and F10 score for best model: