Heaven's Light is Our Guide

Rajshahi University of Engineering & Technology

Course No: CSE 4204

Course Title: Sessional Based on CSE 4203

Experiment No. 2

Name of the Experiment: Design and implementation of single layer perceptron learning algorithm.

Submitted by:

Ashraf-Ul-Alam

Roll: 1803070

Section: B

Department of Computer Science & Engineering

Rajshahi University of Engineering & Technology

Submitted to:

Rizoan Toufiq

Assistant Professor

Department of Computer Science & Engineering

Rajshahi University of Engineering & Technology

Date of Submission: 18 November 2023

Contents

1 Introduction

This report presents a comprehensive analysis of the design and implementation of a single-layer perceptron learning algorithm. The experiment focuses on examining various weight adaptation techniques within the perceptron learning framework. The primary dataset utilized in this study is the "Indian Liver Patient Records"[1] dataset, which includes a range of medical attributes.

2 Dataset

2.1 Dataset Analysis

This analysis began by loading the dataset from the "indian_liver_patient.csv" [1] file. The dataset contains information on various attributes related to liver health, such as age, gender, bilirubin levels, liver enzymes, and more. The dataset has 583 entries and 11 columns.

- Age: An integer representing the patient's age.
- Gender: Categorical feature (0 for female, 1 for male).
- Total Bilirubin, Direct Bilirubin, Alkaline Phosphotase, Alamine Aminotransferase, Aspartate Aminotransferase: Numeric measurements.
- Total Proteins, Albumin, Albumin and Globulin Ratio: Numeric measurements.
- Dataset: The target variable (1 for liver disease, 2 for no liver disease).

2.2 Feature Processing

The original dataset contained the 'Gender' feature as a categorical variable with string values representing male and female. To make this feature compatible with the perceptron learning algorithm, which requires numerical input, the 'Gender' data was transformed into a numeric format using Scikit-Learn's Label Encoder, assigning '0' for female and '1' for male.

2.3 Handling Missing Values

The missing values within the dataset were also checked and it was found that the "Albumin_and_Globulin_Ratio" column had four missing values. To address this, the missing values were filled with the mean of the column.

2.4 Correlation Analysis

A correlation analysis was performed To ensure that the selected features were suitable for the perceptron learning algorithm. The goal was to visually represent any significant connections between variables using a heatmap [1]. Recognizing these relationships is important for a perceptron model because it depends on the independence of features to make accurate predictions. The findings revealed no noteworthy correlations, suggesting that each variable independently affects the prediction of liver disease. This independence is advantageous for the perceptron algorithm as it reduces the chance of multicollinearity, leading to more dependable and understandable model results.

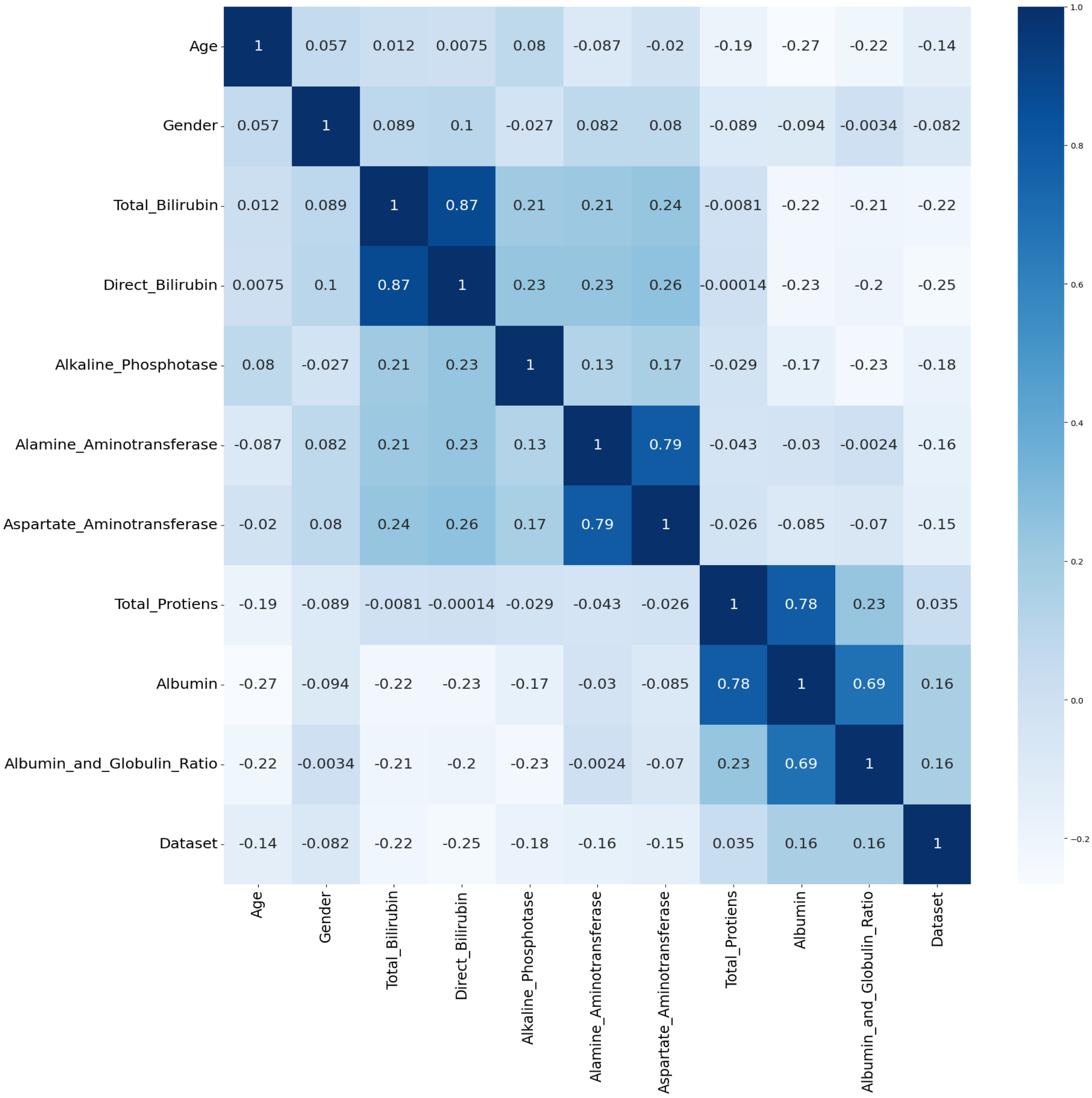

Figure 1: Correlation Heatmap

2.5 Data Type Conversion

To maintain consistency and avoid data loss, all the features were converted to the float data type.

2.6 Data Imbalance

A significant class imbalance can be observed in the target variable ("Dataset") with more instances of one class (1) compared to the other (2). To address this imbalance, Synthetic Minority Over-sampling Technique (SMOTE) was applied to oversample the minority class. The following table [1] illustrates the class distribution before and after applying SMOTE:

Table 1: Class Distribution Before and After SMOTE

Class	Count Before SMOTE	Count After SMOTE
1	416	416
2	167	416

In the original dataset, class 1 (indicating patients with liver disease) had 416 instances, while class 2 (indicating patients without liver disease) had only 167 instances. After applying SMOTE, both classes have an equal number of instances (416), effectively addressing the class imbalance issue.

Balancing the dataset through SMOTE ensures that the machine learning model is trained on a more representative dataset, preventing bias towards the majority class and improving its ability to make accurate predictions for both classes.

2.7 Train-Test Split

The dataset was split into training and testing sets with a 70-30 ratio to evaluate the performance of perceptron learning algorithm.

3 Methodology

3.1 Perceptron Learning Algorithm

The Perceptron learning algorithm is a foundational concept in machine learning, specifically designed for binary classification tasks. Its primary objective is to iteratively adjust the weights and thresholds associated with input features to enable accurate predictions. The algorithm involves presenting input data and desired outputs, calculating the actual output based on the current weights and thresholds, and adapting these parameters based on the correctness of predictions. Through this process, the Perceptron aims to learn optimal weight values that allow it to effectively discriminate between different classes in the given dataset. Here's a breakdown of its steps:

1. **Initialize Weights and Threshold:**
   - Set the random seed for reproducibility: $np.random.seed(seed)$.
   - Determine the number of features n from the input data: $n\_features = data.shape[1]$.
   - Initialize the weights to a random value within a specified range: $weights = np.random.uniform(-0.01, 0.01, n\_features)$.
   - Set the threshold value: $threshold = -1$.

2. **Perceptron Training (without learning rate):**
   - For each epoch, iterate over each training sample x and its corresponding desired output d:
     - Calculate the actual output y using the dot product of weights and input plus the threshold: $y = np.dot(weights,x) + threshold$.
     - **Update weights:**
       (a) **When Learning rate,** $\eta = 1$ **:**
         * If $y \geq 0$ and d is Class B (desired output is 0):
           $weights = weights - x$.
         * If $y < 0$ and d is Class A (desired output is 1):
           $weights = weights + x$.

       (b) **When Learning rate,** $0 <= \eta <= 1$ **:**
         * If $y \geq 0$ and d is Class B:
           $weights = weights - \eta \times x$.
         * If $y < 0$ and d is Class A:
           $weights = weights + \eta \times x$.

       (c) **Widrow - Hoff delta rule** $(0 <= \eta <= 1)$ **:**
         * Compute Δ based on the class:
           · If d is Class A: $\Delta = 1 - y$.
           · If d is Class B: $\Delta = 0 - y$.
         * Update the weights and threshold:
           · $weights = weights + \eta \times \Delta \times x$.
           · $threshold = threshold + \eta \times \Delta$.

   - Return the trained weights and threshold.

The Perceptron learning algorithm iterates through the presented inputs, adjusting weights based on the desired outputs. This process continues until the algorithm converges or reaches a specified number of epochs. Through this adaptation, the Perceptron aims to learn and classify inputs into distinct categories.

4 Results Analysis

Three distinct approaches were employed in the implementation of the Perceptron learning algorithm. Each approach varied in its weight adaptation strategy and learning rate, yielding differing accuracies and performance metrics.

4.1 Basic Weight Adaptation

In this approach, a learning rate of 1 was utilized. The best accuracy achieved was 68.00%, observed at epoch 11 shown in Figure [2]. The confusion matrix shown in Figure [3] depicted a distribution of predictions, where 69 instances of class 1 were correctly identified, yet 45 instances were misclassified. Similarly, 101 instances of class 2 were correctly identified, while 35 instances were misclassified.

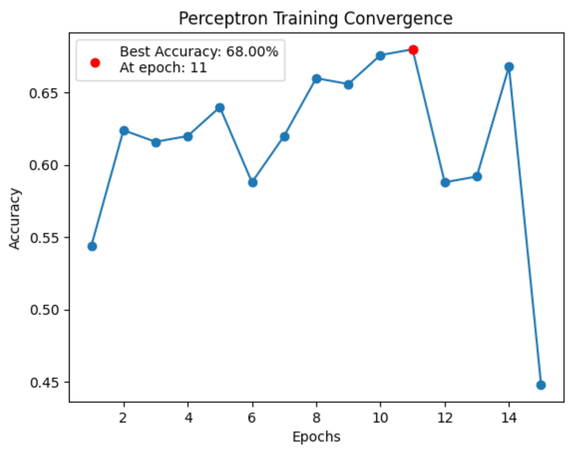

Figure 2: Training progress

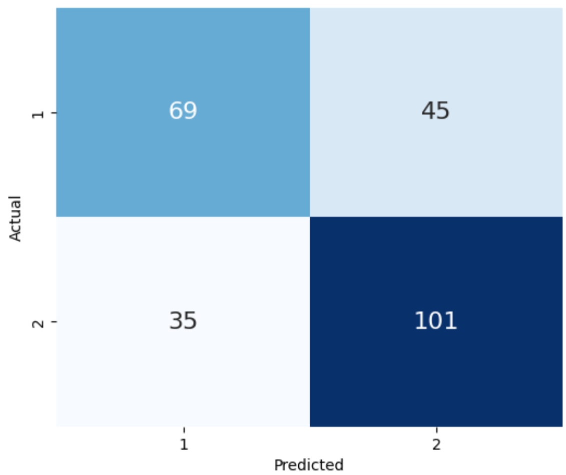

Figure 3: Confusion Matrix of Best Accuracy

4.2 Learning Rate-Adjusted Weight Adaptation

This approach involved a lower learning rate of 0.001. This approach showcased a slightly improved best accuracy of 68.40% at epoch 12 shown in Figure [4]. The confusion matrix shown in Figure [5] demonstrated a different distribution, where 63 instances of class 1 were correctly identified, and 51 instances were misclassified. Additionally, 108 instances of class 2 were correctly identified, while 28 instances were misclassified.

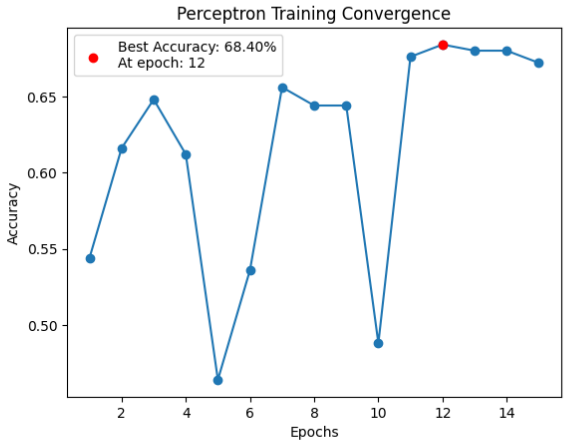

Figure 4: Training progress

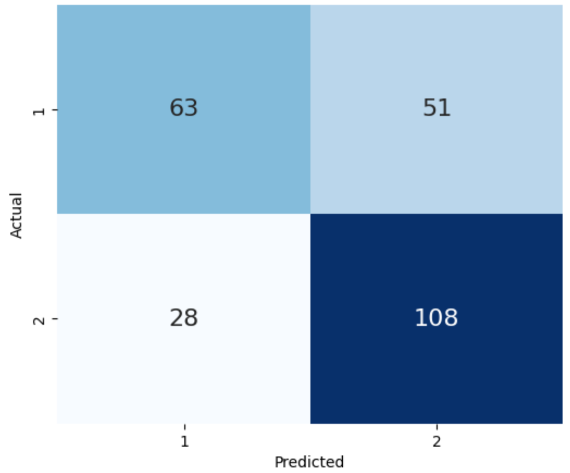

Figure 5: Confusion Matrix of Best Accuracy

4.3 Widrow-Hoff Delta Rule

Due to the potential for significantly large or small 'y' values resulting in rapid weight updates, an exceptionally small learning rate was necessary. The experimentation process revealed that learning rates from 0.001 to 0.00001 predominantly identified class 2 instances. However, a learning rate of 0.0000001 exhibited the best performance, yielding an accuracy of 72.80% at epoch 20 shown in Figure [6].

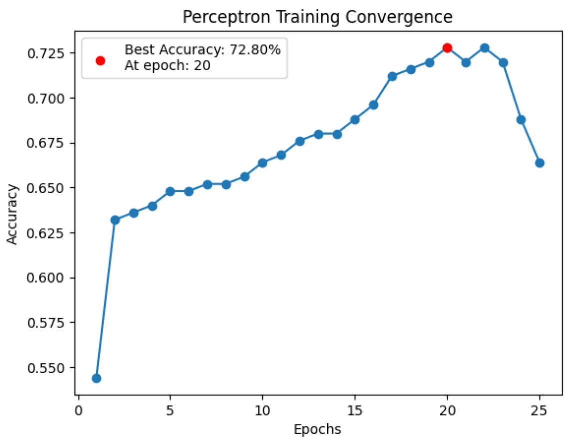

Figure 6: Training progress

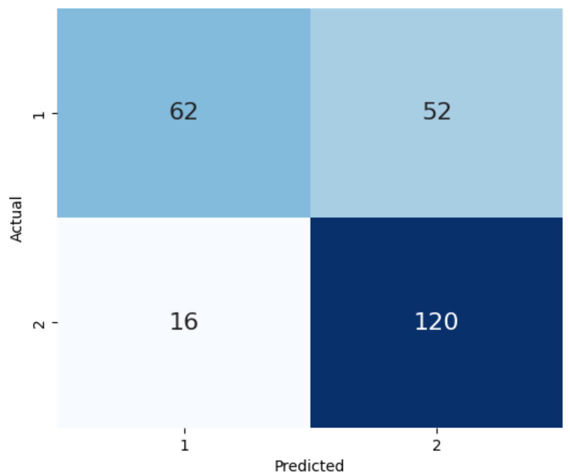

Figure 7: Confusion Matrix of Best Accuracy

The usage of an extremely low learning rate in Widrow-Hoff Delta Rule was essential to mitigate the impact of large 'y' values on weight updates. This precaution aimed to prevent weights from becoming dominated by such values, ensuring a more balanced and effective learning process. In Figure [7], the confusion matrix shows how the model performed. It correctly recognized 62 out of 114 class 1 instances but got 52 of them wrong. For class 2, it got 120 out of 136 instances right but made mistakes in 16 instances.

5 Applying Perceptron Learning Algorithm with Widrow-Hoff Delta Rule to Solve the AND Problem

This section demonstrates the application of the perceptron learning algorithm, employing the Widrow-Hoff delta rule, to solve the binary AND problem. The AND problem, a fundamental operation in binary logic, requires the classification of input pairs into two categories based on the AND logic gate.

5.1 Dataset for AND Problem

The dataset for the AND problem comprises the following input-output pairs, represented in Table [2]:

Table 2: AND Dataset

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

5.2 Experiment and Results

The perceptron was trained over 30 epochs with a learning rate of 0.00001. The model successfully separated the input points, correctly classifying them according to the AND logic as shown in Figure [8].

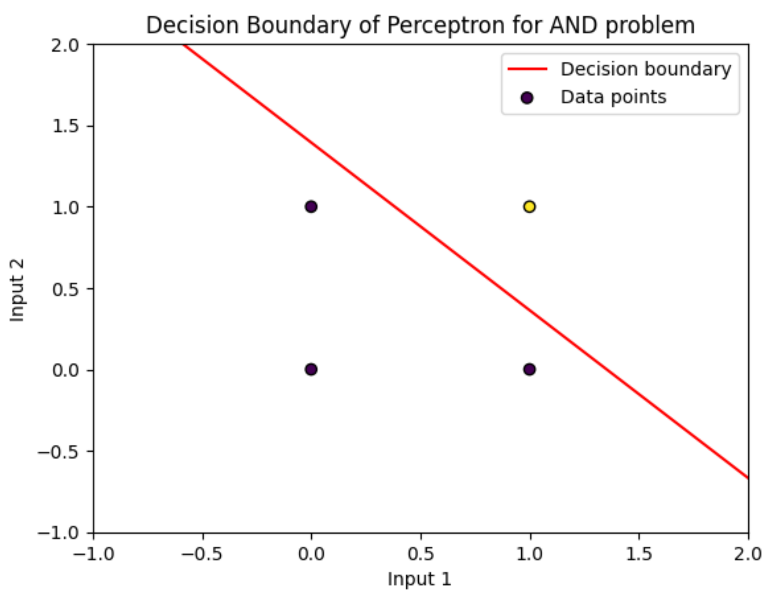

Figure 8: Decision boundary of the perceptron for the AND problem.

The decision boundary, plotted as a red line in Figure [8], clearly delineates the classification done by the perceptron. The Black points are classified as 0, and Yellow points are classified as 1, as per the AND logic.

6 Conclusion

In this study, the perceptron learning algorithm was examined through three different weight adaptation techniques. The first method, Basic Weight Adaptation without a learning rate, reached a peak accuracy of 68.00%. The second strategy, integrating a Learning Rate-Adjusted Weight Adaptation with a learning rate of 0.001, showed a marginal improvement, achieving a best accuracy of 68.40%. The final and most effective approach, employing the Widrow-Hoff Delta Rule, necessitated a highly precise learning rate due to the risk of substantial negative weight adjustments. This method stood out by achieving the highest accuracy of 72.80%. These findings underscore the critical role of adaptive learning rate strategies in enhancing the efficiency and accuracy of the perceptron learning algorithm, especially in complex classification tasks. Lastly, the Widrow-Hoff delta rule's successful application in solving the AND problem highlights its adaptability and effectiveness in perceptron learning.

References

[1] UCI Machine Learning Repository. [Online]. Available: https://archive.ics.uci.edu/ [Accessed: November 2, 2023].