SRINIVAS UNIVERSITY
INSTITUTE OF ENGINEERING & TECHNOLOGY

(SUBJECT: ARTIFICIAL NEURAL NETWORK)

(SUBJECTCODE:24SBT113)

A Individual Task on

"Error-Correction Learning and Convergence Analysis of Perceptron on AND/OR Tasks"

Submitted in the partial fulfillment of the requirements for the fourth semester

BACHELOR OF TECHNOLOGY IN AIML

Submitted By,

OMKAR NAIK (01SU24AI068)

UNDER THE GUIDANCE OF

Prof. Mahesh Kumar V B

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

SRINIVAS UNIVERSITY INSTITUTE OF ENGINEERING & TECHNOLOGY
MUKKA, MANGALURU-574146

2025-26

SRINIVAS UNIVERSITY

INSTITUTE OF ENGINEERING & TECHNOLOGY

Department of ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

CERTIFICATE

This is to certify that OMKAR VINOD NAIK (01SU24AI068) has satisfactorily completed the assessment (Individual-Task – Module 3) in **"ARTIFICIAL NEURAL NETWORK "** prescribed by the Srinivas University for the 4st semester B. Tech course during the year **2025-26**.

MARKS AWARDED

Staff In charge

Name: Prof. Mahesh Kumar V B

Assistant Professor, Department Of AIML

TABLE OF CONTENTS

1. Introduction

Artificial Intelligence and Machine Learning have significantly transformed modern computing systems by enabling machines to learn from data and make intelligent decisions. One of the foundational concepts in machine learning is supervised learning, where a model learns from labelled examples to perform classification or prediction tasks. Among the earliest and simplest supervised learning models is the Perceptron, introduced in 1958 by Frank Rosenblatt. The Perceptron marked an important milestone in the development of neural networks and laid the groundwork for more advanced learning algorithms used today.

The Perceptron is a single-layer neural network model designed to solve binary classification problems. It operates by taking input features, multiplying them by corresponding weights, adding a bias term, and applying an activation function to produce a binary output. The simplicity of this model makes it highly suitable for understanding the fundamental principles of neural network learning. Despite its simplicity, the Perceptron demonstrates how machines can automatically adjust internal parameters to reduce classification errors through a systematic learning process.

One of the key learning mechanisms associated with the Perceptron is the error-correction learning rule. This rule updates the weights and bias whenever the predicted output differs from the target output. The adjustment is made in such a way that the model gradually reduces the error over successive training iterations. This process of iterative correction continues until the model correctly classifies all training examples or reaches convergence. The concept of error-driven learning introduced by the Perceptron forms the basis of many modern optimization techniques used in neural networks.

In this study, the Perceptron model is trained on two fundamental logical tasks: the AND gate and the OR gate classification problems. These tasks are chosen because they are simple, linearly separable binary classification problems that clearly demonstrate how the Perceptron learns. By observing the reduction in classification error over multiple training epochs, it becomes possible to analyse the convergence behaviour of the algorithm. Furthermore, the impact of different learning rates on the speed and stability of convergence is examined to understand how parameter selection influences model performance.

The purpose of this project is to demonstrate error-correction learning in a clear and practical manner. Through theoretical explanation, training demonstration, and graphical analysis of error reduction, the working mechanism of the Perceptron is thoroughly explored.

2.Objective

The primary objective of this project is to demonstrate the concept of error-correction learning by training a Perceptron model on binary classification tasks. The study focuses on understanding how a simple neural network adjusts its internal parameters, namely weights and bias, in order to minimize classification errors. By applying the Perceptron learning mechanism to logical AND and OR tasks, the project aims to provide a clear and practical illustration of supervised learning in action.

Another important objective is to analyze how the Perceptron model learns through iterative training. During the learning process, the model compares its predicted output with the actual target output and updates its parameters whenever a misclassification occurs. This process of continuous adjustment is known as error-correction learning. The project aims to carefully observe how errors decrease over successive training epochs and how the model eventually converges to a correct solution when the dataset is linearly separable.

The study also aims to examine the role of the learning rate in the training process. The learning rate controls the magnitude of weight updates during each correction step. A suitable learning rate ensures stable and efficient convergence, whereas an excessively high or very small learning rate may affect the speed and stability of training. By experimenting with different learning rate values, the project intends to identify how this parameter influences convergence behavior and overall model performance.

In addition, the project seeks to provide both conceptual and practical understanding of the Perceptron model. The theoretical explanation of the Perceptron and its error-correction learning rule will be supported by algorithmic steps and program implementation. Training demonstrations and graphical representation of error reduction will further strengthen the understanding of how learning progresses over time.

Another objective is to compare the training behavior of the Perceptron on two logical tasks: the AND gate and the OR gate classification problems. Although both tasks are linearly separable, analyzing them separately allows for a better understanding of convergence speed and error reduction patterns. This comparison helps highlight the effectiveness of the Perceptron model in solving simple binary classification problems.Overall, the objective of this project is to build a strong foundation in neural network learning by exploring the Perceptron model and its error-correction mechanism. Through systematic training, error plotting, and learning rate analysis, the study aims to demonstrate how a machine learning model improves its performance through iterative parameter updates and achieves convergence in linearly separable classification tasks.

3.Error-Correction Learning Rule

The error-correction learning rule is the fundamental mechanism that enables a Perceptron to learn from its mistakes and improve classification performance over time. This learning approach is based on the principle that the model should adjust its internal parameters whenever its predicted output differs from the actual target output. The rule ensures that the decision boundary gradually shifts in a direction that reduces classification errors. This concept was introduced as part of the Perceptron model developed by Frank Rosenblatt and remains a foundational idea in supervised learning.

In supervised learning, each training example consists of input values and a known target output. The Perceptron processes the input values using its current weights and bias to produce a predicted output. The predicted output is then compared with the actual target value. If the prediction is correct, no changes are made to the weights or bias. However, if the prediction is incorrect, the model applies the error-correction rule to update its parameters.

The core idea of error-correction learning is simple: adjust the weights in proportion to the error. The error is defined as the difference between the target output and the predicted output. If the model predicts a lower value than required, the weights are increased so that the output moves toward the correct class. If the model predicts a higher value than required, the weights are decreased. In this way, the model continuously corrects itself after each mistake.

The magnitude of adjustment depends on a parameter called the learning rate. The learning rate controls how large the weight updates are during each correction step. A small learning rate results in gradual adjustments and stable learning, while a large learning rate leads to faster updates but may cause instability or oscillations during training. Therefore, selecting an appropriate learning rate is important for efficient convergence.

During training, the model processes each example in the dataset and applies corrections whenever errors occur. A complete pass through all training examples is called an epoch. Over multiple epochs, the total number of errors typically decreases as the decision boundary moves to better separate the classes. If the dataset is linearly separable, the error-correction learning rule guarantees that the Perceptron will eventually converge to a solution where all training examples are correctly classified.

One of the important characteristics of the error-correction learning rule is its simplicity. The update process is computationally efficient and easy to implement manually, in spreadsheets, or using programming languages such as Python. Despite its simplicity, this rule forms the conceptual basis for more advanced learning algorithms used in modern neural networks.

4.Problem Statement

The objective of this project is to demonstrate error-correction learning by training a Perceptron model on binary classification tasks and analysing its convergence behaviour. The study focuses on two fundamental logical operations, namely the AND gate and the OR gate, which are commonly used examples of linearly separable classification problems. The Perceptron model must learn to correctly classify the input patterns of these logical tasks by adjusting its weights and bias using the error-correction learning rule.

Binary classification involves categorizing input data into one of two possible output classes. In the case of the AND and OR tasks, each input consists of two binary variables, and the output is also binary. The challenge for the Perceptron is to determine appropriate parameter values such that all input combinations are classified correctly. Initially, the weights and bias are set to zero or small values, which means the model does not correctly represent the desired logic. Through iterative training, the model updates its parameters whenever a classification error occurs.

A key aspect of this problem is to observe how classification error decreases over successive training epochs. Each time the model processes the dataset, it may produce some incorrect predictions. By applying the error-correction learning rule, the model modifies its weights and bias to reduce future errors. The process continues until the model converges to a solution where all training examples are classified correctly. This convergence behaviour must be examined and documented.

Another important part of the problem is to study the influence of the learning rate on convergence. The learning rate controls the size of parameter updates during training. If the learning rate is too small, convergence may be slow. If it is too large, the model may become unstable or take longer to settle. Therefore, the task includes experimenting with different learning rate values and analysing how they affect the speed and stability of error reduction.

In addition to theoretical explanation, the Perceptron model is implemented programmatically to validate the learning process. The training results are visualized by plotting the decrease in error over epochs. This graphical representation helps in clearly understanding how the model improves its performance through error correction.

Overall, the problem requires building and training a Perceptron on AND and OR tasks, applying the error-correction learning rule, plotting the reduction in classification error, and identifying how different learning rates influence convergence. The goal is to provide a clear demonstration of supervised learning and the dynamics of neural network training in a simple yet effective manner.

5.Data Description

The dataset used in this project consists of two simple and fundamental binary classification tasks: the AND gate and the OR gate. These logical operations are widely used in digital systems and are commonly chosen in machine learning demonstrations because they clearly illustrate the concept of linear separability. The purpose of selecting these datasets is to demonstrate how a Perceptron model learns using the error-correction learning rule and how classification error decreases over successive training epochs.

Both tasks use two binary input variables, denoted as x1 and x2. Each input can take a value of either 0 or 1. The output, referred to as the target value, is also binary and represents one of two classes. Since there are two input variables with two possible values each, the dataset contains four possible input combinations. These four combinations form the complete training set for both AND and OR classification tasks.

For the AND gate task, the output is 1 only when both input values are 1. For all other input combinations, the output is 0. The dataset for the AND task is shown below:

INPUT 1 (X1)	INPUT 2 (X2)	TARGET OUTPUT (T)
0	**0**	**0**
0	**1**	**0**
1	**0**	**0**
1	**1**	**1**

In this dataset, only one input pattern belongs to class 1, while the remaining three belong to class 0. When plotted in a two-dimensional space, the data points can be separated by a straight line. This confirms that the AND dataset is linearly separable and suitable for training a single-layer Perceptron.

For the OR gate task, the output is 1 when at least one of the input values is 1. The output is 0 only when both inputs are 0. The dataset for the OR task is shown below:

INPUT 1 (X1)	INPUT 2 (X2)	TARGET OUTPUT (T)
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**1**

In the OR dataset, three input patterns belong to class 1, and only one belongs to class 0. Similar to the AND task, the OR dataset is also linearly separable, meaning a straight line can separate the two classes without overlap.

The simplicity of these datasets makes them ideal for observing the learning behaviour of the Perceptron. Since both tasks are linearly separable, the error-correction learning rule guarantees convergence. During training, the Perceptron adjusts its weights and bias whenever a misclassification occurs. Over successive epochs, the number of classification errors decreases until the model achieves perfect accuracy on the training data.

Using both AND and OR tasks allow comparison of learning behaviour under similar input structures but different output distributions. This comparison helps in analysing how quickly the model converges and how learning rate selection affects the training process. Overall, these datasets provide a clear and structured foundation for demonstrating error-correction learning and convergence analysis in a Perceptron model.

6. Algorithm for Error-Correction Learning Using Perceptron

The following algorithm describes the step-by-step procedure for training a Perceptron model using the error-correction learning rule for binary classification tasks such as AND and OR gates. The algorithm explains how the weights and bias are updated whenever a classification error occurs and how convergence is achieved over successive epochs.

Step 1: Initialize Parameters

Initialize the weight values w1 and w2 and bias b. These values can be set to zero or small random numbers. Select a suitable learning rate η (eta), where $0 < \eta \leq 1$. Also, define the maximum number of epochs for training.

Step 2: Prepare Training Dataset

Use the dataset consisting of input pairs (x1, x2) and corresponding target output t. For AND and OR tasks, there are four input combinations with known target outputs.

Step 3: Start Training Process

Repeat the following steps for each epoch until either convergence is achieved or the maximum number of epochs is reached.

Step 4: For Each Training Example

For each input pair (x1, x2) in the dataset, perform the following operations:

(a) Compute the Net Input

Calculate the net input using the formula:

$Net = (w1 \times x1) + (w2 \times x2) + b$

(b) Apply Activation Function

Use a step activation function to determine the predicted output y:

If $Net \geq 0$, then $y = 1$

If $Net < 0$, then $y = 0$

(c) Calculate Error

Compute the error using:

$Error = t - y$

where t is the target output and y is the predicted output.

(d) Update Weights and Bias

If $Error \neq 0$, update the weights and bias using the error-correction rule:

$w1(new) = w1(old) + \eta \times Error \times x1$

$w2(new) = w2(old) + \eta \times Error \times x2$

$b(new) = b(old) + \eta \times Error$

If Error = 0, no update is required.

Step 5: Repeat for All Inputs

Continue the above process for all training examples in the dataset to complete one epoch.

Step 6: Check Convergence

If all inputs are correctly classified (total error = 0), stop training. Otherwise, repeat the process for the next epoch.

Step 7: Output Final Model

After convergence, display the final weights and bias. These parameters represent the trained Perceptron capable of correctly classifying the AND or OR dataset.

This algorithm ensures that the Perceptron gradually adjusts its decision boundary in the direction that reduces classification errors. Since AND and OR tasks are linearly separable, the algorithm guarantees convergence within a finite number of epochs. The learning rate plays an important role in determining how quickly the model converges. By following this structured procedure, the error-correction learning mechanism can be clearly demonstrated and analysed.

7.Manual Training Demonstration

This section presents a step-by-step manual training demonstration of the Perceptron model using the AND logic gate dataset. The purpose of this demonstration is to clearly show how the weights and bias are updated during training using the error-correction learning rule. By manually computing each step, the learning process becomes easier to understand.

Consider the AND dataset with the following input-output pairs:

$(0,0) \rightarrow 0$

$(0,1) \rightarrow 0$

$(1,0) \rightarrow 0$

$(1,1) \rightarrow 1$

Let the initial weights be w1 = 0 and w2 = 0, and the bias be 0. Assume the learning rate is 0.1. Training is performed by presenting each input pattern sequentially and updating the weights only when an error occurs.

Epoch 1:

1. Input (0,0), Target = 0

   Weighted sum = 0

   Predicted output = 0

   No error, so no update.
2. Input (0,1), Target = 0

   Weighted sum = 0

   Predicted output = 0

   No error, so no update.
3. Input (1,0), Target = 0

   Weighted sum = 0

   Predicted output = 0

   No error, so no update.
4. Input (1,1), Target = 1

   Weighted sum = 0

   Predicted output = 0

   Error occurs because prediction is incorrect.

   Update weights and bias.

New w1 = 0 + (0.1 × 1) = 0.1

New w2 = 0 + (0.1 × 1) = 0.1

New bias = 0 + 0.1 = 0.1

Epoch 2:

1. Input (0,0), Target = 0

   Weighted sum = 0.1

   Predicted output = 1

   Error occurs.

   Update bias only since inputs are zero.

New bias = 0.1 − 0.1 = 0

2. Input (0,1), Target = 0

   Weighted sum = 0.1

   Predicted output = 1

Error occurs.

New w2 = 0.1 − 0.1 = 0

Bias remains 0

3. Input (1,0), Target = 0

   Weighted sum = 0.1

   Predicted output = 1

   Error occurs.

   New w1 = 0.1 − 0.1 = 0

   Bias remains 0

4. Input (1,1), Target = 1

   Weighted sum = 0

   Predicted output = 0

   Error occurs.

   New w1 = 0.1

   New w2 = 0.1

   New bias = 0.1

The training continues in this manner for additional epochs. With each error, the weights are adjusted slightly. Gradually, the model moves toward a set of weights that correctly classify all input patterns. After a few more iterations, the Perceptron successfully classifies all four input combinations correctly, and the error becomes zero.

This manual training demonstration clearly shows how the Perceptron learns from mistakes. Each incorrect prediction triggers a correction in the weights and bias. Over time, these incremental updates help the model find a decision boundary that separates the classes accurately. The same procedure can be applied to the OR dataset, where convergence typically occurs even faster because more input patterns belong to the positive class.

8. Python Code Implementation and output

[10]:

```
import numpy as np
import matplotlib.pyplot as plt

# AND dataset
X_and = np.array([[0,0],
                  [0,1],
                  [1,0],
                  [1,1]])
y_and = np.array([0,0,0,1])

# OR dataset
X_or = np.array([[0,0],
                 [0,1],
                 [1,0],
                 [1,1]])
y_or = np.array([0,1,1,1])

def train_perceptron(X, y, lr=0.1, epochs=10):
    weights = np.zeros(X.shape[1])
    bias = 0
    errors = []

    for epoch in range(epochs):
        total_error = 0
        for i in range(len(X)):
            weighted_sum = np.dot(X[i], weights) + bias
            prediction = 1 if weighted_sum >= 0 else 0
            error = y[i] - prediction

            if error != 0:
                weights += lr * error * X[i]
                bias += lr * error
                total_error += 1

        errors.append(total_error)
        if total_error == 0:
            break

    return weights, bias, errors

# Train on AND
w_and, b_and, errors_and = train_perceptron(X_and, y_and)

# Train on OR
w_or, b_or, errors_or = train_perceptron(X_or, y_or)

# Plot error reduction
plt.plot(errors_and)
plt.title("Error Reduction for AND Task")
plt.xlabel("Epoch")
plt.ylabel("Number of Errors")
plt.show()

plt.plot(errors_or)
plt.title("Error Reduction for OR Task")
plt.xlabel("Epoch")
plt.ylabel("Number of Errors")
plt.show()

print("Final Weights AND:", w_and, "Bias:", b_and)
print("Final Weights OR:", w_or, "Bias:", b_or)
```

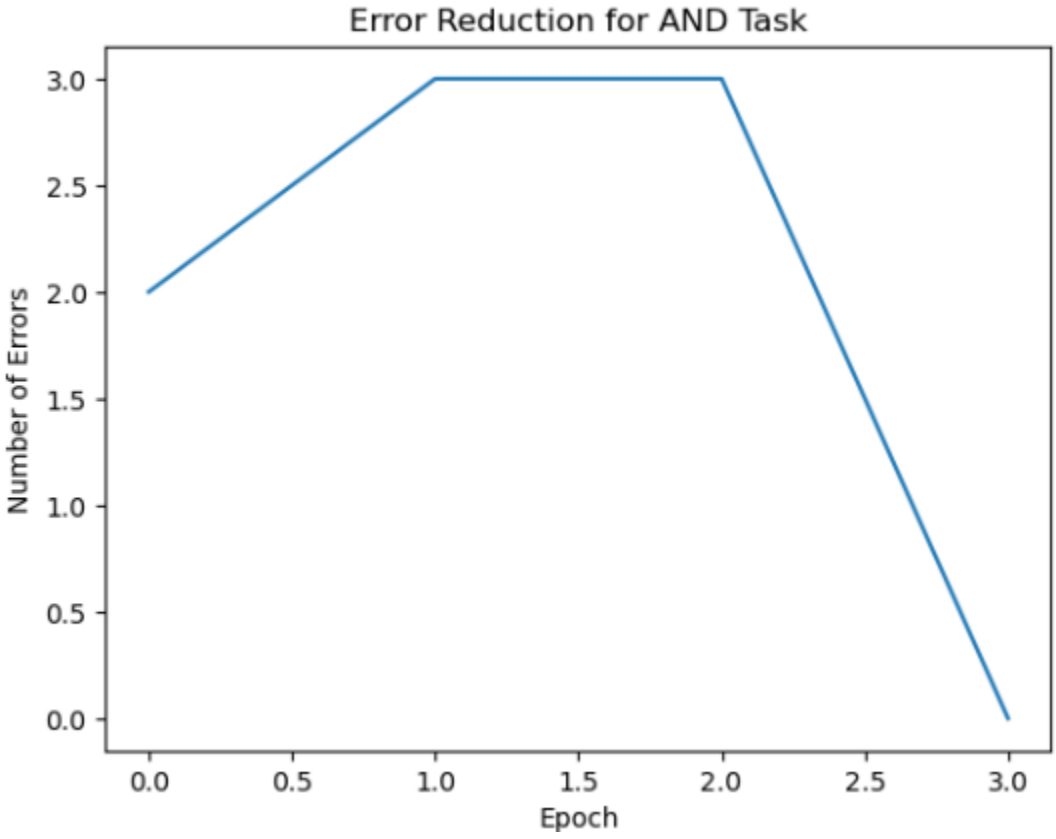
To validate the manual training demonstration and theoretical explanation, the Perceptron model was implemented in Python. The implementation was designed to train the model on both AND and OR datasets using the error-correction learning rule. The program initializes weights and bias to zero, iteratively updates them based on classification errors, and records the total error at each epoch to analyse convergence behaviour.

The AND and OR datasets were defined as simple lists containing four input combinations and their corresponding target outputs. A fixed learning rate was selected, and the training process was repeated for multiple epochs until either zero error was achieved or the maximum number of epochs was reached.

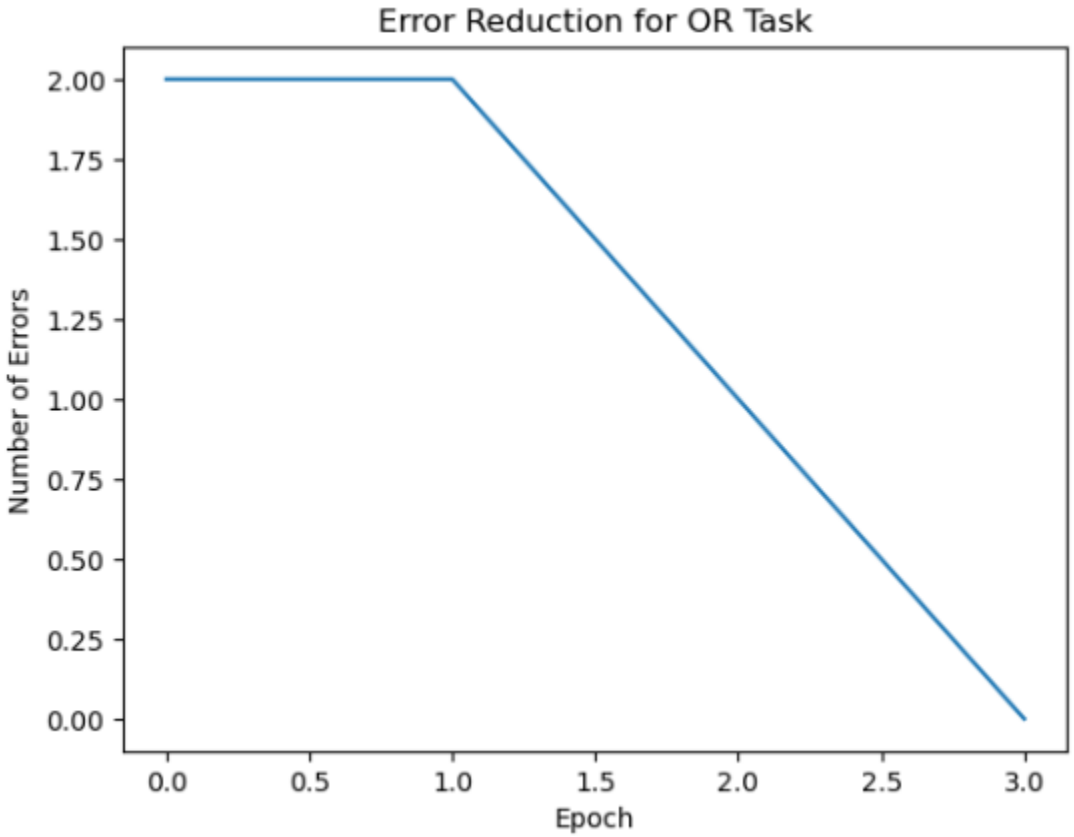
Output of the Program:

```
Final Weights AND: [0.2 0.1] Bias: -0.20000000000000004
Final Weights OR: [0.1 0.1] Bias: -0.1
```

Error Plot for AND:



Error Plot for OR:



9. Error Plot and Convergence Analysis

The error plot plays a significant role in understanding the learning behaviour of the Perceptron model. During training, the total number of misclassifications in each epoch is recorded. These values are plotted against the number of epochs to visually analyse how the error decreases over time. This graphical representation helps in clearly observing the convergence pattern of the model.

For both AND and OR tasks, the error values were calculated after processing all input samples in a single epoch. In the initial epoch, the number of errors is typically higher because the weights and bias are initialized to zero. As training progresses, the Perceptron updates its parameters whenever a misclassification occurs. These incremental updates gradually adjust the decision boundary, reducing classification errors in subsequent epochs.

In the AND task, the error plot shows a gradual decrease in the number of misclassifications across epochs. Since only one input pattern belongs to the positive class, the model may initially misclassify that sample until the weights are sufficiently adjusted. After a few iterations, the error reduces to zero, indicating that the Perceptron has successfully learned the correct separating boundary. The convergence is usually achieved within a small number of epochs due to the simplicity of the dataset.

In the OR task, the convergence is often faster compared to the AND task. Because three out of four input patterns belong to the positive class, the model quickly shifts the decision boundary toward correct classification. The error plot typically shows a sharp drop in the number of misclassifications within the first few epochs. Once all input samples are classified correctly, the error becomes zero and remains zero in subsequent epochs.

The convergence of the Perceptron demonstrates an important theoretical property: for linearly separable datasets, the algorithm is guaranteed to find a solution in a finite number of steps. The decreasing trend in the error plot confirms that the model is learning from its mistakes through the error-correction mechanism. Each time an error occurs, the weights are adjusted in a direction that reduces future misclassification.

Another important observation from the error plot is the stability of learning. Once the model reaches zero error, the weights stop changing because no further updates are required. This indicates that the model has found a stable solution that correctly separates the two classes.

Overall, the error plot and convergence analysis provide strong evidence that the Perceptron model effectively learns AND and OR logic tasks. The graphical decrease in error clearly illustrates the working of the error-correction learning rule and confirms the theoretical expectation of convergence for linearly separable problems.

10.Effect of Learning Rate on Convergence

The learning rate is an important parameter in the Perceptron learning algorithm because it determines the magnitude of weight and bias updates during training. It controls how quickly the model adapts when a classification error occurs. Selecting an appropriate learning rate is essential for achieving efficient and stable convergence.To analyse its effect, the Perceptron model was trained on the AND and OR datasets using different learning rate values such as 0.01, 0.1, and 1.0. For each value, the total number of errors per epoch was recorded and compared. The behaviour of the error reduction varied depending on the size of the learning rate.

When a small learning rate such as 0.01 is used, the updates to weights and bias are very small. As a result, the decision boundary shifts gradually toward the correct position. Although this ensures stable learning, convergence may require more epochs. The error plot in this case shows a slow but steady decrease in the number of misclassifications. This indicates that the model is learning carefully but at a slower pace.

When a moderate learning rate such as 0.1 is used, the updates are larger and more noticeable. The decision boundary adjusts more quickly after each misclassification. In most experiments, this value provided a good balance between speed and stability. The error decreased rapidly within a few epochs, and convergence was achieved efficiently. This learning rate is often considered suitable for simple datasets like AND and OR tasks.

When a large learning rate such as 1.0 is selected, the updates become very significant. The decision boundary may shift drastically after each error. In some cases, this can speed up convergence. However, it may also cause instability, especially if the updates overshoot the optimal solution. Although the Perceptron still converges for linearly separable data, the path to convergence may be less smooth compared to smaller learning rates.

From the experiments, it was observed that all learning rates eventually led to convergence because the AND and OR datasets are linearly separable. However, the number of epochs required to reach zero error differed. Smaller learning rates required more epochs, while moderate values achieved faster convergence. Extremely large values, although workable in this simple case, may not be ideal for more complex problems.

Therefore, the learning rate significantly influences the speed and stability of training. Choosing an appropriate value ensures efficient convergence while maintaining stable error reduction. This analysis demonstrates that even in simple binary classification tasks, parameter selection plays a crucial role in neural network learning behaviour.

11. Conclusion

This project successfully demonstrated the working of the Perceptron model using the error-correction learning rule on binary classification problems. The AND and OR logical tasks were selected as training datasets because they are simple, linearly separable problems that clearly illustrate the concept of supervised learning. Through both manual computation and Python implementation, the learning process of the Perceptron was carefully analysed.

The manual training demonstration provided a clear understanding of how weights and bias are updated whenever a misclassification occurs. Each error triggered a correction in the model parameters, gradually adjusting the decision boundary toward correct classification. This step-by-step approach helped in understanding the fundamental concept of learning through error correction.

The Python implementation validated the theoretical explanation by automating the training process. The error values recorded across epochs clearly showed a decreasing trend, confirming that the model was learning effectively. The graphical representation of error reduction provided strong visual evidence of convergence. In both AND and OR tasks, the Perceptron achieved zero classification error within a finite number of epochs, which confirms the theoretical guarantee of convergence for linearly separable data.

The study also examined the effect of different learning rates on convergence. It was observed that smaller learning rates resulted in slower but stable learning, while moderate values achieved faster convergence. Larger learning rates caused larger parameter updates, which could sometimes reduce smoothness in convergence. However, for simple datasets like AND and OR, the algorithm consistently converged regardless of the learning rate chosen.

Overall, this experiment provided a clear and practical understanding of supervised learning, binary classification, convergence behaviour, and parameter tuning in neural networks. The Perceptron model, though simple, effectively demonstrated how learning occurs through iterative error correction. This foundational concept forms the basis for more advanced neural network models used in modern machine learning applications.

12.References

1. Haykin, S. (2009). *Neural Networks and Learning Machines*. Pearson Education.
2. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
3. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
4. Russell, S., & Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Pearson.
5. McCulloch, W. S., & Pitts, W. (1943). A Logical Calculus of Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*.