Perceptron

-Vignesh Pitchaiah-

Introduction

In this report, I will explain the concept behind each version of the algorithm and compare their performance based on various metrics, and conclude one which is best fitting.

Versions of Perceptron Algorithm

Now, let's discuss the three different versions of the perceptron algorithm that we implemented and compare their performance:

Version 1: Normalizing the Data using Mean Centering and using $w \pm X$

In the first version of the algorithm, we have implemented mean centering to normalize the data and introduced a bias term as 0 to improve the interpretability and stability of the algorithm. Mean centering involves subtracting the mean of each feature from all the data points. The motivation behind this is that it can make it easier to interpret the weights of the model as the impact of each feature on the output, and the weights will not be confounded by the mean of each feature. Additionally, mean centering can improve the numerical stability of the algorithm.
We have reported that the algorithm converges during the 9th iteration and the equation of the decision boundary is:

$$(1.7)x_1 + (-0.2)x_2 + (0.0)\text{bias} = 0.$$

This version of the algorithm has the simplest decision boundary equation among the three versions.

Version 2: Without Normalizing the Data

In the second version of the algorithm, we did not normalize the data and reported that our algorithm converged during the 19th iteration, and the equation of the decision boundary is:

$$(3.3)x_1 + (-0.89)x_2 + (0.0)\text{bias} = 0.$$

Version 3: Implemented Perceptron using updated Weight through error rate and Learning Rate

In the third version of the algorithm, we have incorporated updated weights and a learning rate. The learning rate determines the step size of the weight updates, while the updated weights take into account the misclassified points and their corresponding feature vectors. The algorithm converged during the 19th iteration, and the equation of the decision boundary is:

$$(4.59)x_1 + (-2.14)x_2 + (0.60)\text{bias} = 0.$$

Comparison of Performance

Based on the results we have reported, all three algorithms achieved the same accuracy and produced decision boundaries that correctly classify all the data points with maximum learning rate and maximum number of iterations. However, there are some differences in their performance:

- Version 1 of the algorithm has the simplest decision boundary equation among the three versions. Additionally, mean centering can improve the numerical stability of the algorithm.
- Version 2 of the algorithm did not involve normalizing the data. This version converged slightly later than Version 1 and Version 3.
- Version 3 of the algorithm incorporated updated weights and a learning rate, which can improve the convergence speed and generalization of the model. The equation of the decision boundary for this version is more complex than the previous two versions.

Algorithm	**Accuracy**	**Iterations**
1	83%	9
2	100%	19
3	100%	21

Table 1: Algorithm comparison

Conclusion

In general, normalizing the data can help improve the interpret ability and numerical stability of the model. Incorporating updated weights and a learning rate can also improve the performance of the algorithm, especially in cases where the data is not linearly separable and huge in volume.

However, considering solely on the information we obtained, it is difficult to determine which algorithm is the best. Algorithm 2 and 3 have a higher accuracy than Algorithm 1, which suggests that they are better at making correct predictions. However, Algorithm 1 has fewer iterations than Algorithms 2 and 3, which suggests that it may be faster at making predictions