The Catholic University of America

CSC 527

Fundamentals of Neural Networks

Project1: Double moon classification

Hoang Cao

Computer Science and Electrical Engineering Deparment

Student ID: 5163768

**Content**

1. Rosenblatt perceptron concept 3
2. Double Moon Classification Task 3
3. Task 1 Implementation 4
4. Task 2 Implementation 4
5. Conclusion 4
6. Github Repository 5
7. References 6
8. Rosenblatt perceptron concept

The Rosenblatt’s perceptron is an implementation of McCulloch-Pitts neuron’s model. It is a binary single neuron model. The inputs integration is updated through addition of the weighted inputs every epoch of training process. If the sum is larger than the given threshold then the neuron fires else it won’t fire. The neuron output equals to 1 when it fires, else it’s 0.

The equation is shown below:

H(x) = {1 if w1\*x1 + w2\*x2 + … + wn\*xn >= theta

0 if w1\*x1 + w2\*x2 + … + wn\*xn < theta

The model implements the functioning of a single neuron that can classify linear classification problem through a very simple learning algorithm. It is the first generation of artificial neural network.

1. Double Moon Classification Task

A picture containing shape

Description automatically generated

Figure.1: Double half moon

We have a moon function which generates a number of points that shapes into two half-moons (Figure. 1). The task is to implement Rosenblatt’s perceptron model to linearly classify these two-half-moons with given parameters in Chapter1(page 61 -62).

1. Task 1: Implementation

In order to successful perform classification task, we need to handle our data points in such a way that the model can read and classify based on the input data. The data we have is simply a function which can generate points’ coordinates that shape into two half-moons. The given data does not have any label, so it is impossible to use it for classification and implementation task. Therefore, we need to assign label for these coordinates that corresponding to the moons. We will assign 1 for those points belong the upper moon and 0 for the other. Then we need to create two datasets for training and testing purposes.

Text

Description automatically generated

Figure. 2: Labeling data points

Then we create a Rosenblatt model to read in inputs and classify these labeled points. A Rosenblatt model is implemented in below:

Text

Description automatically generated

Figure. 3: Rosenblatt’s perceptron model

Now we can use the model to train and predict with training data and testing data.

Text

Description automatically generated

Figure. 4: Training and Testing model

Using this parameter, result is obtained below:

Shape, arrow

Description automatically generated

Figure. 5: Classification with distance = 1, radius = 10, and width = 6

The model is able to classify the data points into two parts; however, the decision boundary is not working well as result in Figure 1.9. There are many data points from class1 are misclassified to class 0, which cause the high error rate in this result. The problem lies in the model. We will continue to try the model on the Task2 implementation.

1. Task 2 Implementation

TaskShape

Description automatically generated  
Figure. 6: Classification with d = 0

For the Task2, the model did not perform well on this classification, more points from both classes are misclassified on this implementation. The error rate are even higher than the one in the first task. The model could not reach the convergence point in the given parameter.

1. Conclusion:

My Perceptron model seems to seriously misclassify the upper half moon with high error rate. I had tried every possible learning rate for the model, but it did not seem to work better. The boundary decision does not clearly divide the two classes. I need to perform some work on reducing error rate of my model and reach the convergence point in the require parameters.

1. Repository

https://github.com/caoh96/CSC-527/tree/master

1. Preference

<https://towardsdatascience.com/rosenblatts-perceptron-the-very-first-neural-network-37a3ec09038a>