Contents

Project 5: Neural Network with Backpropagation Algorithm

Jaudat Raza JAUDAT.RAZA@JHU.EDU
Master of Computer Science
John Hopkins University
Baltimore, MD 21218

Editor: Jaudat Raza

Abstract

This document will cover the implementation process of Backpropagation Algorithm. The resource used will be linked below in the resource section of the document. The document will follow a simple pattern for which a less than 5-minute video will be submitted as well. The Jupyter Notebook starts with getting the data into Pandas Data from the link provided in the data source. After that, first the first three data set will be processed through backpropagation for the classification and the other 3 will be processed through backpropagation for regression.

1 Problem Statement

The purpose of this assignment is to give us experience developing one of the main algorithms for training feedforward neural networks. The one we are going to be implementing is backpropagation. We will evaluate the performance of a feedforward network trained with backprop. We will be using six data sets provided by UCI. Breast Cancer, Glass, and Soybean will be used for classification. Then Abalone, Computer Hardware, and Forest Fires will be using Regression implementation. The we will compare the performance compare to the project we have done before without the back prop. My hypothesis is that the neural network with back prop will perform better than our previous implementation. With the ability to adjust model weights and biases each forward pass through the network, the model will be able to train the model more efficiently.

2 Algorithm Studied

In this project we are covering Neural Network with Backpropagation. I will cover in text how the math behind the Back prop works without going into deep math.

2.1 Neural Network with Backpropagation Algorithm

Neural network with backpropagation is a Machine learning algorithm that was created to mimic the brain. Just like the brain neuron sending the message to another neuron, neural networks send information from one node to another. One layer of node is connected to another layer of node by weights. Then the network could have zero to multiple layers.

2.1.1 Hidden Layer

So Neural Network could have zero or several hidden layer. The job of each hidden layer is to receive the input from the previous layer or the input. Then compute the weighted sum of all the input values. Send the output through some activation function like logistic activation function or hyperbolic tangent function for the classification problem. For the Regression problem pass the output through a linear function. Once that's done, the result are sent to each node in the next layer is there is any.

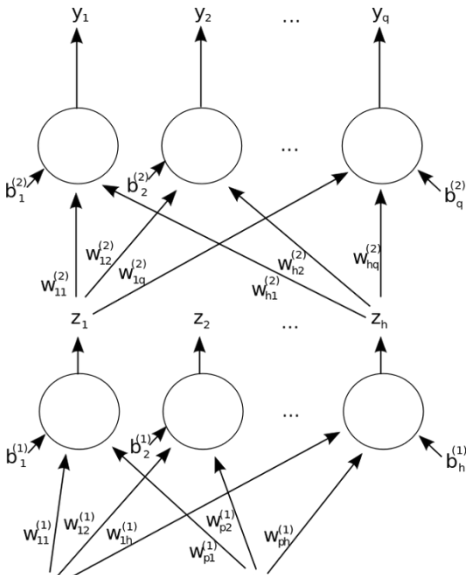

Figure 1 Double Layer Neural Network

Figure 1 Single Layer Neural Network

2.1.2 Training Phase

The main part of the algorithm. First start with weights initialized to zero or something close to it, then it proceeds to forward propagation an after that it will also go through backpropagation

2.1.3 Forward Propagation

During the forward propagation, the network takes the input. Then the hidden layer takes the steps as explained above till it reaches the output layer. The output is the predicted class. We have one- hot encoded class prediction vector, where each index corresponds to a class

2.1.4 Back Propagation

Now once the forward propagation is completed using the prediction vector from the output layer an error is calculated. This show the difference in actual vs predicted, then the algorithm propagates backward from the output later to the input layer. The error causes the weight of each node to be updated using Gradient decent. The goal of this updated weight is to make more accurate prediction for the next round. This continues for a certain number of epochs, which in our case has been set to 100.

2.1.5 Activation Function

2.1.5.1 Logistic Sigmoid Activation Function

$$sig(t) = \frac{1}{1 + e^{-t}}$$

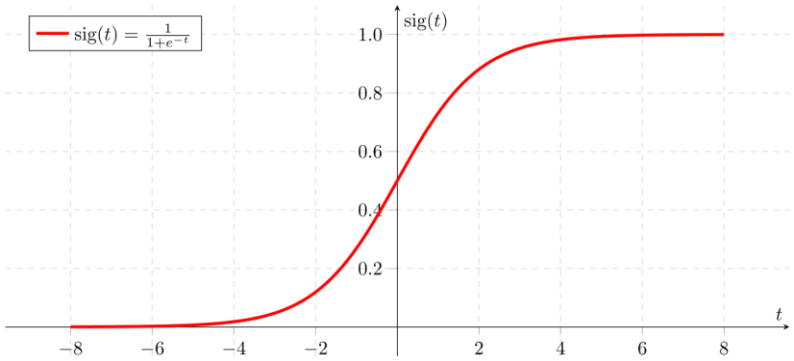

Figure 2 Logistic Sigmoid Activation Function

2.1.6 Linear Activation Function

$$A = Cx$$

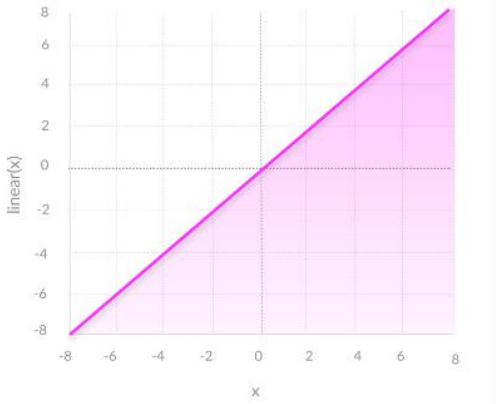

Figure 3 Linear Activation Function

2.1.7 Stochastic Gradient Descent

It was an iterative method for optimizing the function to get to the goal by smoothing the properties by jumping faster toward the goal and slows down as it gets close to the 0. The learning rate is set to 0. In high dimensional optimization problem reduces the computational burden achieving faster iteration in trade for a lower convergence rate.

3 Process

The runs are done with 5-fold cross-validation so we can compare results statistically. The project allowed us to pick from classification error, cross entropy loss, or mean squared error (as appropriate) for the loss function.

4 Tuning

4.1 Learning Rate

The learning Rate in the Gradient Decent was set to 0.1. From the previous project learned that setting the learning rate to very small value wasn't really improving the results, but was greatly impacting the time it took, so for this time It is set to 0.1 for classification and regression.

4.2 Number of Nodes per Hidden Layer

To get the number of nodes I should use per hidden layer. I set the learning rate to 0.1 and the number of iterations to 100. For the regular results, the number of epochs is set to 1000. With these

constant I got the accuracy for the classification data sets which included Breast Cancer, Glass, and Soybean. Then I got the accuracy for layer 1 and 2 with each of them with number of nodes from 1 to 10 Results in the table Below

1 Hidden Layer			
Number of Nodes per Hidden Layer	Breast Cancer	Glass	Soybean
1	100%	67.52%	36.83%
2	100%	76.17%	36.83%
3	100%	75.11%	41.27%
4	100%	77.03%	41.27%
5	100%	77.54%	41.83%
6	100%	76.78%	53.22%
7	100%	78.87%	52.22%
8	100%	79.38%	58.72%
9	100%	79.38%	53.22%
10	100%	79.38%	50.16%

2 Hidden Layer			
Number of Nodes per Hidden Layer	Breast Cancer	Glass	Soybean
1	100%	35.74%	36.83%
2	100%	35.74%	36.83%
3	100%	39.29%	36.83%
4	100%	39.83%	36.83%
5	100%	40.28%	36.83%
6	100%	40.18%	36.83%
7	100%	35.74%	36.83%
8	100%	35.74%	36.83%
9	100%	35.74%	36.83%
10	100%	35.74%	36.83%

4.3 Data Clean Up

4.3.1 Breast Cancer

1) Dropped missing data from the set
2) Change the class, where 2 is 1 and 4 is 0 now
3) Converted BareNuclei to numeric type so the math could be done on it.

4.3.2 Glass

1) Changes the possible Glass Types to in order from 0 to 5

4.3.3 Soybean

1) Removed columns that either had a mean or std of 0. Meaning they didn't change.
    a) plantgrowth
    b) leafspotshalo
    c) leafspotsmarg
    d) leafspotsize
    e) leafshread
    f) leafmalf
    g) leafmild

h) stem
i) fruitspots
j) seed
k) moldgrowth
l) seeddiscolor
m) seedsize
n) shriveling

2) Updated the Distribution to 1, 2, 3, 4

4.3.4 Abalone

1) The only change made in the data was the "Sex" Column. The infants were changed to 1, Female were changed to 2, and Male were changed to 3.

4.3.5 Computer Hardware

1) Model column in the data is serving the unique identifier code purpose, so all of them were updated from their name to integer from 1 to 208.
2) Vendor column had 29 models in it, so they were changed to 1 to 29.
3) Vendor column was set as the last column of the data.

4.3.6 Forest Fires

1) The Month column was updated from 1 to 12, where 1 is January and 12 was December.
2) The Day column was updated from 1 to 7, where 1 was Monday and then 7 was Sunday.

5 Results

For all the results below, the Number of layers was set to 1 and the number of nodes was set to eight. Epochs was set to 100 and the Learning rate was set to 0.1. Learning Rate does not impact the result much, but the number of Epochs does have an impact. From 50 to 100 was almost double the improvement, but after 100 the results start to flatten. Further detail results are in with the source code. The Regression Data Set results are in regression folder and the classification results are in the classification problem folder.

5.1 Breast Cancer

Epochs	100	1000
Fold 1	100.0%	100.0%
Fold 2	100.0%	100.0%
Fold 3	100.0%	100.0%
Fold 4	100.0%	100.0%
Fold 5	100.0%	100.0%
Average	100.0%	100.0%

5.2 Glass

Epochs	100	1000
Fold 1	77.77%	100.0%
Fold 2	77.27%	95.45%
Fold 3	79.06%	93.02%
Fold 4	83.33%	97.61%
Fold 5	79.48%	89.74%
Average	79.39%	95.17%

5.3 Soybean

Epochs	100	1000
Fold 1	40.00%	90.00%
Fold 2	60.00%	100.0%
Fold 3	55.55%	100.0%
Fold 4	55.55%	100.0%
Fold 5	50.48%	100.0%
Average	72.23%	98.00%

5.4 Forest Fire

Mean Squared Error	4.61

5.5 Machine

Mean Squared Error	16.70

5.6 Abalone

Mean Squared Error	5.85

6 Conclusion

My hypothesis for the classification problem was right on, but for the regression problem I feel like the activation function with linear return and its derivative needed some more work. This would be the next step, but as we see that even with not that stellar activation and back prop derivative, the results were almost same as what we got in our previous project or better.

Now for the classification data set, the results were amazing for all three data sets. Breast Cancer with 1 layer and least number of hidden nodes, the results were almost perfect. The reason being that there are total of 699 instances of data in the set and there were only 2 classes. For Glass data set, we see that there are 7 possible classes, so the result isn't as amazing as breast cancer, and instead of having 699 instances, it only had 214. Now for classification, Soybean data set amazed me. We only had total of 47 instances and 4 classes possible. Even with such low data, getting to that 100% was amazing. With having so many attributes in it. The other thing I noticed was that with so many attributes adding extra hidden layer drops the result by good numbers. I know that there is a large data set available for soybean. I would like to see how that reacts to the algorithm.

7 Resources

- StatQuest with Joh Starmer. “Gradient Descent, Step-by-Step.” YouTube, 05 Feb. 2019, https://www.youtube.com/watch?v=sDv4f4s2SB8.
- Srinivasan, Aishwarya V. “Stochastic Gradient Descent - Clearly Explained !!” *Medium*, Towards Data Science, 7 Sept. 2019, towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31.
- “7 Types of Activation Functions in Neural Networks: How to Choose?” *MissingLink.ai*, missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/.