Hacettepe University

BBM 409: Introduction to Machine Learning Laboratory

Fall 2016

# Assingment III - Neural Networks

*Author:*
Batuhan Yaman

*Instructor:*
Aykut Erdem
*TAs:*
Burçak Asal
Aysun Koçak

December 5, 2016

# 1 Introduction

This report consists two parts.Part one contains answers for theoretical questions. Part two is the analysis and results of Neural Networks.

# 2 Part I: Theory Questions

## 2.1 Maximum Likelihood Estimation

### 2.1.1 Question 1

**Problem:** What are differences between logistic regression and linear regression?
**Solution:** Linear Regression is used to determine the relation between variables. i.e. If we have values about height and weight to determine either this person is girl or not, linear regression gaves us a equation to calculate.

### 2.1.2 Question 2

**Problem:** What are differences between logistic regression and naive bayes methods?
**Solution:** Naive Bayes is a generative model, while Logistic Regression is discriminative. Naive bayes is worked well only if variables are independent from each other. On the other hand, Logistic regression will do fine if the variables are dependent. Naive bayes can work well with small training data, but Logistic regression will overfit with small training data.

# 3 PART II:Neural Networks

## 3.1 Introduction

In this section I implemented a neural network with n Layers for part 2. For part 1 I simply removed all hidden layers. So, while part 2 is fully parameterized, part1 is not.

## 3.2 Algorithm

For first part, we simply randomize weight initially, and we multiplied with our data including bias. Calculated the error and updated the weights accordingly. We did this for our given iteration count. For part two, we added hidden layers to this algorithm. Our class is fully parameterized;

- Iteration Count
- Hidden Layer Node Count
- Hidden Layer Width
- Decay
- Precision
- Learning Rate

Our program sets node counts same for all hidden layers. This could be changed but since we are not manuplating our image to get better results this is not crucially important.

Decay is multiplied with learning rate on every iteration on to lower to learning rate over time.

Precision is for early-stop. If the change in error rate is smaller then this rate, iterations are over.

Activation function is hard coded TanH and Sigmoid. TanH is used as default.

As Loss function, only SSE (Sum of Squared Error) is used.

### 3.3 Analysis

Contructing the NN successfully was not enough for prediction. Choosing the correct parameters was really hard. First of all, I calculated the train error with experimental parameters without bias. As activation function TanH is used untill Table 5. At table 5 both are used.

Parameters: $decay = 0.999 \quad precision = 1e-5$
TSize = Training Size , Width = Hidden Layer Count, Height = Hidden Layer Node Size, LR = Learning Rate, IC = Iteration Count

| # | TSize | Width | Height | LR | IC | Results |
|---|---|---|---|---|---|---|
| 1 | 1000 | 4 | 4 | $1e-4$ | 500 | 0.279 |
| 2 | 1000 | 4 | 4 | $1e-3$ | 500 | 0.358 |
| 3 | 1000 | 4 | 4 | $1e-2$ | 500 | 0.117 |
| 4 | 1000 | 4 | 4 | $1e-5$ | 500 | 0.206 |
| 5 | 1000 | 4 | 4 | $1e-5$ | 2500 | 0.200 |
| 6 | 1000 | 4 | 16 | $1e-3$ | 500 | 0.869 |
| 7 | 1000 | 4 | 32 | $1e-3$ | 500 | 0.937 |
| 8 | 1000 | 4 | 64 | $1e-3$ | 500 | 0.973 |
| 9 | 1000 | 8 | 64 | $1e-3$ | 500 | 0.116 |
| 10 | 1000 | 8 | 64 | $1e-4$ | 500 | 0.299 |
| 11 | 1000 | 8 | 128 | $1e-4$ | 500 | 0.306 |

Table 1: Training Succes Results without bias.

In this tests, we can clearly see that LR effects our results. If it is too high we will probably over step our local minumum. If its too low either it will take too long, or we will think that we found the local minumum but it might be just a pletau.

As we can see from test 1,2,3 increasing LR improved our results but over increasing lowered again. So we need to find a sweet spot.

From test 4,5 we can see that even if we increase our IT, if we have a too small LR it won't be very effective. Plus, it will take much more time.

From test 6,7 and 8, we have some very good improvements. By increasing the Height and using a nice LR our Results are almost perfect. Maybe too perfect? Yes, probably we have some overfitting problems over here.

From test 9,10 and 11, it is obvious that increasing height and width is not always a good thing for our network, even though we tried to lower our

LR. It is making our network unnecessarly too complex, especially with our 1K training data.

| # | TSize | Width | Height | LR | IC | Results |
|---|---|---|---|---|---|---|
| 1 | 50000 | 2 | 32 | $1e-4$ | 100 | 0.134 |
| 2 | 50000 | 4 | 64 | $1e-4$ | 100 | 0.365 |
| 3 | 50000 | 4 | 16 | $1e-5$ | 300 | 0.579 |
| 4 | 50000 | 4 | 64 | $1e-5$ | 300 | 0.408 |

Table 2: Training Succes Results without bias.

In this tests TSize is changed to 50K. We can see from 2nd and 3rd tests that similiar parameters that gave very good results on $Table1$ are now doing rather poorly results. This means we should set our parameters according to our TSize.

Then I added bias and also calculated test results to see if we are indeed experiencing overfit.
TS Rate = Training Success Rate, VS Rate = Validation Success Rate

| # | TSize | Width | Height | LR | IC | TS Rate | VS Rate |
|---|---|---|---|---|---|---|---|
| 1 | 1000 | 8 | 32 | $1e-4$ | 250 | 0.421 | 0.335 |
| 2 | 1000 | 8 | 32 | $1e-4$ | 500 | 0.879 | 0.469 |
| 3 | 1000 | 2 | 32 | $1e-5$ | 250 | 0.820 | 0.564 |

Table 3: Results with bias.

As we can see from the first three tests, even though we tried couple of different parameters and got fine training errors, our tests results are bad. This means we are having an overfit. We have too much complex network for such a low train examples. So we can try to increase or training size.

| # | TSize | Width | Height | LR | IC | TS Rate | VS Rate |
|---|---|---|---|---|---|---|---|
| 1 | 25000 | 2 | 100 | $1e-4$ | 150 | 0.470 | 0.307 |
| 2 | 25000 | 2 | 64 | $1e-4$ | 150 | 0.462 | 0.503 |
| 3 | 25000 | 2 | 32 | $1e-4$ | 150 | 0.618 | 0.461 |
| 4 | 25000 | 3 | 12 | $1e-4$ | 150 | 0.622 | 0.607 |
| 5 | 25000 | 5 | 32 | $1e-4$ | 150 | 0.604 | 0.549 |
| 6 | 25000 | 5 | 64 | $1e-4$ | 150 | 0.466 | 0.454 |
| 7 | 25000 | 4 | 32 | $1e-4$ | 150 | 0.639 | 0.655 |
| 8 | 25000 | 4 | 32 | $1e-4$ | 250 | 0.768 | 0.756 |
| 9 | 25000 | 4 | 32 | $1e-4$ | 500 | 0.841 | 0.842 |
| 10 | 25000 | 4 | 32 | $1e-4$ | 1000 | 0.896 | 0.879 |

Table 4: Results with bias.

If we look at first three test, as we increased TSize our TS Rate and Vs Rate got closer to each other. This means we are not overfitting so badly than before. But still, over rates are bad. So we need to tune our parameters.

As we can see from test 4,5 and 6, tuning parameters well effects the results very well. Also they are not overfitting. But Results are just slightly increased.

I got a good result as test7, So our parameters must be good fit for our Training Set. So I tried to increase our IC to see if we can improve our results. As we can see from following tests 8,9 and 10, our results improved superbly. Notice that improvement is not linear, even though we double the IC, Rates increase decreases as we are at higher IC.

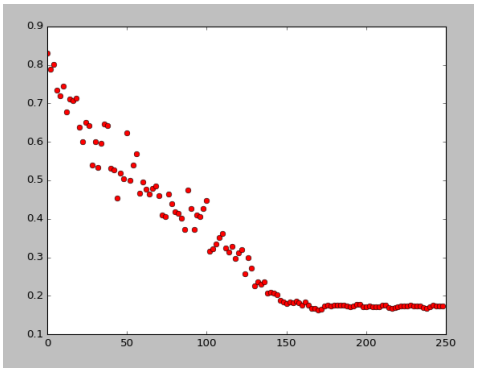


Figure 1: Error graph of a test

| # | Activation Type | TSize | Width | Height | LR | IC | TS Rate | VS Rate |
|---|---|---|---|---|---|---|---|---|
| 1 | Sigmoid | 25000 | 4 | 32 | $1e-4$ | 50 | 0.214 | 0.215 |
| 2 | Sigmoid | 25000 | 4 | 32 | $1e-4$ | 150 | 0.434 | 0.442 |
| 3 | Sigmoid | 25000 | 4 | 32 | $1e-4$ | 250 | 0.615 | 0.631 |
| 4 | Sigmoid | 25000 | 4 | 32 | $1e-4$ | 500 | 0.814 | 0.828 |
| 5 | Sigmoid | 25000 | 4 | 32 | $1e-4$ | 1000 | 0.891 | 0.888 |
| 6 | TanH | 25000 | 4 | 32 | $1e-4$ | 50 | 0.168 | 0.301 |
| 7 | TanH | 25000 | 4 | 32 | $1e-4$ | 150 | 0.639 | 0.655 |
| 8 | TanH | 25000 | 4 | 32 | $1e-4$ | 250 | 0.768 | 0.756 |
| 9 | TanH | 25000 | 4 | 32 | $1e-4$ | 500 | 0.841 | 0.842 |
| 10 | TanH | 25000 | 4 | 32 | $1e-4$ | 1000 | 0.896 | 0.879 |

Table 5: Results with bias.

And finally, I tried to change the activation function to see what difference will it make.

We can see at lower IC, TanH did a better job than Sigmoid. But we can see that At higher IC, Sigmoid can provide slightly better results. So this teaches us that we should choose our activation function according to our data and IC, in other words, spent time.