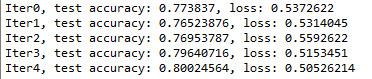
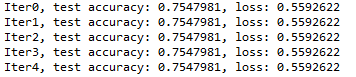
What I expected is something like this(this is also one of my result):



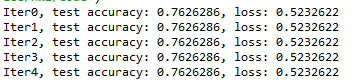
But I found, the accuracy is not always increase as I expected. Sometimes, the accuracy just remains the same, like this: (cry)



From what I learn currently, there would be several reasons, let’s figure out what’s the problems:

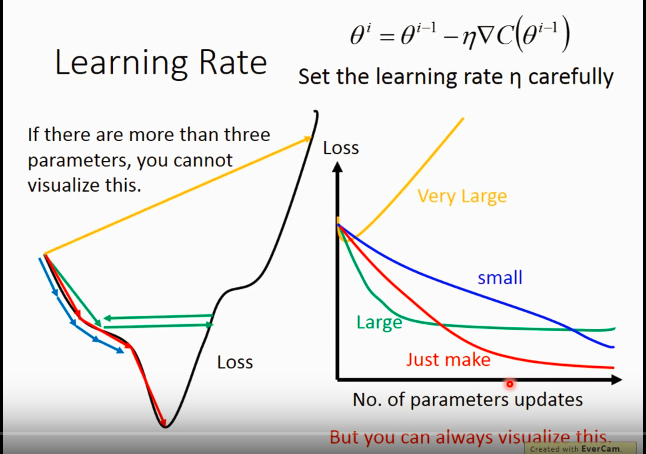
1. If the network is too simple

My network just have 2 layer, wide but not deep. Then make it deeper to 5 layers.



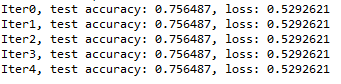
Emmmmm…… not this problem…sad

1. Tuning the learning rate:



I’m thinking of: if I’m very unlucky and my selected learning rate just like the green arrow, so that the it doesn’t change.(but in my case, it remains the same even at the beginning).

So I changed my learning rate. But the only thing that changes is the accuracy at the beginning, but it also remains the same. So it is not this problem.



I’m not sad, I’m not angry.

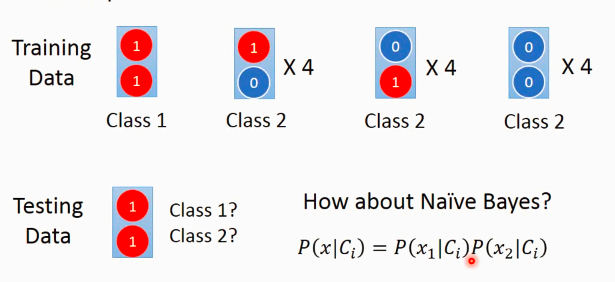
I’m not sad, I’m not angry.

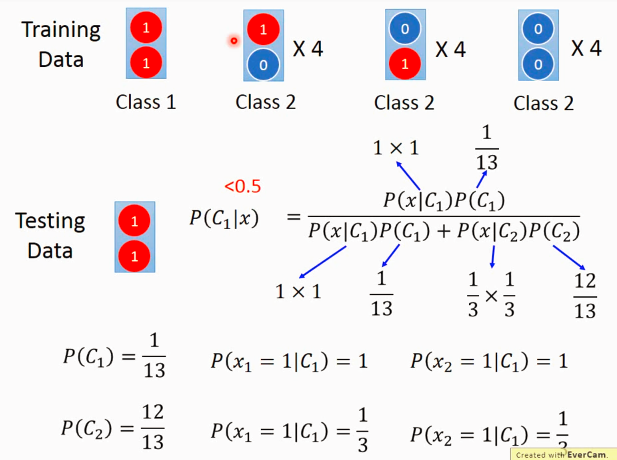
I’m not sad, I’m not angry.

I’m not sad, I’m not angry.

I’m not sad, I’m not angry.

I see someone said if the proportion of training data from different classes are way too different, it would affect the final result, like this:





In this condition, we can see that the generative method does not work well.

**So, when using generative method for classification, the training data are better evenly distributed.**

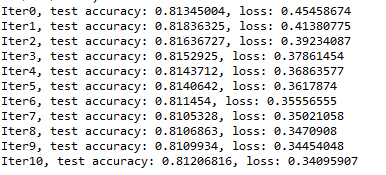
**BUT… I’m not using generative model but discriminative method…\_(:з)∠)\_**

**GG**

Then, what I do is turning off the computer, and go back home.

Lucky Monday morning!!!

When again, I run this, magic!!!



I really don’t know what happened.

Well, I know something, but not sure.

Although I don’t think is the problem of training data distribution, but I still tried to select the same number of training data in each classes.

But, no surprise, the accuracy remains the same!

Then, I modified the learning rate to 1e-5, the magic happens as I showed above…

Emmmmm………….weird…..

But maybe next time, when I tried all possible solutions for a problem and still get stuck, I’ll try the most simple but effective way: restart!!!

I find, when I do normalization for both training test data, the result is as expected. But if not doing it, the accuracy and loss remain the same.

What I’ve done, and does it work?

|  |  |
| --- | --- |
| Deeper the network (5 to 7) | X Accuracy almost the same ~8.2 |
| Normalize the data before put into network | * Yes, it works! |
| Different initialization method for w, b  From truncated\_normal to xavier | X even worse |
| Change the batch size | The smaller seems the better accuracy |