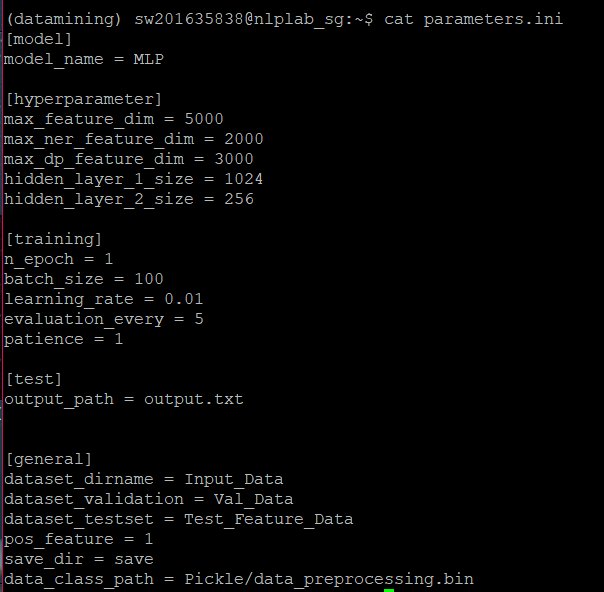
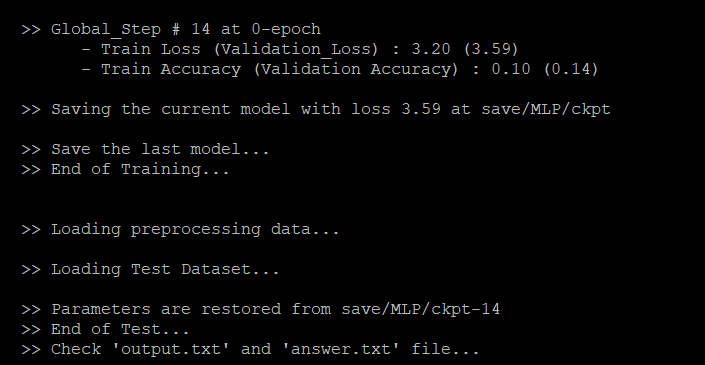
**Datamining HW 3**

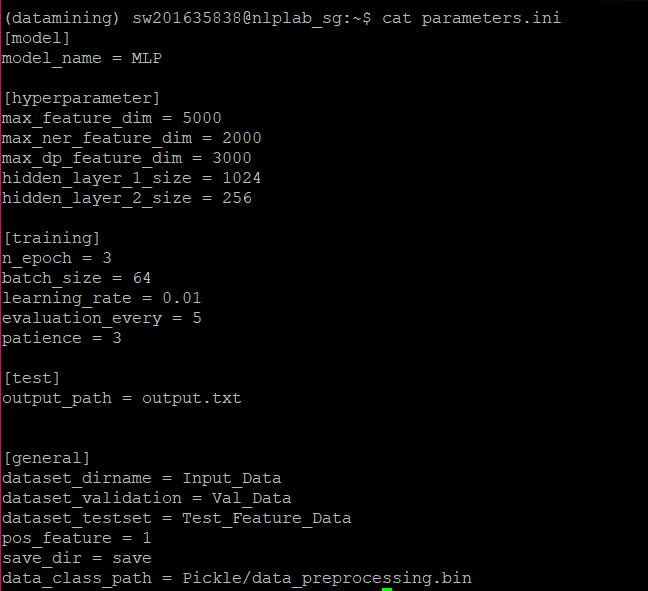
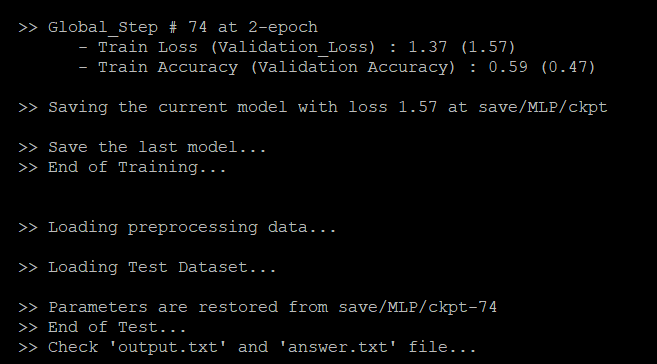
201635838 이예지

Before I showed you the results, I thought that when analyzing learning performance, not only train loss but also validation loss was important. So I tried to lower the validation loss and increase the accuracy.

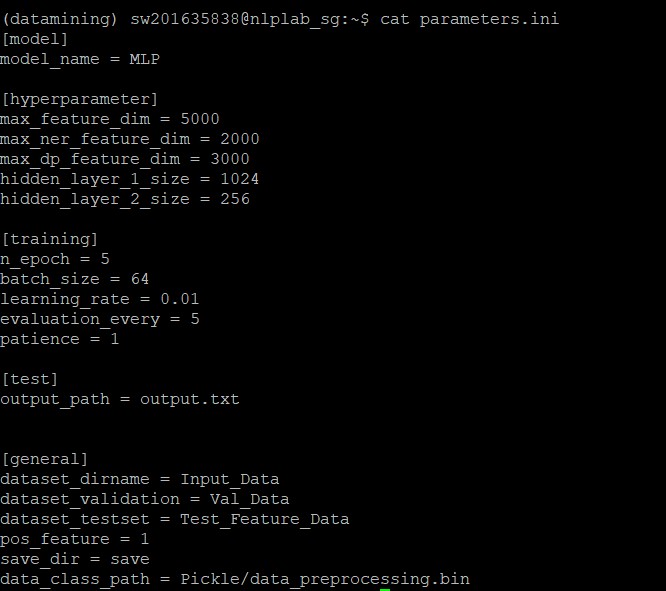
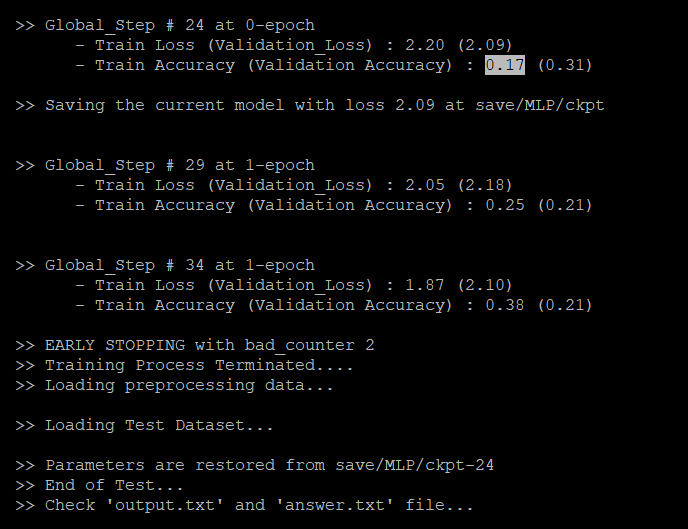
**#1. Default parameters**

This was achieved after the first run with learning data from Implementation2. The total loss was 3.59.

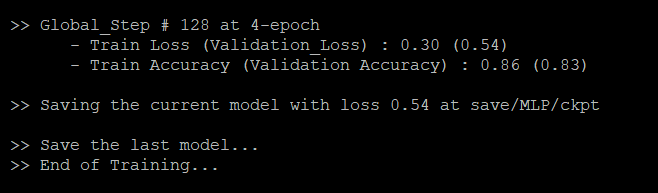
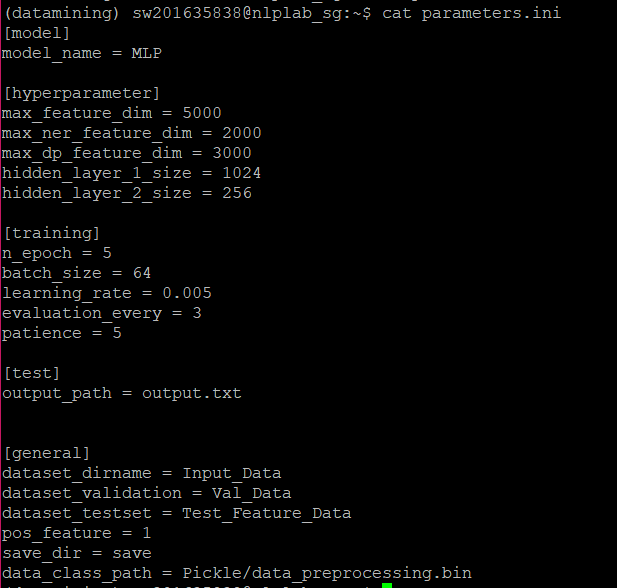
I thought that if we adjust the **N\_epoch**, which indicates the number of learning times in the training set, and the **batch\_size**, which represents the number of data being learned at a time, we'd have less loss.

**#2.**

The loss was 1.57, less than before, but **n\_epoch** was adjusted to make the loss less than 1.

**#3.**

EARLY STOPPING occurred when **N\_epoch** was increased. This is due to a lack of value in **patience**, so I increased it to the same value as n\_epoch. Then, I reduced the **learning\_rate**, which is a parameter that determines how much to learn once you learn, to produce more optimal results for that data.

**#4. Conclusion**

**Total validation loss is 0.54 and accuracy is 0.83.**

In order to optimize Neural network learning, it can be said that we are looking for hyper parameter values in a direction that **minimizes losses**, but the parameters shown in Figure #4 not only derive the minimum loss values from my data, but also derive a high accuracy. For this reason, I think my hyper parameter is perfect for model learning.