Introduction to week3

I bought a book, written by the designer and original implementer of Keras. I put the book in the repository for your reference, please do not distribute it.

This week I did two things:

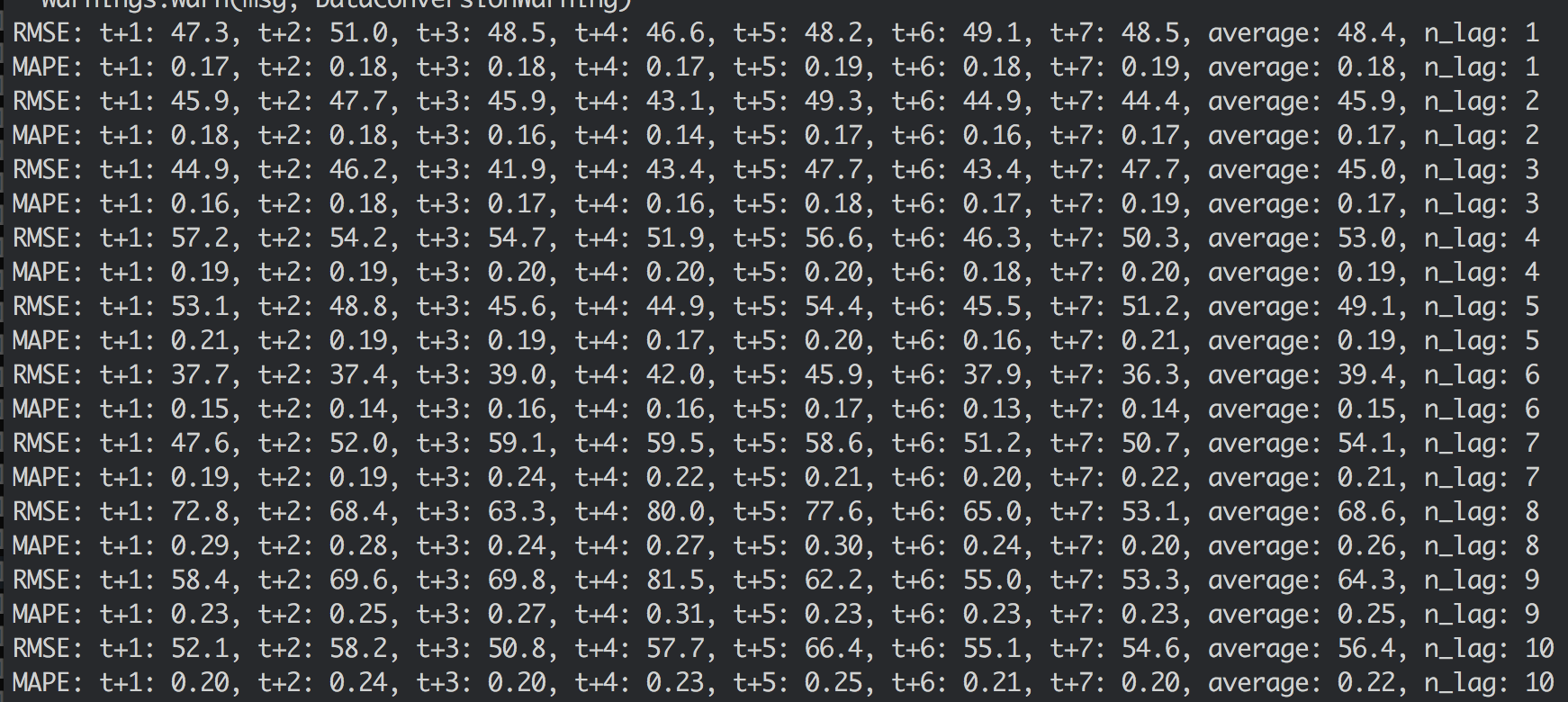
* change the number of input features, to find the differences on the result.
* Read the book “deep learning with python” chapter 1-3, and knows that the less training data we have, the worse overfitting will be (page 86). Using a small network is one way to mitigate overfitting. Our project has 300+ samples, is a typical case of a few training samples.

“A small network” should be a network with at most 2 hidden layers. From next week on, I will try to do the following:

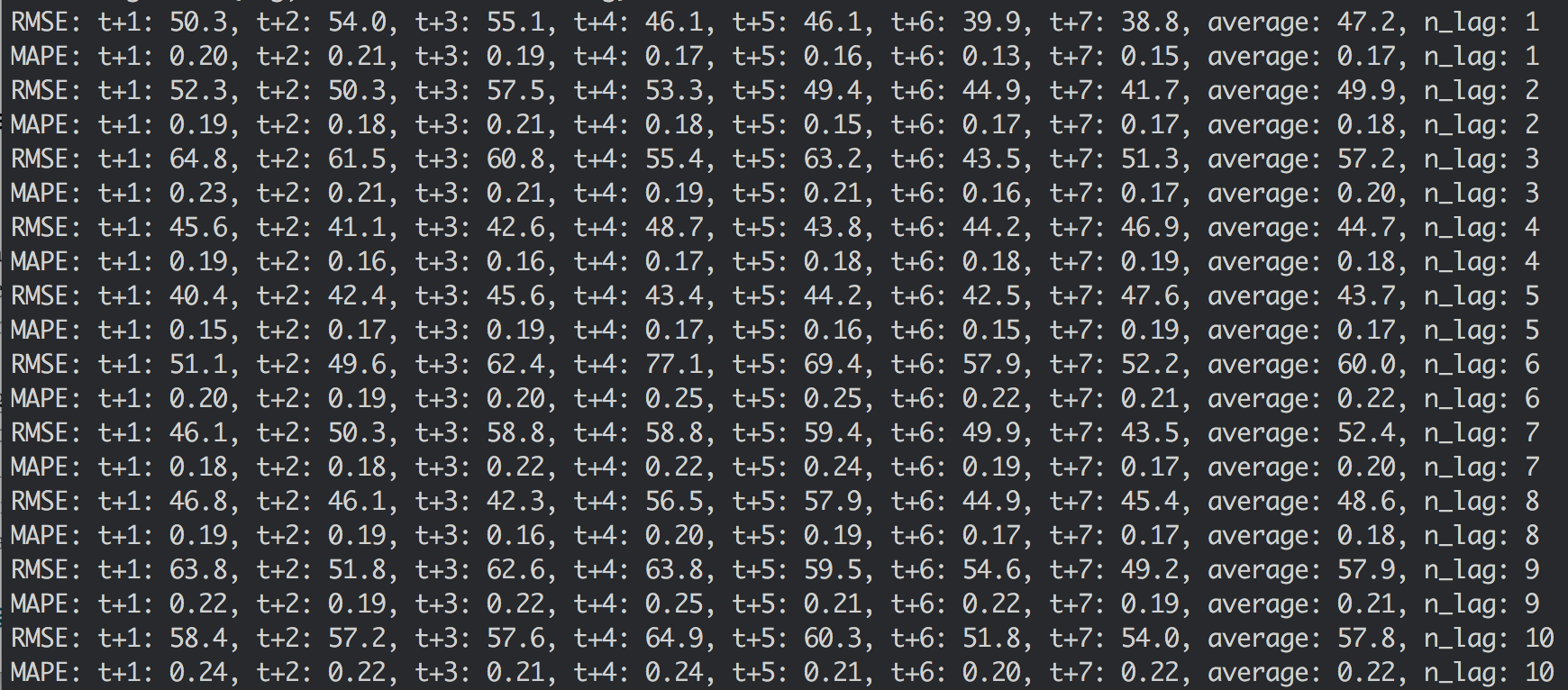
* change size of hidden layer from 1 to 2, and get the most suitable neuron number.
* Change number of epochs, figure out the overfitting zone

# Performance under different number of features

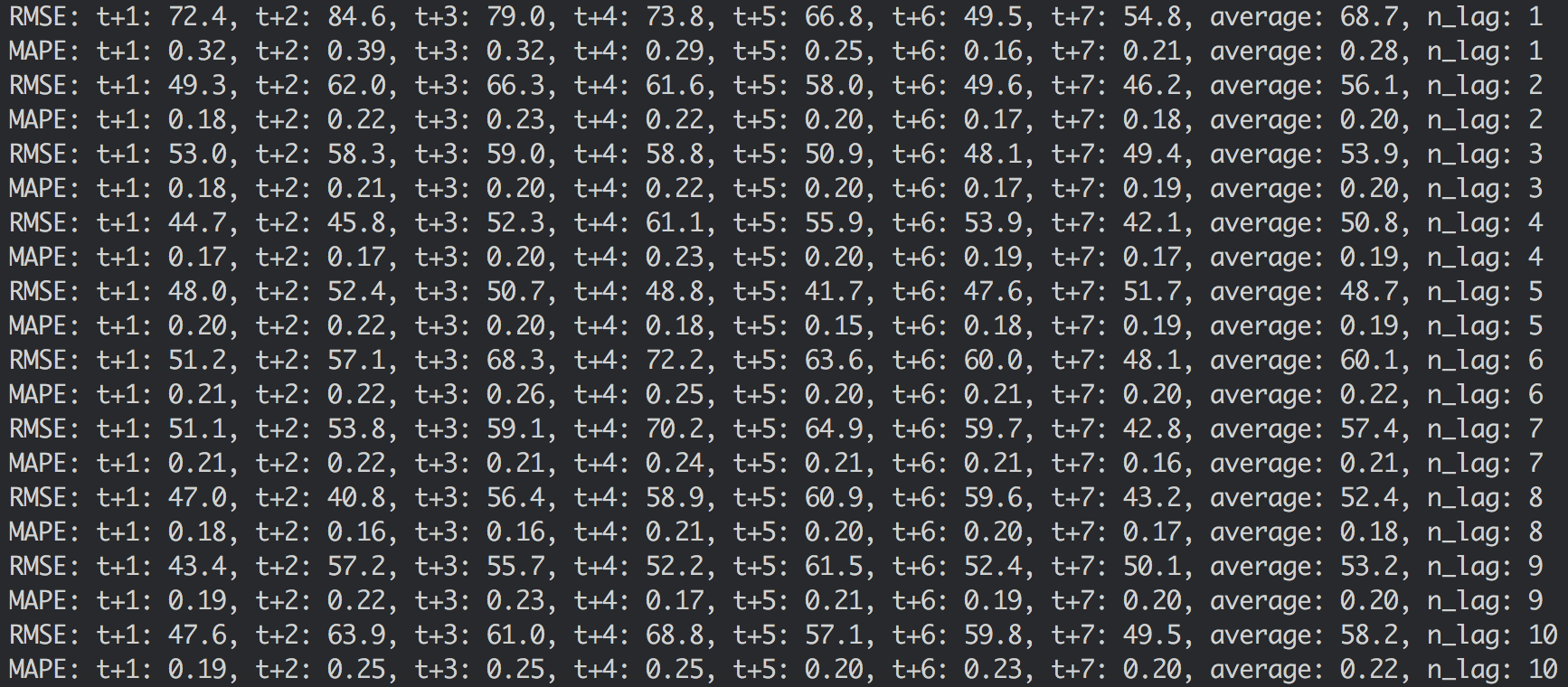
We set the n\_lag’s scope from 1 to ten, and limit the randomness of the code (make the result reproducible), get the following result (50 epoch’s training each turn):



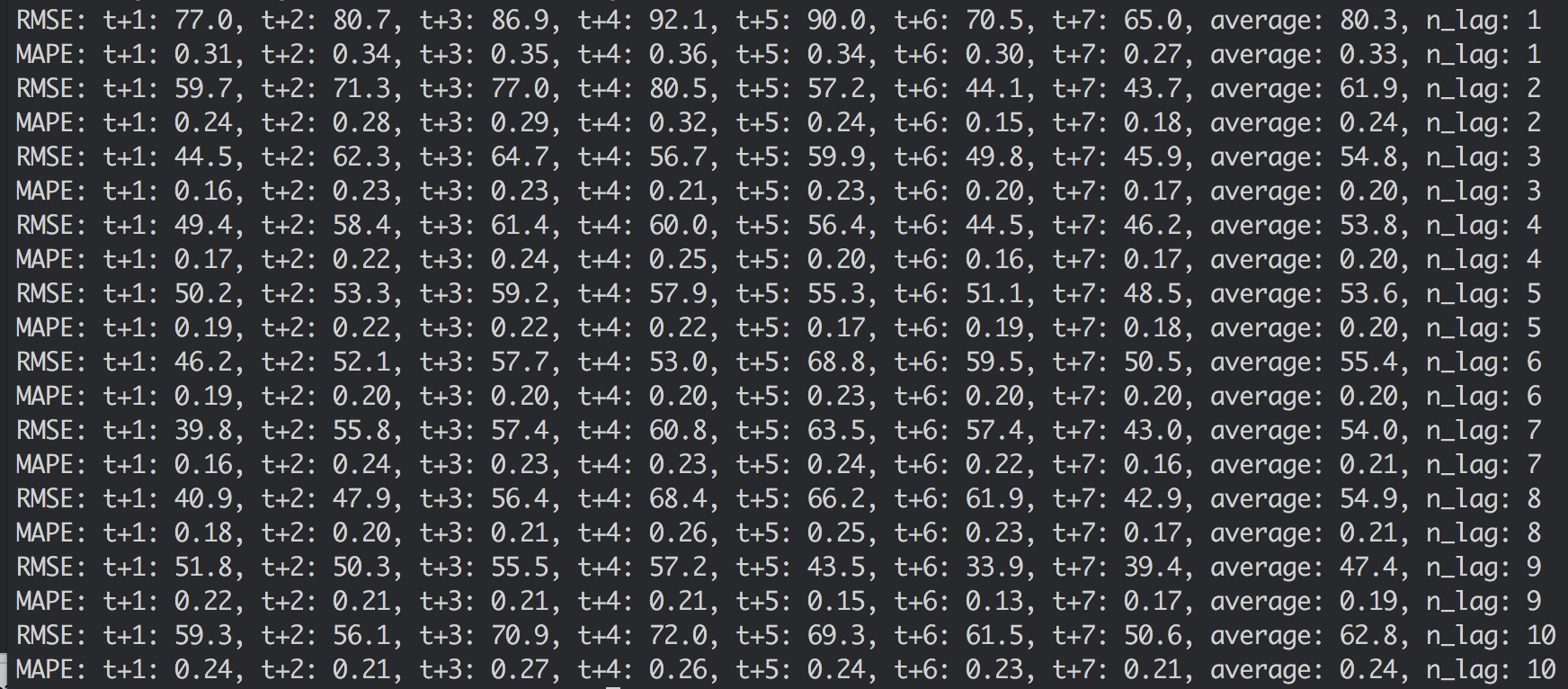
Now we change the initial settings of random number (still keep the result reproducible), get the following result (50 epoch’s training each turn), but can’t find any clue, it seems that the number of n\_lag has no influence on the result:

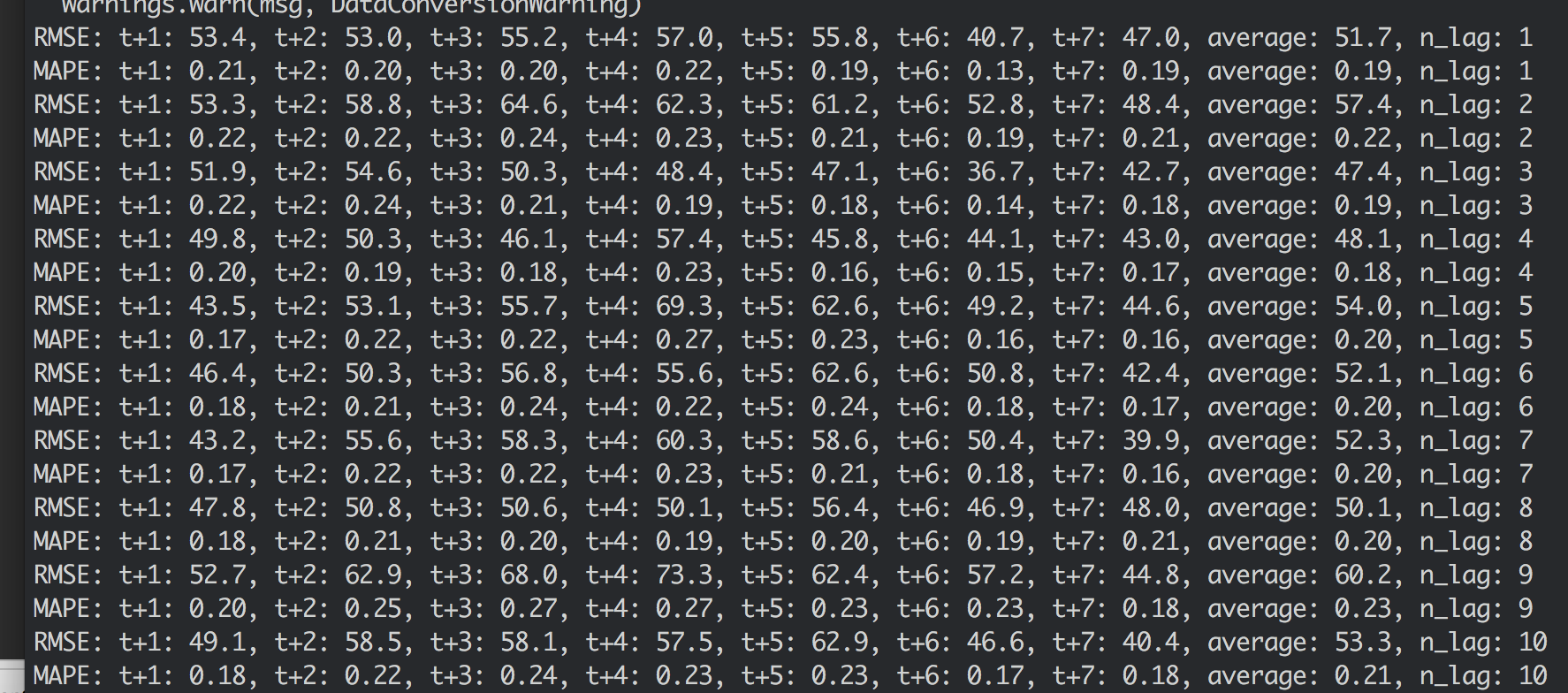


Now we increase the number of training epoch to 500 (a long time to wait), and get the following result:



Change random numbers, another result of 500 epoch:





I changed the random number’s combination (20 combinations), using mstep\_loop\_lag.py, and made 500-epoch tests, the result could be found in lags\_log.docx.

The result showed that there’s no performance gain if we increase the number of features.