ITE4005 Data Science course

Assignment 4

2016026080 한다진

**Experiment environments**

OS Window 10

Language python 3

Tools Jupyter notebook

Pure python script

**Collaborative Filtering**

Collaborative Filtering algorithm is a content-based recommendation algorithm. Collaborative filtering algorithm is. Method of judging and. Ecommending similarities between vectors. In the basic way, if vectors of two users are similar, we judge those similar and recommend the movice each other.

To solve assignment\_4, I embedded user id and. Movie id to the. Vectors with same shape. And I assume that the vector would be similar to the genre. Vector of user would be similar to the genre user likes, and vector of movie would be similar to the genre of movie. And I try to calculate the similarity of movie vector and user vector. If the scalar multiplication of user vector and movie vector is big, I judge the genre user likes and the genre movie belongs to overlaps a lot. So I assume Collaborative Filtering Model using Deep Neural Network would work.

I make a CFModel class with tf library

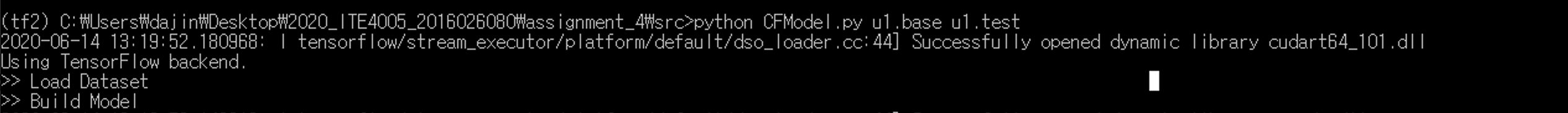
1. Embedding input user id and input movie id
2. Scalar multiplication of embedded vectors
3. Compare the pred\_y and answer\_y.
4. Back propagation with MeanSquaredError loss function

**Quick Start**

* + Run CFModel.py

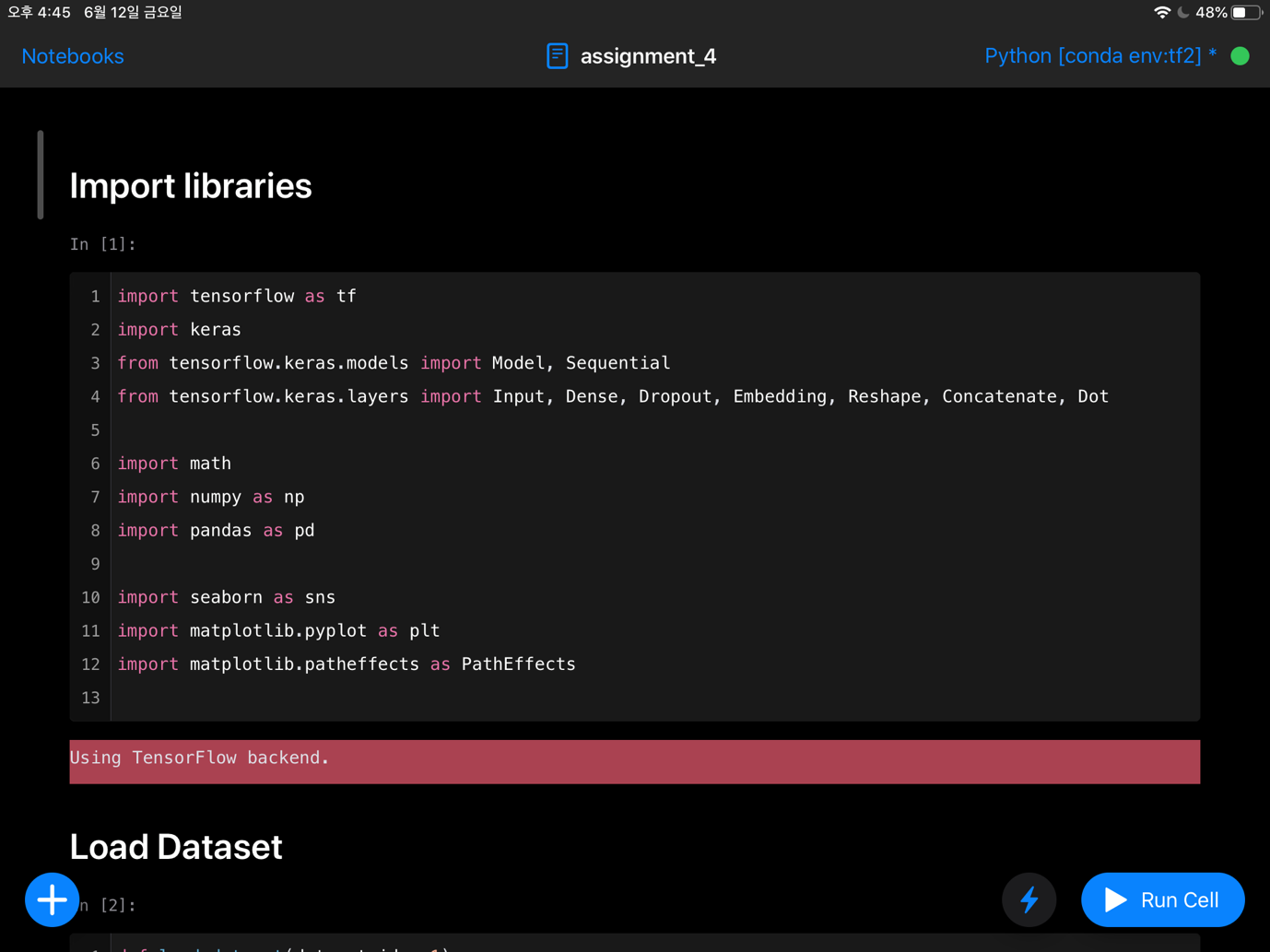
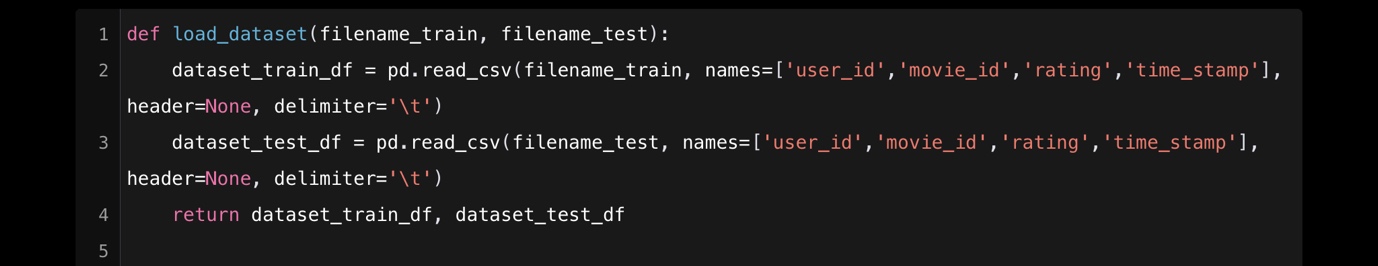
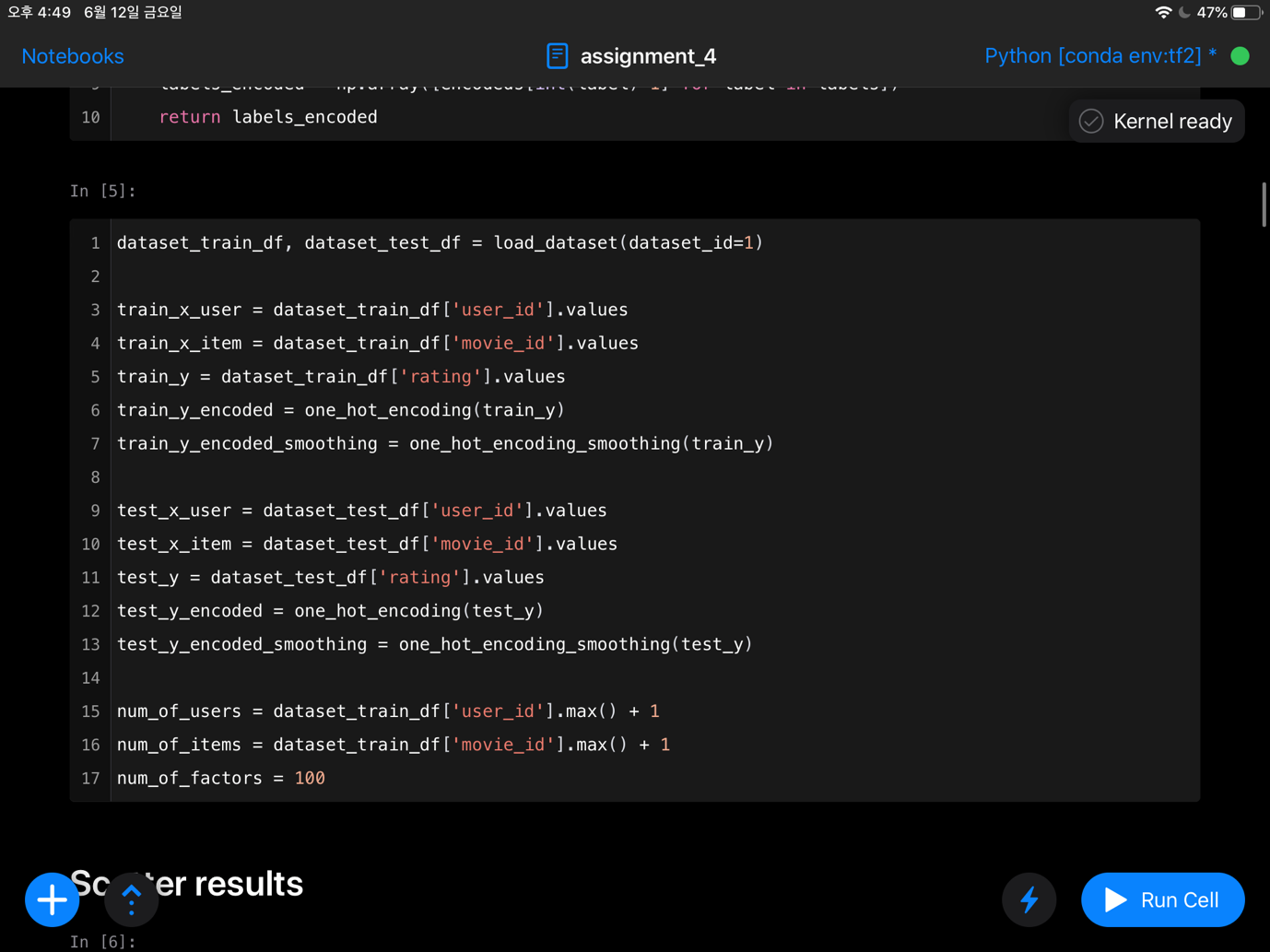
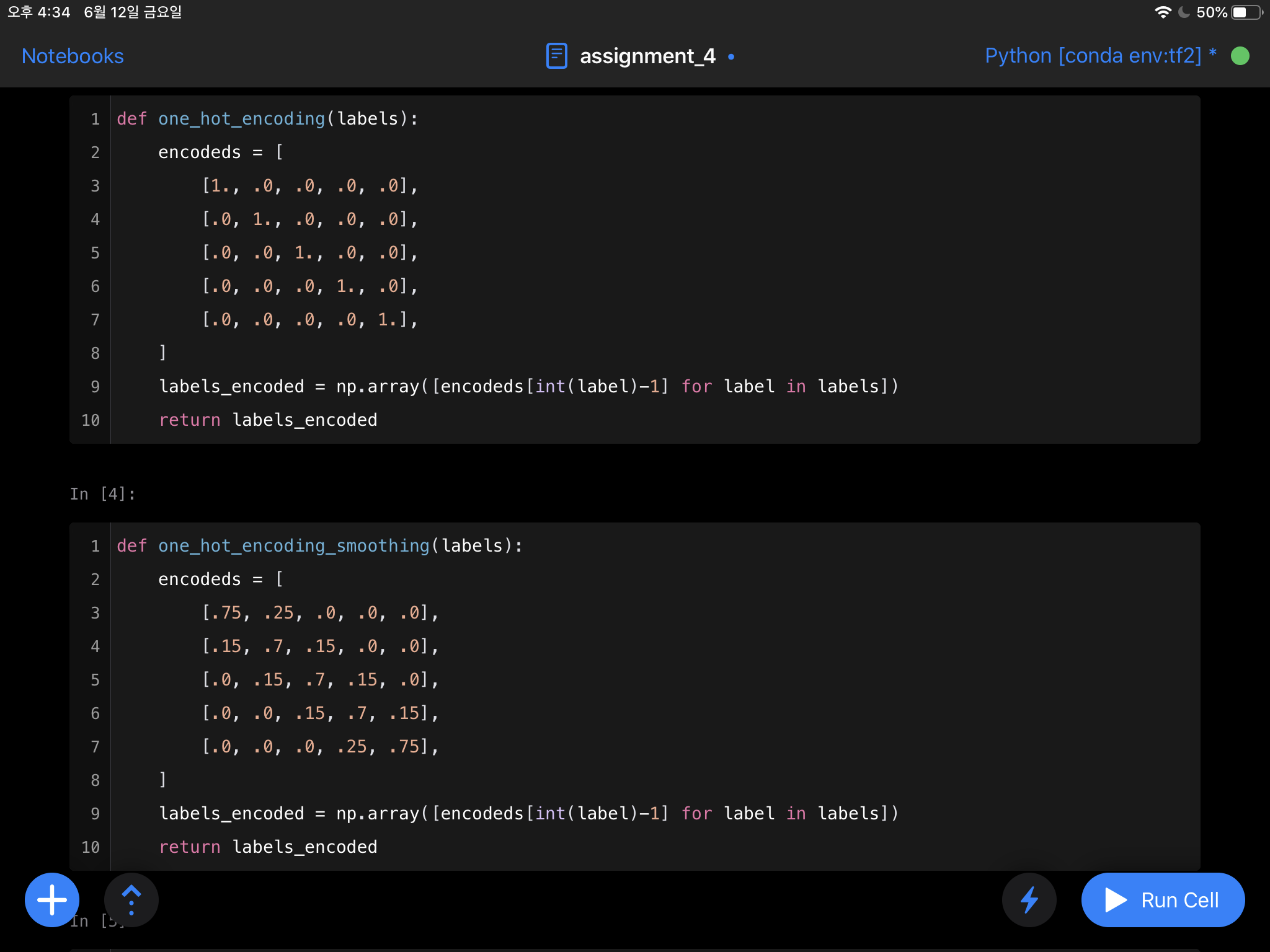
(use trained model and test) Python CFModel.py u1.base u1.test

(train model and test) python CFModel.py u1.base u1.test train

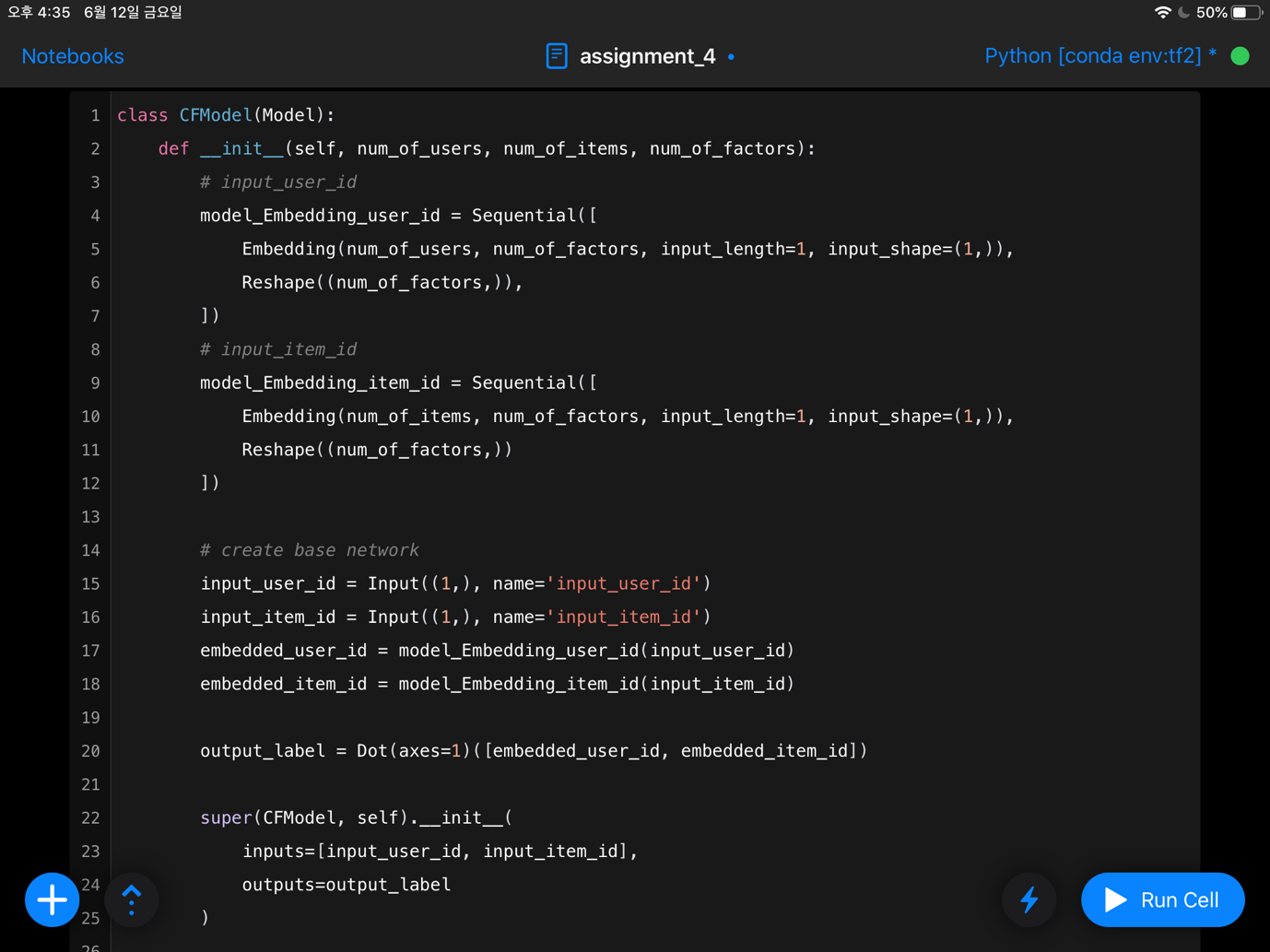
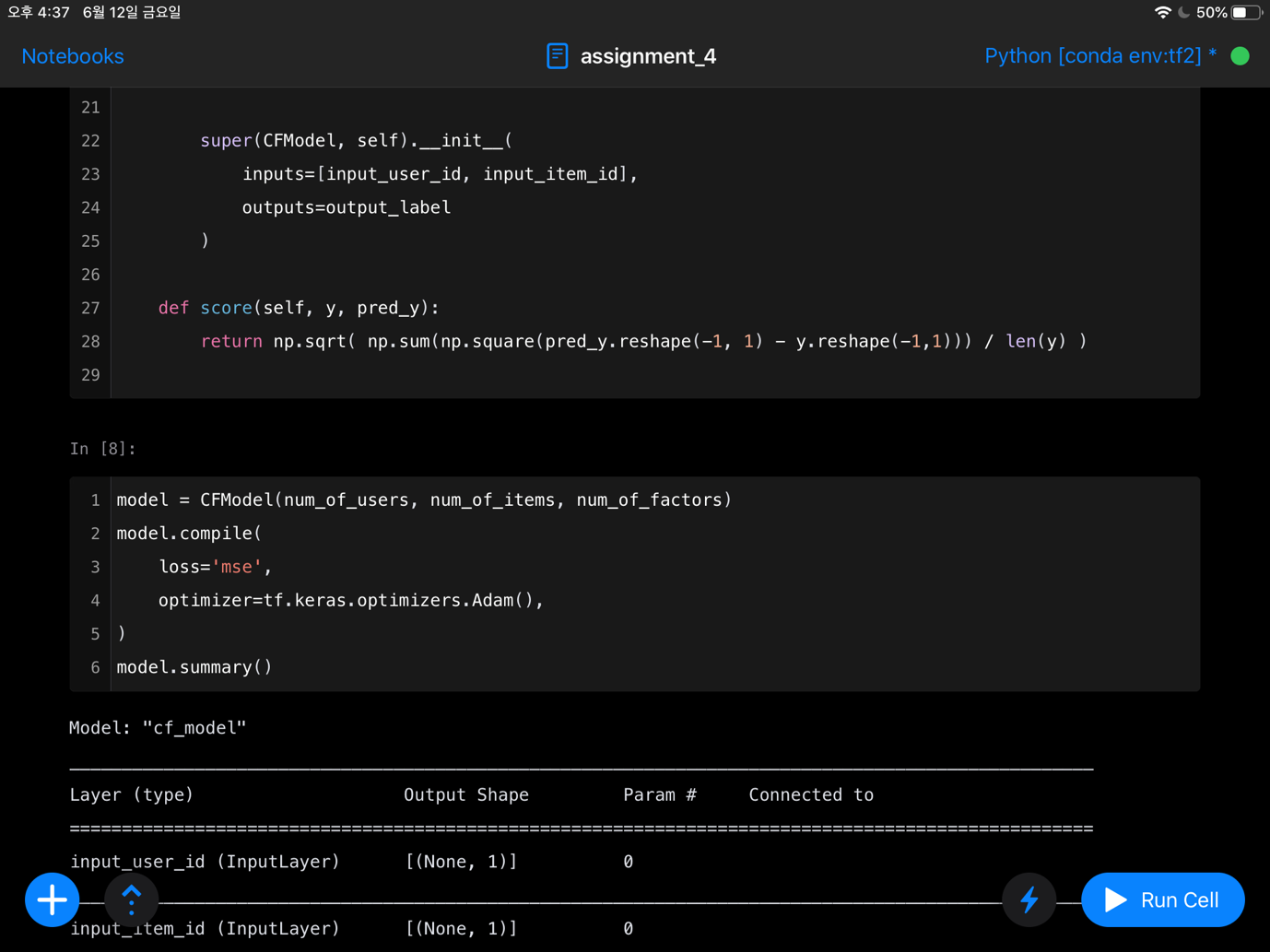




**Code explanation**

* + Import Libraries  
      
    Import keras to make a CollaborativeFilterling model.  
    Seaborn, matplotlib libraries are used to check the distribution of dataset.
  + Load Dataset  
      
    Load dataset and make preprocess it as a training sample. x\_user variable and x\_item variable is a id of input dataset, and y is an target label, in this case ‘rating of movie’. Y\_encoded is an one-shot encoded target, and y\_encoded\_smoothing is an one-shot encoded target with smoothing technique.
  + Encoding target data to use for standard DNN model.

Encode target label for DNN model(my failed models). These functions are not used in CFModel.

* + **Collaborative Filtering Model Class**

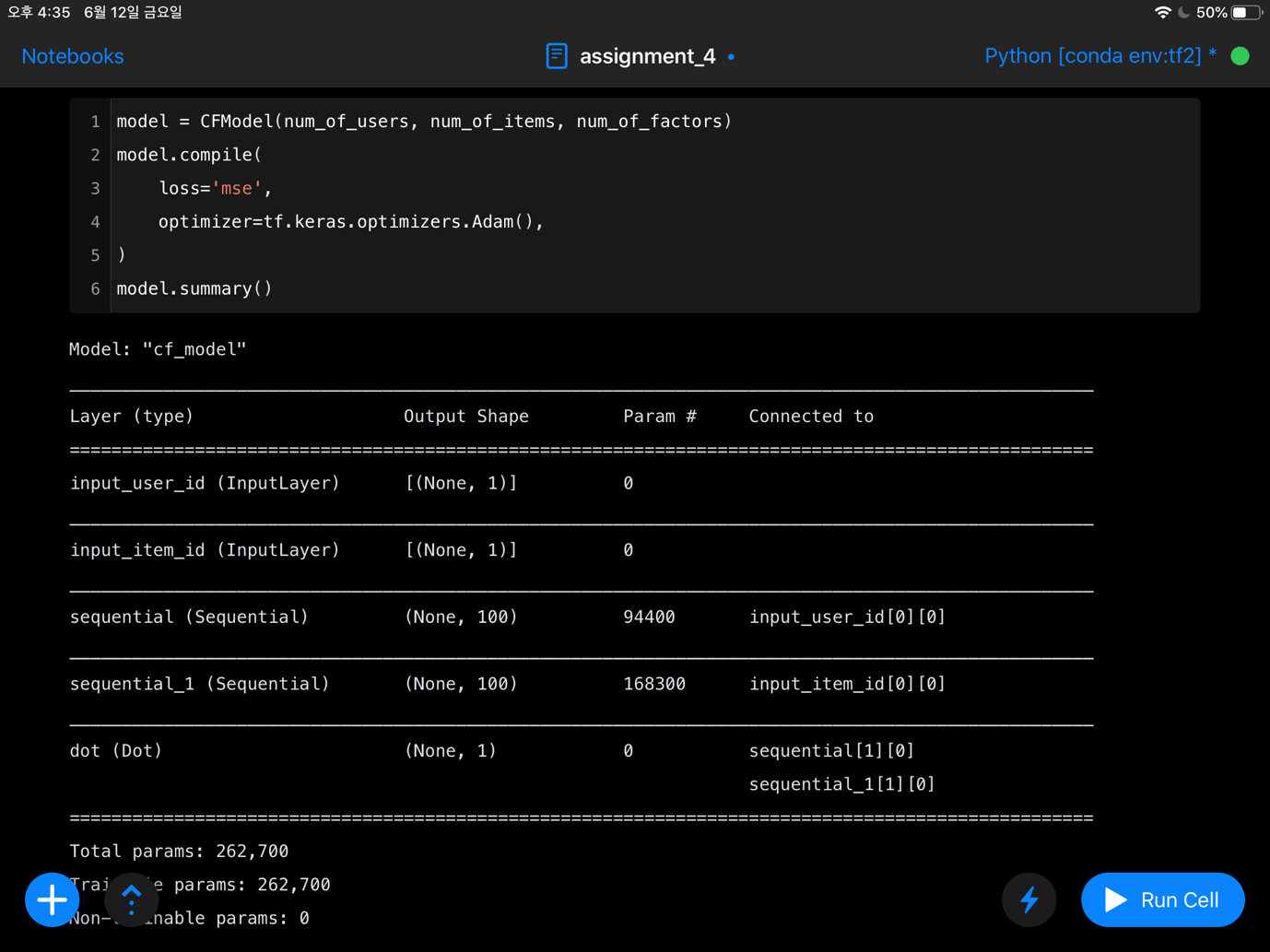
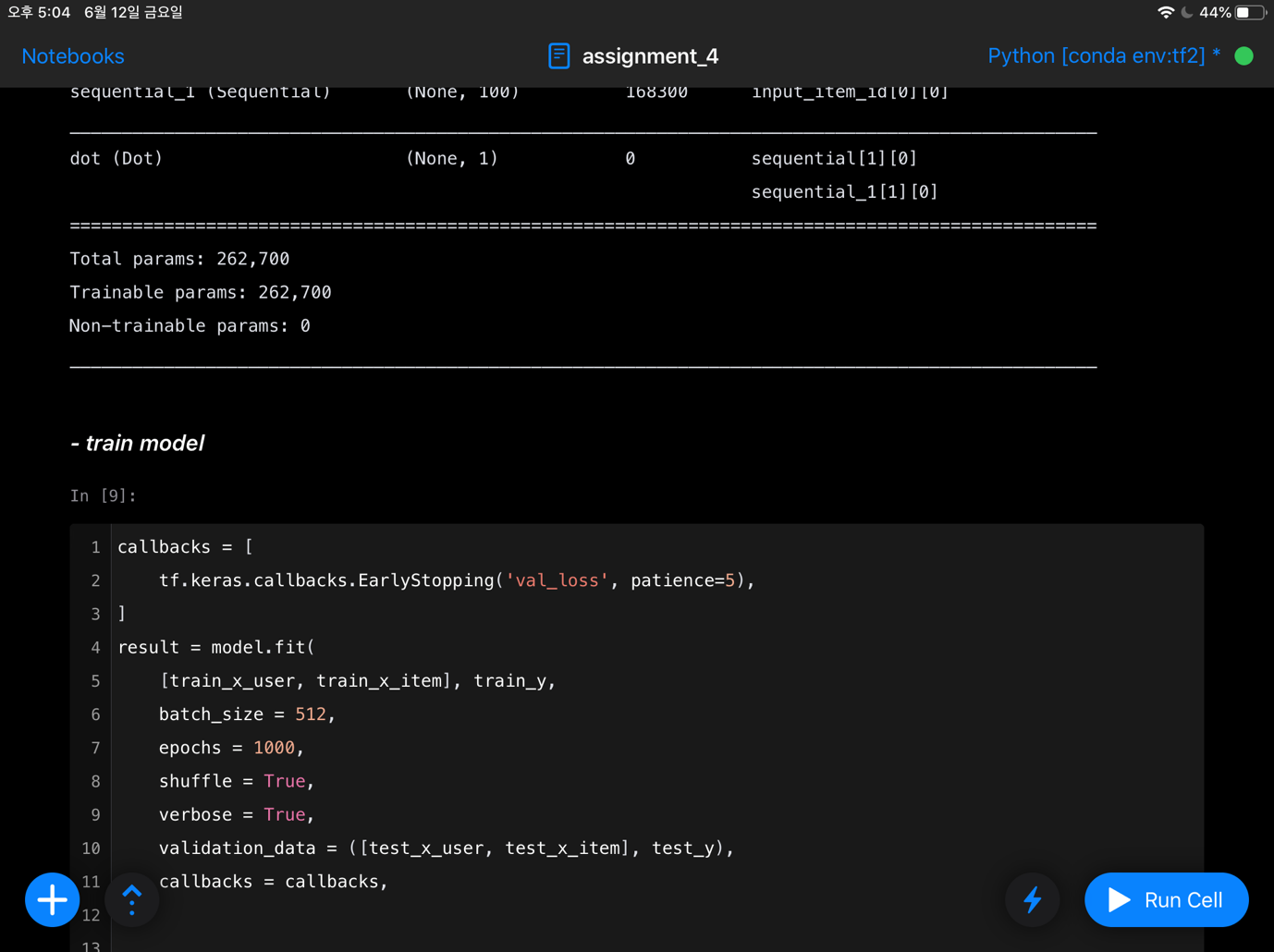
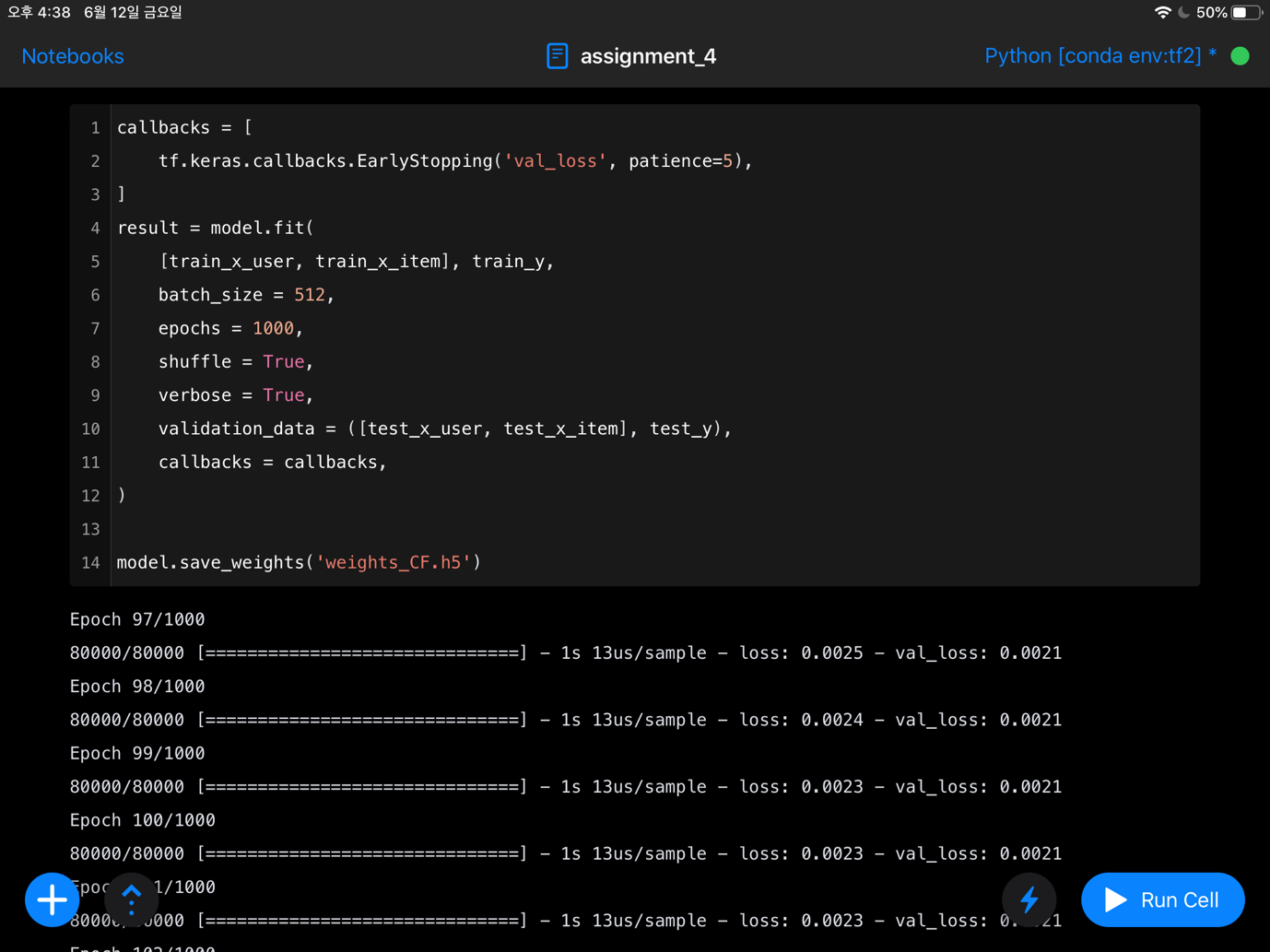
Make CFModel class using super function to inherit the attribute of tf.keras.models.Model class. Just using this class, I can make a completed tensorflow model.

There are two inputs. One is the id of user, and the other is id of movie. CFModel embedding these inputs to vector. The length of vector is num\_of\_factors. The reason for embedding with vectors is that the deep learning model has the characteristic of trying to learn the continuous characteristics of its value together when the type of input is number. Because the ID is an individual number, it is embedded in vector value.

After embedding input id to vector, Do scalar multiplication and get the pred\_y. train the model with mean\_squared\_error loss function. And activation function is relu and linear.

And make score function to get the RMSE value.

Details are mentioned above.

* + Create CFModel and summary
  + Train model and results

스크린샷, 모니터, 화면, 앉아있는이(가) 표시된 사진

자동 생성된 설명모니터, 화면, 앉아있는, 사진이(가) 표시된 사진

자동 생성된 설명Train model and save the weights.

* + **Get RMSE of predictions  
    텔레비전, 음식, 화면, 플레이어이(가) 표시된 사진

    자동 생성된 설명**

**Experiment**

앉아있는, 스크린샷, 병, 테이블이(가) 표시된 사진

자동 생성된 설명

Average RMSE value is 0.925 for u1 to u5 test samples.

**Other Trials**

I tried to build Deep Neural Network with other shape. Using Dense layers, using one-shot encoding, using smoothing technique, using different loss function… but standard DNN model works worse. CFModel works best. Maybe its because of the power of collaborative filtering. DNN has good performance only if the theory is good and the implementation is successful.

In most other trials, the value of RMSE is near 1.0

Optimizer attempted : adam, adamax, bce, RMSDrop

DNN model attempted : CFModel, CFModel\_onehot, DNNModel\_onehot

Dataset attempted : label encodings, one-hot encodings, one-hot encoding with smoothing

Activation functions attempted : ReLu, softmax, linear,

With 4 x 3 x 3 x 3 x (num of learning rates) trials, I found that combination of CFModel and Adamax optimizer has the best performance. With 0.92 value of RMSE. Others trained until the RMSE value is 1.2

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명