Source:

- 신경망 첫걸음, 타리크 라시드, 한빛미디어
- 딥러닝 첫걸음, 김성필, 한빛미디어

20170534

Introduction to machine learning

Lecture #12 Python programming for ANN – 2

Instructor: Jeon, Seung-Bae

Assistant Professor
Hanbat National University
Department of Electronic Engineering

9th week quiz solution

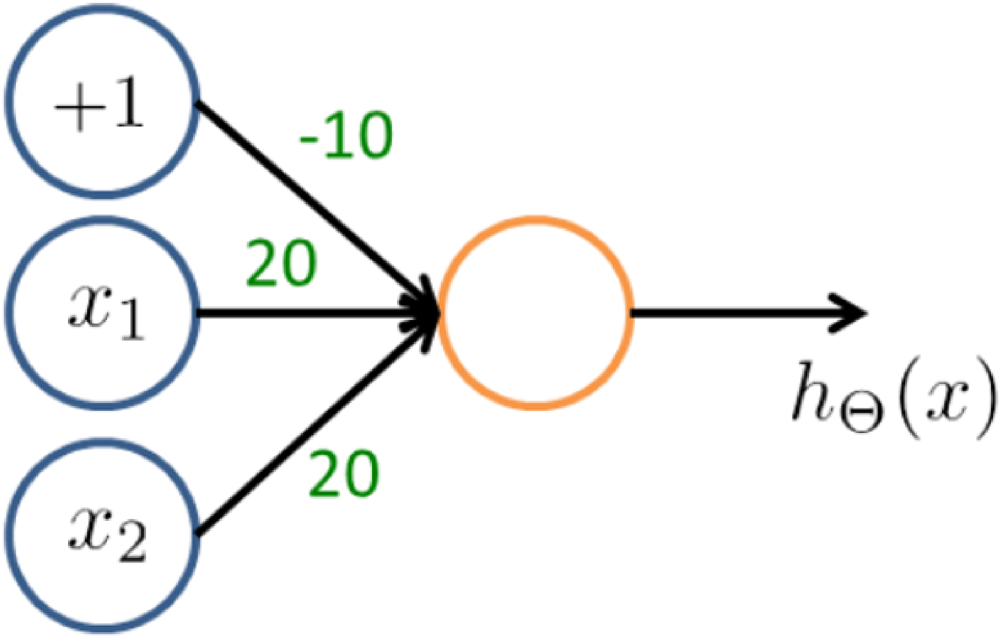
1. **Logic operation with the logistic unit**

1.1 Write the name of the logic operation which is made by the following logistic unit.

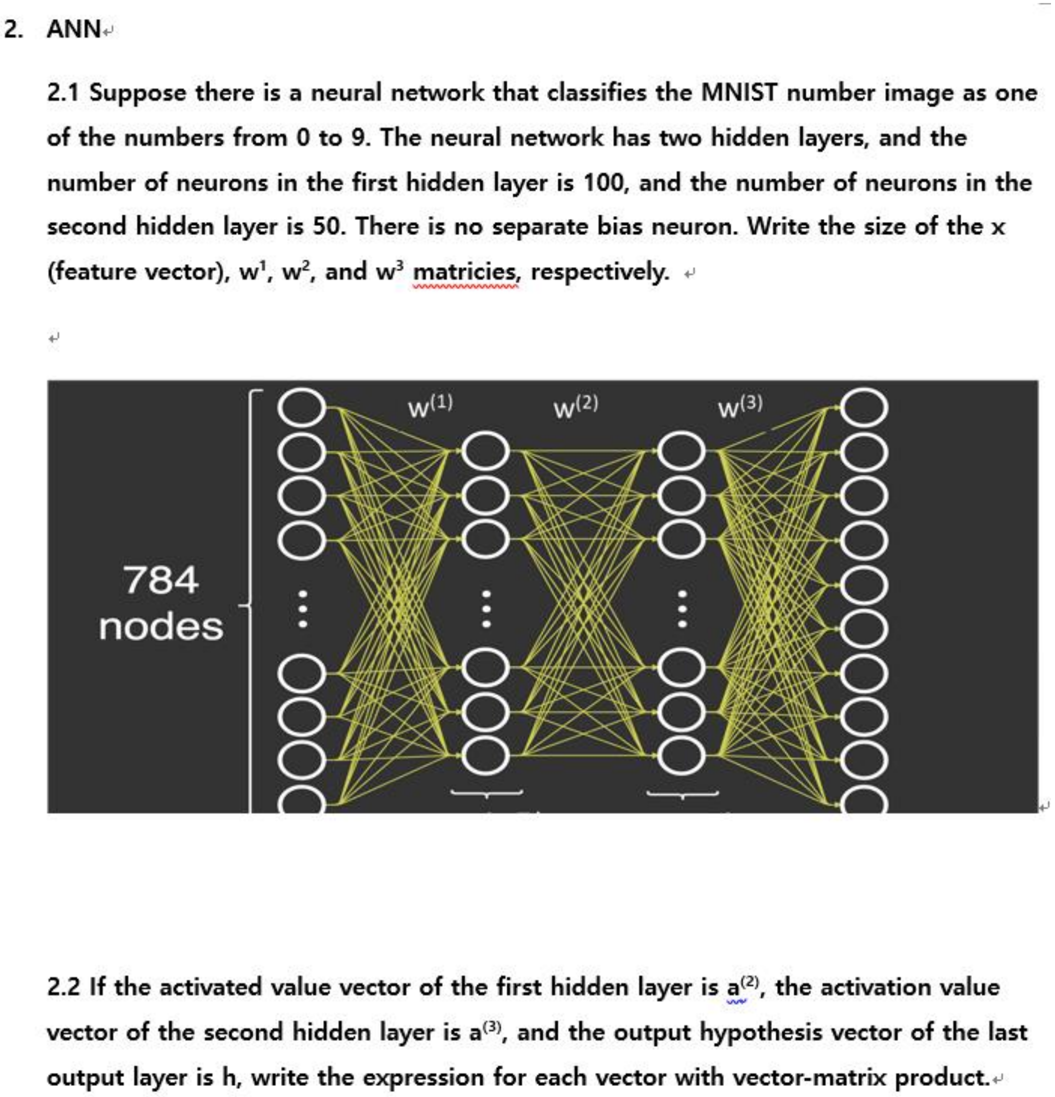
1.2 Suggest new weight values for the same operation.

(Now, $w_{01}^{(1)}$= -10, $w_{11}^{(1)}$=20, $w_{21}^{(1)}$= 20)

x_1	x_2	z	



9th week quiz solution



2. ANN

2.1 Suppose there is a neural network that classifies the MNIST number image as one of the numbers from 0 to 9. The neural network has two hidden layers, and the number of neurons in the first hidden layer is 100, and the number of neurons in the second hidden layer is 50. There is no separate bias neuron. Write the size of the x (feature vector), w¹, w², and w³ matricies, respectively.

2.2 If the activated value vector of the first hidden layer is $a^{(2)}$, the activation value vector of the second hidden layer is $a^{(3)}$, and the output hypothesis vector of the last output layer is h, write the expression for each vector with vector-matrix product.

$W_{(1)} = 784 \times 100$

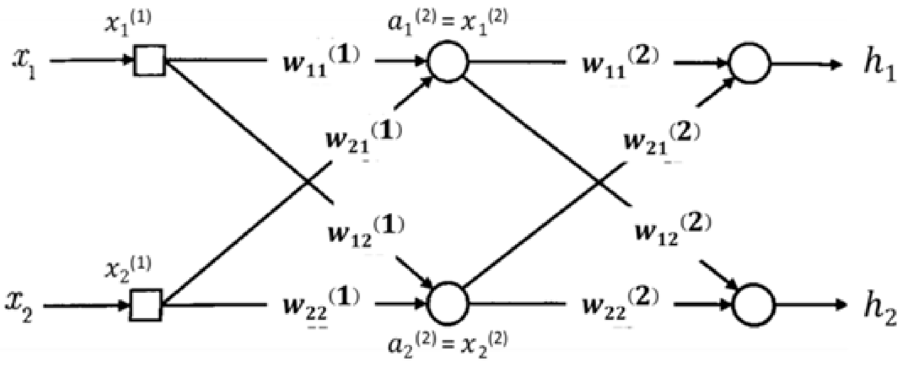
$w^{(2)} = \cancel{784} \times 50$
100

$w^{(3)} = 50 \times 10$

$z^{(2)} = x \cdot W^{(1)}$

$a^{(2)} = g(z^{(2)})$

10^{th} week quiz solution

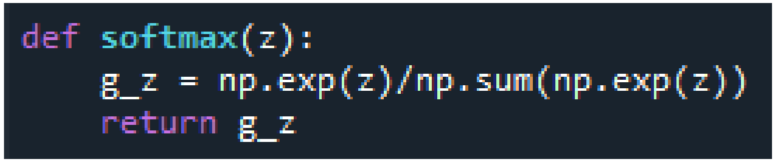


1.1 Express the learning process of the neural network with one hidden layer as above in a matrix form (in the same form as written in the lecture notes 27-28 pages)

※ Use the Cross entropy as the cost function, and tanh is used as the activation function, not sigmoid. Try to write Tanh's derivative equation at the top of. e.g.) g'(x) = ~~. After that, you can just represent that with g'()

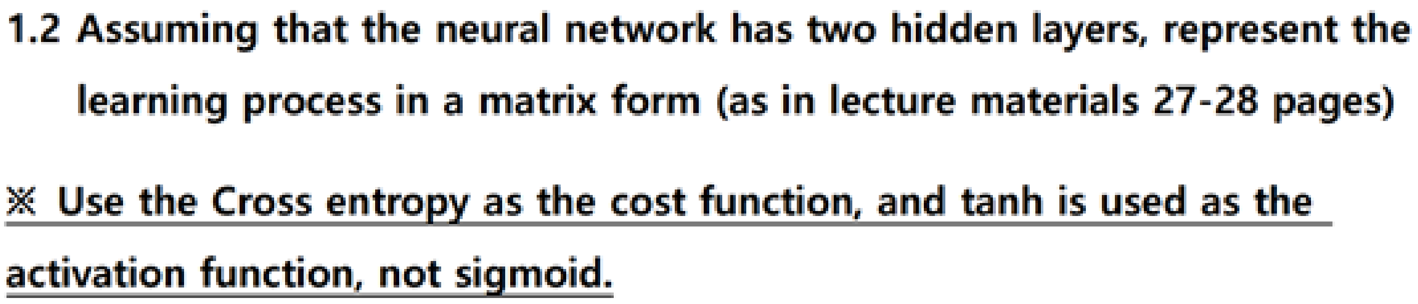
보통 출력단에서만 활성화함수로 softmax 함수를 사용!

$$\text{Softmax(z)} = \frac{\exp(zi)}{\sum \exp(zj)}$$



```
def softmax(z):
    g_z = np.exp(z)/np.sum(np.exp(z))
    return g_z
```

10th week quiz solution



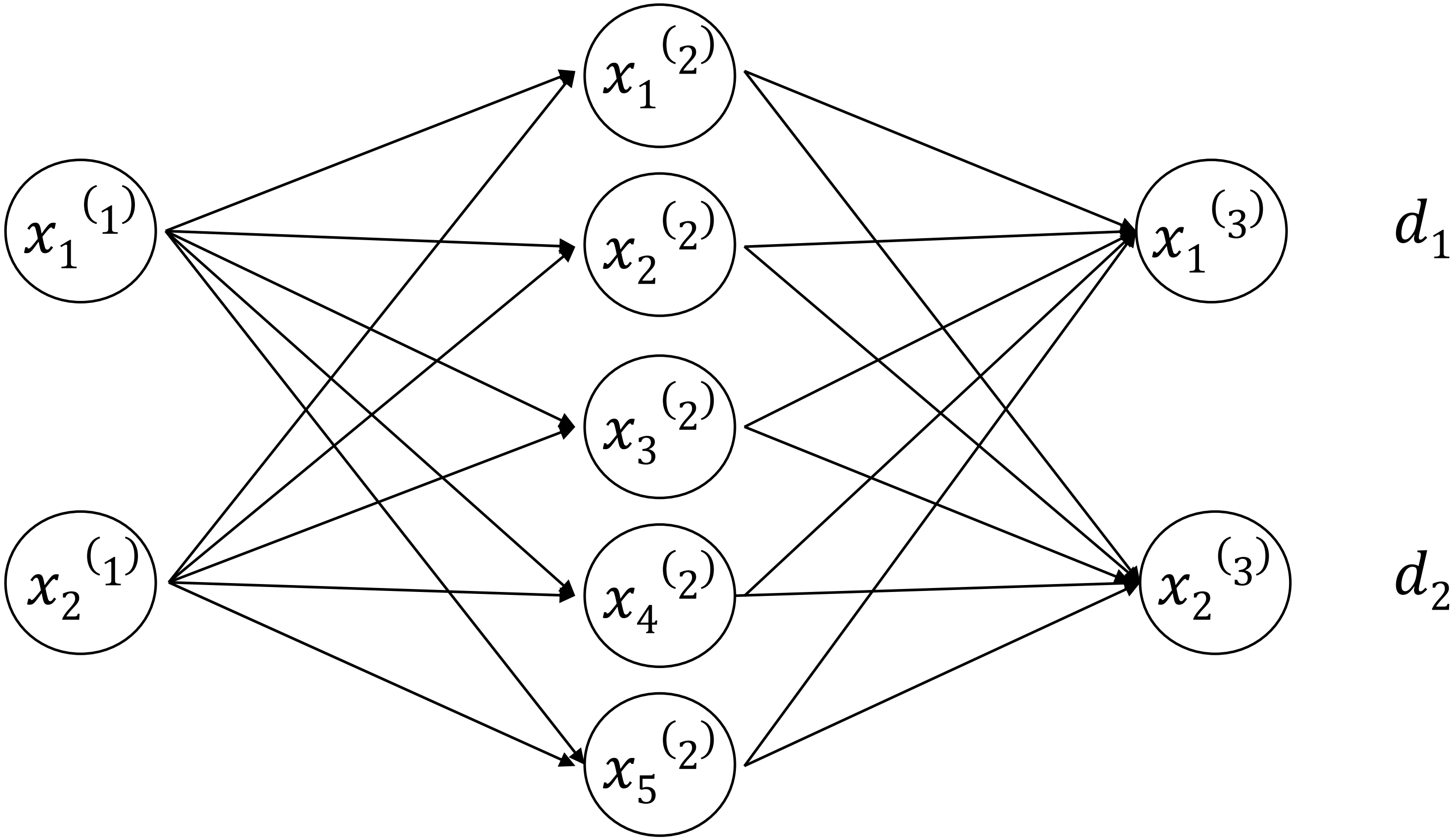
1.2 Assuming that the neural network has two hidden layers, represent the learning process in a matrix form (as in lecture materials 27-28 pages)

※ Use the Cross entropy as the cost function, and tanh is used as the activation function, not sigmoid.

ANN programming – XOR

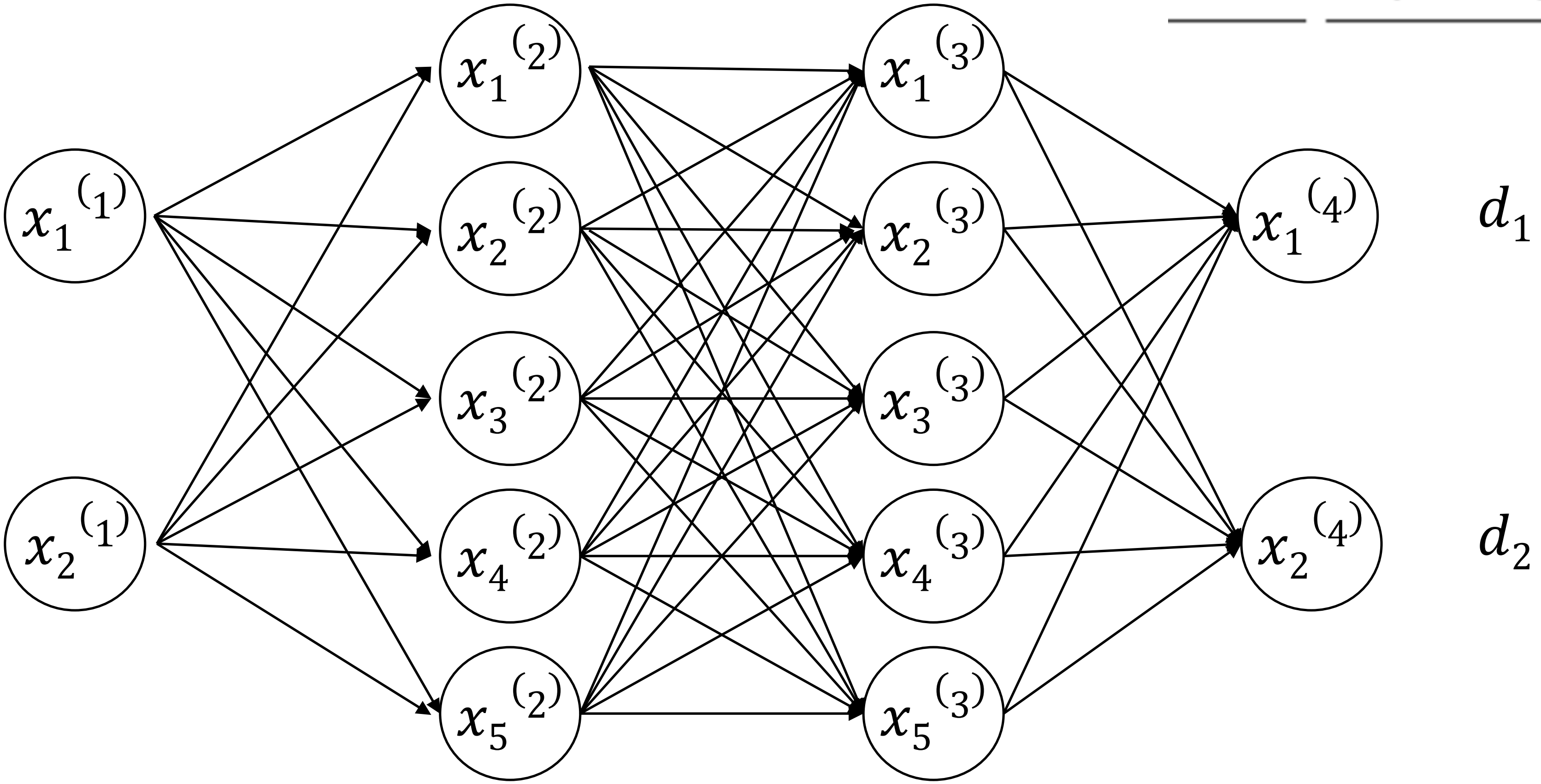
XOR	Input 1	Input 2
0	1	1
1	1	0
1	0	1
0	0	0

- **Let's make the ANN with the 5 neurons in the hidden layer. (there is no bias)**



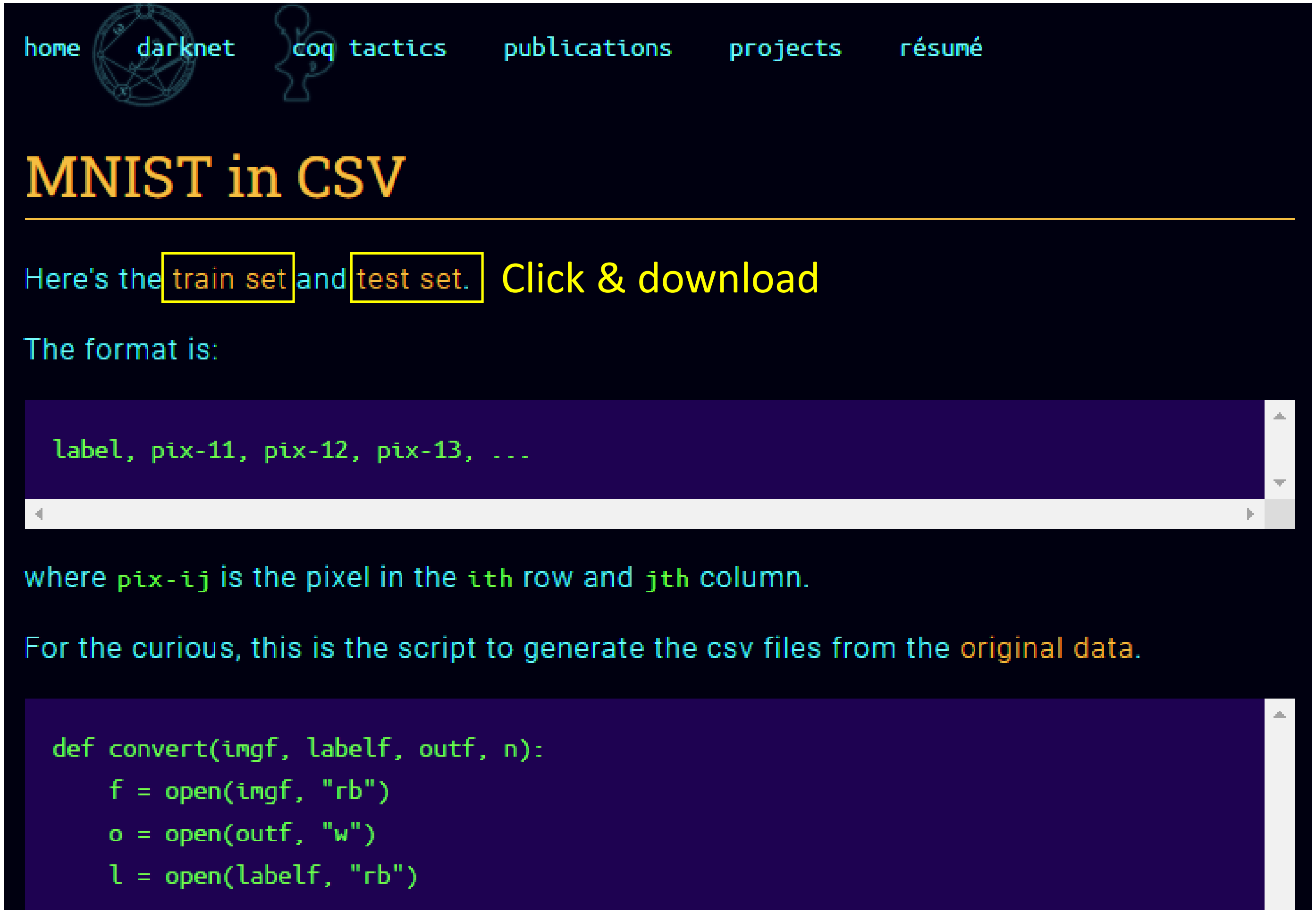
ANN programming – XNOR

XNOR	Input 1	Input 2
1	1	1
0	1	0
0	0	1
1	0	0



MNIST data

- https://pjreddie.com/projects/mnist-in-csv/



MNIST in CSV

Here's the train set and test set. Click & download

The format is:

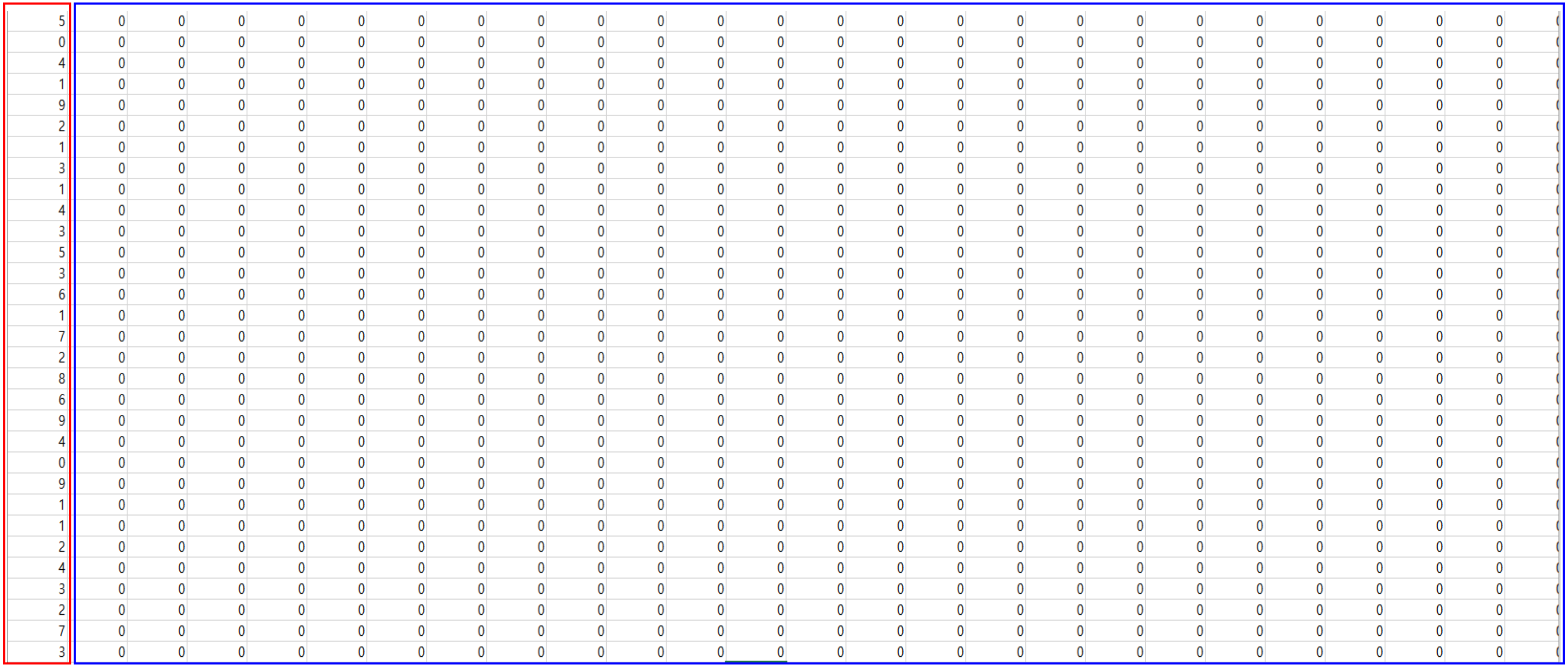
```
label, pix-11, pix-12, pix-13, ...
```

where `pix-ij` is the pixel in the `ith` row and `jth` column.

For the curious, this is the script to generate the csv files from the original data.

```
def convert(imgf, labelf, outf, n):
    f = open(imgf, "rb")
    o = open(outf, "w")
    l = open(labelf, "rb")
```

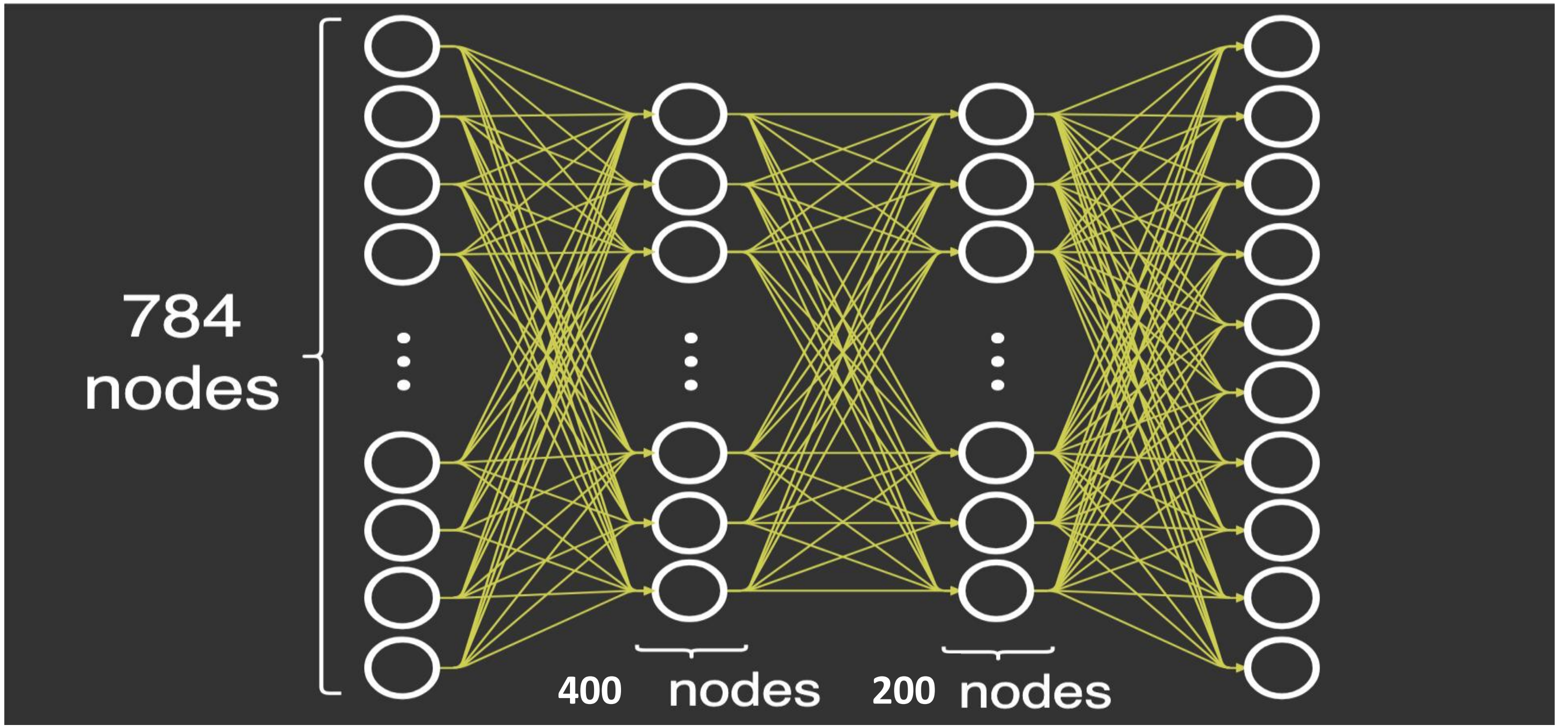
MNIST data



5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Label Pixel value (x_1~x_{784})

MNIST Programming (Project 2)



MNIST Programming (Project 2)

- We should modify the number of neurons

- The test should be carried out with the test set
  → Statesments for Test set open, readline & close are needed.
  → Classification accuracy & cost calculation should be conducted with the test set.

- Use the mini-batch (start from the batch size of 5)

- Activation function : tanh, use the softmax only for the output layer

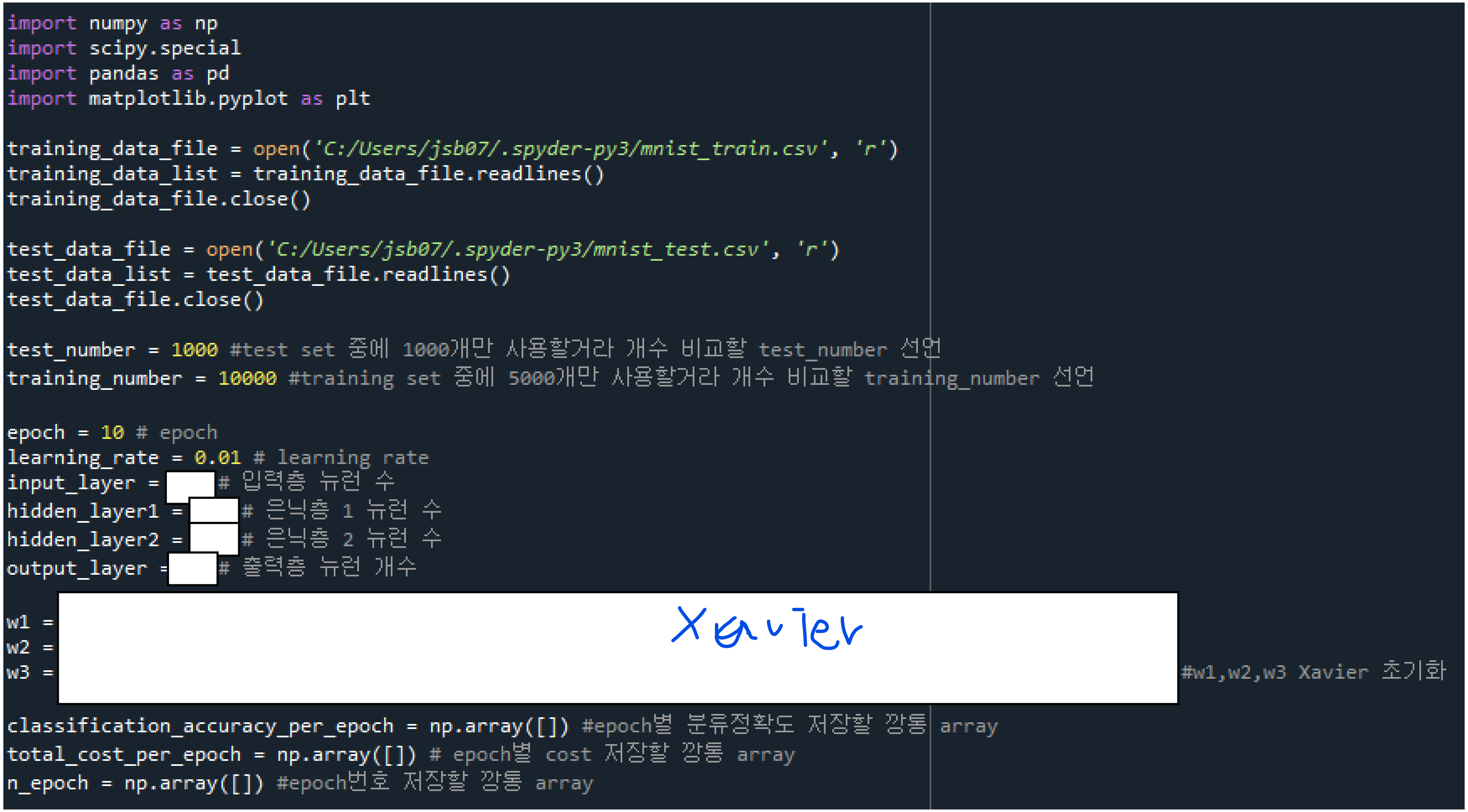
- We will only use the first 10,000 examples in the training set and the first 1,000 examples in the test set (because it will take so long time if we use the entire data set.)

- If you finish to implement the code with the guideline above, change the activation function (expect the softmax for the output layer) and compare the results (tanh vs sigmoid)

- Change the batch size to 100 and 200 then compare the results (5 vs 100 vs 200)

Project 2 – MNIST classification

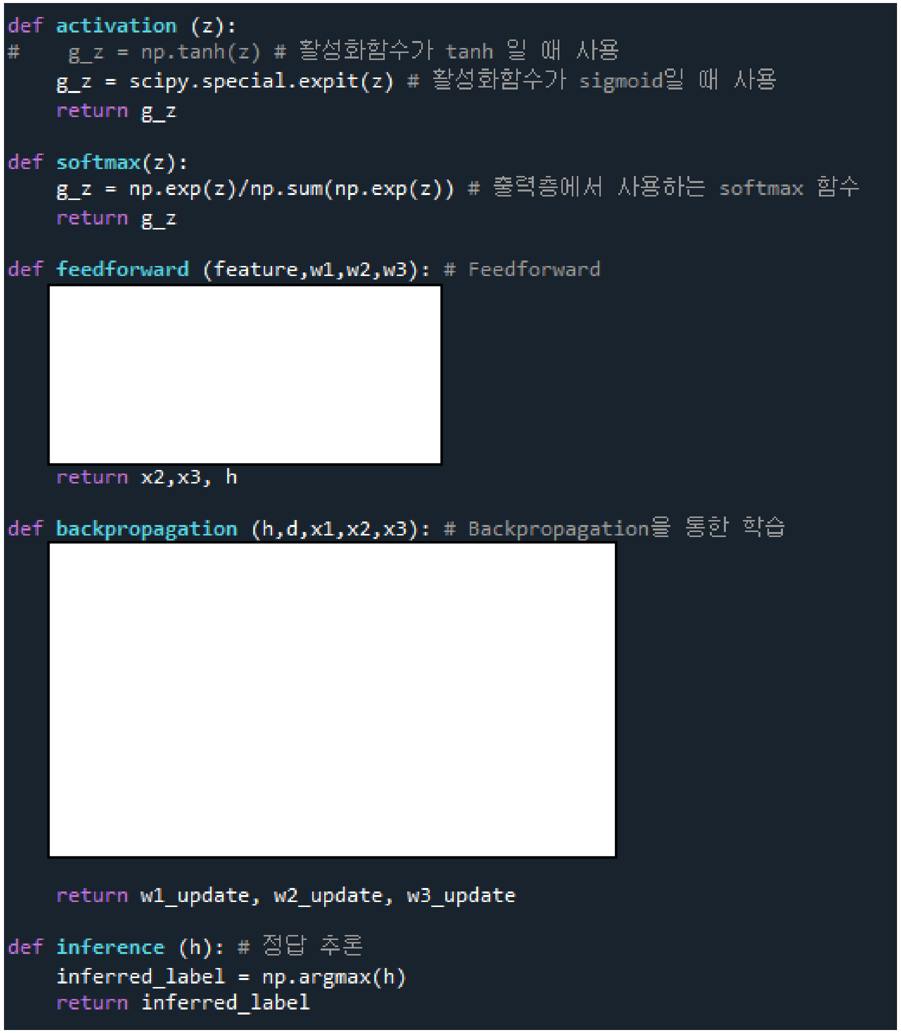
Library import, declaring the variables, weight initialization, empty arrays



```
import numpy as np
import scipy.special
import pandas as pd
import matplotlib.pyplot as plt

training_data_file = open('C:/Users/jsb07/.spyder-py3/mnist_train.csv', 'r')
training_data_list = training_data_file.readlines()
training_data_file.close()

test_data_file = open('C:/Users/jsb07/.spyder-py3/mnist_test.csv', 'r')
test_data_list = test_data_file.readlines()
test_data_file.close()

test_number = 1000 #test set 중에 1000개만 사용할거라 개수 비교할 test_number 선언
training_number = 10000 #training set 중에 5000개만 사용할거라 개수 비교할 training_number 선언

epoch = 10 # epoch
learning_rate = 0.01 # learning rate
input_layer =      # 입력층 뉴런 수
hidden_layer1 =      # 은닉층 1 뉴런 수
hidden_layer2 =      # 은닉층 2 뉴런 수
output_layer =      # 출력층 뉴런 개수

w1 =                                        Xavier
w2 =
w3 =                                                          #w1,w2,w3 Xavier 초기화

classification_accuracy_per_epoch = np.array([]) #epoch별 분류정확도 저장할 깡통 array
total_cost_per_epoch = np.array([]) # epoch별 cost 저장할 깡통 array
n_epoch = np.array([]) #epoch번호 저장할 깡통 array
```

Project 2 – MNIST classification

Function define (activation, softmax, feedforward, backpropagation, interence)



```
def activation (z):
#    g_z = np.tanh(z) # 활성화함수가 tanh 일 때 사용
    g_z = scipy.special.expit(z) # 활성화함수가 sigmoid일 때 사용
    return g_z

def softmax(z):
    g_z = np.exp(z)/np.sum(np.exp(z)) # 출력층에서 사용하는 softmax 함수
    return g_z

def feedforward (feature,w1,w2,w3): # Feedforward



    return x2,x3, h

def backpropagation (h,d,x1,x2,x3): # Backpropagation을 통한 학습



    return w1_update, w2_update, w3_update

def inference (h): # 정답 추론
    inferred_label = np.argmax(h)
    return inferred_label
```

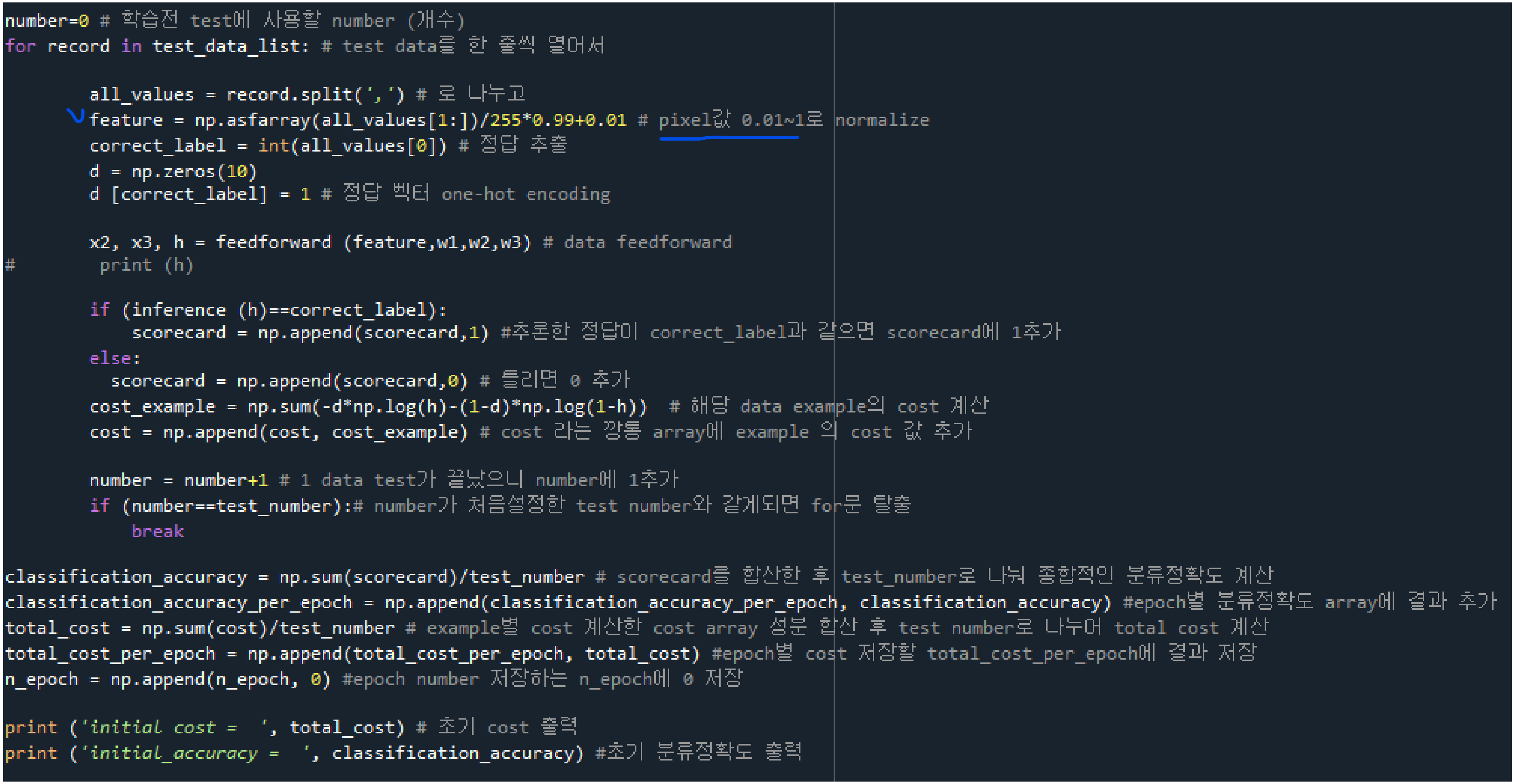
In fact, all of the functions were defined in Week 11. Those can be used in the same form as the XNOR neural network with two hidden layers. (except the softmax at output layer)

→ If you made it well in the 11th week, you can use it as it is.

→ However, if the activation function is changed to tanh instead of sigmoid, the backpropagation needs to be modified a little, right?

Project 2 – MNIST classification

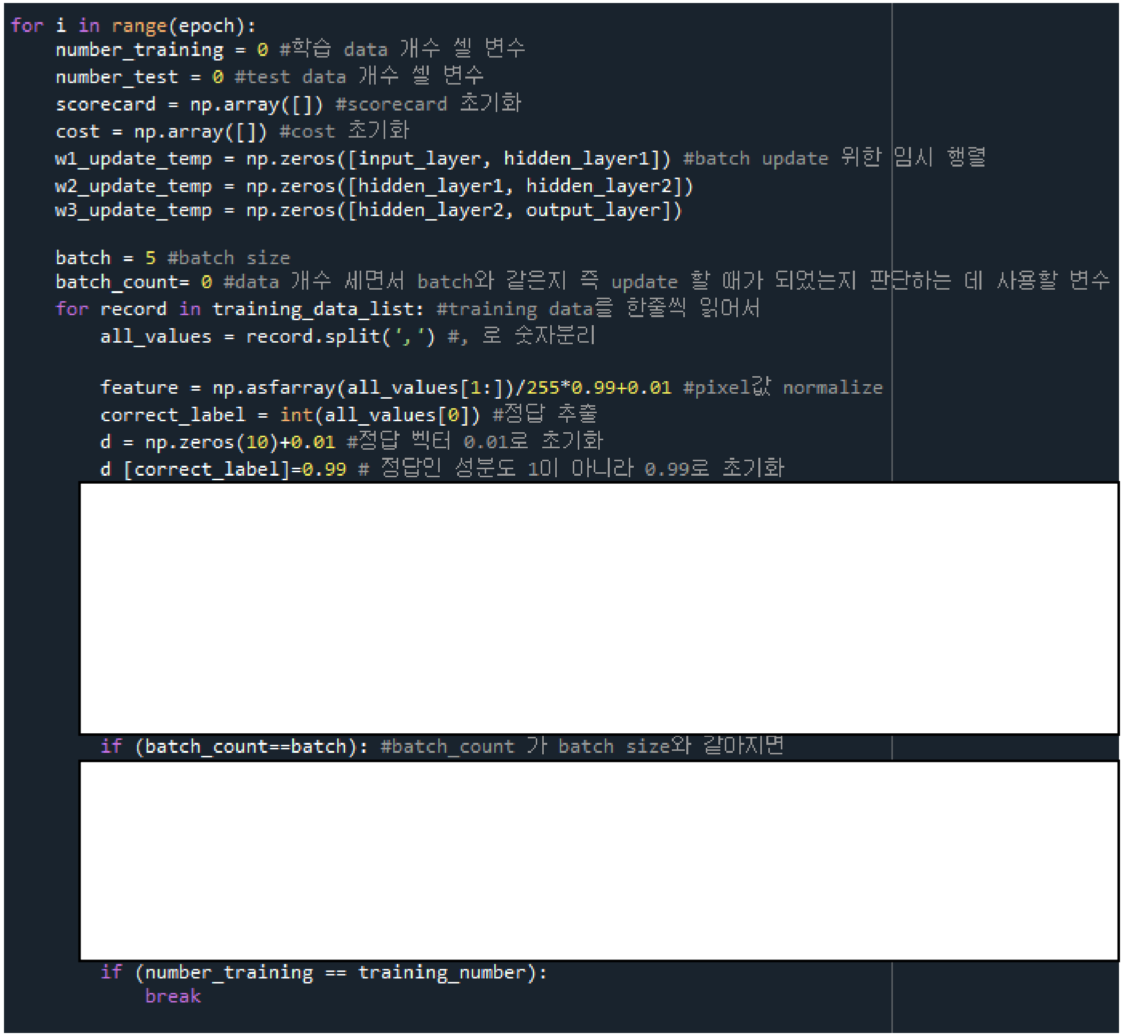
Evaluation before the training



```
number=0 # 학습전 test에 사용할 number (개수)
for record in test_data_list: # test data를 한 줄씩 열어서

        all_values = record.split(',') # 로 나누고
        feature = np.asfarray(all_values[1:])/255*0.99+0.01 # pixel값 0.01~1로 normalize
        correct_label = int(all_values[0]) # 정답 추출
        d = np.zeros(10)
        d [correct_label] = 1 # 정답 벡터 one-hot encoding

        x2, x3, h = feedforward (feature,w1,w2,w3) # data feedforward
#        print (h)

        if (inference (h)==correct_label):
            scorecard = np.append(scorecard,1) #추론한 정답이 correct_label과 같으면 scorecard에 1추가
        else:
          scorecard = np.append(scorecard,0) # 틀리면 0 추가
        cost_example = np.sum(-d*np.log(h)-(1-d)*np.log(1-h))  # 해당 data example의 cost 계산
        cost = np.append(cost, cost_example) # cost 라는 깡통 array에 example 의 cost 값 추가

        number = number+1 # 1 data test가 끝났으니 number에 1추가
        if (number==test_number):# number가 처음설정한 test number와 같게되면 for문 탈출
            break

classification_accuracy = np.sum(scorecard)/test_number # scorecard를 합산한 후 test_number로 나눠 종합적인 분류정확도 계산
classification_accuracy_per_epoch = np.append(classification_accuracy_per_epoch, classification_accuracy) #epoch별 분류정확도 array에 결과 추가
total_cost = np.sum(cost)/test_number # example별 cost 계산한 cost array 성분 합산 후 test number로 나누어 total cost 계산
total_cost_per_epoch = np.append(total_cost_per_epoch, total_cost) #epoch별 cost 저장할 total_cost_per_epoch에 결과 저장
n_epoch = np.append(n_epoch, 0) #epoch number 저장하는 n_epoch에 0 저장

print ('initial cost =  ', total_cost) # 초기 cost 출력
print ('initial_accuracy =  ', classification_accuracy) #초기 분류정확도 출력
```

Project 2 – MNIST classification

Weight update



```
for i in range(epoch):
    number_training = 0 #학습 data 개수 셀 변수
    number_test = 0 #test data 개수 셀 변수
    scorecard = np.array([]) #scorecard 초기화
    cost = np.array([]) #cost 초기화
    w1_update_temp = np.zeros([input_layer, hidden_layer1]) #batch update 위한 임시 행렬
    w2_update_temp = np.zeros([hidden_layer1, hidden_layer2])
    w3_update_temp = np.zeros([hidden_layer2, output_layer])

    batch = 5 #batch size
    batch_count= 0 #data 개수 세면서 batch와 같은지 즉 update 할 때가 되었는지 판단하는 데 사용할 변수
    for record in training_data_list: #training data를 한줄씩 읽어서
        all_values = record.split(',') #, 로 숫자분리

        feature = np.asfarray(all_values[1:])/255*0.99+0.01 #pixel값 normalize
        correct_label = int(all_values[0]) #정답 추출
        d = np.zeros(10)+0.01 #정답 벡터 0.01로 초기화
        d [correct_label]=0.99 # 정답인 성분도 1이 아니라 0.99로 초기화



        if (batch_count==batch): #batch_count 가 batch size와 같아지면



        if (number_training == training_number):
            break
```

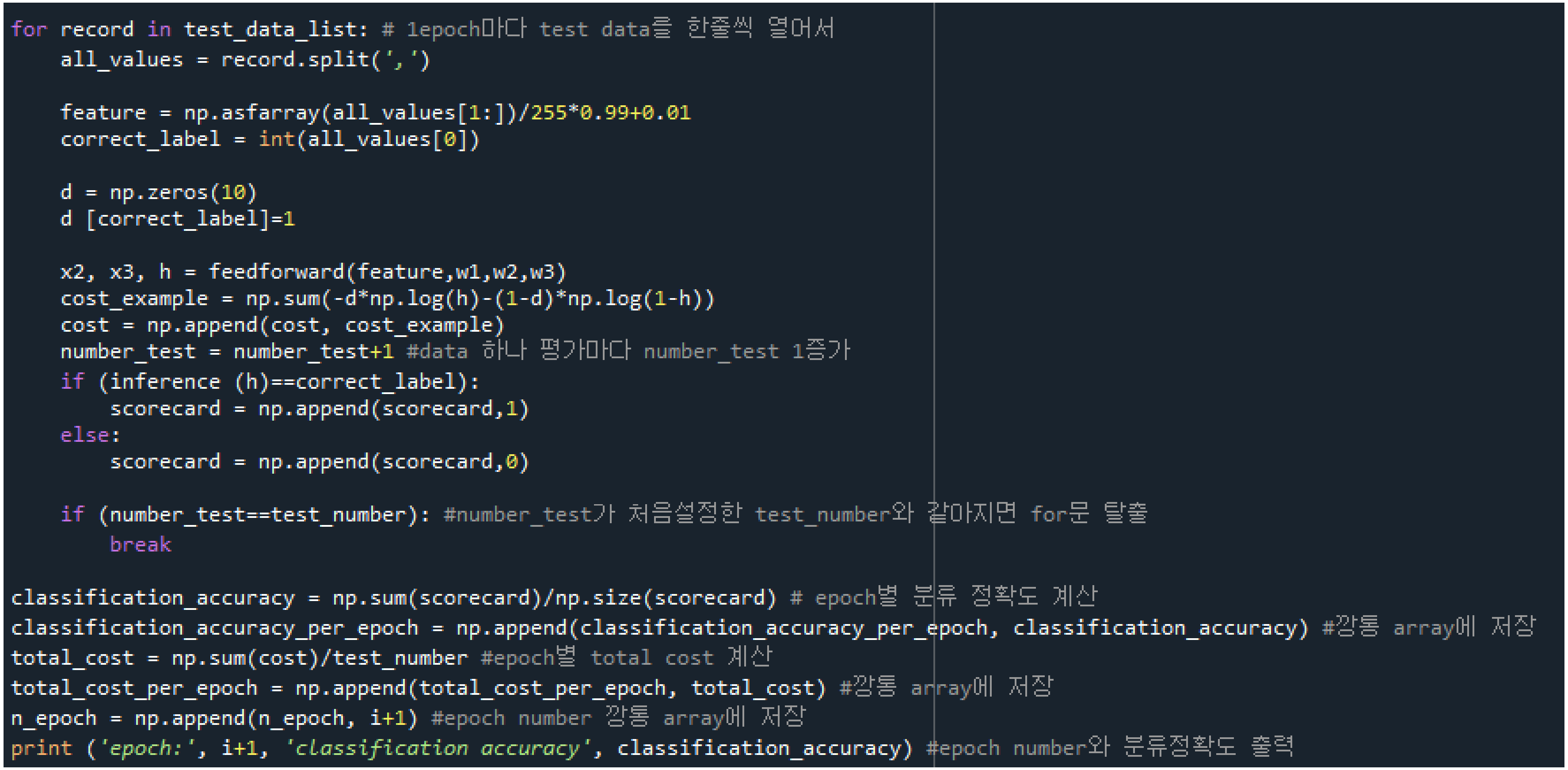
Please fill in the blanks.

Flow is the same as Week 11.

However, we would like to use only the first 10,000 out of 60,000 training sets. Use number_training well for this purpose.

Project 2 – MNIST classification

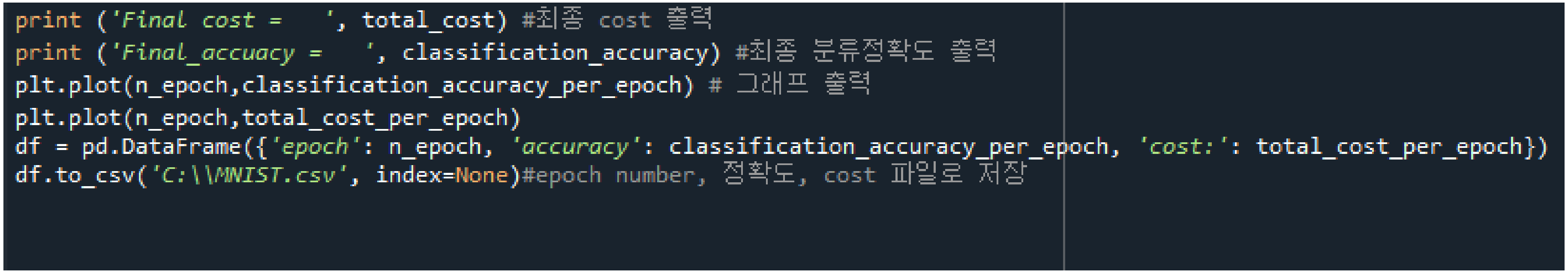
Evaluation per each epoch and save



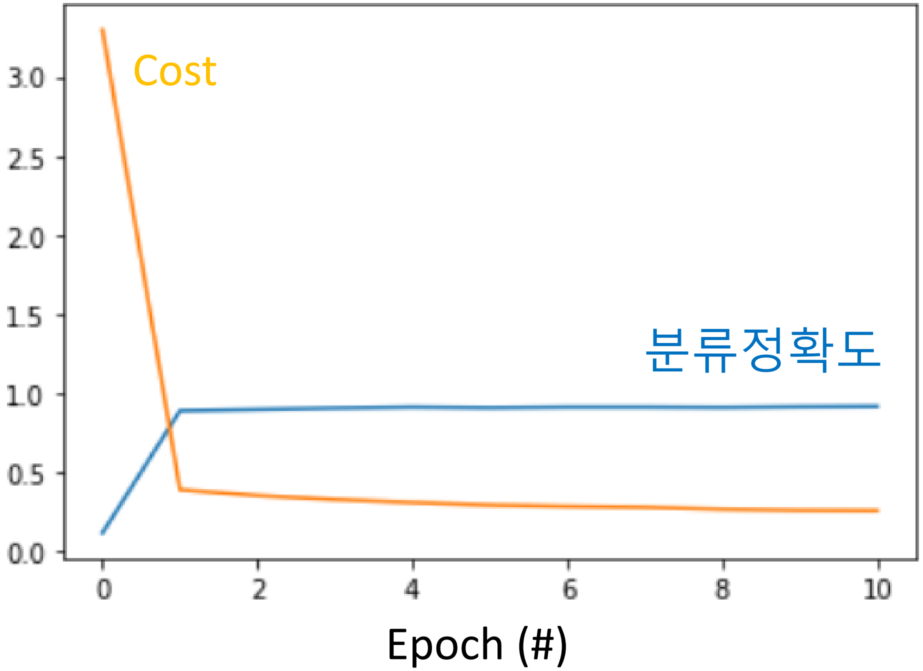
```
for record in test_data_list: # 1epoch마다 test data를 한줄씩 열어서
    all_values = record.split(',')

    feature = np.asfarray(all_values[1:])/255*0.99+0.01
    correct_label = int(all_values[0])

    d = np.zeros(10)
    d [correct_label]=1

    x2, x3, h = feedforward(feature,w1,w2,w3)
    cost_example = np.sum(-d*np.log(h)-(1-d)*np.log(1-h))
    cost = np.append(cost, cost_example)
    number_test = number_test+1 #data 하나 평가마다 number_test 1증가
    if (inference (h)==correct_label):
        scorecard = np.append(scorecard,1)
    else:
        scorecard = np.append(scorecard,0)

    if (number_test==test_number): #number_test가 처음설정한 test_number와 같아지면 for문 탈출
        break

classification_accuracy = np.sum(scorecard)/np.size(scorecard) # epoch별 분류 정확도 계산
classification_accuracy_per_epoch = np.append(classification_accuracy_per_epoch, classification_accuracy) #깡통 array에 저장
total_cost = np.sum(cost)/test_number #epoch별 total cost 계산
total_cost_per_epoch = np.append(total_cost_per_epoch, total_cost) #깡통 array에 저장
n_epoch = np.append(n_epoch, i+1) #epoch number 깡통 array에 저장
print ('epoch:', i+1, 'classification accuracy', classification_accuracy) #epoch number와 분류정확도 출력
```

Project 2 – MNIST classification

Print, plot and save



```
print ('Final cost =    ', total_cost) #최종 cost 출력
print ('Final_accuacy =    ', classification_accuracy) #최종 분류정확도 출력
plt.plot(n_epoch,classification_accuracy_per_epoch) # 그래프 출력
plt.plot(n_epoch,total_cost_per_epoch)
df = pd.DataFrame({'epoch': n_epoch, 'accuracy': classification_accuracy_per_epoch, 'cost:': total_cost_per_epoch})
df.to_csv('C:\\MNIST.csv', index=None)#epoch number, 정확도, cost 파일로 저장
```


Cost
분류정확도
Epoch (#)