11주. Multi layer Neural network			
학번	3 2 1 7 0 7 5 8	이름	김산

(8장 후반부)

Q1 (3점) 강의 slide 43~45 에 있는 코드를 완성하여 실행 결과를 보이시오
(실행 결과가 길므로 처음 10개와 끝 10 개 정도를 보인다)

Source code :

```
from sklearn import datasets
import random
import numpy as np

## prepare dataset ######################################
iris = datasets.load_iris()
X = iris.data
target = iris.target

# one hot encoding
num = np.unique(target, axis=0)
num = num.shape[0]
y = np.eye(num)[target]

def SIGMOID(x):
    return 1/(1 + np.exp(-x))

def SLP_SGD(tr_X, tr_y, alpha, rep):
    # initialize w
    n = tr_X.shape[1] * tr_y.shape[1]
    random.seed = 123
    w = random.sample(range(1, 100), n)
    w = (np.array(w) - 50) / 100
    w = w.reshape(tr_X.shape[1], -1)

    for i in range(rep):
        for k in range(tr_X.shape[0]):
            x = tr_X[k, :]
            v = np.matmul(x, w)
            y = SIGMOID(v)
            e = tr_y[k, :] - y
            w = w + alpha * \
                np.matmul(tr_X[k, :].reshape(-1, 1),
```

```
                    (y*(1-y)*e).reshape(1, -1))

        print("error", i, np.mean(e))
    return w


## Training (get W) ####################################
W = SLP_SGD(X, y, alpha=0.01, rep=1000)

## Test ################################################
def test(tr_X, tr_y, W) :
    pred = np.zeros(X.shape[0])
    for i in range(X.shape[0]):
        v = np.matmul(X[i,:],W)
        y = SIGMOID(v)

        pred[i] = np.argmax(y)
        print("target, predict", target[i], pred[i])
    print("accuracy :", np.mean(pred==target)) ]

test(X, y, W)
```

실행화면 캡쳐:

```
error 0 -0.27851013798512353
error 1 -0.03648509448286317
error 2 -0.02962064357207574
error 3 -0.028015536059935824
error 4 -0.02670343146390765
error 5 -0.025450455644563313
error 6 -0.02425871707412815
error 7 -0.023134781992855236
error 8 -0.022080653856199534
error 9 -0.021095421113117335
error 10 -0.02017649266753076
```

```
error 989 5.1281210131583675e-05
error 990 4.2836284810522186e-05
error 991 3.4394388336939086e-05
error 992 2.5955526822848036e-05
error 993 1.751970633483609e-05
error 994 9.086932888762353e-06
error 995 6.572124544440561e-07
error 996 -7.76944904743356e-06
error 997 -1.619304574317082e-05
error 998 -2.4613571804494676e-05
error 999 -3.303102145181536e-05
```

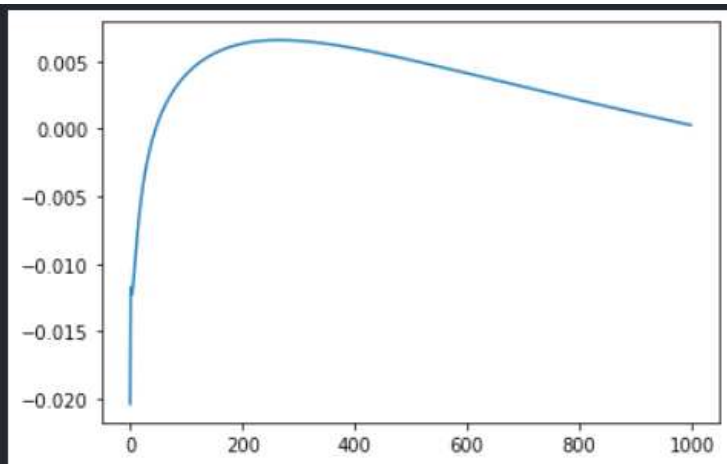
```
target, predict 2 2.0
target, predict 2 2.0
target, predict 2 2.0
target, predict 2 2.0
accuracy : 0.9133333333333333
```

Q2 (2점) (slide 47) Practice 1 에서 α 값을 0.05, 0.1, 0.5 로 하여 테스트 하여 보시오
- 에러가 줄어드는 추세를 비교하여 보시오
- 최종 예측 accuracy 가 어떻게 되는지 비교하여 보시오

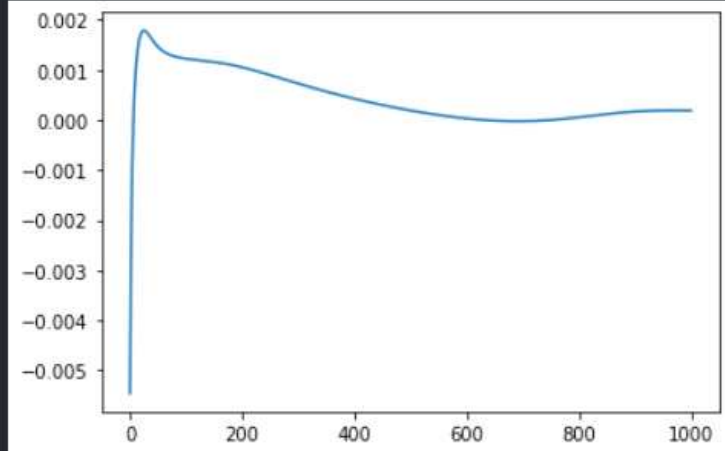
Source code :

```
import matplotlib.pyplot as plt

def SLP_SGD_PLT(tr_X, tr_y, alpha, rep):
    errors = []
    # initialize w
    n = tr_X.shape[1] * tr_y.shape[1]
    random.seed = 123
    w = random.sample(range(1, 100), n)
    w = (np.array(w) - 50) / 100
    w = w.reshape(tr_X.shape[1], -1)

    for i in range(rep):
        for k in range(tr_X.shape[0]):
            x = tr_X[k, :]
            v = np.matmul(x, w)
            y = SIGMOID(v)
            e = tr_y[k, :] - y
            w = w + alpha * \
                np.matmul(tr_X[k, :].reshape(-1, 1),
                          (y*(1-y)*e).reshape(1, -1))

        errors.append(np.mean(e))

    plt.plot(errors)
    plt.show()
    return w

alphas = [0.01, 0.1, 0.5]
for i in alphas:
    W = SLP_SGD_PLT(X, y, alpha=i, rep=1000)
    test(X, y, W)
```

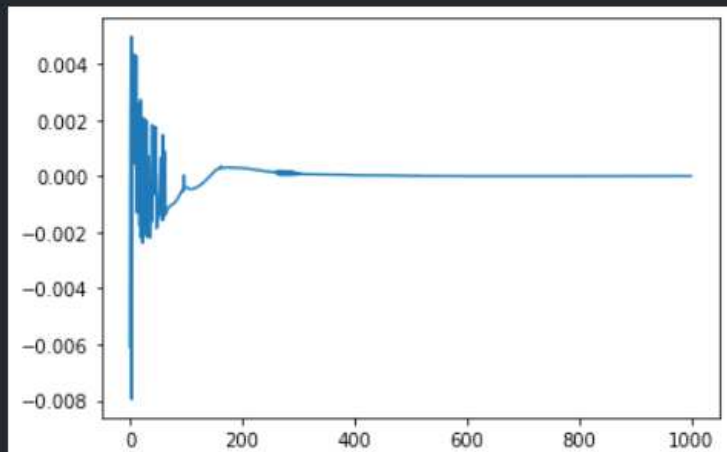
실행화면 캡쳐:




```
accuracy : 0.9133333333333333
```




```
accuracy : 0.8733333333333333
```




```
accuracy : 0.7533333333333333
```

Q3 (5점) Practice 1을 수정하되 학습률 α=0.01, epoch= 50, batch size=10 으로 하고 dataset을 train/test 로 나누되 test의 비율은 30%로 하시오.
- training accuracy 와 test accuracy를 보이시오

Source code :

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
import random
import numpy as np

iris = datasets.load_iris()
X = iris.data
y = iris.target

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=123)

def one_hot_encoding(target) :
    num = np.unique(target, axis=0)
    num = num.shape[0]
    y = np.eye(num)[target]
    return y

target_train = one_hot_encoding(y_train)
target_test = one_hot_encoding(y_test)

def SIGMOID(x):
    return 1/(1 + np.exp(-x))

def SLP_BATCH(tr_X, tr_y, alpha, epoch, batch_size):
    # initialize w
    n = tr_X.shape[1] * tr_y.shape[1]
    random.seed = 123
    w = random.sample(range(1, 100), n)
    w = (np.array(w)-50)/100
    w = w.reshape(tr_X.shape[1], -1)

    for i in range(epoch):
        batch_mask = np.random.choice(tr_X.shape[0], batch_size)
```

```
        delta = np.full((4, 3), 0)
        for k in batch_mask:
            x = tr_X[k, :]
            v = np.matmul(x, w)
            y = SIGMOID(v)
            e = tr_y[k, :] - y
            delta = delta + alpha * \
                np.matmul(tr_X[k, :].reshape(-1, 1),
                          (y*(1-y)*e).reshape(1, -1))

        w += delta / batch_size

    return w

def test(X, target, w):
    pred = np.zeros(X.shape[0])
    for i in range(X.shape[0]):
        v = np.matmul(X[i,:],w)
        y = SIGMOID(v)
        pred[i] = np.argmax(y)
        # print("target, predict", target[i], pred[i])
    print("accuracy :", np.mean(pred==target))

W = SLP_BATCH(X_train, target_train, alpha=0.01, epoch=50,
batch_size=10)


test(X_train, y_train, W)
test(X_test, y_test, W)
```

실행화면 캡쳐:

```
accuracy : 0.8761904761904762
accuracy : 0.8444444444444444
```

(9장)

Q1 (2점) Neural Network에서 과적합을 방지하는 두가지 기법에 대해 설명하시오

- Drop out : 매학습마다 랜덤하게 노드간 연결을 끊어 학습을 진행한다. 학습속도가 느려지는 대신 과적합을 방지할 수 있다.
- Weight decay : w갱신할때마다 0과 1사이의 값을 가지는 가중치를 곱함으로써 학습을 방해하는 대신 과적합을 방지할 수 있다.

Q2 (2점) Deep neural network 에서 back propagation 시 발생하는 문제와 이를 해결하기 위한 방법을 제시하시오

- **기울기 소실 문제** : 레이어가 많은 신경망일 수록 깊은 레이어 까지 에러 전파가 잘 이루어지지 않는다. 이를 해결하기 위해서는 적절한 활성화 함수를 사용하면 된다.

Q3 (1점) Neural network에서 momentum 의 역할에 대해 설명하시오

- 가중치 값의 변동 (oscillating) 을 줄이기 위해 사용하며, 학습속도가 빨라지는 효과가 있다.