Assignment9

May 30, 2019

1 Information

Writer : *Junhyuck Woo*
Std.ID : 20145337
Project : *Build a binary classifier to classify digit* 0 *against all the other digits at MNIST dataset.*

2 import library

```
In [1]: import numpy as np
        import pandas as pd
```

3 Load files

Training data, Testing data

```
In [2]: file_data   = "mnist_train.csv"
        handle_file = open(file_data, "r")
        train_data  = handle_file.readlines()
        handle_file.close()

        file_data   = "mnist_test.csv"
        handle_file = open(file_data, "r")
        test_data   = handle_file.readlines()
        handle_file.close()
```

4 Global Parameter

```
In [3]: count = 0
        size_row = 28
        size_col = 28
        num_true = 0
        num_true_t = 0
        num_train = len(train_data)
        num_test = len(test_data)
        train_image  = np.empty((size_row * size_col + 1, num_train), dtype=float)
        train_label  = np.empty(num_train, dtype=int)
        test_image  = np.empty((size_row * size_col + 1, num_test), dtype=float)
        test_label = np.empty(num_test, dtype=int)
```

5 Reshape data & Set label

```
In [4]: count = 0
        num_true = 0;
        for line in train_data:
            line_data= line.split(',')
            label = line_data[0]
            im_vector = np.asfarray(line_data[1:])
            im_vector = np.insert(im_vector, 0, 1)

            if (int(label)==0):
                train_label[count]      = 1
                num_true += 1
            else:
                train_label[count]      = -1

            train_image[:, count] = im_vector
            count += 1
```

```
In [5]: count = 0
        num_true_t = 0
        for line in test_data:
            line_data= line.split(',')
            label = line_data[0]
            im_vector = np.asfarray(line_data[1:])
            im_vector = np.insert(im_vector, 0, 1)

            if (int(label)==0):
                test_label[count]      = 1
                num_true_t += 1
            else:
                test_label[count]      = -1

            test_image[:, count] = im_vector
            count += 1
```

```
In [6]: A = np.asmatrix(train_image.transpose())
        y = np.asmatrix(train_label)
        pinv_A = np.linalg.pinv(A)
```

```
In [7]: A_t = np.asmatrix(test_image.transpose())
        y_t = np.asmatrix(test_label)
```

```
In [8]: x = pinv_A * y.transpose()
In [9]: y_ = A * x
        y_t_ = A_t * x
```

```
In [10]: tp=0; fp=0; tn=0; fn=0
         for i in range(num_train):
             if(y_[i] < 0):
                 y_[i] = -1
                 if(y.T[i] < 0):
                     tn += 1
                 else:
                     fn += 1
             else:
                 y_[i] = 1
                 if(y.T[i] < 0):
                     fp += 1
                 else:
                     tp += 1
```

```
In [11]: tp_t=0; fp_t=0; tn_t=0; fn_t=0
         for i in range(len(y_t_)):
             if(y_t_[i] < 0):
                 y_t_[i] = -1
                 if(y_t.T[i] < 0):
                     tn_t += 1
                 else:
                     fn_t += 1
             else:
                 y_t_[i] = 1
                 if(y_t.T[i] < 0):
                     fp_t += 1
                 else:
                     tp_t += 1
```

```
In [14]: acc_tp = tp / num_true
         acc_fp = fp / (num_train-num_true)
         acc_fn = fn / num_true
         acc_tn = tn / (num_train-num_true)
         print("Train data set")
         train_result = np.array([[acc_tp, acc_fp],[acc_tn, acc_fn]])
         chart = pd.DataFrame(train_result, index=['Possitive', 'Negative'], columns=['True',
         chart
```

```
Train data set
```

```
Out[14]:                 True     False
         Possitive  0.872531  0.003310
         Negative   0.996690  0.127469
```

```
In [15]: acc_tp_t = tp_t / num_true_t
         acc_fp_t = fp_t / (num_test-num_true_t)
         acc_fn_t = fn_t / num_true_t
         acc_tn_t = tn_t / (num_test-num_true_t)
         print("Test data set")
         test_result = np.array([[acc_tp_t, acc_fp_t],[acc_tn_t, acc_fn_t]])
         chart = pd.DataFrame(test_result, index=['Possitive', 'Negative'], columns=['True', '
         chart
```

```
Test data set
```

```
Out[15]:                True     False
         Possitive  0.883673  0.004767
         Negative   0.995233  0.116327
```

```
In [ ]:
```