Assignment10

June 4, 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()


        num_train = len(train_data)
        num_test = len(test_data)
        train_image  = np.zeros((28 * 28, num_train), dtype=float)
        train_label = np.zeros(num_train, dtype=int)
        test_image  = np.zeros((28 * 28, num_test), dtype=float)
        test_label = np.zeros(num_test, dtype=int)

        count = 0
        for line in train_data:
            line_data = line.split(',')
```

```
            label = line_data[0]
            train_label[count] = label
            im_vector = np.asfarray(line_data[1:])
            train_image[:,count] = im_vector
            count += 1
        count = 0
        for line in test_data:
            line_data = line.split(',')
            label = line_data[0]
            test_label[count] = label
            im_vector = np.asfarray(line_data[1:])
            test_image[:,count] = im_vector
            count += 1
```

In [3]:
```
class classifier:
    def reshape_data(self, data):
        num = max(data.shape)
        reshape_data = np.zeros((28 * 28 + 1, num), dtype=float)
        for i in range(num):
            reshape_data[:,i] = np.insert(data[:,i], 0, 1)
        return(reshape_data)

    def reshape_label(self, label, select):
        num = len(label)
        reshape_label = np.zeros(num, dtype=int)
        for i in range(num):
            if(int(label[i])==select):
                reshape_label[i] = 1
            else:
                reshape_label[i] = -1
        return(reshape_label)


    def train(self, train_data, label, num_digit=10):
        count = 0
        num_digit
        self.x = np.zeros((num_digit,785), dtype=float)
        train_image = self.reshape_data(train_data)
        A = np.asmatrix(train_image.transpose())
        pinv_A = np.linalg.pinv(A)

        for i in range(num_digit):
            train_label = self.reshape_label(label, i)
            y = np.asmatrix(train_label)
            buf = np.array(pinv_A * y.transpose())
            self.x[i,:]= buf.T
        self.x = np.asmatrix(self.x)
        return(self.x)
```

```
        def predict(self, test_data):
            test_image = self.reshape_data(test_data)
            A = np.asmatrix(test_image.transpose())
            y = A * self.x.T
            label = []
            for i in range(max(test_data.shape)):
                label.append(np.argmax(y[i,:]))
            label = np.array(label)
            return(label)

        def evaulation(self, prediction, label):
            tp = 0
            error = 0
            result =np.zeros((11,11), dtype=int)
            for i in range(len(prediction)):
                result[prediction[i]][label[i]] +=1
                if(prediction[i] == label[i]):
                    tp += 1
                else:
                    error += 1

            for i in range(10):
                result[10][i] = sum(result.T[:][i])
                result[i][10] = sum(result[:][i])
            result[10][10] = sum(result[:,10])

            # Plot
            print("True Possitive: ", tp/result[10][10])
            print("Error Rate: ", error/result[10][10])
            chart = pd.DataFrame(result.T)
            return(chart)
```

```
In [4]: binary_classifier = classifier()
        binary_classifier.train(train_image, train_label)
        y = binary_classifier.predict(train_image)
        binary_classifier.evaulation(y, train_label)
```

```
True Possitive:  0.8577333333333333
Error Rate:  0.14226666666666668
```

Out[4]:

	0	1	2	3	4	5	6	7	8	9	10
0	5682	7	18	14	24	43	64	4	61	6	5923
1	2	6548	40	15	19	31	14	12	55	6	6742
2	99	264	4792	149	108	11	234	91	192	18	5958
3	42	167	176	5158	32	125	56	115	135	125	6131
4	10	99	42	6	5212	50	39	23	59	302	5842

5	164	95	28	432	105	3991	192	36	235	143	5421
6	108	74	61	1	70	90	5476	0	35	3	5918
7	55	189	37	47	170	9	2	5426	10	320	6265
8	75	493	63	226	105	221	56	20	4412	180	5851
9	68	60	20	117	371	12	4	492	38	4767	5949
10	6305	7996	5277	6165	6216	4583	6137	6219	5232	5870	60000

In [5]:
```
y = binary_classifier.predict(test_image)
binary_classifier.evaulation(y, test_label)
```

```
True Possitive:  0.8603
Error Rate:  0.1397
```

Out[5]:

	0	1	2	3	4	5	6	7	8	9	10
0	944	0	1	2	2	7	14	2	7	1	980
1	0	1107	2	2	3	1	5	1	14	0	1135
2	18	54	813	26	15	0	42	22	37	5	1032
3	4	17	23	880	5	17	9	21	22	12	1010
4	0	22	6	1	881	5	10	2	11	44	982
5	23	18	3	72	24	659	23	14	39	17	892
6	18	10	9	0	22	17	875	0	7	0	958
7	5	40	16	6	26	0	1	884	0	50	1028
8	14	46	11	30	27	40	15	12	759	20	974
9	15	11	2	17	80	1	1	77	4	801	1009
10	1041	1325	886	1036	1085	747	995	1035	900	950	10000