Assignment03

March 25, 2019

20164245 Hong Jin

1 Load MNIST traning dataset.

```
In [1]: import matplotlib.pyplot as plt
        import numpy as np

        file_data                = "mnist_train.csv"
        handle_file        = open(file_data, "r")
        data                        = handle_file.readlines()
        handle_file.close()

        size_row        = 28    # height of the image
        size_col          = 28    # width of the image

        num_image        = len(data)
        count                  = 0     # count for the number of images
```

2 Nomarlization

```
In [2]: def normalize(data):

            data_normalized = (data - min(data)) / (max(data) - min(data))

            return(data_normalized)
```

3 Compute distance based on L2-norm

```
In [3]: def distance_n2(x):

            d = x ** 2

            return(d)
```

4 Compute distance based on L1-norm

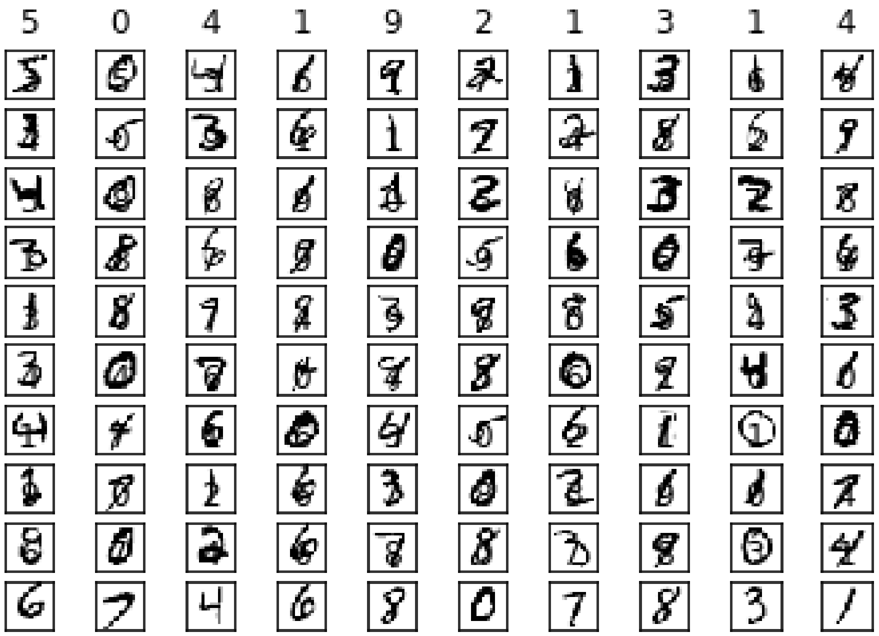
```
In [4]: def distance_n1(x):

            d = np.abs(x)

            return(d)
```

5 Make label, image array

```
In [5]: list_image  = np.empty((size_row * size_col, num_image), dtype=float)
        list_label  = np.empty(num_image, dtype=int)

        for line in data:

            line_data   = line.split(',')
            label       = line_data[0]
            im_vector   = np.asfarray(line_data[1:])
            im_vector   = normalize(im_vector)

            list_label[count]       = label
            list_image[:, count]    = im_vector

            count += 1
```

6 Draw 100 images with label

```
In [6]: f1 = plt.figure(1)

        for i in range(100):

            label       = list_label[i]
            im_vector   = list_image[:, i]
            im_matrix   = im_vector.reshape((size_row, size_col))

            plt.subplot(10, 10, i+1)
            plt.title(label)
            plt.imshow(im_matrix, cmap='Greys', interpolation='None')

            frame   = plt.gca()
            frame.axes.get_xaxis().set_visible(False)
            frame.axes.get_yaxis().set_visible(False)

        plt.show()
```



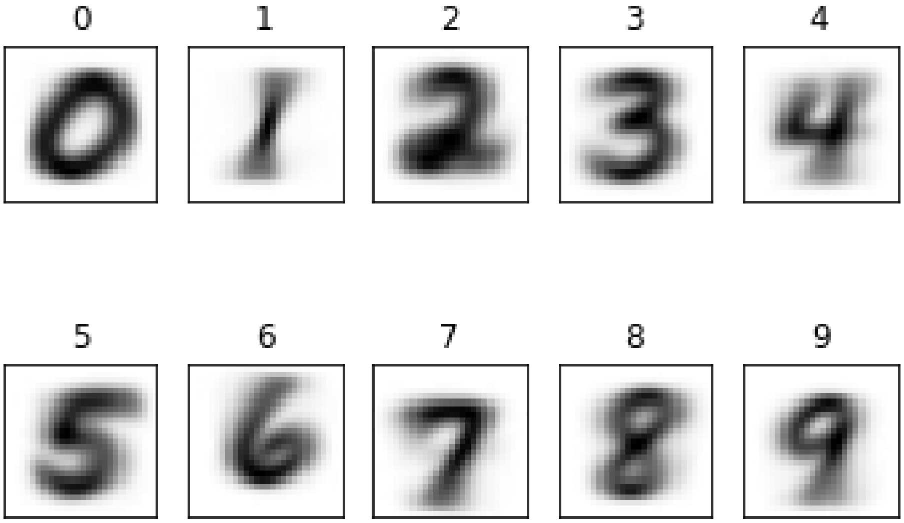
7 Compute sum of the distance based on L2-norm

```
In [7]: f2 = plt.figure(2)

        im_average2  = np.zeros((size_row * size_col, 10), dtype=float)

        for i in range(num_image):
            im_average2[:,list_label[i]] += distance_n2(list_image[:,i])
```

```
<Figure size 432x288 with 0 Axes>
```

8 Visualize the average images (L2-norm)

```
In [8]: for i in range(10) :
            im_average2[:,i] = np.sqrt(im_average2[:,i])
            im_L2matrix = im_average2[:,i].reshape((size_row,size_col))

            plt.subplot(2,5,i+1)
            plt.title(i)
            plt.imshow(im_L2matrix, cmap='Greys', interpolation = 'None')

            frame = plt.gca()
            frame.axes.get_xaxis().set_visible(False)
            frame.axes.get_yaxis().set_visible(False)

        plt.show()
```



9 Compute the sum of distance based on L1-norm

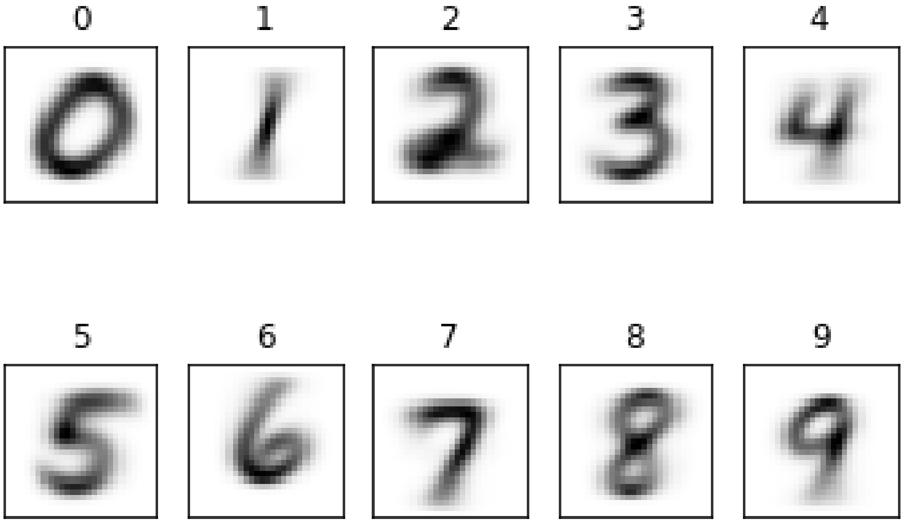
```
In [9]: f3 = plt.figure(3)

        im_average  = np.zeros((size_row * size_col, 10), dtype=float)

        for i in range(num_image):
            im_average[:,list_label[i]] += distance_n1(list_image[:,i])
```

```
<Figure size 432x288 with 0 Axes>
```

10 Visualize the average images (L1-norm)

```
In [10]: for i in range(10) :
             im_L1matrix = im_average[:,i].reshape((size_row,size_col))

             plt.subplot(2,5,i+1)
             plt.title(i)
             plt.imshow(im_L1matrix, cmap='Greys', interpolation = 'None')

             frame = plt.gca()
             frame.axes.get_xaxis().set_visible(False)
             frame.axes.get_yaxis().set_visible(False)
```

```
plt.show()
```


0
1
2
3
4
5
6
7
8
9