Assignment04

April 4, 2019

20163228

```
In [1]: import matplotlib.pyplot as plt
        import numpy as np
        import random
        from collections import Counter
```

```
In [2]: # Normalize the values of the input data to be [0, 1]
        def normalize(data):

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

            return(data_normalized)

        # Reshape vector to matrix (k, 28*28) -> (k, 28, 28)
        # And plot the image
        def reshape_plot_img(k, vector) :
            size = 28
            matrix = np.empty((k, size, size), dtype=float)

            for i in range(k) :
                matrix[i, :, :] = vector[i, :].reshape(size, size)

            f = plt.figure(figsize=(20,2))

            for i in range(k):
                plt.subplot(1, k, i+1)
                plt.imshow(matrix[i, : , :], cmap='Greys', interpolation='None')

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

            plt.show()
```

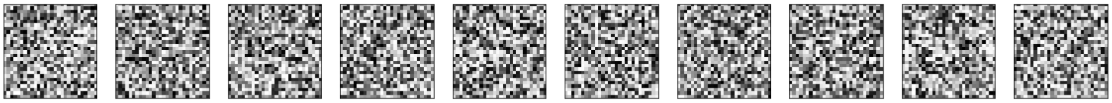
```
In [3]: # Get Majority
        def majority(label_list) :
            counter = Counter(label_list)
            maximum = counter.most_common(1)
            return maximum[0][1]

        # Compute Accuracy
        def accuracy(label_list, k_list) :
            k = len(k_list)
            accuracy = 0.0


            for i in range (k):
                length = len(k_list[i])
                if (length != 0):
                    k_label = np.empty(length, dtype=float)
                    for j in range (length):
                        k_label[j] = label_list[k_list[i][j]]
                    count = majority(k_label)
                    accuracy += count

            return accuracy
```

```
In [4]: #
        # Read Train File
        #
        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


        #
        # Make a vector which represent images
        # and save label in another vector
        #

        # image vector for all images (60000, 28*28)
        train_img   = np.empty((num_image, size_row * size_col), dtype=float)
        train_label = np.empty(num_image, dtype=int)    # label for each image

        for line in data:

            line_data          = line.split(',')    # len(line_data) = 784
            label              = line_data[0]
```

```
    train_label[count] = label

    im_vector = np.asfarray(line_data[1:])
    im_vector = normalize(im_vector)
    train_img[count, :] = im_vector

    count += 1

#
# Read Test File
#
test_file   = "mnist_test.csv"
handle_file = open(test_file, "r")
test_data   = handle_file.readlines()
handle_file.close()

num_test_img = len(test_data)
count_test   = 0

# Make a vector of test data
test_img   = np.empty((num_test_img, size_row * size_col), dtype=float)
test_label = np.empty(num_test_img, dtype=int)

for line in test_data:
    line_data = line.split(',')     # len(line_data) = 784
    label     = line_data[0]
    test_label[count_test] = label

    im_vector = np.asfarray(line_data[1:])
    im_vector = normalize(im_vector)
    test_img[count_test, :] = im_vector

    count_test += 1
```

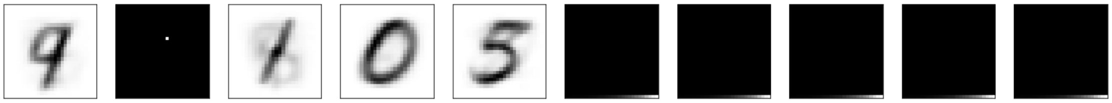
In [5]:

```
# Init Center
k_ten = 10
ten_means_center = np.empty((k_ten, size_row * size_col), dtype=float)
for i in range(k_ten) :
    for j in range (size_row * size_col) :
        ten_means_center[i][j] = random.random()

reshape_plot_img(k_ten, ten_means_center)
```



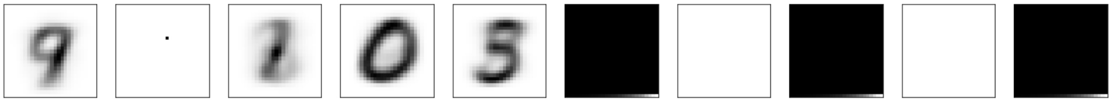
```
In [ ]: iteration = 0

        while True :  # one iteration
            # Cluster
            k_means_sum = np.empty((k_ten, size_row * size_col), dtype=float)
            k_list      = [[]for row in range(k_ten)]
            k_count     = np.empty(k_ten, dtype=int)
            energy      = 0.0

            # for every image data put an image in the appropriage group
            for i in range(num_image):
                new_img = train_img[i, :]
                min_distance = (new_img - ten_means_center[0, :]) ** 2
                min_distance = np.sum(min_distance)
                min_index    = 0

                for j in range (1, k_ten):
                    distance = (new_img - ten_means_center[j, :]) ** 2
                    distance = np.sum(distance)

                    if distance < min_distance:
                        min_distance = distance
                        min_index = j

                k_means_sum[min_index, :] += new_img
                k_list[min_index].append(i)
                k_count[min_index] += 1

            for i in range(k_ten):
                if (len(k_list[i]) == 0) :
                    k_list[i].append(random.randint(0, num_image))


                # Calculate energy
                energy += min_distance

            # Compute Accuracy
            train_accuracy = 0.0
            train_accuracy = accuracy(train_label, k_list)
            train_accuracy /= num_image

            # Print Energy and Accuracy
            print("\nIteration : %d" %(iteration))
            print ("Train Energy : %f" %(energy))
            print("Train Accuracy : %f" %(train_accuracy))


            # Update Center
            new_center = np.empty((k_ten, size_row * size_col), dtype=float)
            for i in range(k_ten):
                new_center[i, :] = k_means_sum[i, :] / k_count[i]


            if np.array_equal(ten_means_center, new_center):
                break
            elif (iteration > 30) :   # just for test
                break
            else :
                ten_means_center = new_center

            iteration += 1

            reshape_plot_img(k_ten, ten_means_center)  # iteration center
```

```
Iteration : 0
Train Energy : 2295.214866
Train Accuracy : 0.247800
```

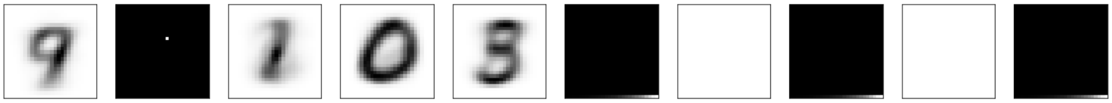


```
C:\Users\ys\Anaconda3\lib\site-packages\ipykernel_launcher.py:18: RuntimeWarning: overflow enc



Iteration : 1
Train Energy : 477.465321
Train Accuracy : 0.345950


C:\Users\ys\Anaconda3\lib\site-packages\ipykernel_launcher.py:51: RuntimeWarning: divide by ze
C:\Users\ys\Anaconda3\lib\site-packages\matplotlib\image.py:405: UserWarning: Warning: converti
  dv = (np.float64(self.norm.vmax) -
C:\Users\ys\Anaconda3\lib\site-packages\matplotlib\image.py:406: UserWarning: Warning: converti
  np.float64(self.norm.vmin))
C:\Users\ys\Anaconda3\lib\site-packages\matplotlib\image.py:413: UserWarning: Warning: converti
  a_min = np.float64(newmin)
C:\Users\ys\Anaconda3\lib\site-packages\matplotlib\image.py:418: UserWarning: Warning: converti
  a_max = np.float64(newmax)
C:\Users\ys\Anaconda3\lib\site-packages\matplotlib\colors.py:916: UserWarning: Warning: conver
  dtype = np.min_scalar_type(value)
```

```
C:\Users\ys\Anaconda3\lib\site-packages\numpy\ma\core.py:715: UserWarning: Warning: converting
  data = np.array(a, copy=False, subok=subok)
```



```
Iteration : 2
Train Energy : 464.466650
Train Accuracy : 0.366667
```



```
Iteration : 3
Train Energy : 465.381999
Train Accuracy : 0.371567
```