Prof. Dr. Rudolf Mathar, Dr. Arash Behboodi, Emilio Balda

Exercise 3

Friday, November 3, 2017

Problem 1. *MNIST dataset (raw data version)*

In this example we load the MNIST dataset from the provided files, without the aid of MNIST specific loading APIs.

```
In [1]: # %matplotlib inline
        from matplotlib.pyplot import imshow
        import matplotlib.pyplot as plt
        import numpy as np
        import csv
```

We first define the file names corresponding to the dataset as well as the directory where they are stored.

```
In [2]: FILES_DIR = 'MNISTcsv_files/'
        TRAIN_FILE = 'train.csv'
        TEST_FILE = 'test.csv'
```

We now open the provided dataset files and output their sizes. The size of training and test set is the same as before. Note that the validation set should be manually constructed from the training set.

```
In [3]: with  open(FILES_DIR + TRAIN_FILE) as csvfile:  # Open this file
            rowreader = csv.reader(csvfile, delimiter=',')
            data = list(rowreader)
            print('Size of '+ TRAIN_FILE +': ' + str(len(data)))
            print('Size of Training Set: ' + str(len(data)-1))

        with  open(FILES_DIR + TEST_FILE) as csvfile:  # Open this file
            rowreader = csv.reader(csvfile, delimiter=',')
            data = list(rowreader)
            print('Size of '+ TEST_FILE +': ' + str(len(data)))
            print('Size of Training Set: ' + str(len(data)-1))
```

```
Size of train.csv: 42001
Size of Training Set: 42000
Size of test.csv: 28001
Size of Training Set: 28000
```

Now we would like to take a look on the format of the provided data. To that end, we print an specific rows of the data to take a look at its format.

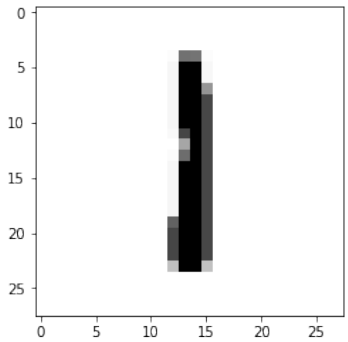
```
In [4]: SAMPLE_INDX = 1 #ROW number to be displayed
        with  open(FILES_DIR + TRAIN_FILE) as csvfile:  # Open this file
            rowreader = csv.reader(csvfile, delimiter=',')
            interestingrows = [row for idx,  \
                               row in enumerate(rowreader) \
                               if idx == SAMPLE_INDX ]
            for row in interestingrows:
                print('Raw data format:')
                print(row)
                print('Length of 1 row of ' + TRAIN_FILE + ' : ' \
                      + str(np.array(row).shape[0]))
```

```
Raw data format:
['1', '0', '0', '0', '0', '0', '0', 0', '0', ..., '0', '0', '0', '0']
Length of 1 row of train.csv : 785
```

We now extract the label value and its corresponding gray-scale image.

```
In [5]: SAMPLE_INDX = 3
        with  open(FILES_DIR + TRAIN_FILE) as csvfile:  # Open this file
            rowreader = csv.reader(csvfile, delimiter=',')
            interestingrows = [row for idx, \
                               row in enumerate(rowreader) \
                               if idx == SAMPLE_INDX]
            for row in interestingrows:
                print('Label: '+ row[0])
                mnist_image = np.array(row[1:]).reshape(28,28).astype(int)
                imshow(mnist_image ,cmap='binary')
                plt.show()
```

```
Label: 1
```



We repeat this process for the test data.

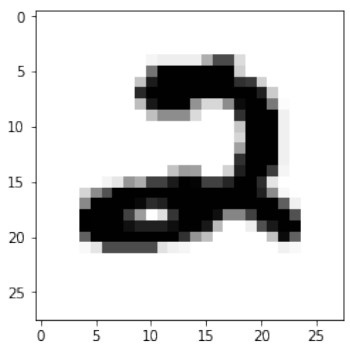
```
In [6]: SAMPLE_INDX = 1 #ROW number to be displayed
        with  open(FILES_DIR + TEST_FILE) as csvfile:  # Open this file
            rowreader = csv.reader(csvfile, delimiter=',')
            interestingrows = [row for idx, \
                               row in enumerate(rowreader) \
                               if idx == SAMPLE_INDX]
            for row in interestingrows:
                print('Raw data format:')
                print(row)
                print('Length of 1 row of ' + TEST_FILE + ' : ' \
                      + str(np.array(row).shape[0]))
```

```
Raw data format:
['0', '0', '0', '0', '0', '0', '0', 0', '0', ..., '0', '0', '0', '0']
Length of 1 row of test.csv : 784
```

As expected, this file has no labels associated to the images. We now plot the an image of this dataset.

```
In [7]: SAMPLE_INDX = 1
        with  open(FILES_DIR + TEST_FILE) as csvfile:  # Open this file
            rowreader = csv.reader(csvfile, delimiter=',')
            interestingrows = [row for idx, \
                               row in enumerate(rowreader) \
                               if idx == SAMPLE_INDX ]
            for row in interestingrows:
                print('Length of 1 row of ' + TEST_FILE + ' : ' \
                      + str(np.array(row).shape[0]))
                print('(NO Label)')
                mnist_image = np.array(row[0:]).reshape(28,28).astype(int)
                imshow(mnist_image ,cmap='binary')
                plt.show()
```

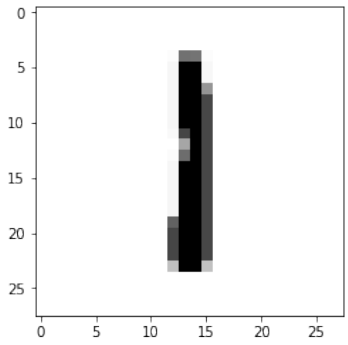
```
Length of 1 row of test.csv : 784
(NO Label)
```



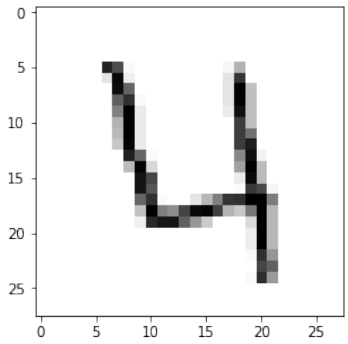
In applications where the datasets are large, it is not feasible to load all the data into memory at once. Therefore, it is common practice to read only small portions data at a time. This small chunks of data are commonly refered to as 'batch'. For example, in the following script we load a batch of 7 images and display them along with their labels.

```
In [8]: INDX_START = 3 # Start of the batch
        INDX_END = 9 # End of the batch
        with  open(FILES_DIR + TRAIN_FILE) as csvfile:  # Open this file
            rowreader = csv.reader(csvfile, delimiter=',')
            interestingrows = [row for idx, \
                               row in enumerate(rowreader) \
                               if idx in range(INDX_START, INDX_END) ]
            for row in interestingrows:
                print('Label: '+ row[0])
                mnist_image = np.array(row[1:]).reshape(28,28).astype(int)
                imshow(mnist_image ,cmap='binary')
                plt.show()
```

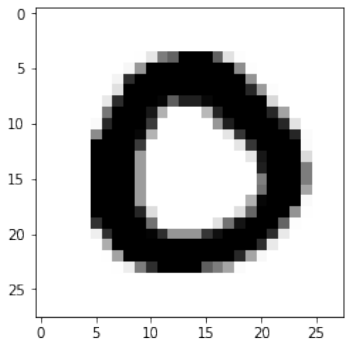
```
Label: 1
```



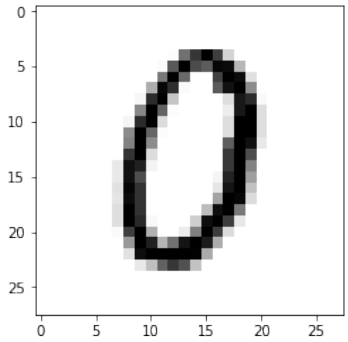

```
Label: 4
```



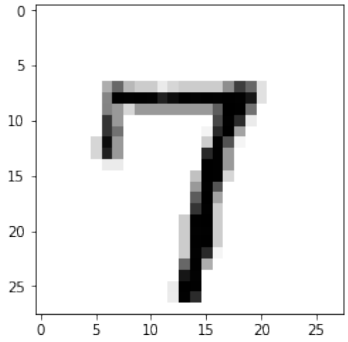

```
Label: 0
```



Label: 0



Label: 7



Label: 3