4. Construct a K-Nearest Neighbors classifier for the MNIST dataset.

My code is as follows:

```
“import numpy as np
import scipy.io as sio
from collections import Counter

mnist_data = sio.loadmat('mnist_data.mat')
train_data = mnist_data['train']
test_data = mnist_data['test']

train_images = np.asarray(train_data[:,1:785])
train_labels = np.asarray(train_data[:,0])
test_images = np.asarray(test_data[:,1:785])
test_labels = np.asarray(test_data[:,0])

errorcount = 0
klist = [1, 5, 9, 13]
for k in klist:
 for i in range(100):
  randnum = np.random.choice(test_labels.size)
  test_image = test_images[randnum]
  test_label = test_labels[randnum]
  distances = [(np.linalg.norm(test_image - image), label) for (image, label) in zip(train_images,
train_labels)]
  distsort = sorted(distances, key = lambda tup: tup[0])
  k_labels = [label for (_, label) in distsort[0:k]]
  win_label, freq = Counter(k_labels).most_common()[0]
  if (int(win_label) != int(test_label)):
   errorcount += 1
 print('for k = ', end =' ')
 print(k, end =' ')
 print(', the error is ', end =' '))
 print(errorcount/100)”
```

Sample output is:

```
for k = 1 , the error is 0.02
for k = 5 , the error is 0.03
for k = 9 , the error is 0.07
for k = 13 , the error is 0.11
```

We thus conclude that k=9 gives the best classification accuracy.

b) Consider the L1-Norm, and choose between L1- and L2-normed classifiers.

I change the distance metric from L2-Norm to L1-Norm. This is simply done by changing the “np.linalg.norm(test_image – image)” to “np.linalg.norm((test_image – image), 1). Doing this gives the following output:

for k = 1 , the error is 0.03
for k = 5 , the error is 0.08
for k = 9 , the error is 0.12
for k = 13 , the error is 0.15

We note that for all values of k in our set, the L2-Norm produces a lower error, so we select this as our distance measure.