

The accuracy for my KNN implementation peaked at 90% with a k of 7 and 9. I used a loop to figure out the optimal K by calculating the accuracy for each K.

To implement KNN what I did was read in the CSV and then pass it into my KNN function. From there I take the test data and iterate over each row which represents an image. From there I iterate over each "image" in the training data and subtract the test image from the training image. Since numpy by default will do element wise arithmetic we can subtract the two vectors normally. We then square the difference to get rid of the negatives, sum up all the elements, then take the square root. This gives me the "distance" from the images. Once I process all training images I then sort the distances and take the "K" shortest images. From there I find what these images represent and return the majority representation. Then I check if it was correct or not

```
In [66]: import numpy as np
import pandas as pd
from collections import defaultdict
```

```
In [67]: def find_majority(labels):
    c = defaultdict(int)
    for l in labels:
        c[l] += 1

    majority = max(c.values())
    for k, v in c.items():
        if v == majority:
            return k

def knn(test_data, training_data, k):
    prediction = []
    for idx, img in test_data.iterrows():
        distance = []
        d = 0.0
        for i, t in training_data.iterrows():
            d = np.math.sqrt(sum((img.values.astype(float)[1:] - t.values.as
            distance.append((t[0], d))
        label, distance = zip(*(sorted(distance, key=lambda tup: tup[1])[:k]
        guess = find_majority(label)
        prediction.append(img[0] == guess)

    return prediction.count(True) / float(len(prediction)) * 100
```

```
In [68]: training_data = pd.read_csv('MNIST_training.csv', skiprows=[0],header=None)
test_data = pd.read_csv('MNIST_test.csv', skiprows=[0],header=None)
```

```
In [69]: for k in range(1,41,2):  
         p = knn(test_data, training_data, k)  
         print "Accuracy for k %i is %f" % (k, p)
```

```
Accuracy for k 1 is 84.000000  
Accuracy for k 3 is 86.000000  
Accuracy for k 5 is 86.000000  
Accuracy for k 7 is 90.000000  
Accuracy for k 9 is 90.000000  
Accuracy for k 11 is 84.000000  
Accuracy for k 13 is 82.000000  
Accuracy for k 15 is 82.000000  
Accuracy for k 17 is 82.000000  
Accuracy for k 19 is 82.000000  
Accuracy for k 21 is 82.000000  
Accuracy for k 23 is 82.000000  
Accuracy for k 25 is 80.000000  
Accuracy for k 27 is 80.000000  
Accuracy for k 29 is 78.000000  
Accuracy for k 31 is 76.000000  
Accuracy for k 33 is 74.000000  
Accuracy for k 35 is 72.000000  
Accuracy for k 37 is 72.000000  
Accuracy for k 39 is 72.000000
```

In []:

In []:

In []:

In []: