**Design**

**1 Program structure and flow:**

First, when the program starts, the computer first call the function “load\_data”.

In this function, the computer will call the function “read\_digit”. Then it will load all the information of “digit-training.txt” into the dict “g\_dataset”.

In this dict, the keys are digits from “0” to “9”. Each digit corresponds to a large list.

Each list has a lot of elements which are also lists.

Each smaller list store all the index of “1” in each 32\*32 data.

I choose to store index of “1” instead of vector because I want to improve system efficiency. I will tell you why.

In knn model, the important step is to find the distance.

I use Euler distance formula to solve this problem.

However, we know that the testing file has about 2000 digits and each digit has 32\*32 index. If we use Euler distance formula to compare each index of two vectors and calculate the sum of the squares of the difference. We will have to calculate about 32\*32\*2000 times. It is so large that we will waste a lot of time for testing. So I try to change the way of it. (But the principle is still the Euler distance formula.)

When I load the data from every text file, I just load the index of all the “1” for every digit. You can see that by using Euler distance formula:

For every index in these two vectors, if they are both 1 or 0, the square of the difference will be 0.

Only when they are different, (one is 1 and the other is 0) the subdistance will be 1. So we just use the differences between two vectors to find the distance.

I use the property of set and list to transform between them to find the number of all the index in which these two vectors are different from each other. Then I take the square root of this number, I can get the distance. The process has been greatly simplified.

Second, after the program load all the data from “training.txt” file, the computer will call the function “validate”, the program will then load digits from “testing.txt” file one by one until all the digits are loaded. Each digit will be used as a parameter to “knn\_by\_most\_common” function, then in this function it will call the function “knn”.

In “knn” function, it will compare all the digits in g\_dataset with the parameter digit by calling the function “distance” to find the distance between two vectors by using the way what I have said in the first step. It will return a list whose elements are all tuple whose first element is a digit and second element is the distance.

I choice 7 as the value of k.

Then the knn function will sort the list and choose the first k elements. Then it will try to find the most common digit.

After that the “knn\_most\_common” function will return the target value.

The “validate” will compare the target value with the true value.

If they are the same, the value of corresponding number in “g\_test\_good” dict will plus one, or the the value of corresponding number in “g\_test\_bad” dict will plus one.

Then the validate will test next digit in “testing.txt” file until all the digits in it are tested.

After that, finally, the program will call the “predict” function to predict all the digits in “predict.txt” file and then get a target value of each digit.(the process is similar to the process in “validate” function. What is different is that it does not have the process to judge whether the target value is right or not. In fact, it just gets the target value and show them in the terminal.)

Finally, what I want to say is that by using “show\_train”, “show\_test” and “lineBoundary” function to show all the information of training, testing and prediction in the terminal.

**2 Python objects (global variables):**

I import libraries of Python to solve this problem.

I create a dictionary called “g\_dataset” to store training data.

I create a dictionary called “g\_test\_good” to store number of correct data identified.

I create a dictionary called “g\_test\_bad” to store number of incorrect data identified.

I create a constant called “NUM\_ROWS” to symbol the number of rows of each identified data.

I create a constant called “NUM\_COLS” to symbol the number of columns of each identified data.

I create a constant called “DATA\_TRAINING” to symbol the training file that need to be used.

I create a constant called “DATA\_TESTING” to symbol the testing file that need to be used.

I create a constant called “DATA\_PREDICT” to symbol the prediction file that need to be used.

I create a constant called “KNN\_NEIGHBOR” to symbol the knn parameter.

**3 Functions:**

I create a function “read\_digit” to convert next digit from input file as a list. This list contains all the index for “1” in each digit. And this function will return (digit, indexList) or (-1, “”) on end of file. The parameter “p\_fp” is the file which will be read.

I create a function “load\_data” function to parse all digits from training file and store all digits (as lists). Each list contains all the index for “1” in each digit in dictionary g\_dataset. The parameter “p\_filename” is the file which will be loaded.

I create a function “knn” to give it a digit list and return the k nearest neighbor by vector distance. The parameter “p\_v” is the vector list and the parameter “size” is the number of k.

I create “knn\_by\_most\_common” function to base on the knn model (nearest neighbor) and return the target value. The parameter “p\_v” is the vector list.

I create a function “predict” to make prediction based on knn model. It parses each digit from the predict file and print the predicted value. The parameter “p\_filename” is the file which will be predicted.

I create a function “validate” to compile an accuracy report by comparing the data set with every digit from the testing file. The parameter “p\_filename” is the file which will be used to test.

I create a function “data\_by\_random” to randomly select X samples for each digit. X is the new size of each digit data in g\_dataset. It will reduce the kNN computation time. The parameter “size” is the number of X.

I create a function “distance” to return distance between vectors v & w. It will compare two lists which contain the index of all the “1” in each digit and return the distance. The parameter “v” and “w” are the two vector lists.

I create a function “show\_train” to show information for training data set. The parameter “start” is the beginning time of training in the terminal. The parameter “stop” is the end time of training in the terminal.

I create a function “show\_test” to show information for testing data set. The parameter “start” is the beginning time of testing in the terminal. The parameter “stop” is the end time of testing in the terminal.

I create a function “lineBoundary” to show each title with boundary in the terminal. The parameter “content” is the content of each title of each part.

**4 Output:**





