Assignment3: kNN classifier

Saeed Kalateh, Student, Robotic Engineering,

Abstract—In this assignment, the KNN classifier is implemented using Mnsit data set. First, the model is trained and then the accuracy is calculated for different k values. Finally, the results are compared and plotted.

Index Terms—k nearest neighbor algorithm

1 Loading data

The Mnist data and class labels are imported as the first step. *loadMnist* function is used and according to Matlab help, it gets two arguments as input. The first argument, or "which" according to help, is set to 0. This will load the training set for our project. The set has 60000 observations with 784 attributes. The attributes are 28x28 grayscale images of handwritten numbers.
The code asks for k number after execution and the user must set the k number. If k satisfies one of these two conditions, then the algorithm will ask again.

$$k <= 0 \text{ or } k > \text{Number of Observations}$$

2 Similarity Matrix

Similarity matrix has been defined to calculate the distance between train and test data. This matrix is a 2 dimensional 60000x10000 array of distances between train and test data. For example, similarity(1,2) calls the distance between first observation in train set and second item in the test set.
To calculate these distances, a custom function is written and named dist_func, which takes two vectors as input and returns their euclidean distance.
Next, we need to know the closest train example to each of our test examples. This needs a function like argmin in python, and I have written it in Matlab with the same name. This function iterates through a vector and passes the k highest values. Therefore, it has two input arguments, k and a vector (column of similarity matrix).
Finally, we need to check the label of k nearest neighbors from train labels. These values are stored in the first row of *predicted* matrix in the code. To check the accuracy of prediction, the second row contains the true value for test labels.
After comparing the first and second rows of *predicted* matrix, and dividing the true predictions by total number of test labels, the accuracy is obtained. This process is repeated for different number of k values. The accuracy for different k and training - test examples have been summarized in the following figures.
Note: Using the total number of observations in the data set takes a lot of time. Therefore, a limited number of train and test data instances have been used to test the code.
Figure 1 and 2 show that although the accuracy becomes less sensitive to k for larger sets, kNN shows its best performance on Mnist when the value of k is about 5.

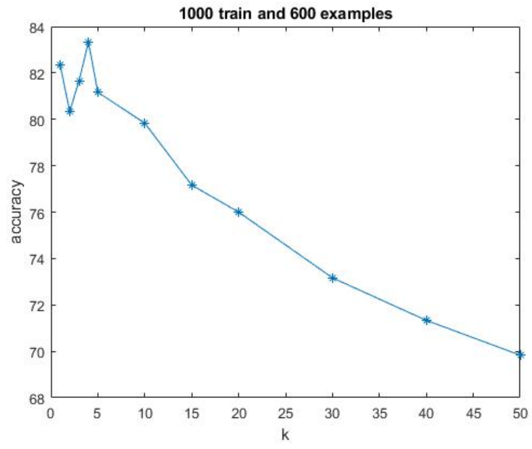

Fig. 1. k-accuracy diagram using 1000 train and 600 test instances

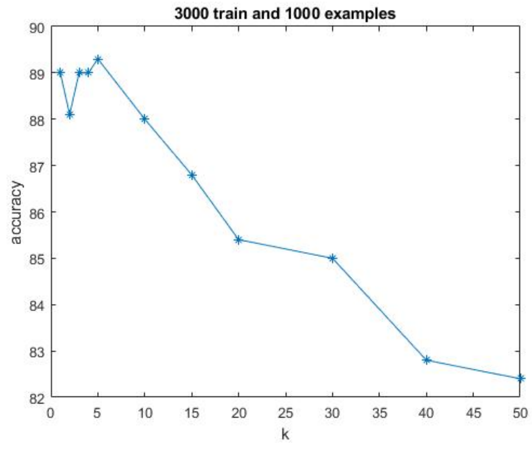

Fig. 2. k-accuracy diagram using 3000 train and 1000 test instances