Machine learning Assignment3

YESHWANTH GURU KRISHNAKUMAR

S5059111

1 Introduction

1.1 kNN classifier

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. It assumes that similar things exist in close proximity. [1] Example:

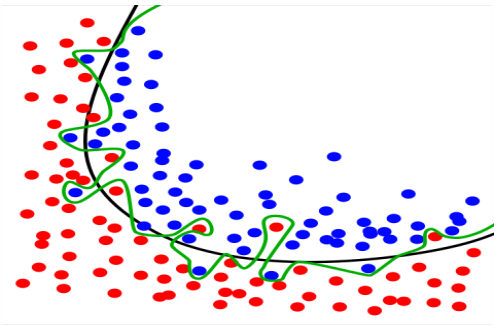

Figure 1: Similar data points typically exist close to each other

In order to implement this algorithm we need a training set X, a query point x and a value k. Then, given a set of n training examples, when the kNN classifier receives a new instance to predict, it firstly identifies the k nearest neighboring training examples of the new instance and then assigns the class label holding by the highest number of neighbors to the new instance. In formulas:

$$\{n_1, n_2, ...n_i, ...n_k\} = top_k||x_i - \bar{x}|| \tag{1}$$

$$y = mode\{t_n1, t_n2, ..., t_nk\} \tag{2}$$

1.2 The dataset

In order to try our code we used the MNIST dataset, which is composed of images of handwritten digits from 0 to 9. The training set has 600000 examples and the test set has 100000 examples.

2 Matlab code

2.1 main.m

The main first loads the MNIST training set and test set. Then it reduces the lines of both datasets by taking random samples, since the code takes a very long time to execute by taking the whole dataset. Then it applies the kNN classifier on the training set and tests it on the test set, by also showing some of the classified images. It averages accuracies and plots them with reference to the k. Finally it tests the kNN by considering each single digit and plots its accuracy.

2.2 knnClassifier.m

Inputs:

- TrainSet: training set + training labels
- TestSet: test set without labels
- k: parameter of the kNN classifier
- TestLabels: test labels (optional input)

Outputs:

- Predictions: predicted class for each observation
- Error: total error of the classifier

This function implements a kNN classifier. It first controls that the number of inputs received is at least equal to the number of required inputs and that the number of columns of the test set has exactly one column less than the training set. Finally it also checks that k >0 and k≤ (cardinality of training set).
Then it uses the Matlab function pdist2() on the training and test sets. This function computes the pairwise distance between two sets of observations by using a certain metric. In here the metric is euclidean for the distance, so the function outputs:

- matrix D of the K smallest pairwise distances to observations in the training set for each observation in the test set in ascending order.
  Size: k x TEST_COLS
- matrix TrainIdx of the indices of the observations in the training set corresponding to the distances in D.
  Size: kxTEST_COLS

The matrix TrainIxd is used to index the training set labels that correspond to the k nearest neighbours. Then, since it is a classification problem, the mode of the k labels is computed in order to define the output. The resulting matrix is the Predictions matrix.
Finally, if the actual labels of the test set are present, the error is computed, thus obtaining the Error matrix.

3 Results

By calculating the accuracy on the reduced training set, we can see by its plot that it decreases with respect to increasing values of k, starting from approximately 93% for k=10 to 86% for k=110. This is due to the fact that a higher k takes into consideration too many neighbouring pixels that are not important to determine the query point's class.
By considering the single digits, the trend is the same, however the accuracy is generally higher. In fact, it's easier to recognize if a digit is for example a 0 or not, compared to actually understanding which digit it is.

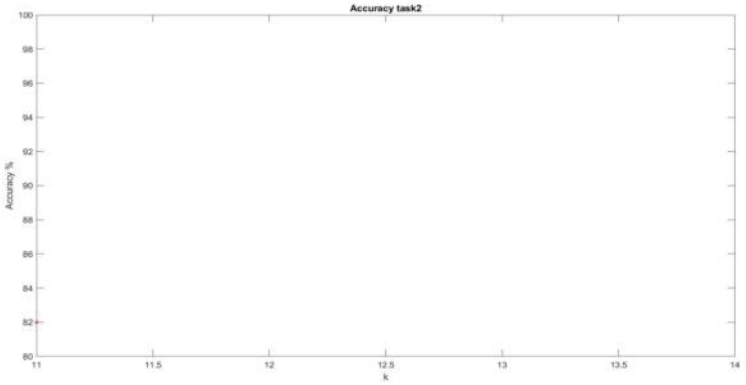

Figure 2: Accuracy

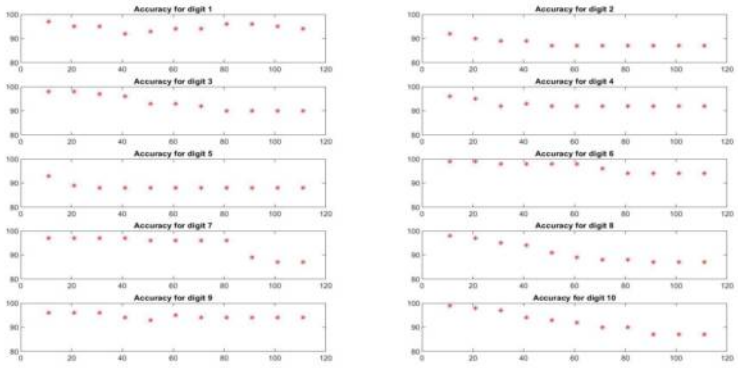

Figure 3: Accuracy on each digit