Assignment 3

Ryan

December 14, 2020

Abstract

In this assignment, I attempt to code a number classifier to classify written numbers converted to digital representations.

Introduction

We were given access to an mnist.mat file containing 60000 training number images and 10000 testing number images along with their labels correctly identifying them. We were asked to create a classifier using K nearest neighbor to attempt to correctly guess the numbers from our test set using the training set for comparison. On a side note, my initial attempts at creating this classifier ended with incredibly low accuracy. I was not sure how accurate my results were supposed to be until I heard from my classmates about their own attempts. Thanks to their suggestions as well as our professor, I was able to correct my mistake. I will discuss this more later in this document.

1 Initial Setup

I initally downloaded the mnist.mat file from the given github profile and used the code given from the same profile to access the data set.

2 Decision Points

2.1 Decision 1

My first decision was for how I wanted to calculate the nearest neighbor with the euclidean distance formula. In class we went over its use and saw that we could remove the square root part of the equation since we were only interested in the minimum distance. The square root wouldn't change which distances were greater or less.

$$dist = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

With Square Root:

$$dist = \sqrt{(2)^2 + (3)^2} = 3.6 \quad > \quad dist = \sqrt{(1)^2 + (2)^2} = 2.23$$

Without Square Root:

$$dist = (2)^2 + (3)^2 = 13 \quad > \quad dist = (1)^2 + (2)^2 = 5$$

For my initial testing, I decided to forgo the square root to decrease the amount of calculations needed each time I ran the program. Once I had finished testing my program I reintroduced the square root, just to keep the standard of this approach. An issue I ran into was large numbers. When I attempted to find the euclidean distance of each point, I would end up with incredibly large numbers which introduced errors into my calculations. Thanks to my classmates suggestions and the direction of my professor, I learned that I needed to normalize the numbers in my data sets in order to keep the numbers in a stable state. I accomplished this by dividing each point by the maximum sized point in each set.

2.2 Decision 2

My next decision was how to calculate the cosine distance for the nearest neighbor method. This one took many bouts of trial and error and I struggled to get it to work correctly using a direct conversion from the formula to python. I did not have a large amount of time to figure out the specifics so I eventually opted to use the built in function from scipy to accomplish this.

2.3 Decision 3

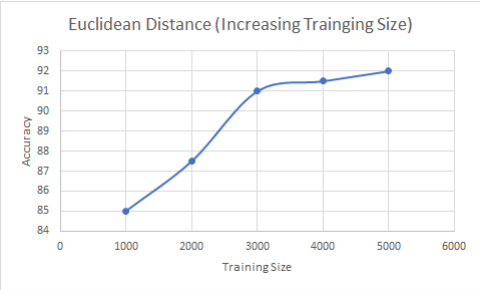
My next decision pertained to creating both the confusion matrix and the histogram. It took me a bit of time to decide how I wanted to do this but I eventually settled on creating a 10x10, 2-dimensional list of zeros initially for the confusion matrix and a list for the histogram. I decided to just create a confusion matrix and histogram for the euclidean distance algorithm since I had a better understanding of it and to reduce the number of graphs I would eventually end up with. For each image from the test set, I ran it through my algorithm and each time it guessed incorrectly I would increase the value at the cell of the incorrect number and the correct number.

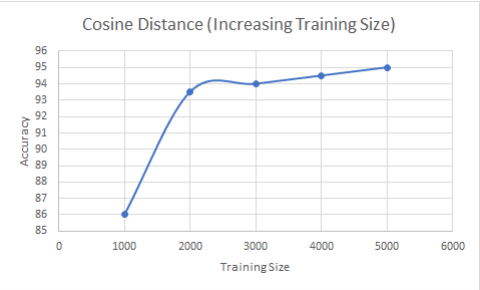
For example, if my algorithm guessed 2 and the correct number was 5, then it would increment the number at cell (2, 5). I would also add 2 to the histogram list. Then, when I created the histogram it would count all occurrences of each number in the list and create the appropriate histogram. I also added some additional parameters to the histogram graph to make it a distribution and to center the bars at their respective numbers.

3 Results

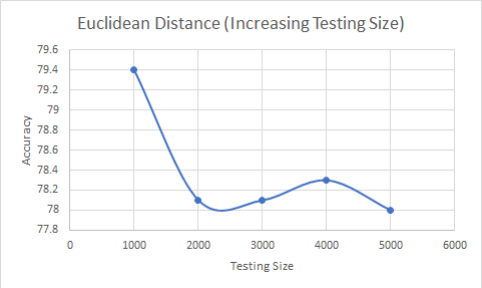
When attempting to classify these numbers without normalizing the data sets, my accuracy dropped to an average of 21%. After normalizing the data, my accuracy rose to an average of 90% when increasing the training size and keeping the test size the same. When keeping the training size the same and increasing the test size, my accuracy was roughly 78% for euclidean distance and 81% for cosine distance. These results are rather good for the simplicity of this method but are not good enough for any type of commercial use due to the consequences of incorrectly guessing digits. My results are given on the next page.

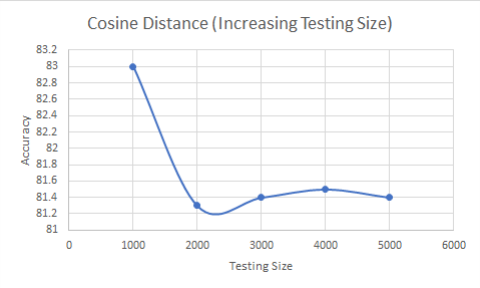
Accuracy with up to 5000 training images and testing size of 200:



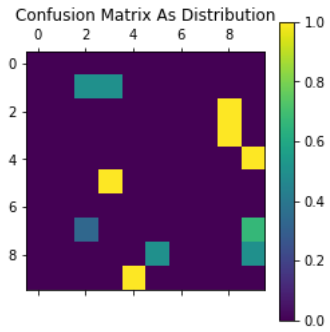


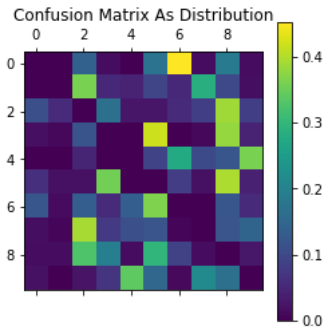
Accuracy with up to 5000 testing images and training size of 500:



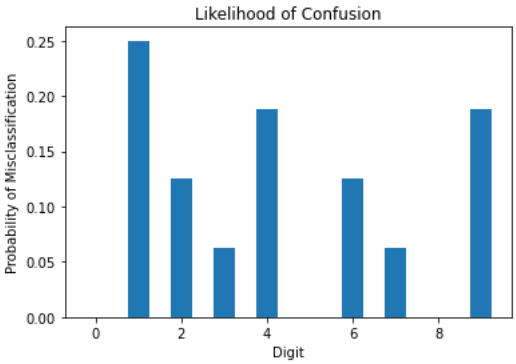


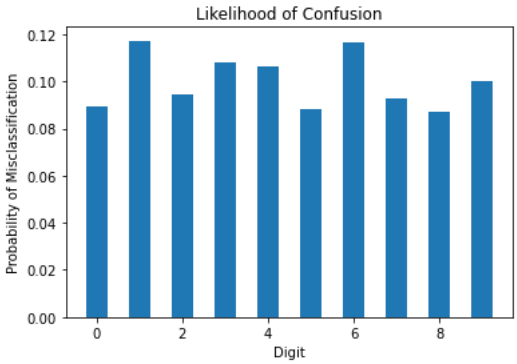
My confusion matrices show the distribution of errors for both a training size of 5000 and test size 200 on the left and a training size of 500 and test size of 5000 on the right. These results give us a visualization of which numbers were misclassified as other numbers. When we keep the testing size the same and increase the training size, we see only a few numbers misclassified as others. When we keep the training size the same and increase the testing size, there is a larger distribution of numbers that are incorrectly misclassified.





Lastly, my histogram graphs show that when there is a large number of training images and small number of test images (5000, 200), the numbers 1, 4, and 9 are misclassified the most and when there is a small number of training images and a large number of testing images (500, 5000), each number is just as likely to be misclassified.





Overall I am happy with these results as they appear to align with the expected results.

4 What I learned

I learned a simplistic approach of K nearest neighbor to classify written numbers and I learned of other ways to optimize the accuracy based on different approaches such as image scaling, although I did not attempt to implement them in this assignment. I learned ways to calculate euclidean distance for large data sets as opposed to just 2 points and I also learned a usage of the scipy library for distance. I also gained a little more knowledge of the plotting functionality in python. Finally, I learned the importance of understanding your data set and how important it is to consider the limitations of the technology being used when writing algorithms and processing data.