MNIST Digit Classifier in PCA Space

Brandon Brodzinski

November 2022

Contents

1 Introduction

In this project, I implemented a deterministic classifier; K-Nearest neighbor in a PCA subspace. The focal point of this assignment is to establish a written digit classifier using the MNIST database and predict the correct number based on a given image. I utilized a Principal Component Analysis subspace and reduced the dimensionality of the original data points to improve accuracy. KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label in the case of classification.

2 Implementation of PCA

In this section, I will examine the process of how I implemented PCA. Principal component analysis, or PCA, is a dimensionality-reduction system. The process is utilized to decrease the dimensions of a big data set by transforming variables into smaller ones that still contain a vast majority of details from the larger set. The methods I used to complete this were standardization, covariance matrix computation, computation of eigenvectors and eigenvalues, and finally projection of the subspace onto the dataset.

3 Implementation of KNN

Furthermore, after implemening PCA I will discuss the process of how I implemented this algorithm. The data set supplied already has a split of training set of 60,000 images and a testing set of 10,000 images. To being with, I wrote a function to calculate the Euclidean distance; which is the length of line segment between two points represented by the formula d = $\sqrt{(x2–x1)^2 + (y2–y1)^2}$. Following, I made a function to compute the KNN by passing it a query point, training data, and a specified K value. It's important to choose an odd K value so that ties within classification are avoided. After the distances are computed within the function it then calculates which digit it is most like by choosing the class that is the max number and uses that as the prediction. For example, if the K value is 5 and 2 labels are the digit 0, and 3 labels are the digit 8, then 8 would be the prediction since 3 is greater than 2.

3.1 Decision Points

After running my first test, I grasped that with a large chunk of data this algorithm performance is hindered. This led me to vectorizing as many things as I could to speed up the process. This was accomplished by avoiding as many for loops as I could and using many built in functions in numpy so that my function could be optimized to the best of its ability. Another key factor I perceived was, I was getting an accuracy of 30% or lower on every run I did. But after augmented research I discerned that 0 values weren't being avoided as inputs, which are capable of preventing weight updates. Subsequently, I corrected this by multiplying each pixel by 0.99 / 255 and adding 0.01 to the result of the training/testing set. This amended my accuracy tremendously and increased it to 90% and above on most runs depending on how many training images were utilized.

4 Evaluation

In this section, I review the results. In Figure 1, I used a fixed number of 1,000 test samples, 10,000 training samples, and a K value of 5. I inferred, as you increase the components used the higher the accuracy you will receive. Figure 1 displays a plateau of roughly 90% after the number of components became greater than 10. This demonstrates a large improvement in accuracy as you use more components versus not using a PCA subspace.

4.1 Accuracy Graph

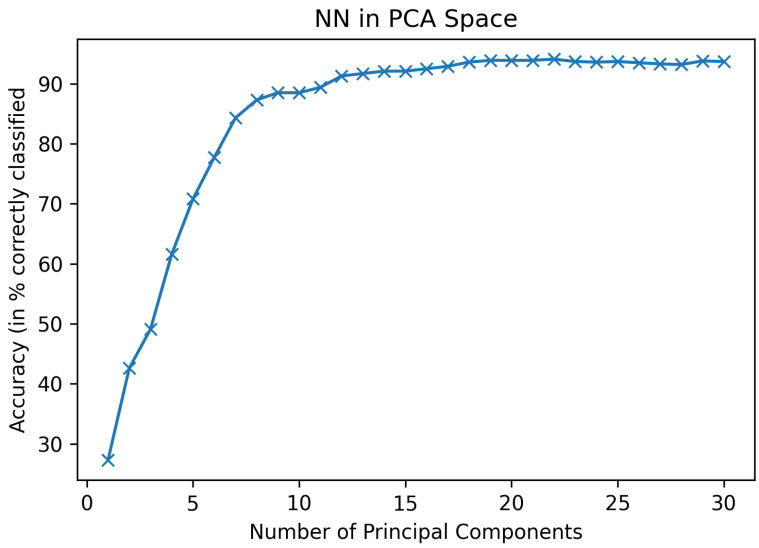

Figure 1: Accuracy as a function of number of principal components, with a fixed number of 1000 test samples.

5 What I've learned

In this project, I became more knowledgeable with the certitude that vectorization is extremely important when it comes to implementing demanding algorithms with large data. Additionally, I discerned the pros and cons of the KNN algorithm. The pros being, no training period is required, it can be used for classification or regression, and a variety of distance criteria can be used. The cons are, decelerated performance in large datsets, does not adequately perform with high dimensions, and is sensitive to noisy data. Briefly, this algorithm solves classification problems easily as there are only two parameters, a K value and a distance. Adjacently, what I also comprehended about PCA is the idea of having x dimensions gives you x principal components. PCA tries to put the maximum information in each component and the remaining information onto the next one. This represents how the maximal amount of variance is dispersed when graphed to envision where most of the data is stored. In the final analysis, PCA removed correlated features and improved algorithm performance swiftly.