# Discriminating Digits 7 & 9

Stat 6560 Final Project

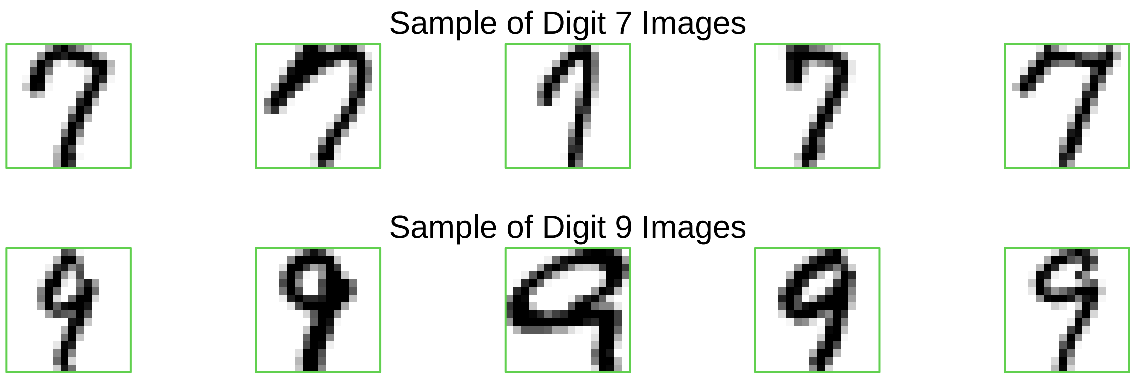
Yousef Qaddura 12-11-2022

## Introduction

We are interested in training a model for distinguishing images of hand-written digits (7 & 9). This is useful for hand-writing recognition systems.

A digit image (resulting from a $16 \times 16$ grayscale matrix) is represented by a 256-dimensional vector of normalized intensity values. There are 645 digit 7 images and 644 digit 9 images.

The following are the first five images of each digit in their corresponding data-sets.

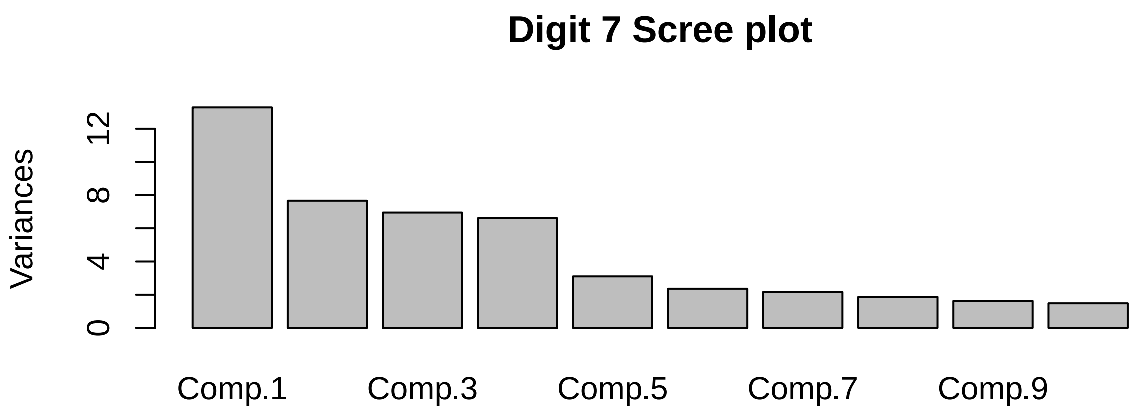


We split that data into a training set of 800 observations (400 for each digit) and a test set of 489 (245 7's and 244 9's).

## Dimensionality Reduction

### Digit 7 Principal Component Analysis

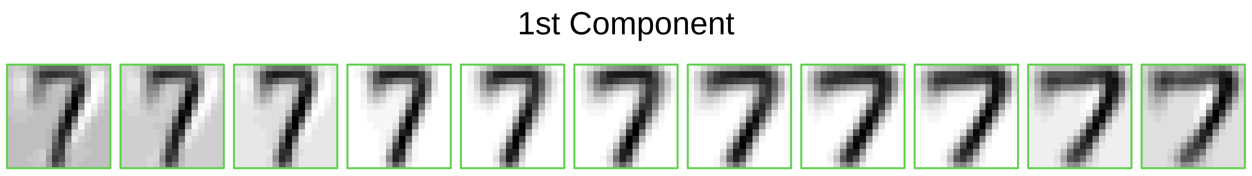
We begin by carrying out a principal component data analysis on the training data for digit 7. The scree plot below shows an elbow taking place at the fifth component. As such, we deem adequate to take the first four principal compoenents to explain variations in the data.

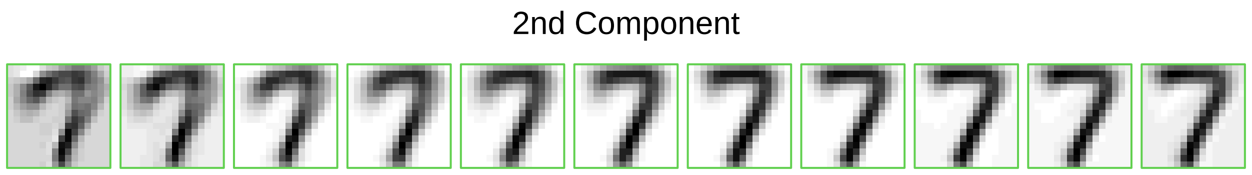


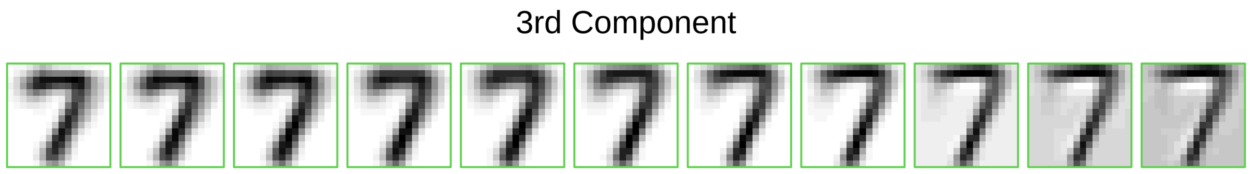
We embark on understanding the kind of variation each of the first few components capture. The first four principal principal components yield mean-centered eigen-images $E_i$ and score standard deviations $s_i$. For each eigen-image, we march starting from the mean using the formula

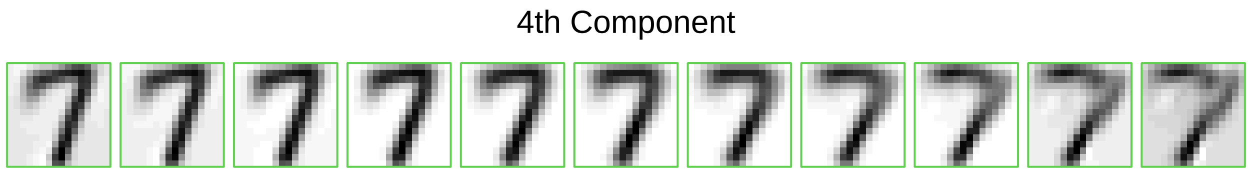
$$\overline{X} + c_i \cdot E_i$$

where $\overline{X}$ is the mean training image and $c_i$ ranges though 11 equally spaced values on the interval $[-2s_i, 2s_i]$ (this includes the endpoints and the value 0). The resulting 11 images for each component are shown below:



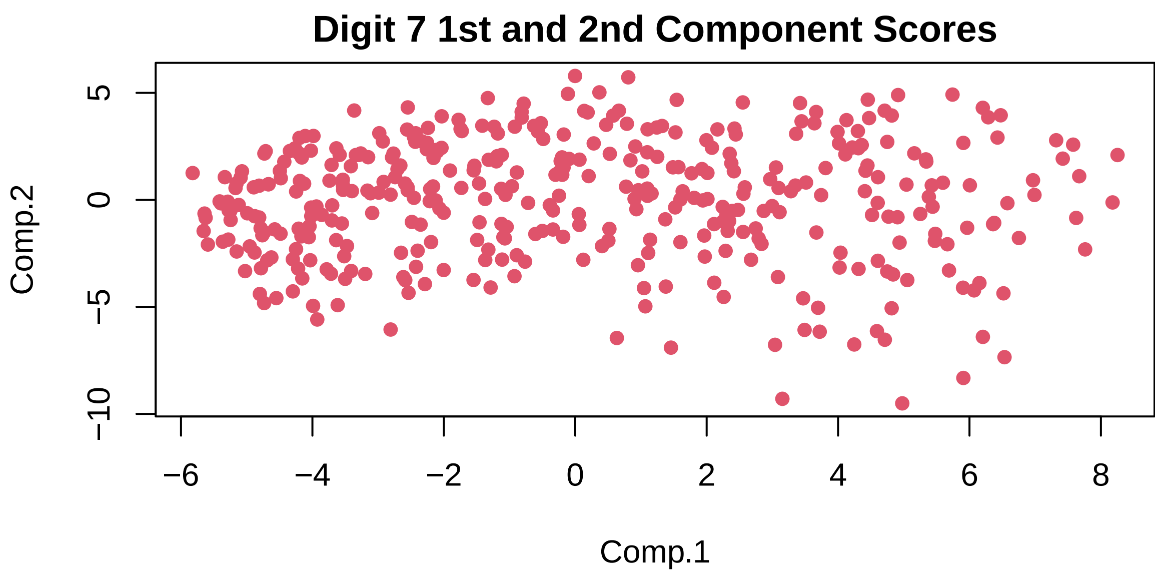




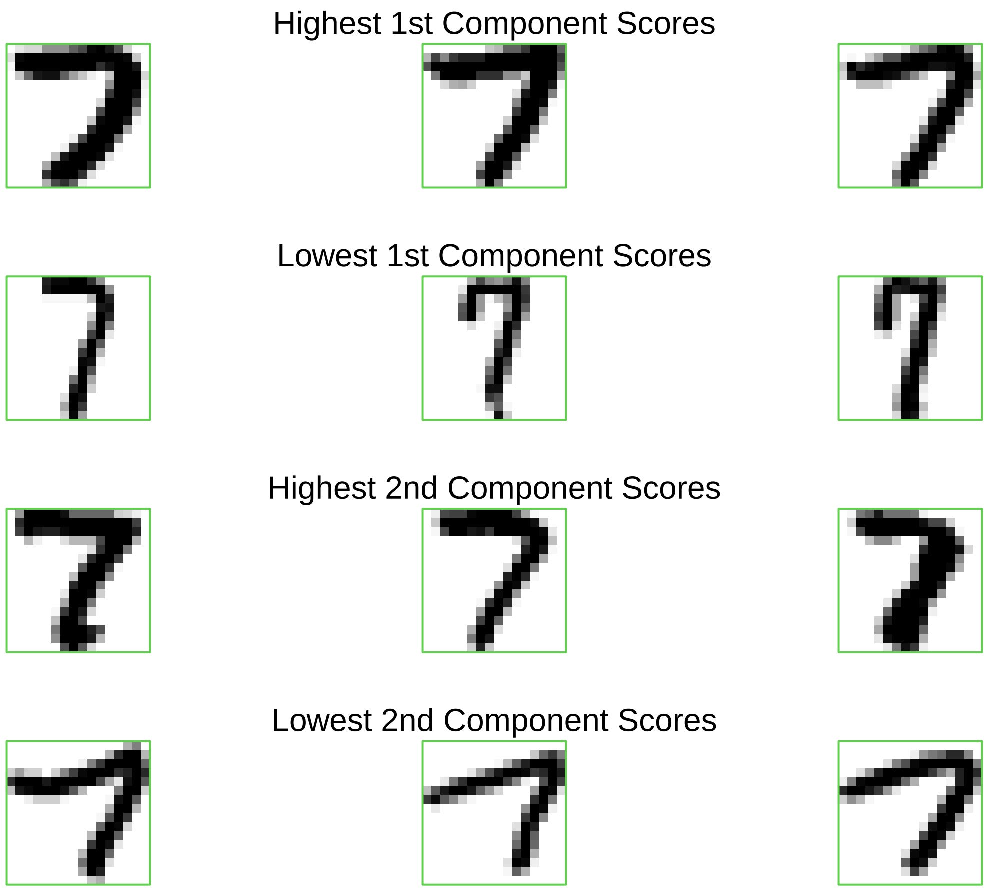


The first principal component describes how wide or thin the seven is. The second describes how obtuse the top edge of the seven is. The third highlights the thickness/thinness of the seven. Lastly, the fourth highlights how rotated the seven is overall.

We now take a look at the score plot for the first two components.



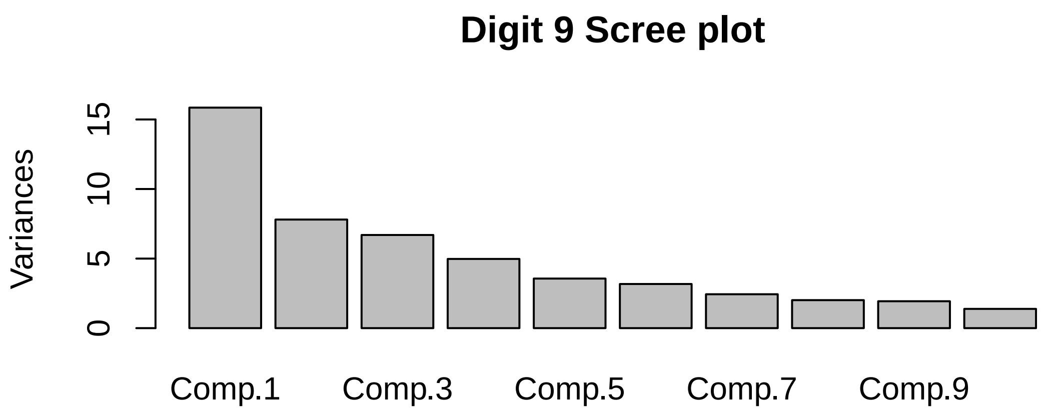
The first component has a longer range of scores. There seems to be a bunch of somewhat stand-outimages with very low score in the second component (nearing $-10$) and a bunch with very high score in the first component (nearing 8). The three extreme images of each component are shown below



The lower 1st component values seem to capture a thinner seven while the higher ones capture a wider seven. The lower 2nd component values seem to capture an acute tilt in the drawing of the seven or an obtuse top edge as opposed to a horizontal top edge captured by the higher 2nd component values.

**Digit 9 Principal Component Analysis**

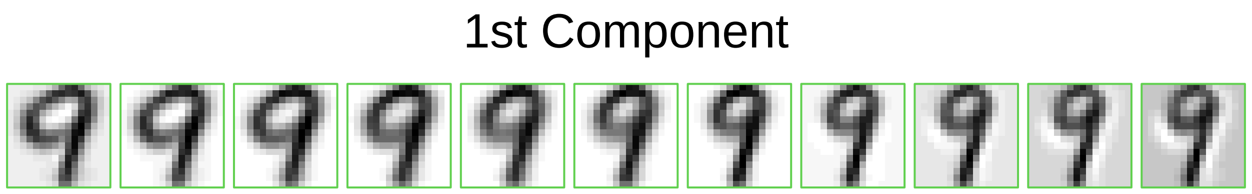
We perform a similar analysis as above to the training data of the digit 9. The scree plot below shows an elbow taking place at the second component, but for the sake of exploration, we deem adequate to take the first four principal compoenents to explain variations in the data.

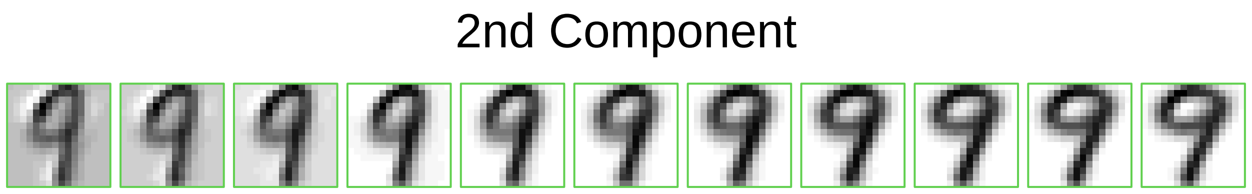


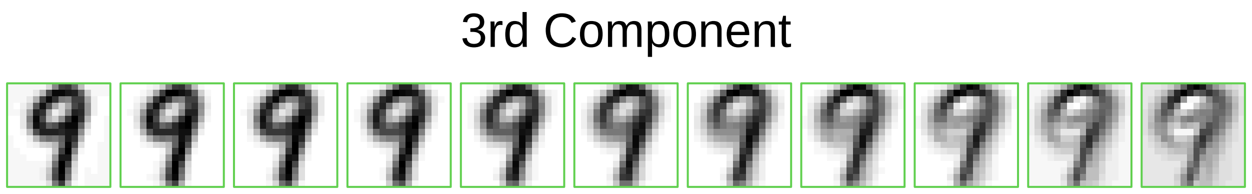
As before, we use the formula

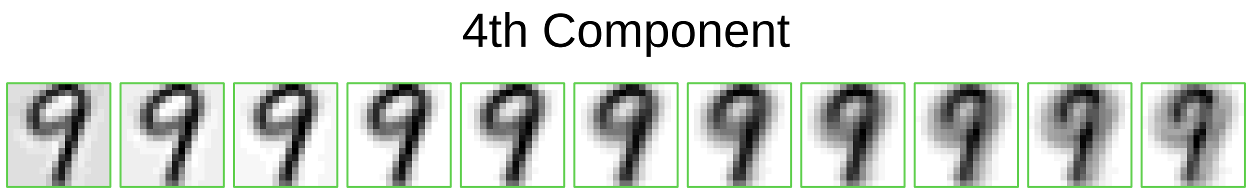
$$\overline{X} + c_i \cdot E_i$$

where $\overline{X}$ is the mean training image and $c_i$ ranges though 11 equally spaced values on the interval $[-2s_i, 2s_i]$ (this includes the endpoints and the value 0). The resulting 11 images for each component are shown below:



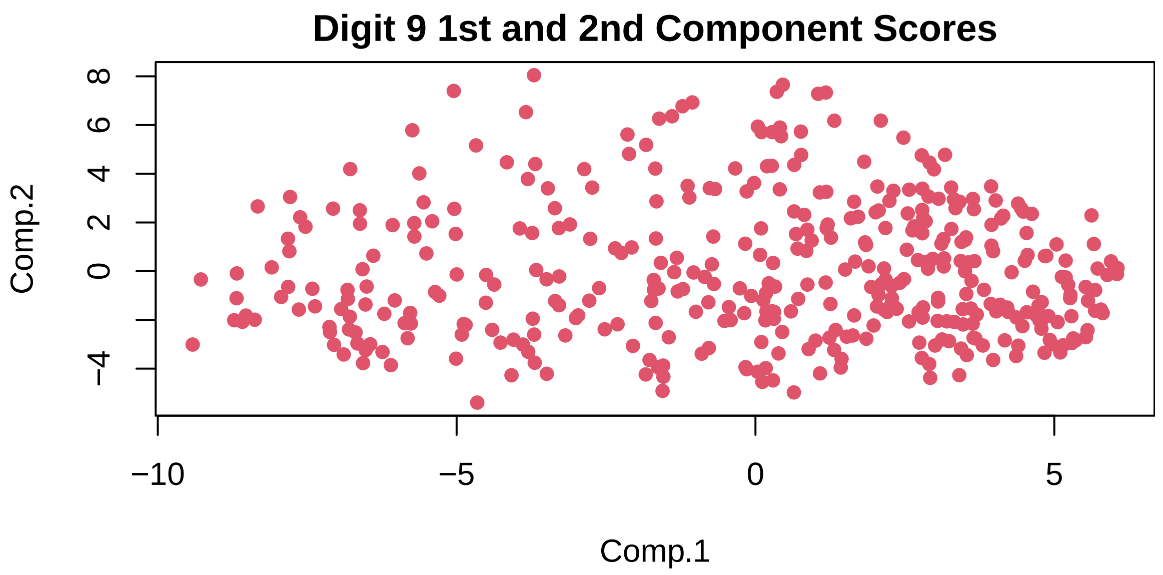




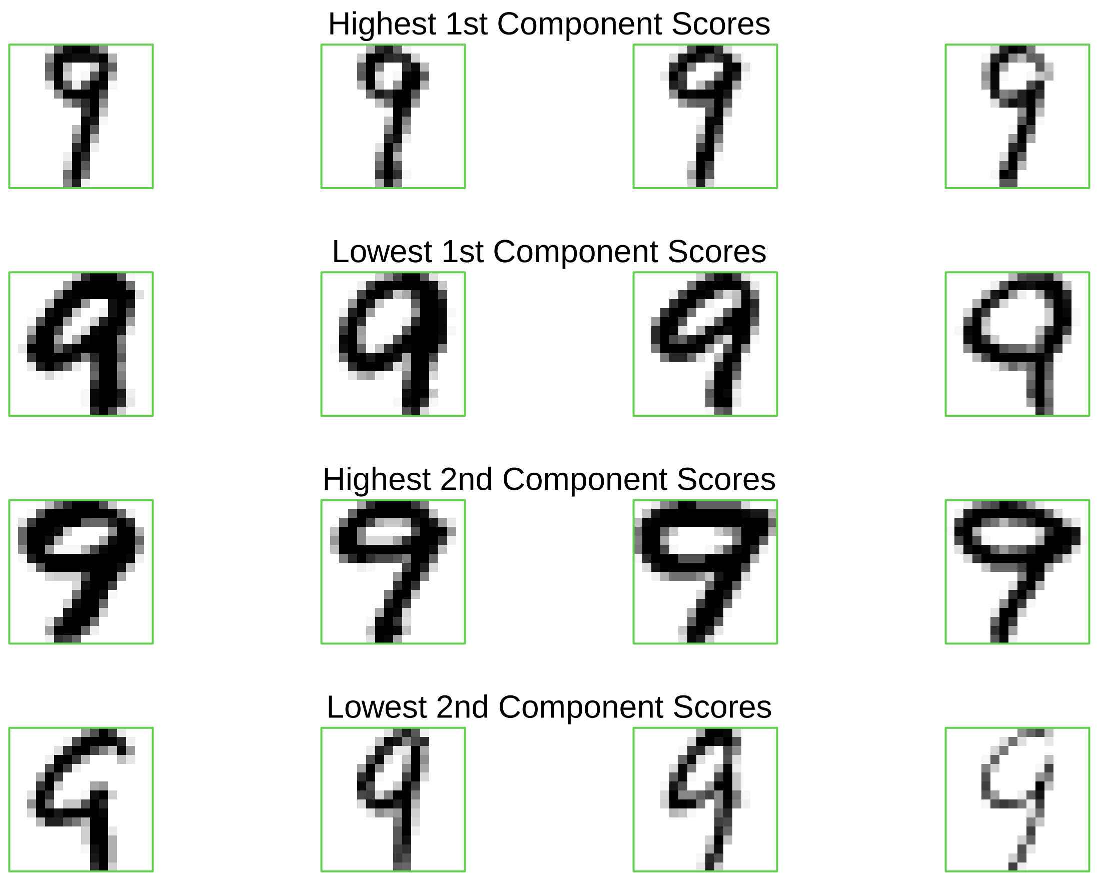


The first component highlights how small the circle of the nine is. The second shows how tilted the oval of the nine is (the oval is more horizontally stretched with higher values and more diagonally stretched with lower values). Next, the third principal component describes how thin/thick the font is. Lastly, the fourth component seems to distinguish between larger nines (higher values) and neater smaller nines (smaller values).

We now take a look at the score plot for the first two components.



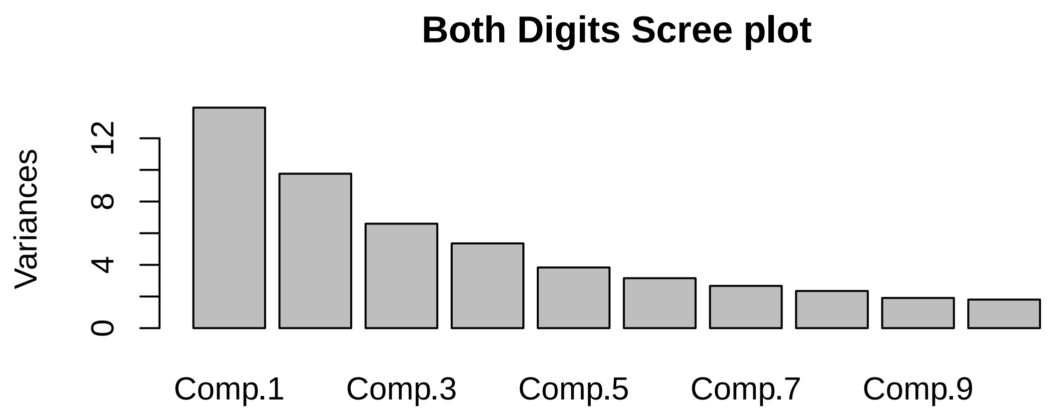
The first component has a longer range of scores. There seems to be a bunch of stand-out images with very high scores in the second component (nearing 8) and a bunch with very low score in the first component (nearing $-10$). The four extreme images from each component are shown below


Highest 1st Component Scores
Lowest 1st Component Scores
Highest 2nd Component Scores
Lowest 2nd Component Scores

Lowest first component scores have nines with very small circles and highest first component scores have nines with largest and more ovally circles. Higher second component scores have nines with horizontally stretched ovals while the lowest scores have more diagonally stretched ovals. One of the lowest scoring nines in the second component looks atypical due to the gap/disconnect in the oval.

**Combined Principal Component Analysis**

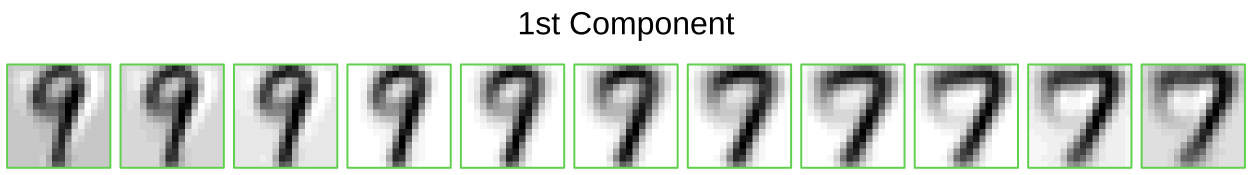
We lastly a perform an analysis as above to the training data of both digits. The scree plot below shows an elbow taking place at the fourth component, so we deem adequate to take the first four principal compoenents to explain variations in the data.

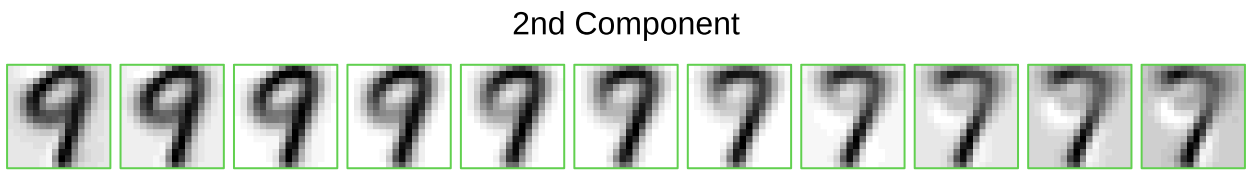


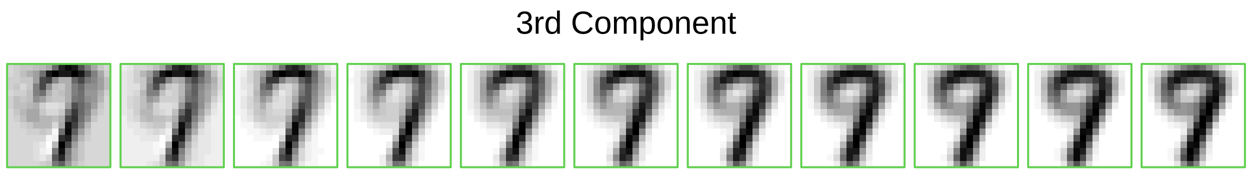
As before, we use the formula

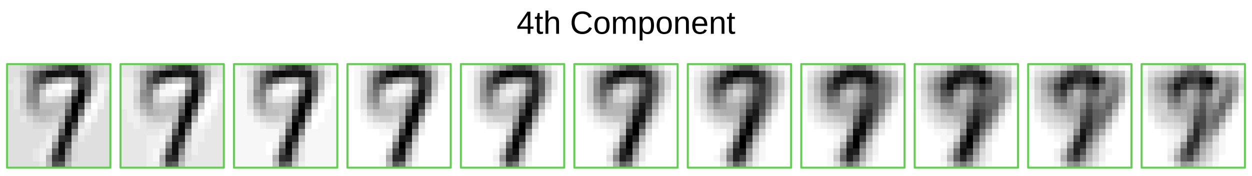
$$\overline{X} + c_i \cdot E_i$$

where $\overline{X}$ is the mean training image and $c_i$ ranges though 11 equally spaced values on the interval $[-2s_i, 2s_i]$ (this includes the endpoints and the value 0). The resulting 11 images for each component are shown below:



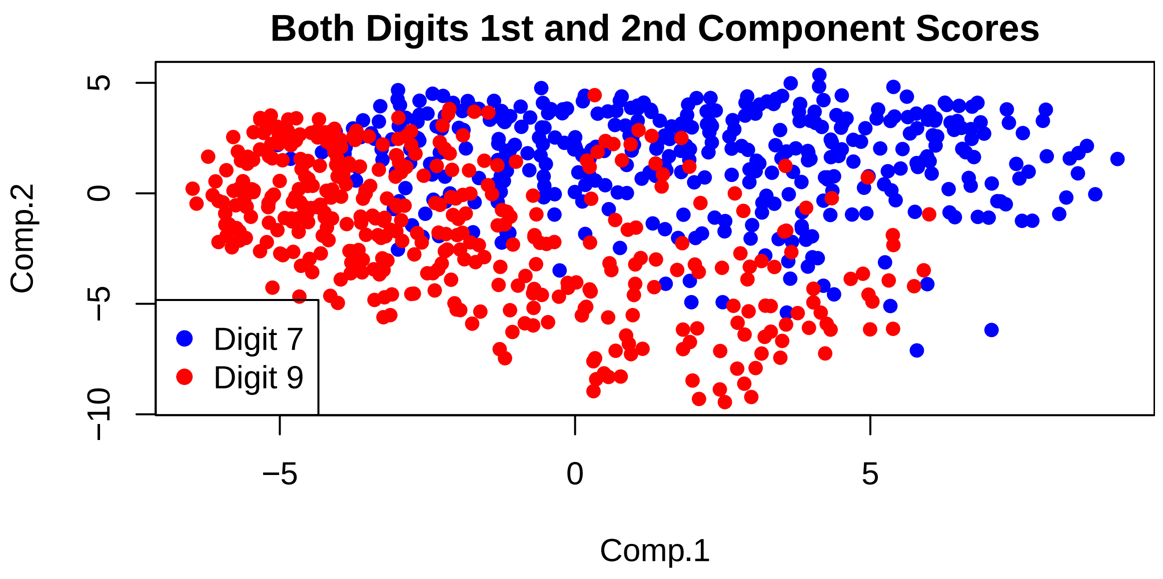






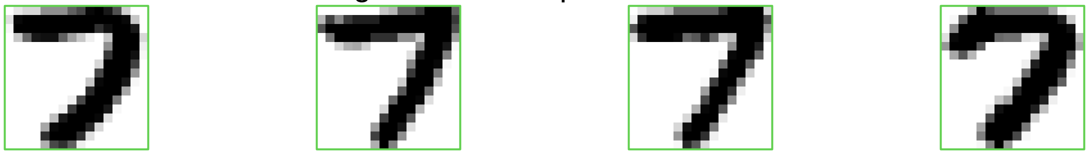
The first component is higher for wider sevens and lower for circular-holed nines. The second component is higher for thinner top-curved sevens and lower for diagonally-oval nines. The third component is lower for sevens that look like 1 on top and higher for thicker nines. The fourth component highlights large digits with higher values.

We now take a look at the labeled score plot for the first two components.

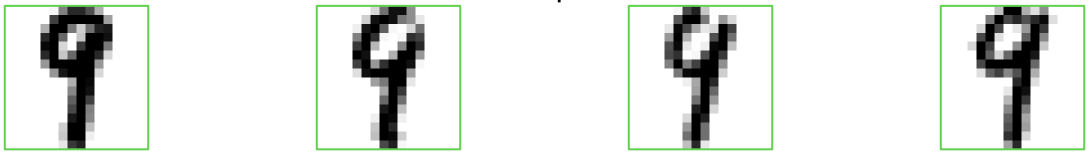


We observe that most 7 digits have higher first and second component scores than most 9 digits. Also, the grouped scores seem to be somewhat linearly separated hinting that linear discriminant analysis is adequate. Similar observations hold by looking at the following extreme score pictures.

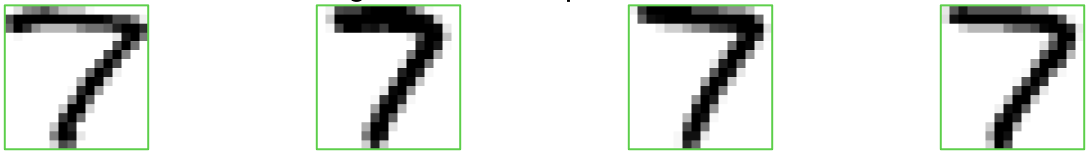
Highest 1st Component Scores



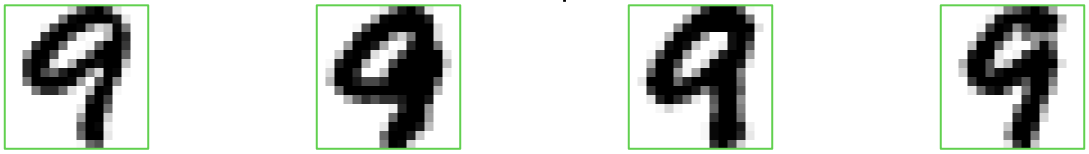
Lowest 1st Component Scores



Highest 2nd Component Scores



Lowest 2nd Component Scores

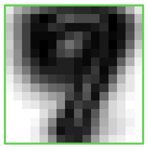


## Classification

### Fisher's Linear Discriminant Analysis through Pixels

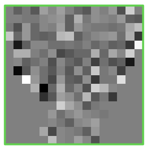
We seek to train a classification model on the data. Our first approach is to perform Fisher's linear discriminant analysis (LDA) on the combined training data, using all the pixels (seen as variables) with sufficient variation. The following is the image whose pixel intensities are given by their standard deviation in the data

Standard Deviation Intensity Image



Using the first quantile of 0.1676, we deem variables with sufficient variation to be precisely the ones with standard deviation above 0.2. By sub-setting such pixels, we perform LDA on the modified labeled training data. The following is an image of the coefficients of the linear discriminant.

Linear Discriminant Coefficient Image



The error rates for the training data, testing data and leave-one-out cross-validation are respectively given in the table below.

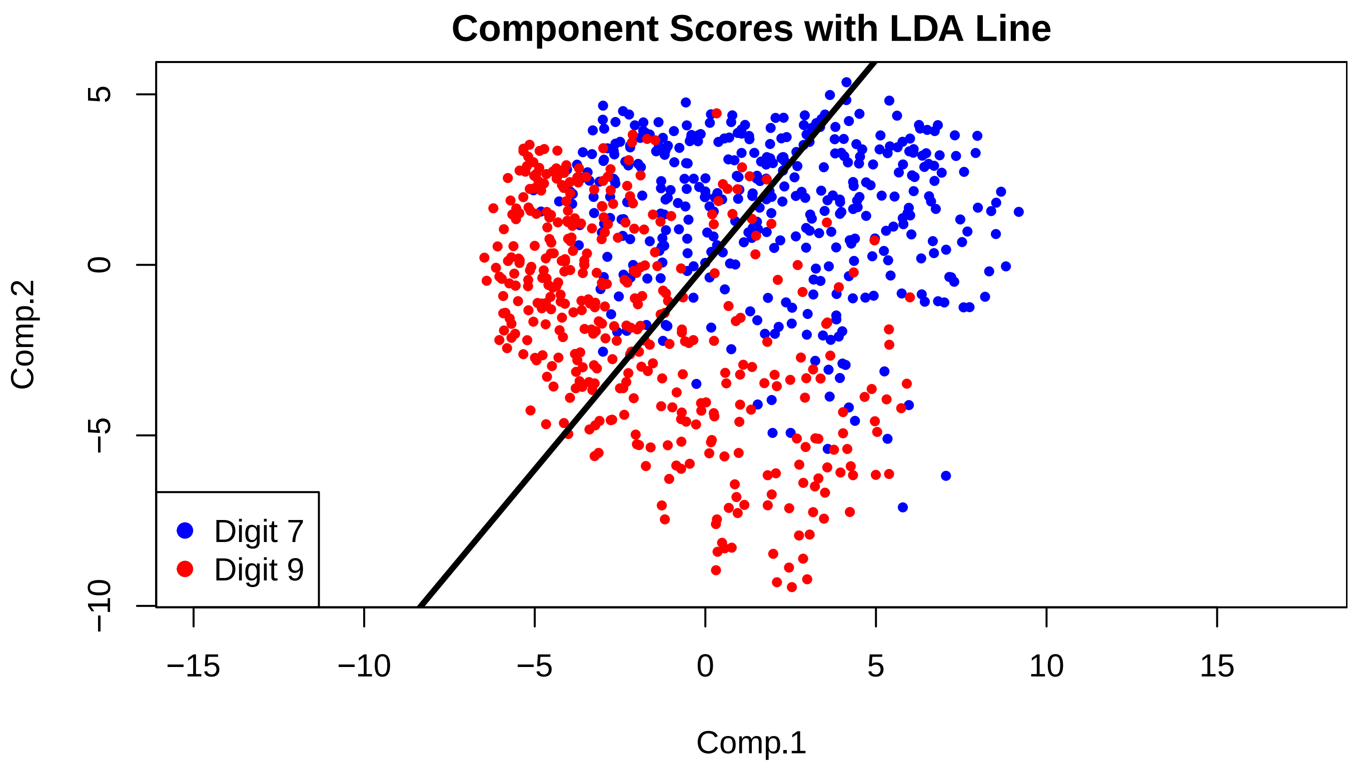
Table 1: Error Rates for LDA with Pixels

| | Apparent | Test | Leave.one.out |
|---|---|---|---|
| Error Rate | 0.00875 | 0.0408998 | 0.0275 |

The model performs very well in terms of the apparent error rate hitting above %99 accuracy. It hits almost %96 accuracy on the testing data and around %97.25 accuracy on the leave-one-out cross-validation method. All in all, we observe that the model has performed well.

**Fisher's Linear Discriminant Analysis through Principal Components**

Instead of looking at pixels as variables for discrimination as above, we simplify matters and focus only on the first two principal components as variables for LDA. We have already seen in the scores plot above that there a chance LDA would work relatively in this simpler setting. The resulting linear discriminant is shown in the plot below.



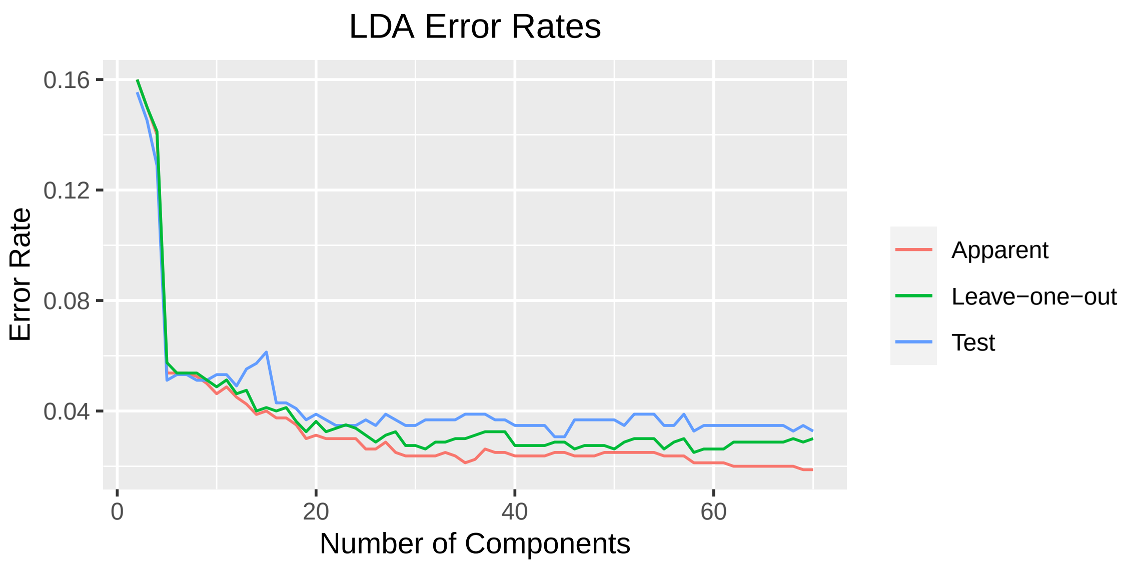
Due to the red dots overlapping in the blue region, we should expect LDA to perform worse here. The following table shows the error rates for the model:

Table 2: Error Rates for LDA with Principal Components

| | Apparent | Test | Leave.one.out |
|---|---|---|---|
| Error Rate | 0.16 | 0.1554192 | 0.16 |

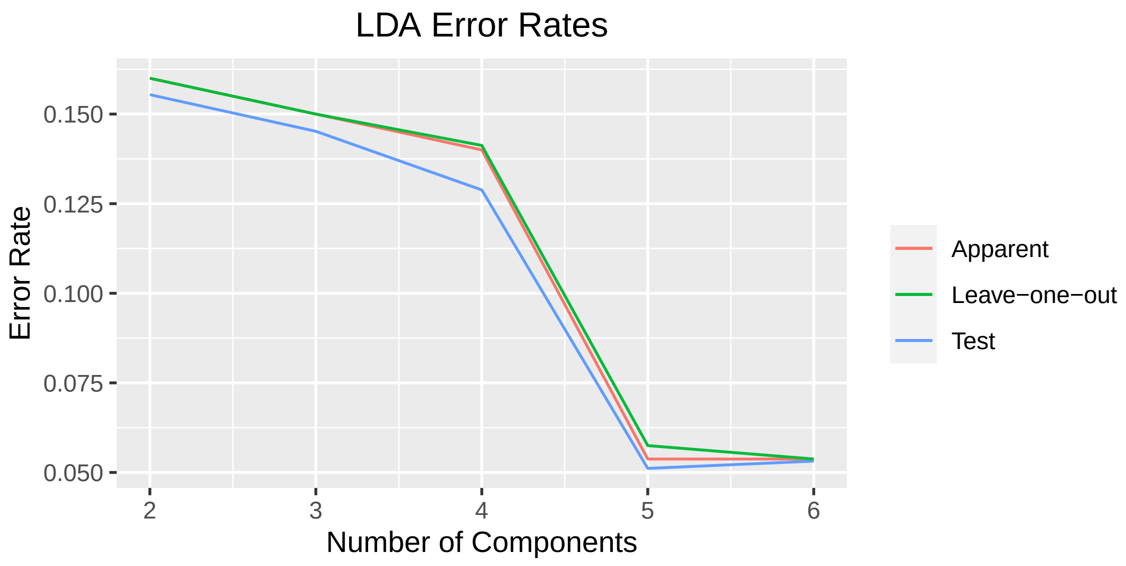
All error rates are not ideal with almost all accuracies being around %84. Indeed, this is expected to be worse than before since we have reduced the number of variables from 182 to 2!!! On the other hand, this shows the power of PCA in choosing two variables that matter the most in contrasting the data.

We now consider the effect of including more principal components in the LDA analysis. The plot below shows the three error rates plotted against how many components are considered in the analysis:



We observe that the testing error stabilizes around 0.04, the leave-one-out error stabilizes around 0.03 and the apparent error slowly decreases and goes below 0.02 with 70 principal components considered. Eventually, including too many components does not change the error rates by much. However, the plot does show that taking only two components may not be good enough.

The plot below shows the scree taking place at five components. Hence, if LDA is to be performed, we recommend performing it on the first five PCA components to achieve decent error rates and keep the model simple.



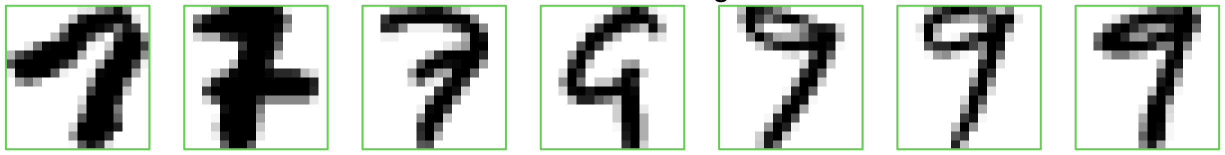
## Discussion

We have remarked how Principal Component Analysis is a powerful tool in picking out the most distinguishing features in the data. In our context of two-way classification, this was apparent in all three PCA analyses conducted on each of digit 7 training data, digit 9 training data and both digits' training data respectively.

We have also remarked that PCA is powerful in reducing model complexity. In our context, the performance of LDA when a sizable portion of pixels (182) are considered as variables is very comparable to the performance of LDA when restricted to only five principal components (both reaching accuracies above %96 in all three measures). This is a huge reduction in dimensionality. In fact, even when only two components were considered, LDA performed relatively well reaching accuracies of around %84 in all three measures!

Now, let us address the limitations in our analysis. First, we take a look at the training data that LDA with sub-setted pixel variables failed to classify:

LDA with Pixels Training Failure



Three of them are nines with too thin of hole that it could look like a seven. Two of the failed sevens had an extra line through the middle. The other nine had a gap in its the drawing. This shows that our model does not perform well in extreme cases of thin nines and extreme cases of sevens with a line through.

One solution to improve on our model is to use the idea of Boosting. This is a sequence of models such that the data of each is generated as a bootstrap which over-samples data points which failed to be classified in the previous model. By doing so, we might be able to improve performance on all of our models above.

Another limitation is the fact that the problem we are considering is a comparison between seven and nine, two digits that have a different topology and rarely look similar. A harder situation would be distinguing between the digits 1 and 7 or even all the digits 1 through 9 combined. A better solution that is usually taken in such kind of circumstance are Convolutional Neural Networks. This is outside the scope of this report and we leave it for future work.