Kevin Shao & Jeffrey Cheng
Mr. Piper
RML B & F
23 May 2018

Proposal for Dimension Reduction

1 Methodology

We had two ideas for dimension reduction of our variables. One was averaging the pixel intensities of the 28x28 image to create a 7x7 with less pixels. Another was simply taking out the pixels that were completely correlated with one another (all intensities were zero for that one pixel for each image in training set). These two methods illustrated above are the "visual" was of thinking about dimensional reduction. Both are a type of PCA, where we essentially take linear combinations of the pixel intensities. However, we also plan to run PCA using R and determine the "mathematical" best way to reduce our dimensions.

2 Pixel Intensity Averaging

Pixel subsampling, in practice, becomes challenging when considering the structure of our data. Namely, our train and test data comes in large matrices, where the i-th observation (i-th row) consists of 785 observations, where the latter 784 are the pixels, we have little information regarding the position of pixels.

Our goal was to subsample from a 28x28 pixel image to a 14x14 pixel image by averaging. That is, for each 2x2 pixel group, we would take the four pixel intensities and take their mean. Intuitively, this would result in a similar, albeit lower quality, image.

```
pixel_positions = matrix(nrow=196,ncol=4)
for (i in 1:14) {
  for (j in 1:14) {
    pixel_positions[i+14*j-14,1] = (i-1)*2 + (j-1)*56 + 1
    pixel_positions[i+14*j-14,2] = (i-1)*2 + (j-1)*56 + 2
    pixel_positions[i+14*j-14,3] = (i-1)*2 + (j-1)*56 + 29
    pixel_positions[i+14*j-14,4] = (i-1)*2 + (j-1)*56 + 30
  }
}
```

The approach we took was first to create a new matrix, called "pixel positions", consisting of 196 rows (since there will be 196 pixels in the new image) and 4 columns (since each pixel in the new image corresponds to 4 pixels in the old image. Essentially, the i-th row consisted of the pixels from the old image to average to obtain the i-th pixel in the new image.

```
test_sub = matrix(nrow = 9999, ncol = 196)

for (i in 1:196) {
  test_sub[,i] = test[,c(pixel_positions[i]) + 1 ]
}

train_sub = matrix(nrow = 59999, ncol = 196)

for (i in 1:196) {
  train_sub[,i] = train[,c(pixel_positions[i]) + 1]
}
```

We then created matrices to hold the subsampled train and test data sets, called "train_sub" and "test_sub", respectively. For each of the 196 **columns** (hence "test_sub[,i]"), we wanted the average of the columns (of test or train) that were given by the i-th entry in our list. The increment at the end of the index serves to eliminate the leading column of the original data, which contains the true value of the image.

3 Removing Collinear Pixels

4 Mathematical Dimensional Reduction

5 Conclusion