Technical Report: Digit Recognition

Group 6: Yilin Li, Hien Nguyen, Lyn Peterson

12/6/2019

Your technical report should be an .Rmd file that contains the following sections. So as not to make the compilation (knitting) of the document not take too long, consider setting cache = TRUE in the curly braces of any R chunk with substantial computing. Please knit both to pdf and github document (.md).

Abstract

A brief overview of the area that you'll be investigating, the research question(s) of interest, your approach to analysis, and the general conclusions.

Overview:

Research question: Can we build classifiers to recognize what the digit (0-9) is in a given image based on 60,000 training images?

Approach to analysis:

General conclusions:

Introduction

Overview of the setting of the data, existing theories/models (particularly if you are working in a descriptive/inferential setting), and your research questions.

The Data

Where does the data come from? How many observations? How many variables? What does each observation refer to (what is the observational unit)? What sorts of data processing was necessary to get the data in shape for analysis?

The data comes from the MNIST database of handwritten digits. The digits have been size-normalized and centered in a fixed-size image.

The data is split into a training set of 60k images and a test set of 10k images.

Data processing: Since the pre-downloaded data is already well-formatted, minimal effort was spent on data processing. We performed two additional steps of data processing:

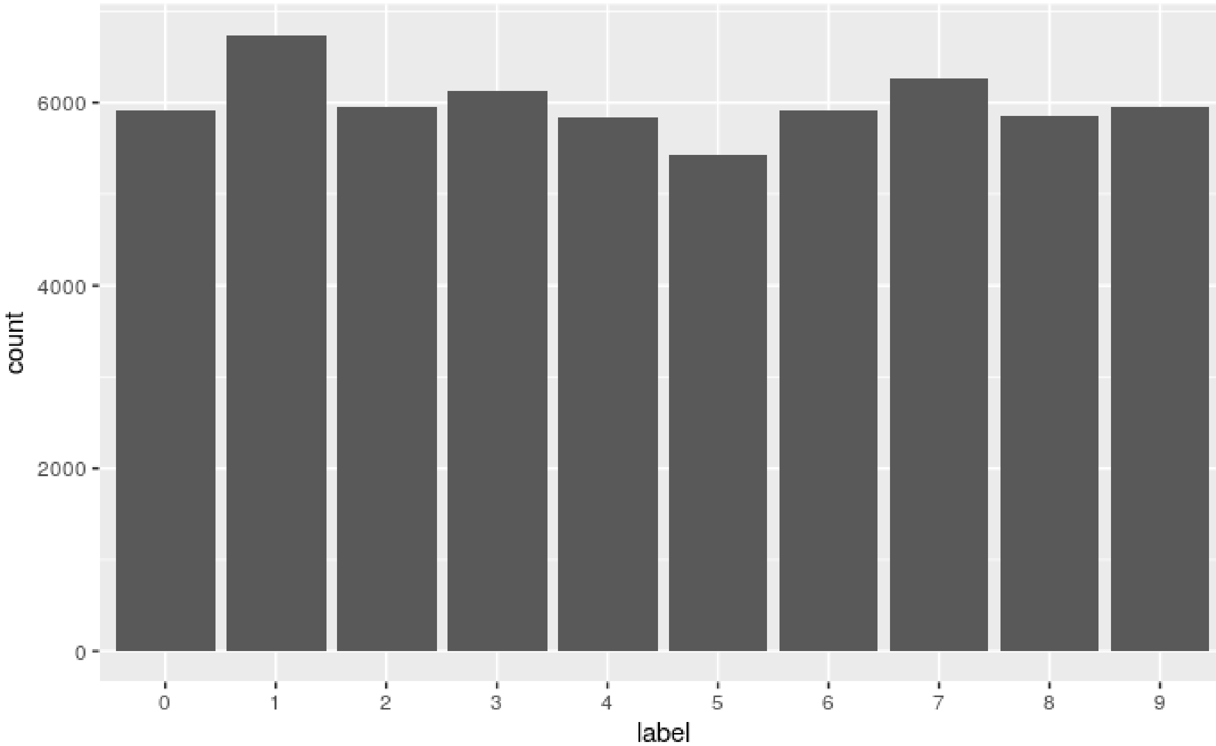
(1) Each 28x28 image was flattened into a single row of 784 pixels.

(2) All pixels were rescaled from 0-255 to 0-1.

Exploratory Data Analysis

Explore the structure of the data through graphics. Here you can utilize both traditional plots as well as methods from unsupervised learning. Understanding the distribution of your response is particular important, but also investigate bivariate and higher-order relationships that you expect to be particular interesting.

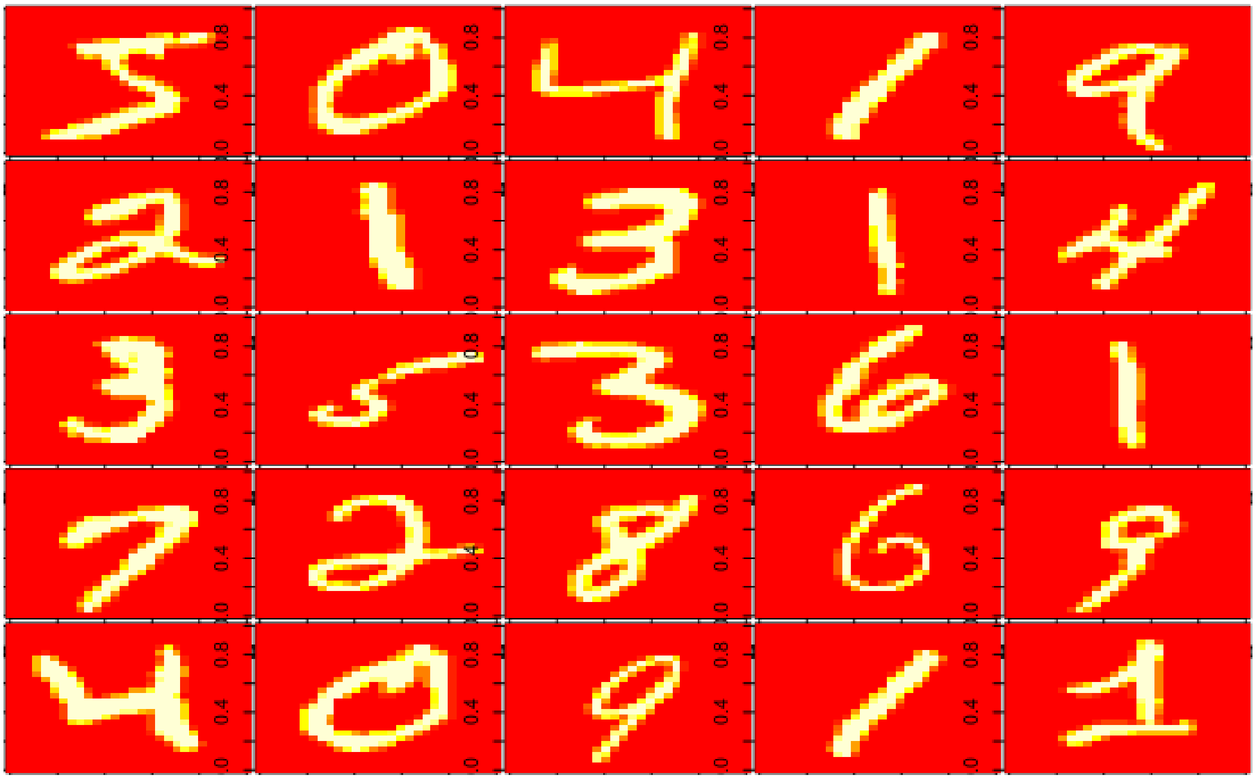
1. The digit classes are well-separated.



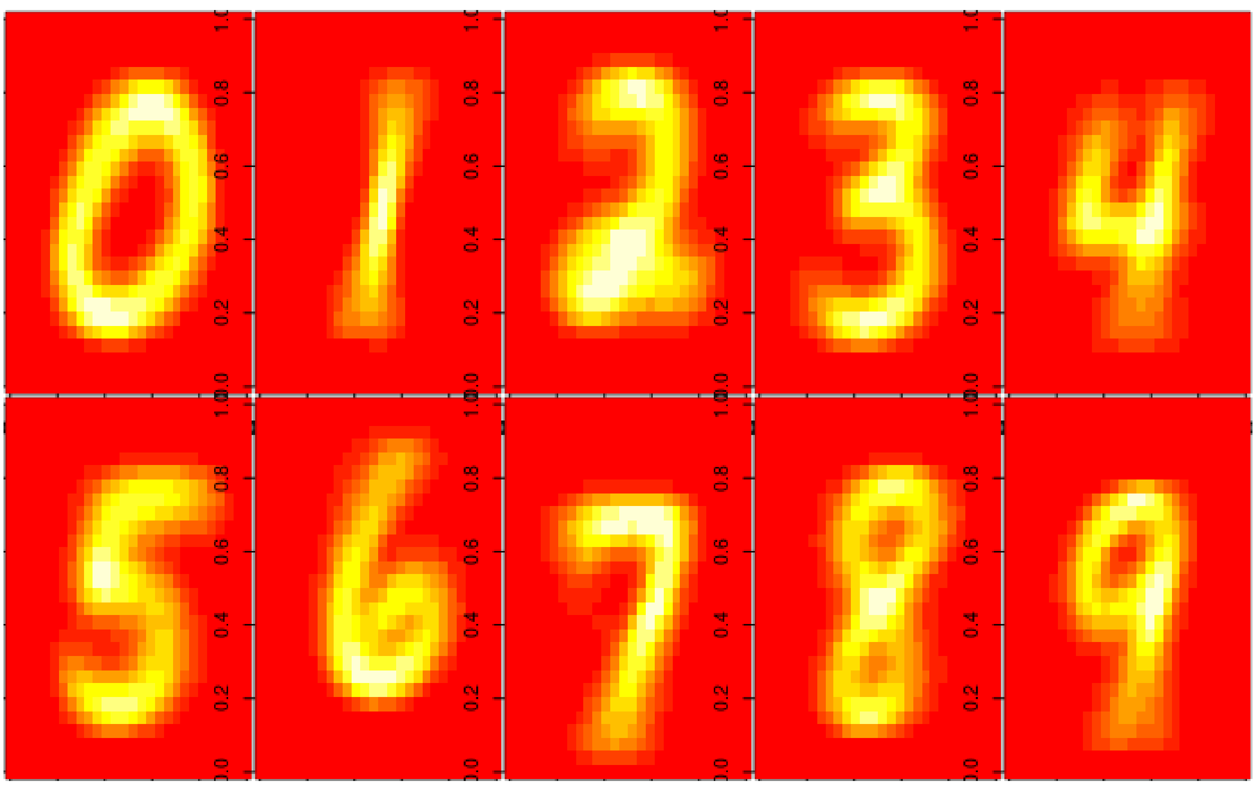
shown by the frequency graph above, about equal amount of images for each digit.

2. Handwriting patterns and variations.

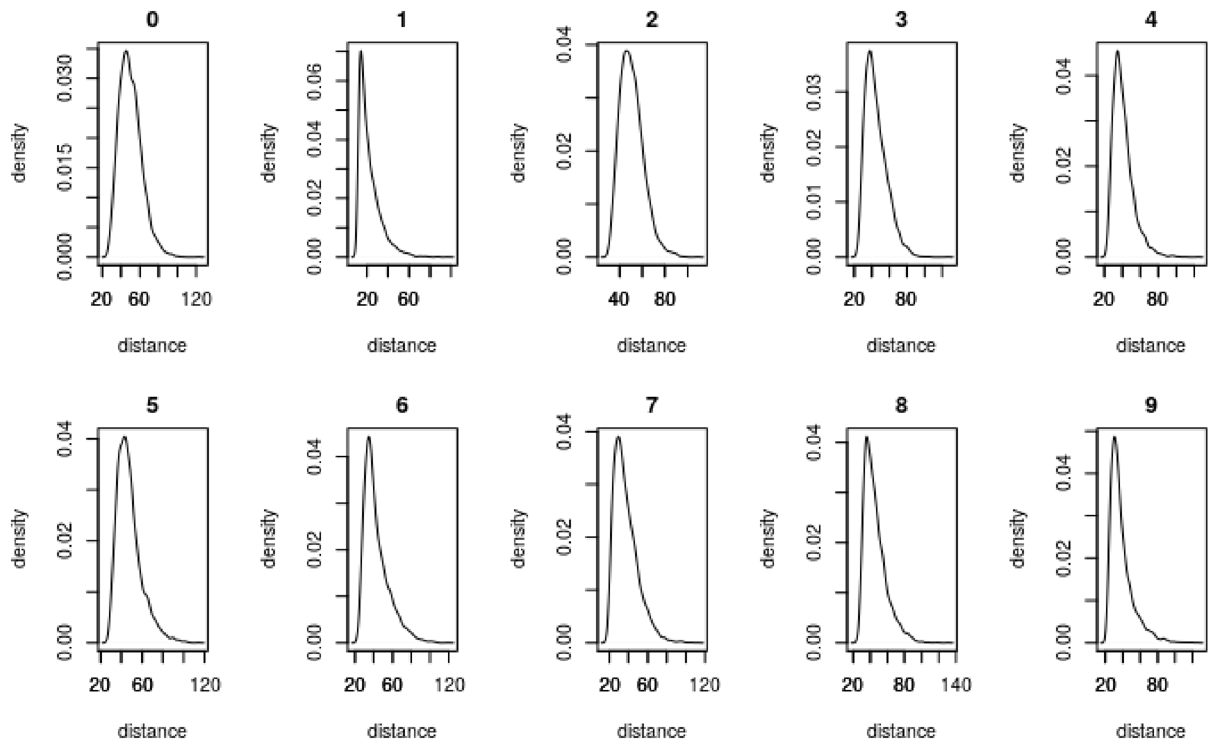
Let's take a look at first 25 images in the training data. It's clear that those digits are centered and manipulated with some anti-aliasing technique. As you might observe, among six 1's below, different styles appear such as italic version and formal version. We can say that the data set has generalization of digit styles.



Then for each digit, we average every pixels and get a mean form for each digit.



We calculate Euclidean distance for each image to its mean form and generate distance(to the mean) distributions for each digit.



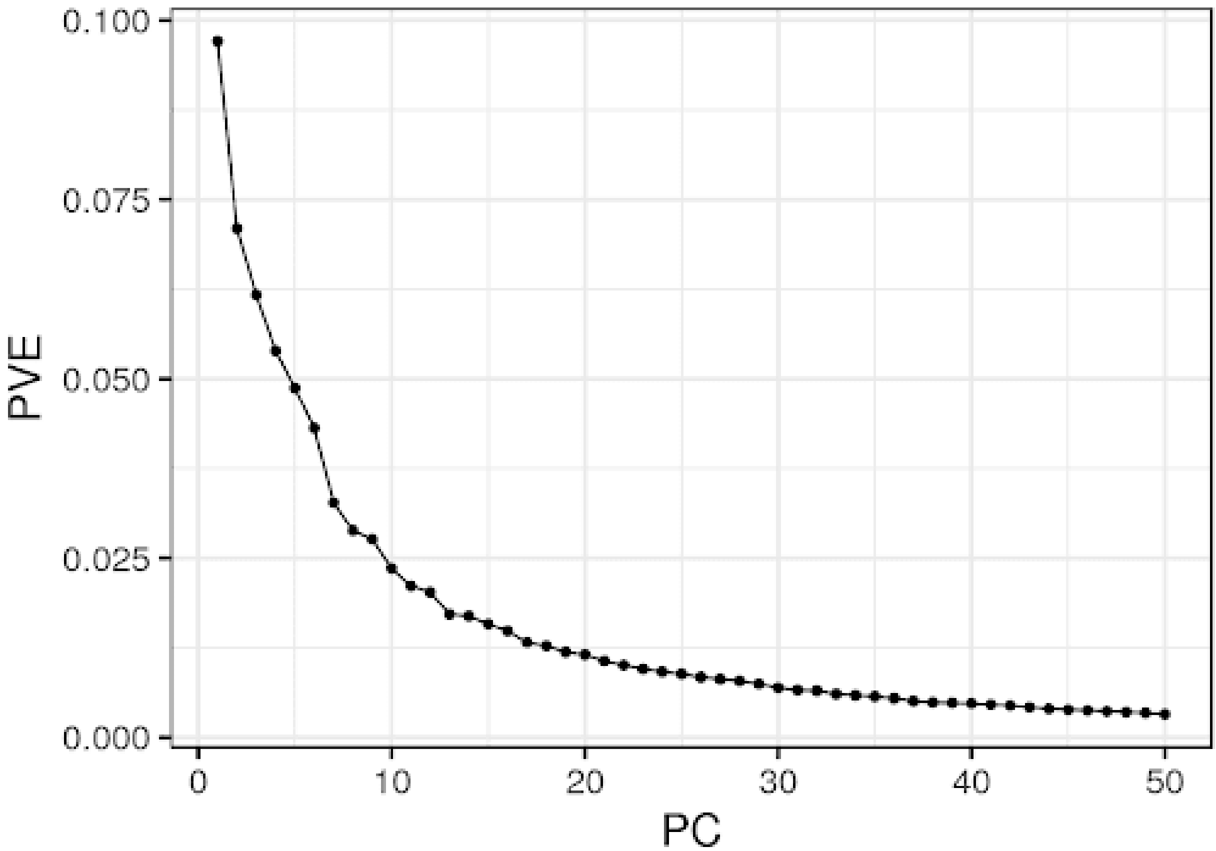

```r
dist_df <- data.frame(x = distance_mean, y = distance_var)
colnames(dist_df) <- c("mean", "var")
rownames(dist_df) <- label
print(dist_df)
```

```
##       mean      var
## 0 49.91889 140.5000
## 1 22.47562 107.3414
## 2 50.76020 107.3001
## 3 44.94435 150.5974
## 4 40.87232 130.9288
## 5 47.58101 142.0700
## 6 43.09167 179.6460
## 7 37.47072 157.1436
## 8 45.59575 165.0296
## 9 38.56896 179.6759
```

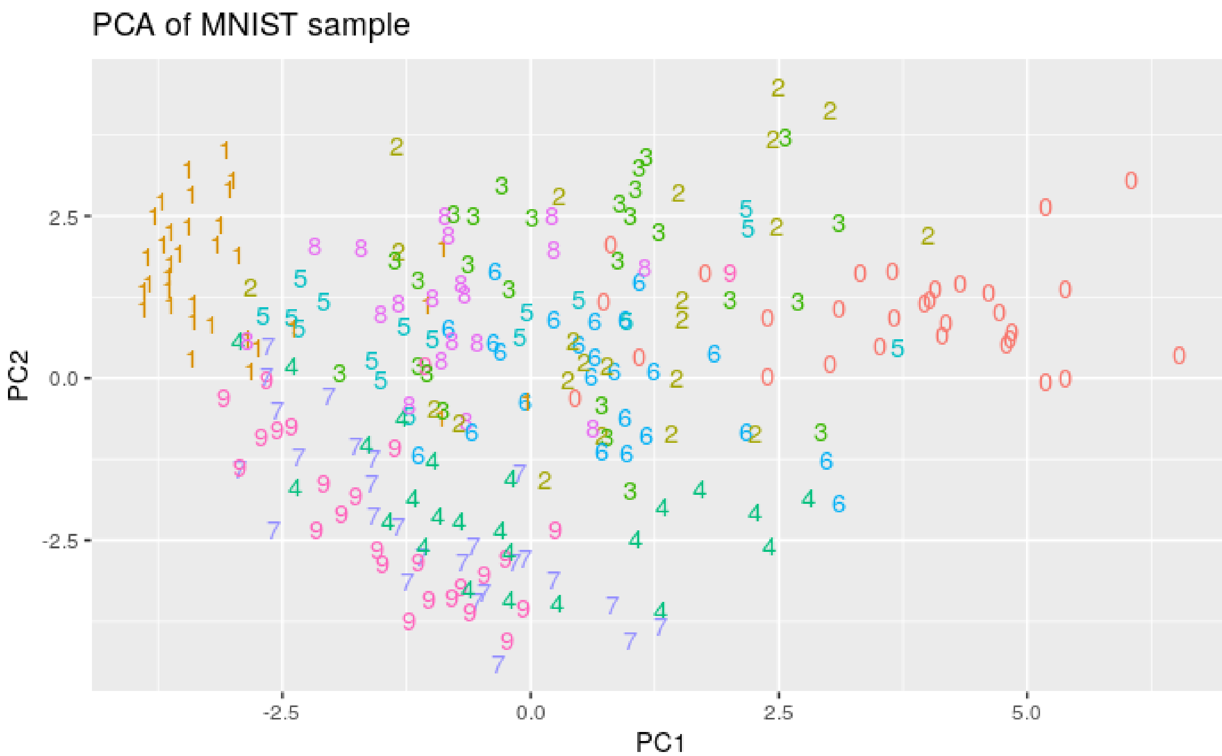
From the summary table, we can see that digit 1 has the lowest mean distance; this indicates that most people write it similarly. 2 has the highest mean distance, so people tend to write 2 in different ways. We also see that 6 and 9 have the highest distance variance, which suggests these two digits have the most variation in people's writing styles.

3. PCA & early predictions.

We use principal component analysis on the original data set and select first 20 PCs to capture 90% of the variance, see the scree plot below.



Let's visualize our digits with the first 2 PCs.



We can observe from the image that 1-0 pair is well seperated, 4-9 pair is overlapping. Based on the distribution of digits, we think PC1 represents centerization of the digit, for example, 0 has more pixels on the outer area of 28 by 28 box. 1 has more pixels in the middle which can be illustrated from the mean images above. Thus, generally 0s has larger PC1 than 1s. We think PC2 is related to horizontal symmetry, for example, 3 and 1 are symmetric horizontally, which have larger PC2. 9 and 7 are not horizontally symmetric, which have lower PC2.

Based on the PC graph above, we predict that 1-0 pair will be easily recognized and 4-9 pair might be the hardest one to seperate.

Modeling

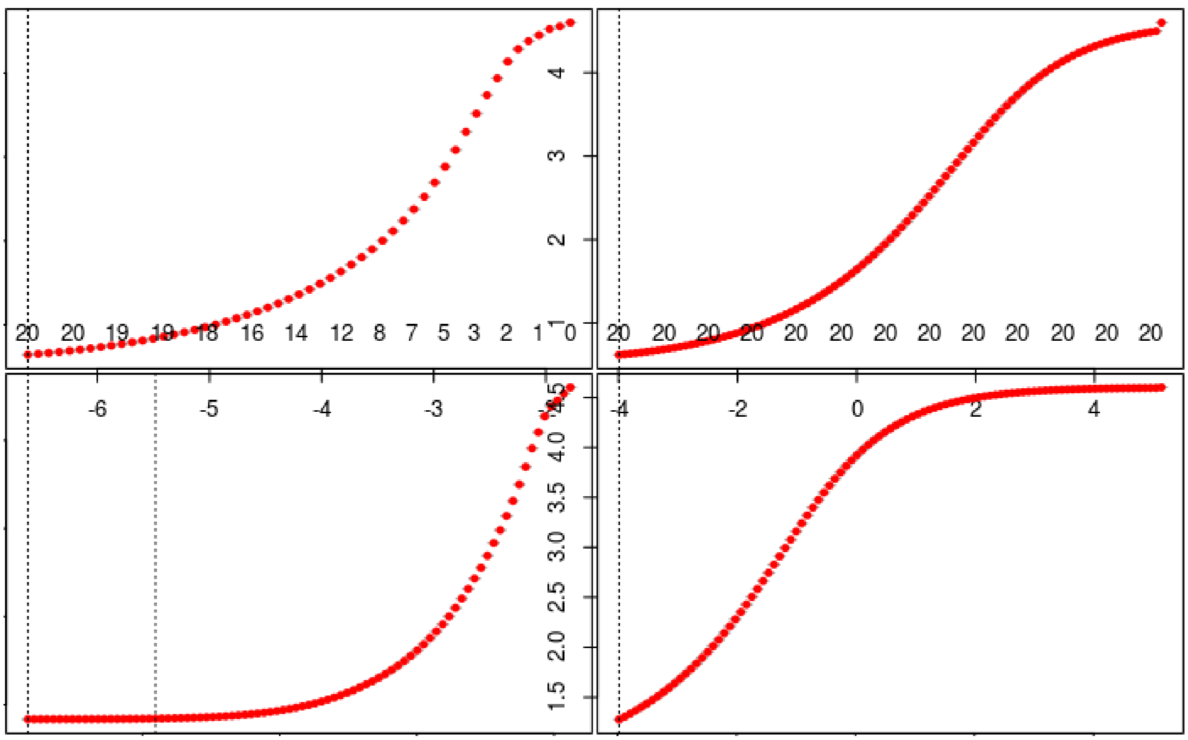
Construct (descriptive and/or predictive) (classification and/or regression) models that address your research questions. You are encouraged to fit many different classes of models and see how they compare in terms the bias/variance tradeoff (do you have a Rashomon effect going on?). Also be sure to guard against overfitting through cross-validation or shrinkage/penalization (don't forget about ridge regression and the lasso).

This will be the most extensive section and will include your results as well.

- **Classification Tree (with pruning)** Our first pruned classification tree had limited success. Even without pruning, the optimal number of nodes was selected (13), and the graph of error against size showed no clear elbow indicating optimal size. The first tree could not classify the class 5 with only 13 splits, so we created a similar tree using another package that successfully modeled all classes. The important pixels (at each split) seem to be near the center of the image (around 400). The misclassification rate is near 40%, as this is a rather weak model with a limited numer of splits.
- **Random Forest (with bagging)** The plot of error against n trees shows that the pixels had variable importance in predicting the random forest. This plot also shows that as long as there are 50-100 trees, error is rather low. The mean decrease accuracy chart shows that the random forest identified 7-13

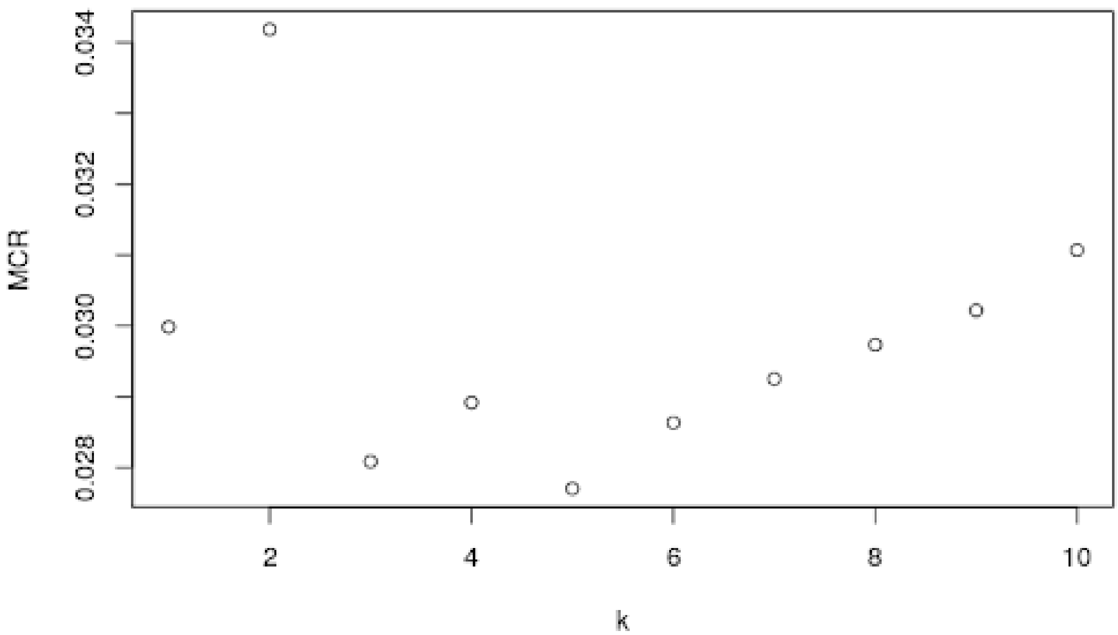
most imortant pixels, as did the simple tree. The bagged random forest (m = p) also identified a similar number of important pixels that seem to be near the center of the image. The error rate was the lowest when m=1 and the highest when m=p showing that there may be overfitting in the random forest, as it fairs better when less predictors are considered at each split.

- **Boosted Tree** The boosted model also identified a similar number of important pixels near the center of the image. The error rate for this model, 8%, is a little higher than the bagged random forest.
- **Logistic Regression for Multinomial Distribution (with Ridge and Lasso)** At the first place, we use 10 one-to-all logistic classifiers with pca data set. We find that the coding for it is too confused and not clean, so we use instead *multinom* function which can do muti-class logistic regression with multinomial distribution. We find the misclassification rates of both methods are about same. Then we use *ridge* and *lasso* regularization to prevent overfitting. It turns out that we don't encounter any overfitting problem shown by the graphs below.



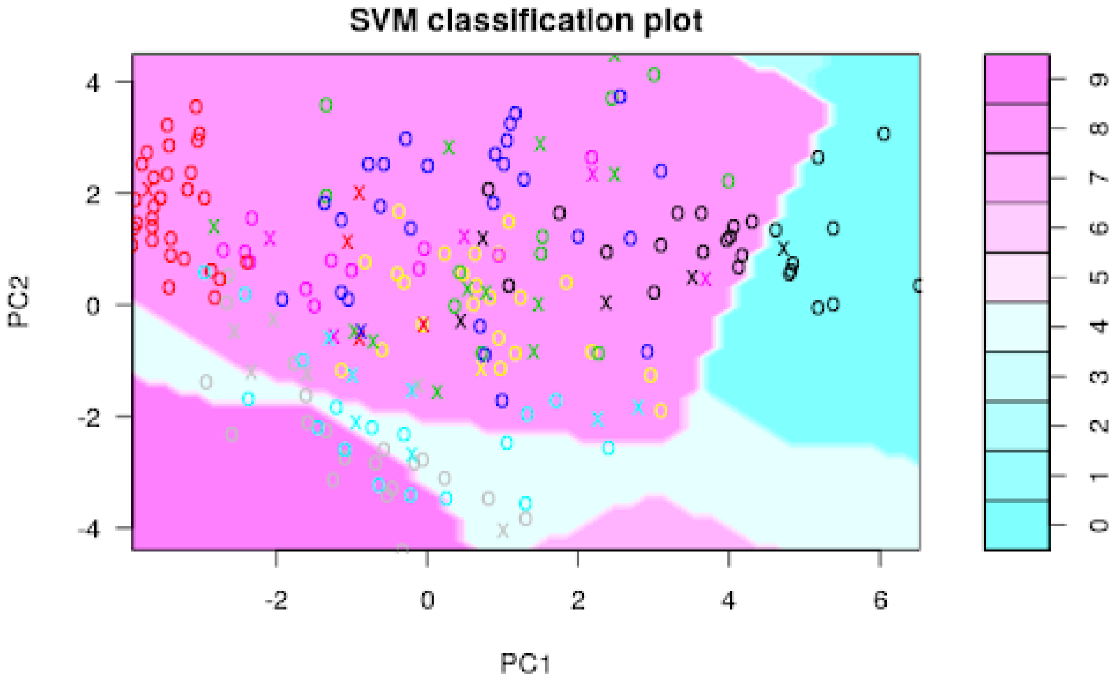
In the graph above, x-axis is lambda values, y- axis is multinomial deviances. Clearly, as lambda increases, the model performs worse, and lambda equals 0 will generate the best classifier. Since the overfitting problem is not obvious, we have ridge and lasso resulting similarly to the original multinomial regression. Our best result for original images is 7.89% error rate and 11.95% error rate for pca data set.

- **KNN** We use *knn* function to predict k nearest neighbors result. We apply leave-one-out cross validation on the pca data set, and have $k = 5$ resulting best.



Because it takes too long to do cross validation on the original data set. We also apply $k = 5$ to the original data set. As, a non-parametric model, k nearest neighbors generate the second best result among all of our models. The misclassification rate of the original data set is 3.06% and the misclassification rate of the pca data set is 3.02%.

- **SVM** We apply one of the most complicated model, support vector machine, to the pca data set. Generally, SVM is finding a seperating hyperpplane that best seperate 2 different classes. With its unique different kernel tricks, it can map raw predictors to a higher dimensions with arbitary complexity. Then the model does linear seperating in that higher dimension, which will generate better results. We can consider this as an automated way of creating polynomial terms or interaction terms in logistic regression. We choose the "radial" kernel and get **2.17%** misclassification rate, which is an awesome result. Here's the region plot for the model with only the first 2 PCs.



The built in plot method doesn't give us a clear plot because we have 10 classes and 20 predictors. But it still sparks us with some seperating region with only 2 predictors, PC1 and PC2. Unfortunately, when we apply support vector machine on the original images, it takes too long to train the model, so we give up this model finally. But, we believe that with a correct kernel and tuned parameters, it will generate a better result than 2.17%.

All models are cross-valiated.

Discussion

Review the results generated above and sythensize them in the context from which the data originated. What do the results tell your about your original research question? Are there any weaknesses that you see in your analysis? What additional questions would you explore next?

- 4-9 is the most difficult pair to predict across models
- No overfitting with PCA, but some with raw data
- Best model
- Compare with published models

References

At minimum, this will contain the full citation for your data set. If you reference existing analyses, they should be cited here as well.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998. Online Version.