Introduction:

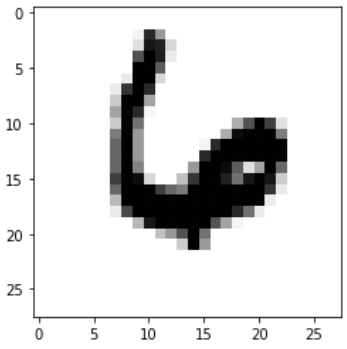
Use random forest for multiclass classification to build a model to predict the value in the MINST data set. Along with accurately predicting the digit that is written, the model should run efficiently on a larger data set.

The steps involved in the following analysis are

1. Fit a random forest classifier using the train data set and predict the values for test dataset
2. Run a principal component analysis (PCA) to reduce the data set
3. Fit a random forest classifier with reduced data set using the train data set and predict the values for test dataset
4. Compare the runtimes, model (F1, Precision and Recall) scores.

Exploratory Data Analysis:

There were two data sets provided. The first data set is train, that is used to create the model and the second data set is to test the model and predict the digit. The train data has 42000 rows of data and 785 variables. The test data set has 28000 rows and 784 variables. The 784 variables represent an image of 28x28 with each pixel value associated, indicating the darkness of that pixel. The pixel value is an integer between 0 and 255. The image on the right is a binary plot of showing a row of data that has a response value of 6.



Data Preparation and Overview of Programming:

We use SciKit learn to create the models. The first step to get the data ready to create the model, is to drop the response variable from the train data set and convert both train and test data set into an array.

Random Forest using all variables

The train data set is split into train and test using train_test_split function. The random forest classifier using all variables on the split train set takes about 10sec to build and test on a data set of size 25200 rows, with a F1/Precision score (train/test) of 92/91.

PCA

The next step is to run principal component analysis on the combined train and test dataset. It takes about 6 seconds to generate the principal components that represent 96 percent of the variability. The number of principal components is reduced to 154 explanatory variables.

Random Forest Classifier with PCA

The train and test data are transformed using the principal components generated. The transform takes about 3 seconds. The train data set is split into train and test using train_test_split function with the number of features to be used as the number of principal components generated. The random forest classifier using all principal components on the split train set takes about 2 min to build and test on a data set of size 25200 rows, with a F1/Precision score (train/test) of 93/92.

Insights & Conclusions:

Random forest classifier with principal components model performs better on both train and test data set with a high F1/Precision score. Based on the analysis above, as random forest classifier with principal components takes longer to run than the random forest classifier with all the variables, I recommend that random classifier without principal components be used.

Appendix:

The ipynb notebook and an html version of the notebook along with the output and Kaggle submission scores are included in the submission.