Assignment 5: Principal Component Analysis by Mimi Trinh

Section 1: Summary and Problem Definition

The purpose of this project is to develop a predictive model that can detect written numbers in the MNIST dataset with 70,000 observations. Specifically, the goal is to build a multi-class classifier to be able to determine all ten digits in contrast to previous examples of binary classifier that can only detect yes/no outcome of a single number. The entire MNIST dataset is used in this project. This is a computer vision problem (classification of images analogous to the MNIST dataset) in which the predictive accuracy of models must be weighed against the costs of model development and implementation.

Section 2: Research Design, Measurement, and Statistical Methods

In this project, we will develop four models below. To compare models, we use two metrics 1) the time it takes to fit the model 2) the evaluation of the model on the hold out test set using the F1 score, which is the harmonic mean of precision and recall.

1. Fit a random forest classifier utilizing the full set of 784 explanatory variables using a train-split test with the first 60,000 observations as the train set and the last 10,000 observations as the hold out test set.
2. Execute principal component analysis (PCA) on the full set of 70,000 observations to get 95% variability.
3. Rerun the random forest classifier with train-split-test using the principal components identified from the second model.
4. Rerun the random forest classifier with train-split test using the principal components and standard scaler. The flaw in the original design is the lack of scaled data.

Section 3: Programming Work

Scikit-Learn is the main environment for this project. We first load the MNIST dataset into Python and do the train-test split. Then we use RandomForestClassifier() to build model #1 and PCA() to build model #2. Combing codes from the first two models, we develop model #3. Finally, we use StandardScaler() and codes from the previous models to build model #4.

Section 4: Results and Recommendations

After loading the MNIST dataset into Python, we conduct a quick exploratory data analysis (EDA) and visualization by showing the 36,000th image, which is a number 5 (exhibit). The goal is to build a model that can identify ten written digits for all 70,000 observations. For each model, we repeat the process 10 times to calculate the time it takes to fit the model in millisecond. The mean of 10 results is one of the two evaluation metrics of the models. The second metric is the F1 score; the higher the score, the better the model. Below are the results.

	Model #1	Model #2	Model #3	Model #4
RF Time	24,862	n/a	52,520	74,139
PCA Time	n/a	17,962	Model #2	18,438
Total Time	24,862	17,962	70,482	92,577
F1 Score	0.77001	n/a	0.76703	0.80254

Model #2 narrows down 784 variables into 154 dimensions whereas model #4 narrows it down to 332 dimensions, which explains the extra time it takes in model #4. Using the time metrics, model #4 is the worst since it takes the most time to fit both PCA and random forest classifier. However, using the F1 metrics, model #4 is the best since it has the highest F1 score. Comparing to model #1, model #3 performs worse whereas model #4 performs 4% better. Time is the cost of model development in this project in which mode #4 is the costliest, and model #1 is the fastest (excluding model #2). Therefore, we recommend against applying PCA and random forest on raw data (model #3). However, it’s recommended to use PCA as a preliminary to random forest classifier on scaled data (model #4). Although it takes longer, the model performs better.

Exhibit