Week 6: Model Speed Using MSINT Handwritten Images

Background and Motivation

This analysis was performed on the MNIST dataset, made available via the "Digit Recognizer - Learn Computer Vision Fundamentals with the Famous MNIST Data" Kaggle competition (Kaggle 2022). The dataset contains 60,000 handwritten numbers, represented as a 28x28 pixel image with each pixel's level of shading captured. The "training" dataset is enriched with the number captured in the image. Successful work with this dataset allows for the application of computer vision to automate scanning of handwritten numbers for, e.g., postal mail or bank check details..

Approaches to Pre-Processing the Data

The original dataset contained 784 pixels, scored from 0-255 representing the depth of shading. For simplicity, these pixels were MinMax transformed, ranging from 0-1.

Other than this basic pre-processing, all other treatments followed explicitly from the assignment instructions. They included::

- Step 1: Use PCA to reduce the dimensionality of data, using a sufficient number of components to capture 95% of dataset variability. The first step we were instructed to execute the PCA using the full train + test dataset, which is conceptually inaccurate
- Step 2: Repeat the PCA in Step 1, using only the training dataset. As above, use a number of components sufficient to capture 95% of the variability.

Note in both the wrong/full PCA and the correct/train PCA, the number of components necessary to represent 95% of the variability was 154.

Modeling Approach

As noted above, the assignment this week was highly prescriptive regarding steps taken to develop models. The models tested included three Random Forest models, one using the full 784 data

elements, one using the wrong/full dataset PCA reduction, and one using the correct/train dataset PCA reduction; as well as using K-Means clustering. Each model's execution time was captured.

Importantly, the focus of analysis was on time to complete, rather than to adjust the hyperparameters to hit a "best" model. This is most apparent in the application of kmeans clustering, where the instructions make clear to only use 10 clusters, which had a remarkably poor model performance.

Results:

Model	Time to Complete	Test Accuracy
RF - Raw Data	30.99	96.60%
RF - Full PCA (wrong; 154 components)	75.87	94.442%
RF - Train PCA (correct; 154 components)	77.14	94.441%
KMeans (k = 10)	1.08	59.61%

Implications

- Random Forest can provide highly accurate results, with nearly 97% accuracy using raw data and default model hyperparameters
- Principal Component Analysis can allow models to punch above their weight. In both applications of PCA, the number of components was set to capture 95% of the variability, but models had a Test Accuracy that was 98% as good as the model using raw data
- PCA *increased* time to generate the random forest. This is prima facie counter-intuitive, since the number of variables considered drops from 784 to 154. However, PCA reduction results in a more information-dense dataset, so finding breakpoints in each tree in the ensemble is likely more difficult
- K-means is a promising technique, which is hamstringed by the assignment's requirement that k = 10. It is naive to believe that with only 10 clusters it would be possible to create a complete 1-1 match. Generating a higher number of clusters would almost certainly result in substantial model performance improvement.

References

"Digit Recognizer - Learn computer vision fundamentals with the famous MNIST data." Kaggle. Accessed February 7, 2022. https://www.kaggle.com/c/digit-recognizer/overview.