Custom MNIST Dataset Modeling

Hosted on Jupyter Notebooks

Collin Sieffert
Electrical and Computer Engineering Department
University of Florida
Gainesville, United States
csieffert@ufl.edu

***Abstract*—This document contains the dataset preprocessing, analysis, and workup of models designed to classify handwritten characters from input images. The models in use will explore various methods for dimensionality reduction in order to improve/reduce the training time of the classifiers developed.**

***Index Terms*—Logistic Regression, Linear Discriminant Analysis, Recursive Feature Extraction, Support Vector Machine, Principle Component Analysis, Manifold Learning**

I. Introduction

This document will cover data exploration and preprocessing choices, hyper parameter tuning and model evaluation.

II. Data Exploration

A. Dataset Descriptions

Here we'll begin with a quick breakdown of the dataset being used for classification.

Much like the MNIST dataset, this one contains numerous handwritten characters falling into 10 categories. However, unlike MNIST, none of these are numbers, instead they are: A, B, C, D, E, F, G, H, $, # All of these characters were handwritten on pieces of paper and then resized to be 300 by 300 pixel images. The training set contains 6720 total examples of the characters written, with just under 700 examples of each character (after the testing sample was removed). The test set contains another 2880 examples that won't be see during training. See an example from each class.

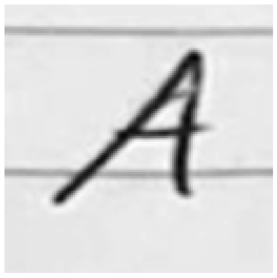
Fig. 1. Class 0, A

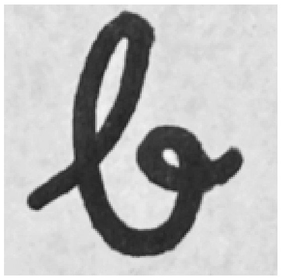
Fig. 2. Class 1, B

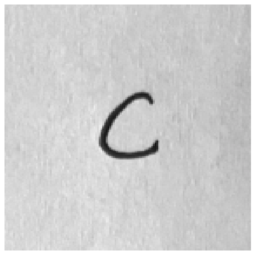
Fig. 3. Class 2, C

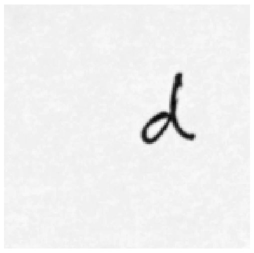
Fig. 4. Class 3, D

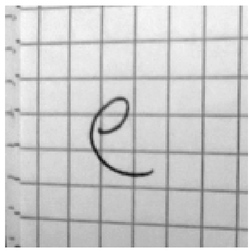
Fig. 5. Class 4, E

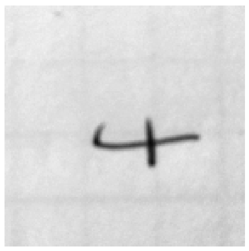
Fig. 6. Class 5, F

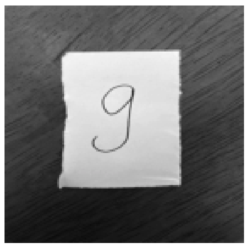
Fig. 7. Class 6, G

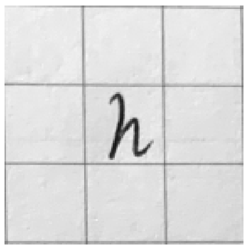
Fig. 8. Class 7, H

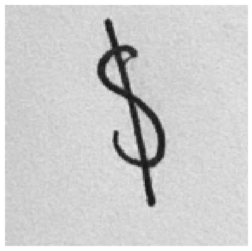
Fig. 9. Class 8, $

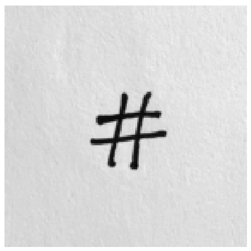
Fig. 10. Class 9, #

Funded by Midnight Oil Co.

B. Preprocessing Selections

As seen in the dataset examples above, the pictures generated are varying in context, some written on lined paper, some written on solid paper, others with textures in the background. When printing out samples from the dataset, there were multiple occurrences of mislabeled images. For example, class 8 contains multiple examples of class 9 and even some examples of class 7. With all 8000 characters it would've been too much work to relabel everything and as such these confounding examples were left in the training and test sets. All of these differences present more challenges to the classification process. On top of the artefacts in the dataset, the images being 300x300 present numerous computational challenges. In order to combat this, all images were down sampled to 28x28 pixels. Whilst this greatly reduces the computational overhead, it also throws out roughly 99% of the image data since it is now about 1% the size of the original. Now that that images have been down sampled to a faster processing size we begin with our classification prep transformations. In this case I employed the MinMaxScaler from sci-kit learn to transform the pixel values to either 0 or 255 to be as sharp of a contrast between the dark ink and the light background. With that bringing up the end of the preprocessing, let's dive into the dimensionality reduction and feature elimination techniques employed to help reduce computational overhead in training classification models.

III. Recursive Feature Elimination

First up we have recursive feature elimination or RFE for short. RFE is a feature selection method which fits a model by removing the weakest feature/features recursively until the desired number of features is reached. This optimal number of features to keep is generally unknown prior to model fitting, and thus cross validation is employed to score and rank various feature subsets on their performance against a held-out validation set. This process in itself is computationally intensive as it scales with the feature space, hence we applied the above pre-processing techniques to reduce the load. In this project both Lasso and Logistic Regression estimators were experimented with as the base estimators for RFE models. After running RFE with the estimators, the masked output shows the central pixels being selected as the most important features. This tracks with intuition as we would expect the background of the image to hold no useful information in classifying the character (generally written in the center of the image). After testing both of the RFE models on the held out test data set, we see that the logistic regression model performed about 28% better than the lasso regularization model.

IV. Principle Component Analysis

One can't pursue feature extraction/dimensionality reduction without discussing principal component analysis, or PCA for short. PCA achieves dimensionality reduction by projecting the data into a new coordinate system through a linear transformation where ideally most of the variation in the data can be described with less dimensions. As such I used PCA in an exploratory data analysis to find the lowest number of components required to explain at least 90% of the variance in the training data.

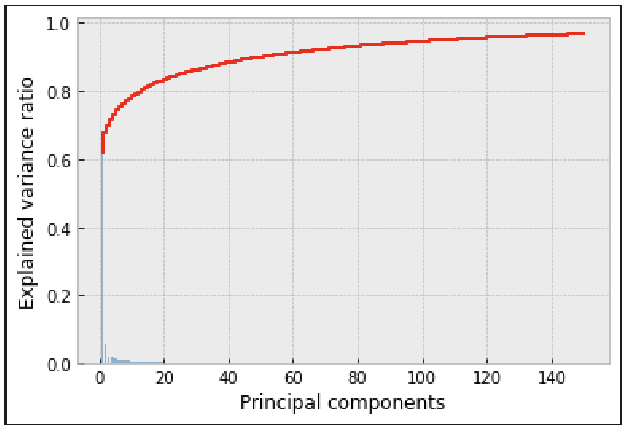

Fig. 11. Variance Explained by Number of Components

As seen in the figure above, the minimum number of principle components required to explain 90% of the variance in the training data was 50. This was quite a large reduction from the number of components in the original data, even as the graph tails off past 150 that's more than 3 times the minimum number for 90% explained variance! In order to test this reduction technique, separate SVM classifiers were trained on the reduced dataset and the original dataset. See the performance below.

Model	Accuracy
SVM	66%
SVM w PCA	55%

As seen in the table, the dimensionality reduction resulted in about an 11% drop in classification accuracy. In addition, both of these models performed terribly anyways and I wouldn't use either of them for this problem in particular.

V. More Dimensionality Reduction

A. Linear Discriminant Analysis

In another attempt to reduce the dimensionality of the dataset I experimented with Linear Discriminant Analysis, or LDA. THe goal of LDA is to find a linear combination of features of the dataset that separates two or more classes/objects. This linear combination can then be used as a dimensionality reduction for a classification model. Based upon the visualizations provided in the training notebook, I would select 4 features to begin the dimensionality reduction as that encompasses a majority of the ”points” in the training dataset. When compared to PCA, LDA seems to provide less discreet classes as it is entirely a centralized lump.

B. Stochastic Neighbor Embedding

In addition to LDA, I experimented with Stochastic Neighbor Embedding, or SNE. SNE is a great way to visualize high dimensional data by locating each point in a 2 or 3 dimensional map. In particular I used the t-distributed version of SNE, aptly named t-SNE. t-SNE begins by calculating a probability distribution across pairs of high dimensional data objects so that similar objects will have a higher probability and dissimilar objects will have a low probability. Then, t-SNE calculates a second probability distribution over all the points in a low dimensional map, minimizes the KL divergence between the distributions in accordance with the locations of points on the map. Based upon the visualization included in the training notebook, I would select 40 or so components as most of the data is contained within that many components. When compared to PCA, t-SNE seems to provide more discreet classes as opposed to a centralized lump.

VI. Manifold Learning Algorithms

In order to fully exhaust the tools in my repertoire, 3 manifold learning techniques were applied and measured on the dataset. For this experiment I selected Multidimensional scaling (MDS), ISOMAP, and Locally Linear Embedding (LLE) all to reduce the dimensionality of the dataset. After fitting models using each of the selected manifold techniques and visualizing them in the training notebook, I selected LLE as it did the best job in separating the classes from each other.

VII. Conclusion

All in all this has been an enjoyable dive into dimensionality reduction in machine learning. As we all know the curse of dimensionality is a very present issue with data, especially image data. As quality of images improve, the need to reduce dimensions and/or speed up the processing techniques increases exponentially. I would say that I struggled a bit on this project with computational resources and balancing the down-sampling from the original data. That and the mislabeled/confounding samples were hard to fight against, and as such the accuracy of my models suffered. Please view the accompanying notebooks for quantitative model evaluation metrics.