AMATH 482 HOME WORK 2
DIGIT CLASSIFICATION

LANG QIN

Amazing Department, University of Washington, Seattle, WA
lq25@uw.edu

Abstract. The goal of this work is to develop and train a classifier to distinguish images of handwritten digits from the famous MNIST data set. This work consists of three sub-tasks in order to achieve the goal: 1) investigate the dimensionality of data by Principle Component Analysis; 2) determine the number of Principle Component Analysis modes to approximate up to certain percentage; and 3) build Ridge regression model with cross validation for classification.

1. Introduction and Overview

The purpose of this work is to develop a digit image classifier by implementing Principle Component Analysis (PCA) over the training data to extract outstanding features of the digit images and by applying Ridge regression with PCA to distinguish digit images. The training set contains 2000 instances of handwritten digits, the "features" are 16×16 black and white images, i.e. $X_{\text{train}} \in \mathbb{R}^{2000\times 256}$, while the "labels" are the corresponding digit, i.e. $Y_{\text{train}} \in \mathbb{R}^{2000}$. The test set has the same attributes except that there are only 500 instances of image, i.e. $X_{\text{test}} \in \mathbb{R}^{500\times 256}$ and $Y_{\text{test}} \in \mathbb{R}^{500}$.

Following the above information, this work consists of three main tasks:

(1) Use PCA to investigate the dimensionality of X_{train} and plot the first 16 PCA modes as 16×16 images.
(2) Compute the number of PCA modes to approximate X_{train} up to 60%, 80%, and 90% in the Frobenius norm.
(3) Train a classifier by Ridge regression with cross-validation between digit pairs $\{1, 8\}$, $\{3, 8\}$, and $\{2, 7\}$.

2. Theoretical Background

To achieve the goal, this work primarily relies on the following three points: 1) singular value decomposition, 2) principle component analysis, 3) Frobenius norm, and 4) Ridge regression with cross-validation.

2.1. **Singular Value Decomposition (SVD).**

The singular value decomposition (SVD) is among the most important matrix factorizations of the computational era, providing a foundation for the principle component analysis which is essential for this work by obtaining low-rank approximations to matrices.

2.1.1. *Definition of the SVD.* Given a large data set $\mathbf{X} \in \mathbb{C}^{n\times m}$, where the columns $\mathbf{x}_k \in \mathbb{C}^n$ may be measurements from simulations or experiments. The SVD of $\mathbf{X}$ is given by $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$ [1] where $\mathbf{U} \in \mathbb{C}^{n\times n}$, $\mathbf{\Sigma} \in \mathbb{C}^{n\times m}$, and $\mathbf{V} \in \mathbb{C}^{m\times m}$. The matrices $\mathbf{U}$ and $\mathbf{V}$ are unitary, so that $\mathbf{U}\mathbf{U}^* = \mathbf{U}^*\mathbf{U} = \mathbf{I}$ and $\mathbf{V}\mathbf{V}^* = \mathbf{V}^*\mathbf{V} = \mathbf{I}$. The matrix $\mathbf{\Sigma}$ has entries along the diagonal corresponding to the singular values that are ordered from largest to smallest [1].

Date: February 11, 2022.

[1] $\mathbf{V}^*$ represents the transpose of $\mathbf{V}$

2.1.2. *Matrix Approximation.* SVD produces a hierarchical matrix decomposition that splits a matrix into a sum of rank-1 matrices given by the outer product of a column vector (left singular vector) with a row vector (conjugate transpose of right singular vector). These rank-1 matrices are ordered by the singular value so that the first r rank-1 matrices:

$$\tilde{\mathbf{X}} = \sum_{k=1}^{r} \sigma_k \mathbf{u}_k \mathbf{v}_k^* = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^* + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^* + \cdots + \sigma_r \mathbf{u}_r \mathbf{v}_r^* \tag{1}$$

form the *best* rank-r matrix approximation of the original matrix in a least-squares sense [1].

2.2. **Principle Component Analysis (PCA).** Principle Component Analysis (PCA) is one of the central uses of the SVD, a decomposition of a data matrix into a hierarchy of principle component vectors that are ordered from the most correlated to least correlated with the data.

PCA is computed by taking the SVD of the data after subtracting the mean. In this case, each singular value represents the variance of the corresponding principle component (singular vector) in the data [1].

2.3. **Frobenius based Energy Truncation.** The norm of a matrix is a scalar that gives some measure of the magnitude of the elements of the matrix. [3] The Frobenius norm can be calcualted directly as follows;

$$||\mathbf{A}||_F^2 = \sum_{j=1}^{\min\{m,n\}} \sigma_j(\mathbf{A})^2 = \sigma_1^2 + \sigma_2^2 + \cdots + \sigma_{\min\{m,n\}}^2, \tag{2}$$

where $\sigma_i = \mathbf{\Sigma}_{i,i}$ from the SVD of $\mathbf{A}$. Frobenius norm could be used as the base of content energy measurement, the required specified contained energy E_k could be represented as follows;

$$E_k = \frac{||\mathbf{A}_k||_F^2}{||\mathbf{A}||_F^2}, \tag{3}$$

where $\mathbf{A}$ is the original data and the $\mathbf{A}_k$ is the approximated or truncated data at rank k of E_k percentage or energy [3].

2.4. **Ridge Regression with Cross-Validation.** Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where linearly independent variables are highly correlated [2]. Mathematically, the ridge regression estimator is

$$\hat{\beta}_{ridge} = (\mathbf{X}^*\mathbf{X} + k\mathbf{I}_p)^{-1}\mathbf{X}^*\mathbf{y}, \tag{4}$$

where $\mathbf{I}_p$ is the $p \times p$ identity matrix and $k > 0$ is small [2].

2.4.1. *Ridge Regression by Scikit Learn.* Ridge regression is a *supervised* learning model, which solves a regression model where the loss function is the linear least squares function and regularization is given by the l2-norm. The ridge coefficients minimize a penalized residual sum of squares:

$$\min_w ||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 + \alpha||\mathbf{w}||_2^2 \tag{5}$$

The complexity parameter $\alpha \geq 0$ controls the amount of shrinkage: the larger the value of α, the greater the amount of shrinkage and thus the coefficients become more robust to collinearity [5].

2.4.2. *Model Evaluation by Mean Squared Error (MSE).* Mean squared error (MSE) is a risk metric corresponding to the expected value of the squared error or loss, which is widely used to measure the performance of a supervised learning model [6].

If $\hat{y}_i$ is the predicted value of the i-th sample, and y_i is the corresponding true value, then the MSE estimated over n_{samples} is defined as

$$\text{MSE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} (y_i - \hat{y}_i)^2. \tag{6}$$

Note: for MSE, the smaller the better.

2.4.3. *Cross-Validation.* The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other $k-1$ subsets are put together to form a training set. Then the average error across all k trials is computed [4].

Define the CV prediction error (CV cost) of prediction function $\hat{f}$ with penalty parameter as

$$CV(\hat{f}, \lambda) := \frac{1}{N} \sum_{k=0}^{k-1} ||\hat{f}_{-k}(\hat{\mathbf{x}}_k, \lambda) - \hat{\mathbf{y}}_k||^2. \tag{7}$$

With above cost function, an optimal choice of λ is obtained by minimizing the CV loss, i.e.

$$\lambda^* = \underset{\lambda \in (0, +\infty)}{\operatorname{argmin}} \; CV(\hat{f}, \lambda). \tag{8}$$

3. Algorithm Implementation and Development

The implementation of this project can be divided into three parts: 1) Environment Setup, 2) Dimensionality of $\mathbf{X}_{\text{train}}$ by PCA, 3) Mode Approximation, and 4) Digit Classifier. This programming work is done by Google Colab. The `Python` libraries imported for this work include: `google.colab.drive`, `numpy`, `matplotlib.pyplot`, `sklearn.linear_model`, and `sklearn.mean_squred_error`.

3.1. **Environment Setup.** Import data from `MNIST_training_set.npy` and `MNIST_test_set.npy` as training and test data set respectively, and extract the "feature" and "label" values. The training set contains 2000 instances of handwritten digits, in which the "features" are 16×16 black and white images, i.e. $X_{\text{train}} \in \mathbb{R}^{2000 \times 256}$, while the "labels" are the corresponding digit, i.e. $Y_{\text{train}} \in \mathbb{R}^{2000}$. The test set has the same attributes except that there are only 500 instances of image, i.e. $X_{\text{test}} \in \mathbb{R}^{500 \times 256}$ and $Y_{\text{test}} \in \mathbb{R}^{500}$.

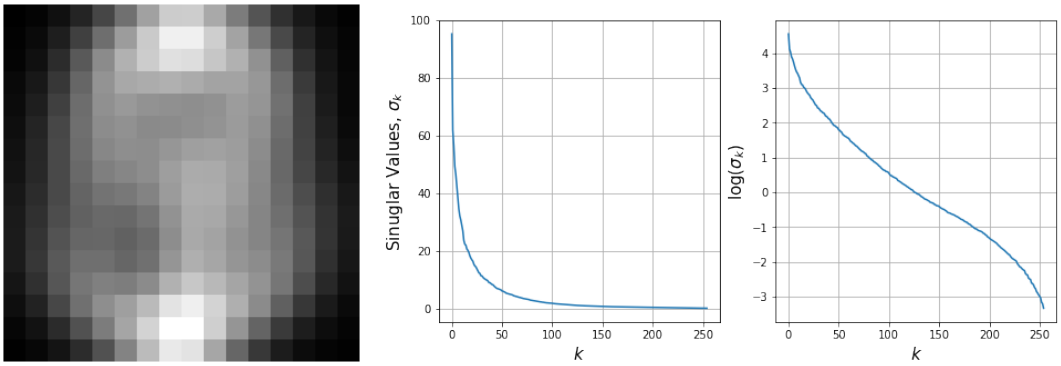
3.2. **Dimensionality of $\mathbf{X}_{\text{train}}$ by PCA.** This part applied PCA on $\mathbf{X}_{\text{train}}$ mentioned in previous section 2.2 with SVD in section 2.1. The first step is to obtain mean-subtracted training data $\mathbf{X}_{\text{train\_centered}}$, where the column mean of $\mathbf{X}_{\text{train}}$ is subtracted from each row in $\mathbf{X}_{\text{train}}$. Then, perform SVD over $\mathbf{X}_{\text{train\_centered}}$, having $\mathbf{X}_{\text{train\_centered}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$. Now, plot the first 16 PCA modes as 16×16 images, which is the first 16 rows of $\mathbf{V}^*$ reshaped into 16×16 matrices (images); and, plot the singular values to see the effective dimensions of the data set and log of singular values to investigate the rate that singular values change.

3.3. **Mode Approximation.** This part computes the number of PCA modes to keep in order to approximate $\mathbf{X}_{\text{train}}$ up to 60%, 80% and 90% in the Frobenius norm, as described in section 2.3. The above is accomplished by calculating the energy E_k using Eq. 3 and Eq. 2 in Section 2.3 at each PCA mode iteratively. Then, plot the energy/approximation percentage with respect to the number of PCA modes and locate the number of modes that restore 60%, 80% and 90% energy.

3.4. **Digit Classifier.** This part develops a binary digit image classifier using 16 PCA modes approximation obtained in Part 3.3 and Ridge regression with cross-validation introduced in Section 2.4, where the model performance is evaluated by MSE defined in Section 2.4.2. The classifier is implemented to three pairs of digits: $\{1, 8\}$, $\{3, 8\}$, and $\{2, 7\}$.

A classifier to distinguish the digit p and q is trained via the following steps:

(1) Extract the features and labels of the digits p and q from the training data set. Name the desired features and labels by $\mathbf{X}_{(p,q)}$ and $\mathbf{Y}_{(p,q)}$.
(2) Computes a matrix $\mathbf{A}_{\text{train}} \in \mathbb{R}^{455 \times 16}$, in which the columns corresponding to the PCA coefficients of each feature and rows corresponding to the total number of 1's and 8's in the training set. Since the column space of V represents PCA modes ($\mathbf{X}_{\text{train\_centered}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$ by Part 3.2) and the row space of $\mathbf{X}_{\text{train\_centered}}$ stores image information, then $\mathbf{A}_{\text{train}}$ is obtained by projecting the row space of $\mathbf{X}_{(p,q)}$ onto the first 16 column space of V, i.e. $\mathbf{A}_{\text{train}} = \mathbf{X}_{(p,q)}V$.
(3) Assign label -1 to the images of the digit p and label $+1$ to the images of the digit q, resulting in a vector $\mathbf{b}_{\text{train}} \in \{-1, +1\}^{455}$.
(4) Use Ridge regression to train a predictor for the vector $\mathbf{b}_{\text{train}}$ by linearly combining the columns of $\mathbf{A}_{\text{train}}$. This step is done by `sklearn.linear_model.RidgeCV`'s functions in practice. In particular, the model uses Leave-One-Out cross-validation in training.



(A) Averaged Digit Images (B) Ordered Singular Values and Log of Singular Values

FIGURE 1. Averaged digit image and Ordered Singular Values (A) and Log of Singular Values of $\mathbf{X}_{\text{train}}$ (B)

(5) Report the training MSE of the fitted classifier

$$\text{MSE}_{\text{train}}(p,q) = \frac{1}{\text{length of } \mathbf{b}_{\text{train}}} \times ||\mathbf{A}_{\text{train}}\hat{\beta} - \mathbf{b}_{\text{train}}||_2^2. \tag{9}$$

(6) Use the fitted classifier to predict using $\mathbf{X}_{\text{test}}$ and report the test MSE

$$\text{MSE}_{\text{test}}(p,q) = \frac{1}{\text{length of } \mathbf{b}_{\text{test}}} \times ||\mathbf{A}_{\text{test}}\hat{\beta} - \mathbf{b}_{\text{test}}||_2^2. \tag{10}$$

Note: The MSE's above are obtained by function `sklearn.mean_squared_error()` in practice.

4. Computational Results

4.1. **Dimensionality of $\mathbf{X}_{\text{train}}$ by PCA.** The subtracted average image for PCA is shown as Figure 1a. The image shows the general shape of all digit images. The shape of an ambiguous digit 8 in the center of the plot is the result by stacking digits on each other and digits such as $2, 3, 5, 6, 7, 8, 9$ could contribute to this shape.

The plot of ordered singular values and the plot log of singular values are shown in Figure 1b. From these plots, one may observe that the singular values decrease significantly from the 1st to 50th and the rate of change is very small from 100th singular value to the rest. In addition, approximately the first 20 singular values decreases extremely fast, which suggests that one may approximate the image using small rank approximation.

The plot of first 16 PCA modes as 16×16 images is shown as Figure 2. The black and dark grey pixels are produced by the mean subtraction step of PCA. From the remaining white and light grey pixels, one could tell the features or characteristics that each mode captures.

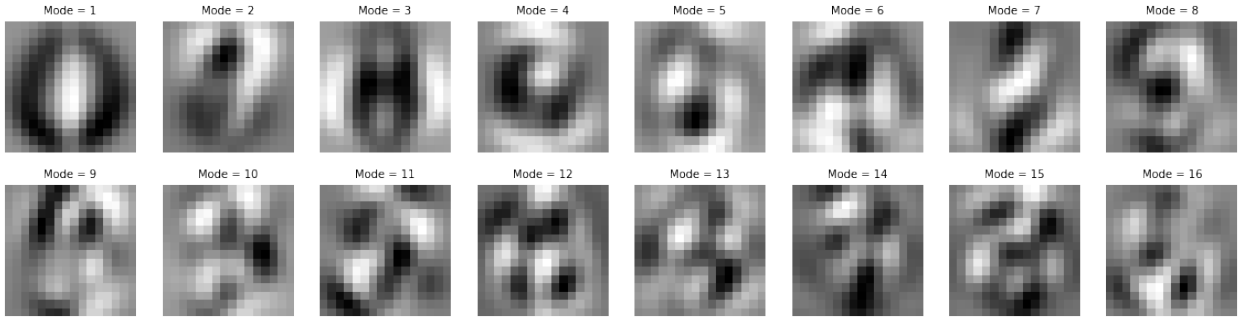


FIGURE 2. First 16 PCA modes of $\mathbf{X}_{\text{train}}$

4.2. **Mode Approximation.** The plot of Cummulative approximation percentage (energy) at rank k approximation is shown as Figure 3. According to the plot, 90% percentage approximation based on Frobenius norm can be achieved by low-rank approximation. By calculation, the number of modes to achieve 60%, 80% and 90% approximation is 3, 7, and 14 respectively. This implies that using 16 modes for further model construction is more than sufficient and safe. In other words, it is unnecessary to use the entire 256 pixels for each data point.

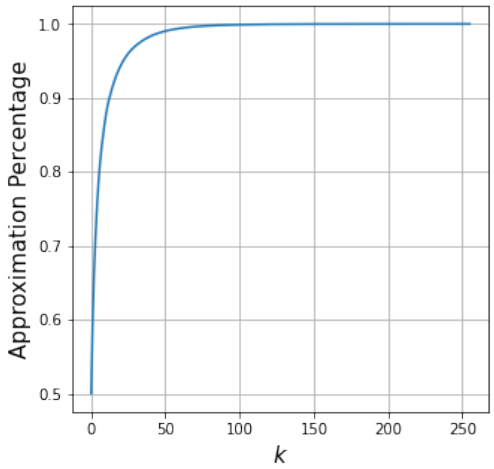


FIGURE 3. Cumulative approximation percentage at rank k

4.3. **MSE of the Classifiers.** Part 3.4 trained and tested supervised model of Ridge regression with cross-validation over digit pairs $\{1,8\}$, $\{3,8\}$, and $\{2,7\}$. The MSE's of the training and test set of each pair are listed in Table 1. From the table, the model performs relatively well in distinguishing digit 1 and 8, while the performance drops significantly when dealing with digit 3 and 8, and digit 2 and 7. In particular, the test MSE of the case {3,8} is higher than 0.25, while 0.083 for {1,8} and 0.133 for {2,7}.

MSE	$\{1,8\}$	$\{3,8\}$	$\{2,7\}$
Training	0.0755	0.1829	0.0921
Test	0.0826	0.2610	0.1336

TABLE 1. Training's and test's MSE's of Pairs

5. SUMMARY AND CONCLUSIONS

Through PCA and Ridge regression with cross-validation, this work successfully develops binary digit classifiers for digit pairs $\{1,8\}$, $\{3,8\}$, and $\{2,7\}$. The algorithms of this work can be applied to images of other objects that shares similarities, e.g. images of dogs and cats.

In addition, one may observe that the inconsistent performance between pairs $\{1,8\}$ and $\{3,8\}$, $\{2,7\}$. This is due to the highly similar shape of 3 and 8, 2 and 7: 3 is the right half counterpart of 8, where 2 and 7 shares very similar upper part especially in handwritten form. This implies that the method and algorithm implemented in this work are not perfect yet, i.e. 16 PCA modes may not be sufficient to distinguish data with coincidences and Ridge regression may not be the best model in this situation. The future work could be done to improve the digit classifier's performance of deceptive cases such as $\{3,8\}$ and $\{2,7\}$.

ACKNOWLEDGEMENTS

The author is thankful to Professor Steve Brunton and Professor Nathan Kutz for their amazing and comprehensive book *Data-Driven Science and Engineering, Dynamical Systems, and Control*. In Additionally, the author grateful to the sunshine on Tuesday 10/2/2022.

References

[1] S. L. Brunton and J. N. Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems and Control*. Cambridge University Press, 2020.
[2] D. E. Hilt and D. W. Seegrist. *Ridge, a computer program for Calculating Ridge Regression estimates*. Dept. of Agriculture, Forest Service, Northeastern Forest Experiment Station, 1977.
[3] R. A. Sadek. SVD based image processing applications: State of the art, contributions and research challenges. *CoRR*, abs/1211.7102, 2012.
[4] J. Schneider. Cross validation, Feb 1997.
[5] 1.1. linear models.
[6] 3.3. metrics and scoring: Quantifying the quality of predictions.