AMATH 582: HOMEWORK 2

VINSENSIUS

Applied Mathematics Department, University of Washington, Seattle, WA
vxvinsen@uw.edu

Abstract. A supervised machine learning is done on a MNIST datasets, with 500 test sets and 2000 training sets [1]. Ridge regression with Cross-Validation is used to train the data using 16 PCA modes. The digit pairs that the data is train are (1,8), (3,8) and (2,7) pairs. The λ value used is 1. Then, the MSE values are calculated for both data sets but some can not be trusted because of uneven distribution of the data point. Based on the scatter plot of the model prediction, it can recognize digit 1 fastest, while having trouble the most to recognize digit 8.

1. Introduction and Overview

Nowadays, machine learning has been widely use in many application, one of them is in data science. It is mainly categorized into two types, supervised learning and unsupervised learning. Supervised learning is used for predict or classify data while the unsupervised is to find meaningful structure of the data. The data that is supplied is MNIST data, which is a data containing handwritten digit and usually used as training dataset for machine learning [1].

In this project, the training data set is 2000 handwritten digits while the test data set is 500 handwritten digits with their respective label. The main task is to classify the pairs of digits (1,8), (2,7), (3,8) using Ridge linear regression. Then, MSE values will be calculated to evaluate how the classifier model performs and validate whether MSE can be trusted.

2. Theoretical Background

Linear algebra is one of the mathematics branches that is important to machine learning, for example the theory of Singular Value Decomposition (SVD). SVD is one of the building blocks of machine learning. Let A $\in \mathbb{R}^{n*m}$ then there exist unitary matrices $U \in \mathbb{R}^{n*n}$ and $V \in \mathbb{R}^{m*m}$ as well as a diagonal matrix, $\Sigma \in \mathbb{R}^{n*m}$ with non-negative entries such that:

$$A = U\Sigma V^T \tag{1}$$

The SVD is valid for any matrix, which makes it very powerful, because it can decompose any matrix to its components. The columns of matrix U are called left singular vectors of A and form a basis in $\mathbb{R}^n$. The columns of V is called right singular vector is the basis for $\mathbb{R}^m$. The diagonal entries are called the singular values of A and is arranged in descending order. The SVD shows the any operation by any matrix A is equivalent to rotation (by unitary matrix V), scaling, i.e. compression and stretching (by diagonal matrix Σ) and new basis formation (by matrix U). From SVD, we can compute the Frobenius norm by:

$$||A||_F = \left(\sum_{j=1}^{min(m,n)} \sigma_j^2 \right)^{\frac{1}{2}} \tag{2}$$

Date: February 12, 2022.

where σ_j is the singular values of matrix A. Principal Component Analysis (PCA) is another method to find approximation to the data. PCA can also be viewed as compression technique. Let X be a data set where $X = \{\underline{x_0}, \underline{x_1}, ..., \underline{x}_{N-1}\}$, where $\underline{x_j} \in \mathbb{R}^d$ and represents a vector of each data point such as image or audio. X is is matrix of dimension $N * d$. To do PCA, the matrix needs to be centered around the mean first. Then, apply SVD on the matrix X. Then, the columns of U matrix of the SVD is the PCA modes of matrix X. We can define a linear regression model, $\hat{f}$ as:

$$\hat{f} = \hat{\beta}_0 + \sum_{j=0}^{d-1} \hat{\beta}_j x_j \tag{3}$$

where $\hat{\beta}$ is the coefficient model to be found by minimizing the least square error of :

$$\hat{\beta} = \text{argmin}_{\beta \in \mathbb{R}^d}(||\hat{f(x)} - y||) \tag{4}$$

where $\hat{f(x)}$ prediction of the model based on the input and y is the true value or the output value of x. We can define the model for regression in general as:

$$f(\underline{x}) = \sum_{j=1}^{j-1} \beta_j \psi_j(\underline{x}) \tag{5}$$

where ψ_j is the feature map define as, $\psi_j : X \to \mathbb{R}$. The model for this can be trained as long as f is linear with respect to β, which means it does not matter the type of function is chosen for ψ. We can define matrix of feature maps that takes in each of $\underline{x} \in X$ evaluated at each basis function ψ_j:

$$\mathbf{A} = \begin{bmatrix} \psi_0(\underline{x}_0) & \cdots & \psi_{j-1}(\underline{x}_0) \\ \vdots & \ddots & \vdots \\ \psi_0(\underline{x}_{N-1}) & \cdots & \psi_{j-1}(\underline{x}_{N-1}) \end{bmatrix} \tag{6}$$

Then, the coefficient $\hat{\beta}$ can be found by:

$$\underline{\hat{\beta}} = (A^T A)^{-1} A^T y \tag{7}$$

The solution exists as long as the inverse of $A^T A$ exists. However, this is not the case in big data. Thus, a regularization parameter, λ is introduced to make the matrix invertible. This is called Ridge linear Regression. However, it is very important to choose the parameter correctly. If the parameter is too large, the model will be biased because the $\hat{\beta}$ is very small resulting in underfitted model. While if the parameter is too small, the model is controlled by the training model, resulting in overfitting model.

The idea of cross-validation (CV) feature of regression is that to break the data into smaller pieces and remove one of the pieces to validate the model. This way the model can be tune better such as number of features. By using CV, the optimal regularization parameter can be found faster rather than testing it by trial and error in normal linear regression model.

To measure the error of the model, the mean-square error (MSE) can be calculated by :

$$MSE_{train} = \frac{1}{N} \sum \left(||A_{train} \hat{\beta} - y_{train}|| \right) \tag{8}$$

$$MSE_{test} = \frac{1}{N} \sum \left(||A_{test} \hat{\beta} - y_{test}|| \right) \tag{9}$$

The MSE only shows how much variance between the model and the value can be affected by the distribution of the data.

3. Algorithm Implementation and Development

Python packages that are used:numpy [2], matplotlib [3] and scikit-learn [4].

Training Data First, transpose the X matrix (data) to follow the convention of the data (feature maps) from the theory. So, now the the rows representing the image pixels while the columns represent the data point (images). Then, center the data around the mean by subtracting the mean around the data points, i.e. take the mean columnwise. Then, take the SVD of the data. This can be done using *np.linalg.svd* function. The PCA mode is the columns of the U matrix from SVD. To plot the image of the modes, the *reshape* function can be used to change the vector of PCA modes into 16x16 images.

The Frobenius norm can be calculated using equation given in theory section. It can be done using *np.sum* function and use the Σ matrix to calculate the norm. To keep certain percentage the Frobenius norm, we can keep only some of the singular values, recalculate and take the ratio of the reduced norm to the full norm.

Next we can do digits recognition task. We need to extract the indices of Y matrix based on the digits that we want the model to train in. Then, construct the label matrix based on the how many data points do each digit have. Next, extract all the handwritten digit from X matrix based on the indices from Y matrix.

After that construct the A matrix by multiplying reduced X matrix by the label matrix. Following that, apply Ridge regression by Cross Validation using *RidgeCV* function, and extract the coefficients of the regressions. Then, multiply the coefficient by the A matrix to get the predicted label matrix. Calculate the mean square error (MSE) between the predicted label matrix with the real label matrix using the formula from the theory section.

Test Data. For the test data, the first part is similar to the training where the data needs to be centered, finding the label matrix and A matrix. The different part is when finding the predicted label matrix based on the model. Use *predict* function to get the predicted label matrix based on the model that is created from the training data. After that, the calculation of MSE will be taking the 2-norm of the difference of predicted label and the true label normalized by the number of data points.

4. Computational Results

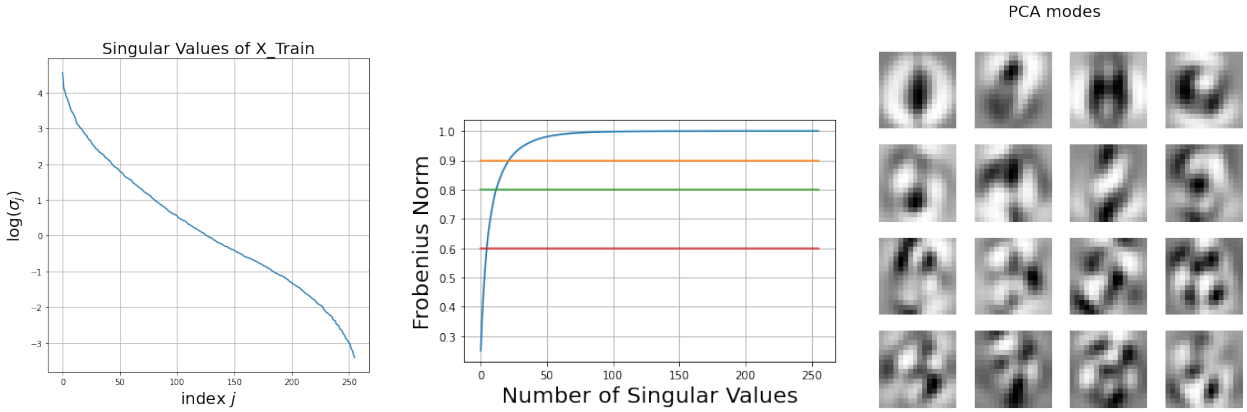
The following section will discuss various result from the data. Figure 1c shows the first 16 PCA modes from X_{train} matrix. The modes represents the rotation of the data in certain directions. In this case, it would be the rotation of the pixel of the image.

Figure 1a shows the singular values of X_{train} matrix. It represents the magnitude of the stretching and compressing the data in the direction of specify by the modes. By looking at the singular values plot, one can know how many modes to take to represent the whole data, i.e. doing compression. From the σ plot, we can roughly see that taking about half of singular values to represent the whole data. This is because above that, the singular values are very small that it has very subtle change to the data (image). Thus, the important information of the data can still be preserve by compressing it to half of its original size.

The compression idea can also be seen from the normalized Frobenius norm, Figure 1b. Frobenius norm is defined as the trace of a covariance matrix, which translates to sum of squares of singular values from SVD. Thus, truncating the singular values can also means decreasing the variance within the data. However, we can truncate the singular values while maintaining high variance in the data such that we can extract the important information. Thus, looking at Frobenius norm we can see that keeping about half of the singular values will still preserve the overall norm.

The horizontal lines represents the objective fraction of the norms that was given in the problem statement. For maintaining about 60% of the norm, only 5 or 6 singular values are needed be kept depending whether one would want higher or lower than the fraction. For maintaining about 80%

of the overall norm, 12 or 13 singular values are needed to be kept. For maintaining about 90% of the overall norm, 22 or 23 are needed.



(A) The plot of singular values in logarithmic scale

(B) The plot of normalized norm and lines of constant fraction

(C) First 16 PCA modes in images

FIGURE 1

Table 1 shows the number of data points for each digit within each of the data set. Table 2 shows the MSE in percentage for each pair of the digits in each dataset. Based on the MSE results, the Ridge regression works the best with the digits 2 and 7. This is possibly because the training set provides a good representation of the digits, leading to a slightly off error.

This can be supported by the Figure 3a and 2c. In Figure 2c, most of the labels cluster around $+1$ and -1 like what they should be doing. By using zero line as dividing line, we can see that most of the labels are clustering in places where they are supposed to be, except for some of the points that are lingering near the zero line. This is where the model is unsure of the digits because of the similarity that they are having. Thus, if the value of the label is close to zero but negative it will straight away be classified the other digit because of the dividing line.

In Figure 3a, the digits are more scatter, but the model still be able to be able to classify, except for some points. This is probably because the training data has most of the representation that is in the training model except for the anomaly. Furthermore, the data points for the digit are more or less equal so that will give a more accurate model because of equal distribution of the data.

Next, analyze the digit pair of 3 and 8. Based on the figures 3b and 3c, the model is having trouble classifying the digits. This is very true for digit 8 in the training data and digit 3 in test data. While for digit 3 of training data, the cluster seems to center around the label, 1. This means that the model mostly recognize digit 3. In the test data, the model is having trouble recognizing the digits because we can see that the points are diffusing from their label line.

Lastly, we can analyze digits 1 and 8. From the training data, the model mostly recognize digit 1 but not digit 8. This is probably some of the handwriting of digit 8 is too narrow that the model classify it as digit 1. Furthermore, like in digit pair (3,8), the model is having the same problem in recognizing digit 8. In the training data, the model is very good at recognizing digit 1 because of the clustering around the label, except for one point of anomaly. The model is doing better in recognizing digit 8 in test model, though there are some anomaly points. This is probably because the number of data points for digit 8 is less than digit 1.

We can see that MSE values do not give us any important information because it is very difficult to make sense what the number means.The MSE values only gives a measure of error by comparing the difference between the predicted label matrix and the true label matrix. However, the value itself is influenced by the variance of the difference and also the distribution of the data points. For

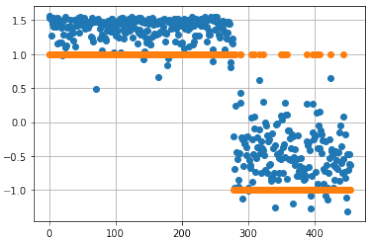
example, in test data, the digit 8 data points are twice as small as data points of digit 3 and three times smaller than data points of digit 1.

Digits (Dataset)	# of data points
1 (Train) & 1 (Test)	278 & 72
2 (Train) & 2 (Test)	180 & 53
3 (Train) & 3 (Test)	174 & 48
7 (Train) & 7 (Test)	171 & 45
8 (Train) & 8 (Test)	177 & 24

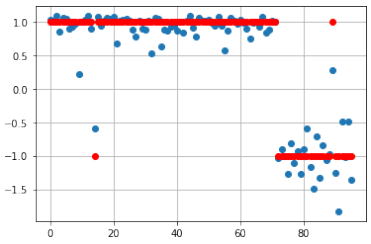
TABLE 1. Number of data points for each digit within each data set

Digits Pair (Dataset)	MSE (%)
1 and 8 (Training)	26.6
1 and 8 (Test)	8.54
2 and 7 (Training)	12.8
2 and 7(Test)	13.7
3 and 8(Training)	24.3
3 and 8 (Test)	26.1

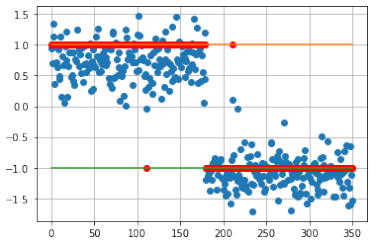
TABLE 2. MSE value for pair of digits for each data set



(A) Label model 1 & 8 training

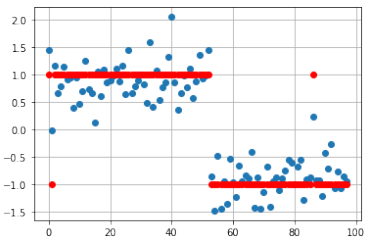


(B) Label model 1 & 8 test

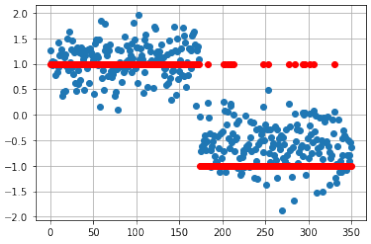


(C) Label model 2 & 7 training

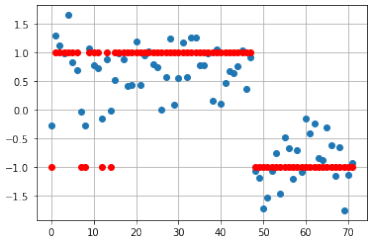
FIGURE 2. The predicted label for each pair of digit and their classification based on the *np.sign* function which classifies them to -1 & $+1$ label. $+1$ label is for digit higher than 5, -1 label is for digits less than equal to 5



(A) Label model 2 & 7 test



(B) Label model 3 & 8 training



(C) Label model 3 & 8 test

FIGURE 3. The predicted label for each pair of digit and their classification based on the *np.sign* function which classifies them to -1 & $+1$ label. $+1$ label is for digit higher than 5, -1 label is for digits less than equal to 5

5. SUMMARY AND CONCLUSIONS

From X_{train}, we obtained the first 16 PCA modes which then represented as images. We also found that to maintain 90%, 80%, and 60% of the Frobenius norm, we need 5,12 and 22 singular values respectively.Then by doing Ridge regression, we found the MSE values for pairs (1,8),(2,7) and (3,8) in each data set. For training data, the MSE values are 26.6%, 12.8%, and 24.4% respectively. For test data, we got 8.54%, 13.6%, and 26.1% respectively. However, the MSE values do not give any qualitative information about the model and also bias due to uneven distribution

in test data, for pair (1,8) and (3,8). It is better to see the scatter plot of the labels predicted by the model to get how the model is doing. To improve the work that has been done, we can use a higher order regression and better labelling system so that the model can account the middle cases when it is unsure about the digits. Another improvement is to account for uneven distribution of the data so that the error can be measured better.

Acknowledgements

The author is thankful to Prof. Hosseni, teaching assistant (TA), Nick, Raymodn Mead, Felipe and discord community of amath 482/582 in providing guidance and help to do the problem and other programming issue.

References

[1] L. Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[2] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020.

[3] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.