

1. In class, you learned that the direction that maximizes the variance of the projection onto a one-dimensional space is the eigenvector corresponds to the largest eigenvalue of the data covariance matrix  $S = \frac{1}{N}X^TX$ , where  $X = \begin{bmatrix} x_1^T - \bar{x}^T \\ \vdots \\ x_n^T - \bar{x}^T \end{bmatrix}$ . Formally, the solution to the following maximization problem

$$\max_{u_1} u_1^T S u_1 \quad \text{subject to } \|u_1\|^2 = 1,$$

is the eigenvector corresponds to the largest eigenvalue of  $S$ .

In this exercise, we use proof by induction to show that the linear projection onto an  $M$ -dimensional subspace that maximizes the variance of the projected data is defined by the  $M$  eigenvectors of the data covariance matrix  $S$  corresponding to the  $M$  largest eigenvalues. Now suppose the result holds for some general value of  $M$  and show that it consequently holds for dimensionality  $M + 1$ . To do this, first set the derivative of the variance of the projected data with respect to a vector  $u_{M+1}$  defining the new direction in data space equal to zero. This should be done subject to the constraints that  $u_{M+1}$  be orthogonal to the existing vectors  $u_1, \dots, u_M$ , and also that it be normalized to unit length. Use Lagrange multipliers to enforce these constraints. Then make use of the orthonormality properties of the vectors  $u_1, \dots, u_M$  to show that the new vector  $u_{M+1}$  is an eigenvector of  $S$ . Finally, show that the variance, i.e.,  $u_{M+1}^T S u_{M+1}$ , is maximized if we choose  $u_{M+1}$  to be the eigenvector that corresponds to the  $M + 1$ -th largest eigenvalue  $\lambda_{M+1}$ , assuming the eigenvalues have been ordered in decreasing value.

2. One application of PCA is compression. Suppose we want to compress a data vector  $x_n \in \mathbf{R}^D$  into  $M$  dimensions. We can write the PCA approximation to a data vector  $x_n$  in the form:

$$\begin{aligned}\tilde{x}_n &= \sum_{i=1}^M (x_n^T u_i) u_i + \sum_{i=M+1}^D (\bar{x}^T u_i) u_i \\ &= \bar{x} + \sum_{i=1}^M (x_n^T u_i - \bar{x}^T u_i) u_i.\end{aligned}$$

, where  $\bar{x}$  is the mean vector of  $\{x_1, \dots, x_N\}$  and  $\{u_1, \dots, u_D\}$  are the eigenvectors (corresponding to the largest to smallest eigenvalues) of the data covariance matrix:

$$S = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T$$

In this exercise, you are given part of the MNIST dataset that has handwritten 3 in it. The data is in the file *MNIST3.csv* which contains a matrix of size  $400 \times 784$ . Each row of the matrix represent an image of size  $28 \times 28$  where each element represents the intensity of each pixel in gray scale. The 400 images are shown in Figure 1.

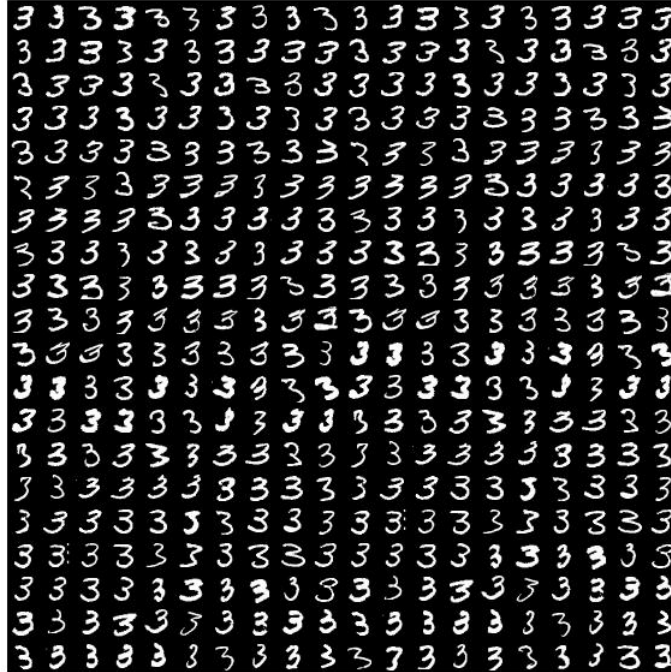


Figure 1: 400 images of 3

- (a) **Eigenvalues** Calculate the eigenvalues and eigenvectors of the data covariance matrix. You may use the function *eig* in MATLAB. Plot the 100 largest eigenvalues.
- (b) **Eigenvectors visualization** Visualize the first 4 eigenvectors by first reshaping the eigenvector into size  $28 \times 28$  and then showing it as an image using *imshow*. For a better visualization result, scale the range of each eigenvector into  $[0 - 255]$ . What do you observe?
- (c) **Compression using PCA** Compress the first image, i.e., the one on the top left corner, into  $M = 1, 10, 50$  and  $250$  dimensions. Plot the compressed images along with the original image in the same figure. Note that the original image corresponds to  $M = 784$ . Comment on the quality of the compressed images. What do you get if  $M = 0$ ?

3. Answer the following questions regarding positive definite matrix. A symmetric real matrix  $M$  is said to be positive definite if the scalar  $z^T M z$  is positive for every non-zero column vector  $z$ .

(a) Consider the matrix

$$A = \begin{bmatrix} 9 & 6 \\ 6 & a \end{bmatrix}.$$

What should  $a$  satisfy so that the matrix  $A$  is positive definite?

- (b) Suppose we know matrix  $B$  is positive definite. Show that  $B^{-1}$  is also positive definite. Hint: use the definition and the fact that every positive definite matrix is non-singular (invertible).

4. Suppose we have  $N$  balls in a jar that are numbered  $1, \dots, N$ .
- (a) We pick the ball randomly one at a time without replacement. What is the probability that ball 1 is not picked in  $N$  realization of this experiment?
  - (b) We pick the ball randomly one at a time with replacement. What is the probability that ball 1 is not picked in  $N$  realization of this experiment?
  - (c) For  $N = 1000$ , verify that the expression you get in (b) is close to  $1/e = 0.3679$ . Show that probability you get in (b) approaches  $\frac{1}{e}$  when  $N \rightarrow \infty$ . Hint: Take the natural log of the limit and then apply the L'Hospital's Rule.

5. In class, we learned that the log likelihood function for the Gaussian mixture model is of this form:

$$J = \ln P(X|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k) \right\}.$$

Here,  $\pi_k$  is the prior probability of the latent variable;  $\mu_k$  and  $\Sigma_k$  are the mean and covariance matrix for the  $k$ -th Gaussian component.

Suppose we want to maximize  $J$  with respect to  $\pi_k$ . Here we must take account of the constraint  $\sum_{k=1}^K \pi_k = 1$ . Use a Lagrange multiplier to enforce this constraint. Show that the  $\pi_k$  that maximize  $J$  is of the form:

$$\pi_k = \frac{N_k}{N},$$

where

$$N_k = \sum_{n=1}^N \gamma(z_{nk}),$$

and

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n | \mu_j, \Sigma_j)}.$$

You may assume that all  $\gamma(z_{nk})$  are known for this step.

6. In this exercise, you will implement the algorithm on page 167 of *A course in Machine Learning* and use it to perform classification on the data in *AdaBoost\_data.csv*.

- (a) **Visualization** The data file contains a matrix in which the 10 rows represent 10 data points. For each row, the first two columns contain the values of  $x_1$  and  $x_2$  and the third column contains the label  $y$  for each data point.

Generate a scatter plot of the dataset where data points from different classes are plotted using different color. Is this dataset linearly separable? Can we use a single layer decision tree to classify all points correctly?

- (b) **Implementation** Consider the decision stump (1 layered decision tree) of the following form as the base classifier for the AdaBoost algorithm.

$$\hat{y} = \text{sign}(s(x_i - t)), s \in \{+1, -1\}, i \in \{1, 2\}, t \in \mathbb{Z}.$$

The above classifier simply classify  $x_i$  to the right of the threshold  $t$  as either  $+1$  or  $-1$  based on the sign of either  $x_i - t$  or  $t - x_i$ . For simplicity, in this problem, we restrict  $t$  to be integer. The data is designed to avoid the evaluation of  $\text{sign}(0)$ .

Implement the AdaBoost algorithm on page 167 of *A course in Machine Learning* using the above base classifier for  $K = 3$ . Use natural log for the log operator in the algorithm and make sure to normalize the weights so that all weights sum to 1. To train the  $k$ -th classifier, for  $i \in \{1, 2\}$  and  $s \in \{+1, -1\}$ , search through all integers in the range of  $x_i$  exhaustively and find  $t^{(k)}$  that minimize the weighted misclassification error  $\hat{\epsilon}^{(k)}$ . To avoid exhaustive search for both  $s$  and  $i$ , we provide the optimal  $s$  and  $i$  for each iteration as follows: for  $k = 1, i = 1, s = -1$ ; for  $k = 2, i = 1, s = -1$ ; for  $k = 3, i = 2, s = 1$ . Choose the smaller  $t$  in the case that multiple  $t$ 's give the same weighted misclassification error.

As a sanity check, you should get  $t^{(1)} = 3$  which give you the first decision stump as:

$$\hat{y} = \text{sign}(3 - x_1).$$

Run the algorithm and report  $d^{(0)}, d^{(1)}$  and  $d^{(2)}$  in a table. What is your  $t^{(k)}$  and  $\alpha^{(k)}$  for  $k = 1, 2$  and  $3$ ? What is the final combined classifier? What is the training accuracy using this combined classifier?

Plot the data using *scatter* in MATLAB for  $k = 1, 2$  and  $3$  with the size of each point being  $1000 \times d^{(k-1)}$ . Draw the decision boundary of each decision stump.