Mini Project 1: Least Squares Based Supervised Classification
ECE 174, Fall 2020
Prof. Piya Pal

Mini Project # 1
Due: November 21, 11:59 pm, on Canvas

Instructions: In this programming assignment, you will implement two different least squares based multi-class classifiers on MNIST data set, and interpret your results. You will also implement an unsupervised classifier based on "K-Means Clustering" heuristic, test your program on the same dataset under various settings, and interpret your results.

- You can write the source code in any language of your choice (Python, MATLAB). The code should be properly commented and you will be required to submit the source code. You should write your own code and not copy from any resource on the internet. If you consult any resource online before writing your own code, you must cite the source, add a comment to your code with a link to the source you got it from. Any uncited code you did not write yourself, or any code which is substantially similar to another student's, will be referred to the Office of Academic Integrity.
- In addition to the source code, you will submit a detailed report that describes the simulations and explains the results using figures and mathematical interpretations. Your report must address all the questions asked in this assignment. We do not have any detailed requirements on the formatting, but we expect the report to be presented in a complete and professional manner, with appropriate labels on items and comments which demonstrate understanding of the results.
- Your grade on the programming assignment will be based on the correctness of the code as well as the quality of the report. Since our evaluation of your code is based on its results as presented in your report, correct code is very important. We have every intention of awarding partial credit, but for instance if a small bug causes your program's output to be unusable throughout the report, we cannot award much credit even if the bug was very small.

- **Reading (Necessary for doing the Programming Assignment):**
  1. **Least Squares Classifier:** Read Sec. 14.1-14.3 from the textbook to get familiar with least-squares based binary and multi-class classifiers. Pay special attention to Sec. 14.2.2, 14.3.1, 14.3.3 to become familiar with different metrics for evaluating classifier performance (such as confusion matrices).

2. **K-means Clustering:** Read Sec. 4.2 and 4.3 from the textbook to review "k-means" clustering. Also study Example 4.4.1 on image clustering. You will implement this example and do several numerical experiments.
3. **Dataset:** We will use the well-known MNIST (Mixed National Institute of Standards) database of handwritten digits which contains grayscale images of size 28×28 which can be represented as vectors of size 784. The dataset is provided as `mnist.mat` which you can directly read into MATLAB. For those working in Python, you can also refer to `http://rasbt.github.io/mlxtend/user_guide/data/loadlocal_mnist/` and `http://rasbt.github.io/mlxtend/installation/` for obtaining the equivalent data from the MNIST images.

1 Problem 1: Least Squares Classifier

(a) **Binary Classifier:** Consider the binary least-squares classifier where the labels can only take two possible values (corresponding to two classes). Suppose we are given data points $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$ where $\mathbf{x}_i$ denotes the feature and $y_i \in \{-1, 1\}$ is the corresponding label for each data point. Given N training data points, we try to find the best linear regression model by solving the following least squares problem

$$\min_{\boldsymbol{\beta}, \alpha} \sum_{i=1}^{N} (y_i - \boldsymbol{\beta}^T \mathbf{x}_i - \alpha)^2 \tag{1}$$

Let $\boldsymbol{\beta}^\star, \alpha^\star$ be the solution of the above problem. Then the **binary least squares classifier** $\hat{f}(\mathbf{x})$ is given by

$$\hat{f}(\mathbf{x}) = \text{sign}(\boldsymbol{\beta}^{\star T} \mathbf{x} + \alpha^\star)$$

for arbitrary test feature data $\mathbf{x}$. Here $\text{sign}(a) = 1$ for $a \geqslant 0$ and -1 for $a < 0$.

(b) **Multi-class classifier: One-versus-all classifier:** Now, we extend this idea to K classes. The **One-versus-all classifier** $\hat{f}(\mathbf{x})$ is given by

$$\hat{f}(\mathbf{x}) = \arg \max_{k=1,\cdots,K} g_k(\mathbf{x}) \tag{2}$$

where $g_k(\mathbf{x}) = \boldsymbol{\beta}_k^T \mathbf{x} + \alpha_k$ is the least squares regression model for kth label against the others. That is, if a training label $y_i = k$, you can treat it as 1 and -1 if $y_i \neq k$. Basically, you are supposed to solve (1) K times given training data set $\{(\mathbf{x}_i, y_i)\}$. Note that you have to transform the original training labels $y_i \in \{1, \cdots, K\}$ to appropriate binary values to solve each of these K problems. The classifier $\hat{f}(\mathbf{x})$ will return the index of the maximum function among $\{g_k(\mathbf{x})\}_{k=1}^K$.

(c) **Multi-class classifier: One-versus-One classifier:** In this part, we design pairwise Boolean classifier (with parameters $\boldsymbol{\beta}_{i,j}, \alpha_{i,j}$) for every pair of classes i, j $(i < j)$ as described in (a). Remember that in this case, in order to design each classifier, you will only consider subsets of data points that have class labels i and j. You can treat

the label corresponding to class i as $+1$ and that corresponding to class j as -1. This gives us a total of $K(K-1)/2$ classifiers called one-versus-one classifiers.

Given a test feature vector $\mathbf{x}$, let $y_{i,j}$ be the prediction of the i-versus-j classifier, with $y_{i,j} = 1$ meaning that the vector belongs to class i and $y_{i,j} = -1$ meaning it belongs to class j. Depending on the outcome of the Boolean classifier, we increment the "vote" of the appropriate class. The final predicted label is decided by picking the class which has maximum number of votes after passing through all $K(K-1)/2$ classifiers. (You can break ties by choosing the smallest index that achieves the largest number of votes.)

(d) **Training and Testing Multi-class Classifiers on MNIST Dataset:** For the least-squares classifier, use the training data from the MNIST dataset for training the classifiers (i.e. learning the least square parameters) and use the testing data to test the performance of the classifier. The objective is of course to identify the digits which can take values between $0-9$ (total of $K = 10$ classes). The dataset consists of $N = 60000$ training images and 10000 test images.

(a) Write your own code to implement the (i) one-versus-one and (ii) one-versus-all multi-class classifier using the binary classifier as building blocks. Your code should take appropriately labeled training data as the input, and determine the weights of the constituent binary classifiers as the output.

(b) Evaluate the training error of both classifiers using error rate and confusion matrix:
- The error rate is the total number of errors in predicted label divided by the total number of inputs being classified.
- Confusion matrix (ref. Pg. 287 and 298 from the textbook)

(c) Evaluate the performance of both classifiers on the test-data. Comment on their performance. How well do they generalize on the test data? Which digits are easy to recognize, which are harder?

(d) Compare the performance of the two classifiers. Explain your observations.

2 Problem 2: K-means Clustering

1. Write a program `k-means` to implement k-means clustering. The functions accepts two inputs: a list of N data vectors to be clustered, and the number of clusters, K. The program outputs three quantities: a list of N group assignment indices $(c_1, c_2, \cdots, c_N)$, a list of K group representative vectors $(\mathbf{z}_1, \cdots, \mathbf{z}_K)$ and the value of J^{clust} after each iteration of the algorithm until convergence or termination.

   Initialization and Termination: Initialize `k-means` by randomly assigning the data points to K group, i.e., randomly selecting each c_i to be an integer between 1 and K. Alternatively, you can also randomly assign K data points as the K group representatives. Explicitly mention your initialization technique in the report. You can follow any of the termination rules suggested in your textbook (Page 76). Again, mention in your report what criterion you used.

2. Let $K = 20$. Implement `k-means` on the MNIST dataset, starting with a random assignment of the vectors to clusters, and repeating the experiment $P = 30$ times. Use the training dataset of $N = 60000$ images to perform clustering using k-means. Save the output of the algorithm for each of the 30 runs. Consider the two runs that produce the maximum and minimum value of J^{clust} upon convergence. For each of these two runs
   - Plot the value of J^{clust} as a function of the number of iterations. Comment.
   - Visualize the K group representatives as images. You should generate something similar to Fig. 4.8 and 4.9 of the textbook. Interpret the group representatives (if which digits they represent, or if they represent something else) and compare them for these two runs. Discuss your results.
   - Find the 10 nearest data points to each of the K group representatives. Manually list what digits they represent (by eye estimation). In your opinion, how many of these 10 images match the corresponding group representative and how many of them are misclassified? Generate a table and interpret your results. Plot a few of the correctly classified as well as misclassified data points as images, alongside the group representatives (also represented as images).
3. Repeat part 2 for $K = 10$ and $P = 20$. Compare with the previous results.
4. Repeat part 2 for $K = 5$ and $P = 10$. Compare with the previous results.
5. Finally, compare the performance of the supervised binary classifierfrom Problem 1 and the unsupervised classifier based on k-means clustering. Comment on your results.