EE 516: Mathematical Foundations of Machine Learning **Winter 2023**

Homework 6
Due: February 24, 2023, 11:59PM PT

Student Name: *Instructor Name: John Lipor*

Problem 1 (2 pts, 4 pts)

Recall that a vector space is a general term, where "vectors" may be other elements (e.g., functions or matrices). Consider the vector space of $n \times n$ real-valued matrices. In this case, we may define an inner product between two "vectors" as

$$\langle A, B \rangle = \operatorname{tr}(B^T A),$$

where $A, B \in \mathbb{R}^{n \times n}$.

(a) Verify that the norm corresponding to this vector space is the Frobenius norm.

(b) Verify the following "orthogonality principle" for low-rank approximation.

$$(A - \hat{B}_K) \perp \hat{B}_K,$$

where $\hat{B}_K = \arg\min_{B:\operatorname{rank}(B)\le K} \|A - B\|_F$. The above states that the error resulting from low-rank approximation is orthogonal to the estimate itself.

Problem 2 (4 pts)

In the setting of slide 6.15 of the Low-Rank Approximation notes, verify the statement

$$\sum_{k=1}^{K+1} \sigma_k^2 \left\|v_k^T x\right\|_2^2 = \|Ax\|_2^2 .$$

Problem 3 (5 pts, 5 pts, 4 pts)

The classic least-squares problem seeks to minimize the ℓ_2-norm of the residual $r = Ax - b$. It is well-known that this norm is not robust to outliers, and hence we may wish to consider a more robust formulation involving the ℓ_1-norm known as the mean absolute error (MAE) or least absolute deviations (LAD) minimization

$$J(x) = \|Ax - b\|_1 = \sum_{i=1}^{m} \left|a_i^T x - b_i\right|, \tag{1}$$

where $b_i \in \mathbb{R}$ denotes the ith element of the vector b and $a_i \in \mathbb{R}^n$ is the (column version of the) ith row of A. Unlike least-squares, there is no closed-form solution to this problem. Further, $J(x)$ is not even differentiable, since the derivative of $|\cdot|$ does not exist at the origin. However, the cost function above is convex, so we do have tools to minimize it.

One popular approach to solving (1) is to apply the majorize-minimize (MM) algorithm, of which the popular expectation-maximization (EM) algorithm is a special case. Given an initialization x_0, the MM algorithm alternates between the following steps.

1. Find a function $\bar{J}_k(x)$ such that *majorizes* $J(x)$, i.e., satisfies

$$J(x_k) = \bar{J}_k(x_k)$$
$$J(x) \leq \bar{J}_k(x) \ \ \forall x.$$

2. Solve $x_{k+1} = \arg\min_x \bar{J}_k(x)$.

The goal is to choose $\bar{J}$ such that the minimization in step 2 can be carried out efficiently.

(a) Let

$$r_i(x) = a_i^T x - b_i.$$

Verify that

$$\bar{J}_k(x) = \sum_{i=1}^{m} \sqrt{r_i^2(x_k)} + \frac{1}{2}\frac{r_i^2(x) - r_i^2(x_k)}{\sqrt{r_i^2(x_k)}}$$

majorizes $J(x)$. Note that if $r_i^2(x_k) = 0$, we set that term in the sum to zero, but this event is extremely unlikely.
Hint: Use the fact that for a convex function $g : \mathbb{R}^n \to \mathbb{R}$

$$g(v) \geq g(u) + \langle \nabla g(u), v - u \rangle .$$

(b) Find and define the matrix $W \in \mathbb{R}^{m \times m}$ such that the minimization in step 2 is reduced to solving

$$\hat{x} = \arg\min_x \|W(Ax - b)\|_2^2 .$$

(c) The resulting algorithm for minimizing (1) is known as *iteratively reweighted least squares*. Implement this algorithm in `mae_irls` and test your implementation on the included `prob3`.
Turn in your `mae_irls` code and the resulting plot.

Problem 4 (4 pts each)

In this problem, you will implement the (classical) Multidimensional Scaling (MDS) algorithm for embedding points into a low-dimensional space given only a notion of distance between points. We derived this algorithm beginning on Pg. 6.31 of the notes. In that case, we considered only Euclidean distances, but in practice we could use any notion of dissimilarity. In this problem, we'll use the *cosine similarity*, i.e., inner products, to get a notion of distance between points on a union of subspaces. To convert this to a distance, we define

$$d(x_i, x_j) = 1 - \left\langle \frac{x_i}{\|x_i\|}, \frac{x_j}{\|x_j\|} \right\rangle , \qquad (2)$$

which is between 0 and 1 due to the normalization of points.

(a) Implement the MDS algorithm derived in class by completing the `MDS` function included. To test your algorithm, we'll consider again the MNIST digits used in the class demo. Run your `MDS` on the `mnistSubset.mat` data using *only digits 1 and 2*. **Turn in** a plot of the data embedded in three-dimensional space. Plot the points from each class in a different color (easily done using `hold all` in MATLAB).

(b) Now we'll compare to the embedding we get by using PCA for dimensionality reduction. Let X be the matrix whose columns are the vectors from classes 1 and 2 only. Using PCA as described in Section 6.2 of the notes, embed these points into three-dimensional space. **Turn in** a plot of the embedded data, again using a different color for each of the two classes. How does this compare to what you get from MDS? Note that PCA operates on the data, while MDS operates only on the pairwise distances.

Problem 5 (5 pts each)

In this problem, you'll use least squares to perform binary classification, an algorithm known as the *least-squares classifier* (see BV, Ch. 14). Suppose you are given training data consisting of N feature-label pairs $\{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathbb{R}^D$ are the MNIST images and $y_i \in \{-1, 1\}$ are the corresponding labels. We can use least-squares to find the vector $w \in \mathbb{R}^D$ such that the sign of $w^T x_i$ is a good predictor of y_i. Define the data matrix and vector of labels as

$$X = \begin{bmatrix} x_1 & x_2 & \dots & x_N \end{bmatrix} \qquad y = \begin{bmatrix} y_1 & y_2 & \dots & y_N \end{bmatrix}^T,$$

respectively. Then we can learn a separator by solving the ridge regression problem

$$\hat{w} = \arg\min_{w \in \mathbb{R}^D} \left\| X^T w - y \right\|_2^2 + \lambda \left\| w \right\|_2^2.$$

The label of any point $x \in \mathbb{R}^D$ can then be predicted as

$$\hat{y} = \operatorname{sign}(\hat{w}^T x).$$

Practical note: We're likely to obtain a better classifier if we add a bias and regularizer to the above. However, this classifier isn't the most practical choice anyway, so consider this problem as being only for instructional purposes of how classification works.

(a) Implement the above least-squares classifier on digits 1 and 2 from the MNIST subset provided using the `prob5` script. Train your classifier using the variables `Xtrain, ytrain`. **Turn in** the resulting error on both the training and test datasets.

(b) Now use PCA on the training data to project your data into two dimensions and re-train your LS classifier on the reduced-dimension training data. **Turn in** your training and test error and a plot of the reduced-dimension data with the learned LS separator (I've given you code for plotting the separator).

Problem 6 (5 pts)

For the remainder of the quarter, we will spend some time thinking about the ethics of machine learning. For this week, either watch the presentation by Timnit Gebru here or listen to the podcast here. After watching/listening, **share one thing you learned and one question you have to the `interesting-reading` channel on Slack**.