Problem Set 1

CS 4375

Due: 9/6/2023 by 11:59pm

Note: all answers should be accompanied by explanations for full credit. All code used as part of your solutions should be included for partial credit. **NO machine learning libraries may be used in your solutions.** Late homeworks will not be accepted.

Warm-Up: Properties of Convex Functions (10 pts)

Recall that a function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if for all $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$, $\lambda f(x) + (1-\lambda)f(y) \geq f(\lambda x + (1-\lambda)y)$. Using this definition, show that

1. $f(x) = wf_1(x)$ is a convex function for $x \in \mathbb{R}^n$ whenever $f_1 : \mathbb{R}^n \to \mathbb{R}$ is a convex function and $w \geq 0$.
2. $f(x) = f_1(x) + f_2(x)$ is a convex function for $x \in \mathbb{R}^n$ whenever $f_1 : \mathbb{R}^n \to \mathbb{R}$ and $f_2 : \mathbb{R}^n \to \mathbb{R}$ are convex functions
3. $f(x) = \max(f_1(x), f_2(x))$ is a convex function for $x \in \mathbb{R}^m$ whenever $f_1 : \mathbb{R}^n \to \mathbb{R}$ is a convex function and $f_1 : \mathbb{R}^n \to \mathbb{R}$.
4. $f(x) = f_1(x)^2$ is a convex function for $x \in \mathbb{R}^n$ whenever $f_1 : \mathbb{R}^n \to \mathbb{R}$ is a convex function and $f(x) \geq 0$ for all $x \in \mathbb{R}^n$.

Problem 1: Linear Regression (20 pts)

Consider the general linear regression problem from class with data observations $x^{(1)}, \ldots, x^{(M)} \in \mathbb{R}^n$ and corresponding labels $y^{(1)}, \ldots, y^{(M)} \in \mathbb{R}$ where the goal is to minimize the squared error,

$$\frac{1}{M}\left[\sum_{m=1}^{M}(a^T x^{(m)} + b - y^{(m)})^2\right],$$

over all possible choices of $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$.

Consider a modified loss function of the form

$$\max_{m \in \{1,\ldots,M\}} \left| a^T x^{(m)} + b - y^{(m)} \right|$$

1. Using the warm-up, argue that the modified loss is a convex function of $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$.
2. Explain how to minimize this loss function using subgradient descent, i.e., write some pseudocode and compute the subgradients. How would you pick the step size?

3. More generally, what are the pros and cons of the modified loss versus the original least squares loss for (polynomial) regression problems?

Problem 2: Perceptron Learning (25 pts)

Consider the data set (perceptron.data) attached to this homework. This data file consists of M rows of data elements of the form $(x_1^{(m)}, x_2^{(m)}, x_3^{(m)}, x_4^{(m)}, y^{(m)})$ where $x_1^{(m)}, \ldots, x_4^{(m)} \in \mathbb{R}$ define a data point in $\mathbb{R}^4$ and $y^{(m)} \in \{-1, 1\}$ is the corresponding class label.

In this problem, you are to implement the perceptron algorithm using two different subgradient descent strategies. For each strategy below, report the number of iterations that it takes to find a perfect classifier for the data, the values of w and b for the first three iterations, and the final weights and bias. Each descent procedure should start from the initial point

$$w^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad b^0 = 0.$$

1. Standard subgradient descent with the step size $\gamma_t = 1$ for each iteration.

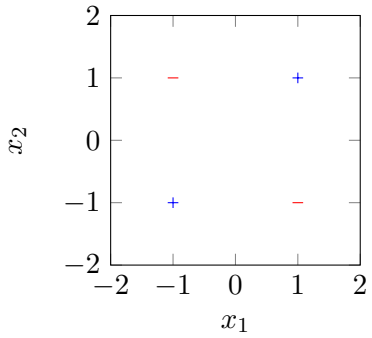
2. Stochastic subgradient descent where exactly one component of the sum is chosen to approximate the gradient at each iteration. Instead of picking a random component at each iteration, you should iterate through the data set starting with the first element, then the second, and so on until the M^{th} element, at which point you should start back at the beginning again. Again, use the step size $\gamma_t = 1$.

3. How does the rate of convergence change as you change the step size? Provide some example step sizes to back up your statements.

4. What is the smallest, in terms of number of data points, two-dimensional data set containing both class labels on which the algorithm, with step size one, fails to converge? Use this example to explain why the method may fail to converge more generally.

Problem 3: Separability & Feature Vectors (15 pts)

1. Consider the following data set.



Under which of the following feature vectors is the data linearly separable? For full credit, you must justify your answer by either providing a linear separator or explaining why such a separator does not exist.

(a) $\phi(x_1, x_2) = \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$ (c) $\phi(x_1, x_2) = \begin{bmatrix} x_1 x_2 \\ 1 \end{bmatrix}$

(b) $\phi(x_1, x_2) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ 1 \end{bmatrix}$ (d) $\phi(x_1, x_2) = \begin{bmatrix} x_1 \cdot \sin(x_2) \\ 1 \end{bmatrix}$

2. Suppose that you wanted to perform polynomial regression using gradient descent, i.e., you want to fit a polynomial of degree k to your data. Explain how to do this using feature vectors. What is the per iteration complexity of gradient descent as a function of the size of your feature representation and the number of training data points?

Problem 4: Support Vector Machines (30 pts)

For this problem, consider the data set (mystery.data) attached to this homework that. This data set contains four numeric attributes per row and the fifth entry is the class variable (either +1 or -1).

In this problem your goal is to find a perfect classifier for this data set that will generalize well to unseen data observations using support vector machines . The quality of your solution will be determined by applying your method to unseen data. What to turn in:

1. An explanation of the optimization problem that you solved to find a perfect classifier for the training data.

2. Your learned parameters (the weights and bias), the optimal margin, and the support vectors.

3. A function `eval` in Python or MATLAB that takes as input a dataset (treated as a two-dimensional array) in the above form excluding the last column (i.e., a matrix with four numeric attributes per row). The function can apply whatever feature transformation you would like, and using whatever weights you learned, it should return a class label for each row of the input matrix.

A total of ten points for this problem will be allocated to the performance of your `eval` function on the unknown test set. These 10 points will be awarded by multiplying your accuracy on the unseen data times 10, i.e., if you get 80% of the labels correct, you will receive 8 points. If the TA is unable to run your code, you will receive zero points.