CSE 151A: Machine learning

Homework 6

Instructions:

- You may discuss problems with your study group, but ultimately all your work (mathematical problems, code, experimental details) must be individual.
- Your solutions must be `typed up` and uploaded to Gradescope by 9.59PM on Thursday November 9. No late homeworks will be accepted under any circumstances, so you are encouraged to upload early.
- A subset of the problems will be graded.

Conceptual and mathematical problems

1. Given a set of data points $x^{(1)}, \ldots, x^{(n)} \in \mathbb{R}^d$, we want to find the vector $w \in \mathbb{R}^d$ that minimizes this loss function:
$$L(w) = \sum_{i=1}^{n} (w \cdot x^{(i)}) + \frac{1}{2} c \, \|w\|^2.$$
Here $c > 0$ is some constant.
   (a) What is $\nabla L(w)$?
   (b) What value of w minimizes $L(w)$?

2. Consider the following loss function on vectors $w \in \mathbb{R}^4$:
$$L(w) = w_1^2 + 2w_2^2 + w_3^2 - 2w_3w_4 + w_4^2 + 2w_1 - 4w_2 + 4.$$
   (a) What is $\nabla L(w)$?
   (b) Suppose we use gradient descent to minimize this function, and that the current estimate is $w = (0, 0, 0, 0)$. If the step size is η, what is the next estimate?
   (c) What is the minimum value of $L(w)$?
   (d) Is there is a unique solution w at which this minimum is realized?

3. Consider the loss function for ridge regression (ignoring the intercept term):
$$L(w) = \sum_{i=1}^{n} (y^{(i)} - w \cdot x^{(i)})^2 \; + \; \lambda \|w\|^2$$
where $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^d \times \mathbb{R}$ are the data points and $w \in \mathbb{R}^d$. There is a closed-form equation for the optimal w (as we saw in class), but suppose that we decide instead to minimize the function using local search.
   (a) What is $\nabla L(w)$?
   (b) Write down the update step for gradient descent.

(c) Write down a stochastic gradient descent algorithm.

4. For each of the following functions of one variable, say whether it is convex, concave, both, or neither.
   (a) $f(x) = x^4$
   (b) $f(x) = x$
   (c) $f(x) = x^3$
   (d) $f(x) = \ln x$, for $x > 0$

5. Show that the matrix $M = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ is not positive semidefinite. *Hint:* Work directly from the definition of positive semidefinite.

6. Show that the matrix $M = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$ is positive semidefinite.

7. For some fixed vector $u \in \mathbb{R}^d$, define
$$F(x) = \|x - u\|^2.$$
   Is $F(x)$ a convex function of x? Justify your answer.

8. Let $p = (p_1, p_2, \ldots, p_m)$ be a probability distribution over m possible outcomes. The *entropy* of p is a measure of how much randomness there is in the outcome. It is defined as
$$F(p) = -\sum_{i=1}^{m} p_i \ln p_i,$$
   where ln denotes natural logarithm. Show that this is a concave function.

Programming problems

9. *Coordinate descent.* In this problem we consider a standard unconstrained optimization problem:
$$\min L(w)$$
   where $L(\cdot)$ is some cost function and $w \in \mathbb{R}^d$. In class, we looked at several approaches to solving such problems—such as gradient descent and stochastic gradient descent—under differentiability conditions on $L(w)$. We will now look at a different, and in many ways simpler, approach:

   - Initialize w somehow.
   - Repeat: pick a coordinate $i \in \{1, 2, \ldots, d\}$, and update the value of w_i so as to reduce the loss.

   Two questions need to be answered in order to fully specify the updates:

   (i) Which coordinate to choose?
   (ii) How to set the new value of w_i?

   Think about these issues and thereby flesh out a coordinate descent method. For (i), you could simply pick a coordinate at random, or do something more adaptive: it is up to you. For (ii), you should try to set w_i so as to get a reasonable improvement in loss, if possible.

   Then implement and test your algorithm on a *logistic regression* problem, using the `heart disease` data set from last week. Your answer should include the following elements.

(a) *A short, high-level description of your coordinate descent method.*

In particular, you should give a concise description of how you solve problems (i) and (ii) above. Do you need the function $L(\cdot)$ to be differentiable, or does it work with any loss function?

(b) *Experimental results.*

- Begin by running a standard logistic regression solver (e.g., from `scikit-learn`) on the training set. It should not be regularized: if the solver forces you to do this, just set the regularization constant suitably to make it irrelevant. Make note of the final loss L^*.
- Then, implement your coordinate descent method and run it on this data.
- Produce a clearly-labeled graph that shows how the loss of your algorithm's current iterate—that is, $L(w_t)$—decreases with t; it should asymptote to L^*.