CS 391L: Machine Learning	Summer 2023

Homework 2 - Programming

Lecture: Prof. Adam Klivans

Keywords: SGD, Boosting

1. [**30 points**] In this problem, we will compare the performance of three different types of algorithms on a synthetic training set. All three algorithms will attempt to learn the a true vector w^* that has generated a noisy dataset. First, though, you will need to generate this dataset. Follow the steps below for the data generation:

   (a) Pick a weight vector $w^* \in \mathbb{R}^{10}$ whose Euclidean norm equals 1 in the following manner:

      i. Generate w^* such that each element of w^* is distributed standard normal, $N(0, 1)$.

      ii. Normalize w^* by $\|w^*\|$ so that it's Euclidean norm is equal to 1.
      Note: For reference, this is the equation for the Euclidian norm of a vector $\boldsymbol{\alpha}$:
      $\|\boldsymbol{\alpha}\| = \sqrt{\sum_i \alpha_i^2}$.

   (b) Next, generate a training set of size m of the form $\{(x^1, y^1), \ldots, (x^m, y^m)\}$ where each $x^i \in \mathbb{R}^{10}$ is a random vector and each $y^i \in \{0, 1\}$ is a label derived from a function of x^i:

      i. To generate the x^is use the same procedure as in 1(a)i, but do not normalize. Do this m times to produce $\{x^1 \ldots, x^m\}$

      ii. For the y^is, you will generate their binary values at random using the following function:
      $$\text{GENLABEL}(x^i) = \begin{cases} 1 & X \leq \sigma(w^{*\intercal}x^i), \quad X \sim \text{UNIFORM}(0, 1) \\ 0 & \text{otherwise} \end{cases}$$
      Where σ is the sigmoid function and $\text{UNIFORM}(0,1)$ is the uniform distribution between 0 and 1. Informally, GENLABEL simply returns a 1 with probability $\sigma(w^{*\intercal}x^i)$ and a 0 otherwise.

   With the data generation complete, the goal is to learn $w*$ using three different algorthms:

   - Algorithm 1 is logistic regression (you may use built-in methods for this).

   - Algorithm 2 is gradient descent where you train a model of the form $\sigma(w' \cdot x)$ (with parameter w') with respect to square loss, i.e. the loss function is $\frac{1}{2}(\sigma(w' \cdot x) - y)^2$, averaged over the points in the training set (code this up yourself, including calculating the gradient).

   - Algorithm 3 is stochastic gradient descent again with respect to square loss, where during each iteration we use the gradient at one random point from the training set.

   You should measure success as follows: compute $\|w^* - w'\|$ where w' is the weight vector output by your algorithm after training. For each value of m, do the following several times (say 10 times) and take the average:

   (a) Generate a fresh w^* and fresh training data using the procedures outlined above

   (b) Train your algorithm and obtain w'

(c) Calculate $\|w - w'\|$

Plot the results for all three algorithms for $m = 50, 100, 150, 200, 250$. For each algorithm, also record the time taken for the entire experiment.

2. [**10 points**] In this problem we will see a toy example of how to use AdaBoost. Please read the documentation on AdaBoostClassifier in sklearn, including how to set the base weak learner (i.e. the `base_estimator`). We will use the Wisconsin breast cancer data set again. For each depth in $1, \ldots, 5$, instantiate an AdaBoost classifier with the base learner set to be a decision tree of that depth, and then record the 10-fold cross-validated error on the entire breast cancer data set. Plot the resulting curve of accuracy against base classifier depth.