Homework 2

Christopher Mertin
CS6966: Theory of Machine Learning

March 8, 2017

6. (Convexity basics) For this problem, let f be a convex function defined over a convex set K, and suppose the diameter of K is 1.

   (a) Let $x \in K$, and suppose $f(x) = 2$ and $||\nabla f(x)|| = 1$. Give a lower bound on $\min_{z \in K} f(z)$.

   Solution:

   (b) Let x^* be the minimizer of f over K (suppose it is unique), and let x be any other point. The intuition behind gradient descent is that the fector: $-\nabla f(x)$ points *towards* x^*. Prove that this is indeed true, in the sense that $\langle \nabla f(x), x - x^* \rangle > 0$ (*i.e.*, the negative gradient makes an acute angle with the line to the optimum).

   Solution:

   (c) Suppose now that the fucntion f is *strictly convex*, *i.e.*, $f(\lambda x + (1 - \lambda)y) < \lambda f(x) + (1 - \lambda) f(y)$ (strictly), for all $x \neq y$, and $0 < \lambda < 1$.

   Prove that all the *maximizers* of f over K lie on the boundary of K.

   [*Hint:* You may want to use the definition that a point x is not on the boundary iff there exists points $y, z \in K$ such that $x = (y + z)/2$]

   Solution:

7. (Gradient Descent Basics)

   (a) Give an example of a function defined over $\mathbb{R}$, for which for *any* step-size $\eta > 0$ (no matter how small), gradient descent with step size η oscillates around the optimum point (*i.e.,* never gets to distance $< \eta/4$ to it), for some starting point $x \in \mathbb{R}$.

   Solution:

   (b) Consider the function $f(x, y) = x^2 + y^2/4$, and suppose we run gradient descent with starting point $(1, 1)$, and $\eta = 1/4$. Do we get arbitrarily close to the minimum? Experimentally, find the *threshold* for η, beyond which gradient descent starts to oscillate.

   Solution:

(c) Why is the behavior similar to that in part (a) (oscillation for *every* η) not happening in part (b)?

Solution:

8. (Stochastic Gradient Descent) Suppose we have points $\{(a_1, b_1), (a_2, b_2), \dots, (a_n, b_n)\}$ in the plane, and suppose that $|a_i| \leq 1$, and $|b_i| \leq 1$ for all i. Let $f(x, y) = \frac{1}{n} \sum_{i=1}^{n} f_i(x, y)$, where $f_i(x, y) = (x - a_i)^2 + (y - b_i)^2$.

(a) What is the point (x, y) that minimizes $f(x, y)$?

Solution:

(b) Suppose we perform gradient descent (on f) with step size $0 < \eta < 1$. Give a geometric interpretation for one iteration.

Solution:

(c) Now suppose we perform stochastic gradient descent with fixed step size $0 < \eta < 1$, and by picking i at random in $\{1, 2, \dots, n\}$, and moving along the gradient of f_i (as in SGD seen in class). After T steps, for T large enough, can we say that we get arbitrarily close to the optimum? (Provide a clear explanation)

[*Hint:* Remeber η is fixed]

Solution:

(d) Pick $n = 100$ random points in $[-1, 1]^2$ (uniformly), and run SGD for fixed $\eta = 1/2$, as above. Write down what the distance to optimum is, after $T = \{10, 100, 1000\}$ iterations (if you want to be careful, you should average over 5 random choices for the initialization). Now consider dropping the step size $\eta_t = 1/t$, and write down the result for T as above.

Solution:

9. (Numeric accuracy in MW updates) Consider the randomized experts setting we saw in class (we maintain a distribution over experts at each time, and the loss of the algorithm at that time is the expected loss over the distribution). Consider the simple setting where the experts predict 0/1, and the loss is either 0 or 1 for each expert. We saw how to update the probabilities (multiply by $e^{-\eta}$ if an expert makes a mistake, keep unchanged otherwise, and renormalize). One of the common problems here is that numeric errors in such computations tend to compound if not done carefully.

Suppose we have N experts, and we start with a uniform distribution overa ll of them. Let $p_t^{(i)}$ denote the probability of expert i at time t, for the "True" (infinite precision) multiplicative weight algorithm, and let $q_t^{(i)}$ denote the probabilities that the "real life" algorithm uses (due to precision limitations).

(a) Consider one simple suggestion: Say we zero out weights that are "too small," specifically, suppose we set $q_t^{(i)} = 0$ if $q_t^{(i)} / \max_j q_t^{(j)} < \epsilon$, for some precision parameter ϵ (such changes frequently occure due to roundoff). Other than this, suppose that the $q_t^{(i)}$ are updated accurately. Prove that in this case, we cannot hope to achieve any non-trivial regret bound. Specifically, for large enough T, the algorithm can have error $T(1-\mathcal{O}(1))$, while the best expert may have error $\mathcal{O}(T)$.

[*Hint:* In this case, we are "losing" all information about an expert]

Solution:

(b) A simple way to avoid this (in this setting) is to avoid storing probabilities, but instead maintaining only the number of mistakes $m_t^{(i)}$. Prove how this suffices to recover the probabilities $p_t^{(i)}$ (assuming infinite precision arithmetic).

Solution:

(c) Suppse we use the idea in part (b) to construct a distribution q_t that differs from p_t by $< \epsilon$ in the ℓ_1 norm, *i.e.*, $\sum_i \left|p_t^{(i)} - q_t^{(i)}\right| < \epsilon$. Then, assuming we construct such a q_t at time t to sample, show that the expected number of mistakes of the algorithm is bounded by $(1+\eta)\min_i m_T^{(i)} + \mathcal{O}(\log(N)/\eta) + \epsilon T$.

Solution:

(d) The bound above is not great if there is an expert who makes very small number of mistakes (compared to T). Noting that we are dealing with binary predictions, can you come up with a way to run the algorithm, so as to obtain a mistake bound of $(1+\eta+2\epsilon)\min_i m_T^{(i)} + \mathcal{O}(\log(N)/\eta)$?

Solution: