IST Austria: Statistical Machine Learning 2018/19
Christoph Lampert <chl@ist.ac.at>
TAs: Nikola Konstantinov <nkonstan@ist.ac.at>,
Mary Phuong <bphuong@ist.ac.at>
Amelie Royer <aroyer@ist.ac.at>
Exercise Sheet 3/6 (due date 29/10/2018)

1 Robustness of the Perceptron

Remember Perceptron training of Lecture 1 (deterministic with samples in fixed order). Look at the dataset with the following three points:

$$\mathcal{D} = \{\ (\begin{pmatrix}2\\1\end{pmatrix}, +1), (\begin{pmatrix}-1\\-2\end{pmatrix}, -1), (\begin{pmatrix}a\\b\end{pmatrix}, +1)\} \subset \mathbb{R}^2 \times \{\pm 1\}.$$

- For any $0 < \rho \leq 1$, find values for a and b such that the Perceptron algorithm converges to a *correct* classifier with *robustness* ρ.
- What's the maximal robustness you can achieve for any choice of a and b?

2 Class Prior Shift

Assume a binary classification setting, $y \in \{-1, 1\}$. Somebody gives you the weight vector, w, and bias term b, of a logistic regression model

$$p_{LR}(y|x; w, b) = \frac{1}{1 + e^{-y(\langle w,x\rangle + b)}},$$

which was trained for an underlying probability distribution $p(x, y)$ that has $p(y = 0) = p(y = 1) = \frac{1}{2}$.

- Derive a logistic regression model, q_{LR}, for a distribution $q(x, y)$ that fulfills $q(x|y) = p(x|y)$, but $q(y = -1) = \frac{1}{3}$, $q(y = 1) = \frac{2}{3}$.
- For any $a \in (0, 1)$, derive a logistic regression model, $q_{LR;a}$ for the same situation as above but with $q(y = -1) = a$, $q(y = 1) = 1 - a$.
- What are the optimal decision functions for p_{LR}, q_{LR}, and $q_{LR;a}$ with 0/1-loss?

3 Hard-Margin SVM Dual

Compute the dual optimization problem to the hard-margin SVM training problem:

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad y^i(\langle w, x^i\rangle + b) \geq 1, \qquad \text{for } i = 1, \ldots, n.$$

(Hint: it should be a quadratic objective function with linear constraints.)

4 Perceptron Training as (Convex) Optimization

The following form of Perceptron training can be interpreted as optimizing a convex, but non-differentiable, objective function by the stochastic subgradient method. What is the objective? What is the stepsize rule? Discuss advantages and shortcomings of this interpretation.

Algorithm 1 Randomized Perceptron Training

input linearly separable training set $\mathcal{D} = \{(x^1, y^1), \dots, (x^n, y^n)\} \subset \mathbb{R}^d \times \{\pm 1\}$

$w_1 \leftarrow 0$
for $t = 1, \dots, T$ **do**
  $(x, y) \leftarrow$ random example from $\mathcal{D}$
  if $y\langle w_t, x\rangle \leq 0$ **then**
    $w_{t+1} \leftarrow w_t + yx$
  else
    $w_{t+1} \leftarrow w_t$
  end if
end for

output w_{T+1}

5 Missing Proofs

- Let $f_1, \dots, f_K$ be differentiable at w_0 and let $f(w) = \max\{f_1(w), \dots, f_K(w)\}$. Let k be any index with $f_k(w_0) = f(w_0)$. Show that any v that is a subgradient of f_k at w_0 is also a subgradient of f at w_0.

- Let f be a convex function and denote by w^* a (global) minimum of f. Let $w_{t+1} = w_t - \eta_t v$, where v is a subgradient of the f at w_t.

  Show: there exists a stepsize η_t such that $\|w_{t+1} - w^*\| < \|w_t - w^*\|$, except if w_t is a minimum already.

- In your above proof, w^* can be *any* minimum of f. Let w_1^* and w_2^* be two different minima, then w_t will approach both of them. Isn't this impossible?

  Note: this is not a trivial question: convex functions *can* have multiple global minima, e.g. $f(w) = 0$ has infinitely many.

- Let $g(\alpha) = \max_{\theta \in \Theta} \left[f(\theta) + \sum_{i=1}^{k} \alpha_i g_i(\theta) \right]$ be the dual function of an optimization problem.

  Show: g is always a convex function w.r.t. α, even if the original optimization problem was not convex.

6 Practical Experiments V

- Implement a *linear support vector machine (SVM)* with training by the subgradient method.

- What error rates do both methods achieve on the datasets from sheet 1?

- For the *wine* data, make a plot of the SVM's objective values and the Euclidean distance to the optimium (after you computed it in an earlier run) after each iteration.