CS 189: Homework 4

William Guss
26793499
wguss@berkeley.edu

April 1, 2016

1. **Ridge Regression**

   (a) Let $J(w, \alpha)$ be a loss function so that

   $$J(w, \alpha) = (Xw + \alpha 1 - y)^T (Xw + \alpha 1 - y) + \lambda w^T w. \tag{1}$$

   The maximizing w, α are derived as follows. The gradient with respect to w is calculated; assuming that J is convex in w, this will give a maximum. A proof is given at the end ofthe assignment.

   $$\begin{aligned} \nabla_w J &= \nabla_w (Xw + \alpha 1 - y)^T (Xw + \alpha 1 - y) + \nabla_w \lambda w^T w \\ &= 2\lambda w + 2(\nabla_w (Xw + \alpha 1 - y))^T (Xw + \alpha 1 - y) \\ &= 2\lambda w + 2X^T (Xw + \alpha 1 - y) \end{aligned}$$

   Letting this quantity be 0 by the convexity of J we get

   $$\begin{aligned} 0 &= 2\lambda w + 2X^T (Xw + \alpha 1 - y) \\ 0 &= 2\lambda w + 2X^T X w + 2\alpha X^T 1 - 2X^T y \\ 0 &= (\lambda I + X^T X) w + \alpha X^T 1 - X^T y \\ X^T y - \alpha \bar{x} &= (\lambda I + X^T X) \\ w &= (\lambda I + X^T X)^{-1} X^T y \end{aligned} \tag{2}$$

   (b) See attached iPython notebook!

2. **Logistic Regression**

   (a) In this case

   $$R(w) = 1.9883724141284103. \tag{3}$$

   (b) Nah

   (c) $w(1) = array([-3., 5.05089812, 0.68363271])$

   (d) $R(w(1)) = 0.066133848540594925$

   (e) Nah

(f) $w(2) = array([-4., 9.10179623, 1.36726541])$

(g) $R(w(2)) = 0.0015778803918752944$

3.

4. **Revisiting Logistic Regression**

(a) Show the following theorem:

Theorem 1. *If we define* $g(z) = \frac{\tanh z + 1}{2}$ *then*

$$g(z) = \frac{e^z - e^{-z}}{2(e^z + e^{-z})} + \frac{1}{2}. \tag{4}$$

Proof. Consider the following algebraic manipulation of $g(z)$ for any $z \in \mathbb{R}$.

$$\begin{aligned} g(z) = \frac{\tanh z + 1}{2} &= \frac{(2s(2z) - 1) + 1}{2} \\ &= \frac{2s(2z) - 1}{2} + \frac{1}{2}. \end{aligned}$$

Therefore we must only show that

$$\frac{e^z - e^{-z}}{2(e^z + e^{-z})} = \frac{2\frac{1}{1+e^{-2z}} - 1}{2}. \tag{5}$$

Observe that since $e^z \neq 0$,

$$\begin{aligned} \frac{2s(2z) - 1}{2} &= \frac{1 - e^{-2z}}{2(1 + e^{-2z})} \\ &= \frac{e^z - e^{-z}}{2(e^z + e^{-z})} \end{aligned}$$

and so we know that,

$$g(z) = \frac{e^z - e^{-z}}{2(e^z + e^{-z})} + \frac{1}{2}. \tag{6}$$

□

(b) We calculate $g'(z)$ as follows,

$$g'(z) = \frac{1}{2} \tanh' z = \frac{1}{2}\left(1 - \tanh^2 z\right)$$

by the definition of $s'(z)$.

(c) Let $J(w) = \sum_{i=1}^{n} y_i \ln(g(X_i \cdot w)) + (1 - y_i)\ln(1 - g(X_i \cdot w))$. We derive the batch gradient descent learning rule as follows.

$$
\begin{aligned}
\nabla J(w) &= \sum_{i=1}^{n} y_i \nabla \ln(g(X_i \cdot w)) + \nabla(1 - y_i) \ln(1 - g(X_i \cdot w)) \\
&= \sum_{i=1}^{n} \frac{y_i}{g(X_i \cdot w)} \nabla g(X_i \cdot w) - \frac{1 - y_i}{1 - g(X_i \cdot w)} \nabla g(X_i \cdot w) \\
&= \sum_{i=1}^{n} \frac{y_i}{g(X_i \cdot w)} g(X_i \cdot w)(1 - g(X_i \cdot w)) \nabla X_i \cdot w - \\
&\qquad \frac{1 - y_i}{1 - g(X_i \cdot w)} g(X_i \cdot w)(1 - g(X_i \cdot w)) \nabla X_i \cdot w \\
&= \sum_{i=1}^{n} \left(\frac{y_i}{g(X_i \cdot w)} - \frac{1 - y_i}{1 - g(X_i \cdot w)} \right) g(X_i \cdot w)(1 - g(X_i \cdot w)) \nabla X_i \cdot w \\
&= \sum_{i=1}^{n} \left(\frac{y_i}{g(X_i \cdot w)} - \frac{1 - y_i}{1 - g(X_i \cdot w)} \right) g(X_i \cdot w)(1 - g(X_i \cdot w)) X_i \\
&= \sum_{i=1}^{n} \left(\frac{y_i(1 - g(X_i \cdot w)) - (1 - y_i) g(X_i \cdot w)}{g(X_i \cdot w)(1 - g(X_i \cdot w))} \right) g(X_i \cdot w)(1 - g(X_i \cdot w)) X_i \\
&= \sum_{i=1}^{n} \left(y_i(1 - g(X_i \cdot w)) - (1 - y_i) g(X_i \cdot w) \right) X_i \\
&= \sum_{i=1}^{n} \left(y_i - y_i g(X_i \cdot w) - g(X_i \cdot w) + y_i g(X_i \cdot w) \right) X_i \\
&= \sum_{i=1}^{n} \left(y_i - g(X_i \cdot w) \right) X_i
\end{aligned}
$$

Therefore for every epoch perform the following weight update rule.

$$
w \leftarrow w - \lambda \sum_{i=1}^{n} \left(y_i - g(X_i \cdot w) \right) X_i. \tag{7}
$$

It is important to note that this loss function although rooted in a probabilistic model, has some other quantitatively adventageous properties. For example, if square loss function were used, the Δw term would be limited, the gradient would vanish as $w \cdot X_i$ approached large negative or positive values, which in some cases may not be adventageous. This avoids such vanishing gradient problems in a manner that again is rooted in the probabilistic model.

5. **Daniel's Email Problem.** For this problem we will denote $X \subset \mathbb{R}^n$ as the manifold to which the feature vectors for all probable email scenarios are close. Let x_t denote the component of some $x \in X$ such that x_t describes Daniel's time since midnight feature.

   From the problem it is observed that the restriction of X to the dimension of x_t is essentially dense in $S = [0, \epsilon_1) \sqcup (\epsilon_2, 24]$ for x in the spam class. Clearly in the dimension of x_t, this is not a linearly separable problem. One might consider a quadratic kernel allowing the classification of S^C inside of the maximal boundary of S with respect to the origin centered norm ordering of $\mathbb{R}$, (take for example (ϵ_1, ϵ_2)). However a simpler solution arises by changing the feature itself.

Let $d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be the normal euclidean distance on $\mathbb{R}$. Then defining the feature x_t so that

$$x_t = \min\{d(x, 0), d(x, 24)\}. \tag{8}$$

This function then implies that the restriction of X to the dimension of x_t has a roughly contiguous interval of spam (positive) samples, ie. $[0, \max\{\epsilon_1, \epsilon_2\}$. This interval is of course linearly separable with respect to $[0, 24]$, and therefore we can use a Linear SVM.