Exercises 2: Online learning

1 Stochastic gradient descent: background

In the previous set of exercises, we learned about gradient descent. To review: suppose we have a loss function $l(\beta)$ that we want to minimize, like a negative log likelihood or negative log posterior density. In gradient descent, we start with an initial guess $\beta^{(0)}$ and then repeatedly take steps in a downhill direction:

$$\beta^{(t+1)} = \beta^{(t)} - \gamma^{(t)} g^{(t)} \,,$$

where $g^{(t+1)} = \nabla l(\beta^{(t)})$ is the gradient of the loss function evaluated at the previous iterate, which points locally uphill, and where $\gamma^{(t)}$ is a scalar step size (which might or might not actually depend on the iteration number t).

To calculate the gradient, we need to sum over the contributions from all n data points. But what if instead $g^{(t)}$ were not the actual gradient of the loss function, but merely an approximation to that gradient? In this context, by an approximation, we mean that the (negative) step direction $g^{(t)}$ is a random variable that satisfies

$$E(g^{(t)}) = \nabla l(\beta^{(t)}) \,.$$

Thus $g^{(t)}$ is an unbiased estimate of the gradient, but has some error. If you used such a random $g^{(t)}$ in your update instead of the actual gradient, some individual steps would lead you astray, but each step would take you in the right direction, on average. This is called *stochastic gradient descent,* or SGD.

Does SGD actually converge to the minimum of $l(x)$? It's easy to convince yourself that, if your step sizes $\gamma^{(t)}$ were constant, then you'd never get to the minimum. You might get close, but the randomness in the search directions would have you perpetually bouncing around the minimum like a moth around a flame. It follows that, if you ever hope to get to the minimum, a necessary condition is that your step sizes $\gamma^{(t)}$ get smaller, at least on average.

So assuming you handle the step sizes properly—a big if, as we'll see—here are two follow-up questions.

1. How random can these $g^{(t)}$'s be and still end up getting us to minimum of $l(\beta)$?

2. Why on earth would we want to inject randomness into the gradient-descent direction in the first place?

The answers are: 1) pretty darn random, and 2) so that we only ever have to touch one data point at a time! Let's explore.

2 SGD for logistic regression

(A) Let $l(\beta)$ be the negative log likelihood associated with the logistic regression model, which was a sum over n terms (one term for each data point). Earlier you derived the gradient of $l(\beta)$. If you haven't already, show that this gradient can be written in the form

$$\begin{aligned}\nabla l(\beta) &= \sum_{i=1}^{n} g_i(\beta) \\ g_i(\beta) &= (\hat{y}_i - y_i)x_i \\ \hat{y}_i &= E(y_i \mid \beta) = m_i \cdot w_i(\beta) = m_i \cdot \frac{1}{1+\exp(-x_i^T\beta)}\,.\end{aligned}$$

If $y_i \sim \text{Binomial}(m_i, w_i(\boldsymbol{\beta}))$, then:

$$p(y_i \mid w_i(\boldsymbol{\beta}), m_i) = \frac{m_i!}{y_i!(m_i-y_i)!} w_i(\boldsymbol{\beta})^{y_i}(1-w_i(\boldsymbol{\beta}))^{m_i-y_i}\,.$$

So:

$$\begin{aligned}l(\boldsymbol{\beta}) &= -\log\left\{\prod_{i=1}^{N} p(y_i \mid \boldsymbol{\beta})\right\} \\ &= -\sum_{i=1}^{N} \log\left\{p(y_i \mid \boldsymbol{\beta})\right\} \\ &= -\sum_{i=1}^{N} \log\left\{\frac{m_i!}{y_i!(m_i-y_i)!} w_i(\boldsymbol{\beta})^{y_i}(1-w_i(\boldsymbol{\beta}))^{m_i-y_i}\right\} \\ &= -\sum_{i=1}^{N} \log\left\{\frac{m_i!}{y_i!(m_i-y_i)!}\right\} - \sum_{i=1}^{N} y_i \log\left\{w_i(\boldsymbol{\beta})\right\} - \sum_{i=1}^{N}(m_i-y_i)\log\left\{1-w_i(\boldsymbol{\beta})\right\}\end{aligned}$$

Let $-\sum_{i=1}^{N} \log\left\{\frac{m_i!}{y_i!(m_i-y_i)!}\right\} = c$, then:

$$\begin{aligned}l(\boldsymbol{\beta}) &= c - \sum_{i=1}^{N} y_i \log\left\{\frac{1}{1+\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}}\right\} - \sum_{i=1}^{N}(m_i-y_i)\log\left\{1-\frac{1}{1+\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}}\right\} \\ &= c - \sum_{i=1}^{N} y_i \log\left\{\frac{1}{1+\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}}\right\} - \sum_{i=1}^{N}(m_i-y_i)\log\left\{\frac{\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}}{1+\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}}\right\} \\ &= c + \sum_{i=1}^{N} y_i \log\left\{1+\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}\right\} + \sum_{i=1}^{N}(m_i-y_i)\mathbf{x_i}^T\boldsymbol{\beta} + \sum_{i=1}^{N}(m_i-y_i)\log\left\{1+\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}\right\} \\ &= c + \sum_{i=1}^{N}(m_i-y_i)\mathbf{x_i}^T\boldsymbol{\beta} + \sum_{i=1}^{N} m_i \log\left\{1+\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}\right\}\end{aligned}$$

The gradient is:

$$\begin{aligned}
\nabla l(\boldsymbol{\beta}) &= \nabla \left\{ c + (m_i - y_i)\mathbf{x_i}^T\boldsymbol{\beta} + \sum_{i=1}^{N} m_i \log\left\{1 + \exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}\right\} \right\} \\
&= \left\{ \sum_{i=1}^{N}(m_i - y_i)\mathbf{x_i} - \sum_{i=1}^{N} m_i \frac{\exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}}{1 + \exp\{-\mathbf{x_i}^T\boldsymbol{\beta}\}}\mathbf{x_i} \right\} \qquad (1) \\
&= \left\{ \sum_{i=1}^{N}(m_i - y_i)\mathbf{x_i} - \sum_{i=1}^{N} m_i(1 - w_i(\boldsymbol{\beta}))\mathbf{x_i} \right\} \\
&= \left\{ \sum_{i=1}^{N}(m_i w_i(\boldsymbol{\beta}) - y_i)\mathbf{x_i} \right\}
\end{aligned}$$

(B) Optional but interesting. Suppose that you draw a single data point at random from your sample, giving you the pair $\{y_i, x_i\}$. If you can, show that the random vector $ng_i(\beta)$ is an unbiased estimate of $\nabla l(\beta)$:

$$E\{ng_i(\beta)\} = \nabla l(\beta)\,,$$

where the expectation is under random sampling from the set of all $\{y_i, x_i\}$ pairs. Note: when we apply SGD using this fact, we typically drop the leading term of n in front of $g_i(\beta)$ and absorb it implicitly into the step size $\gamma^{(t)}$.

For a sample of size n, the probability of sampling pair $\{y_i, x_i\}$ under simple random sampling is $\frac{1}{n}$: $p(\{y_i, x_i\}) = \frac{1}{n}$. So,

$$\begin{aligned}
E\{ng_i(\beta)\} &= E\{(n\hat{y}_i - y_i)x_i\} \\
&= nE\{(\hat{y}_i - y_i)x_i\} \\
&= nE\{\hat{y}_i x_i\} - nE\{y_i x_i\} \\
&= n\frac{1}{n}\sum_{i=1}^{n}\{\hat{y}_i x_i\} - n\frac{1}{n}\sum_{i=1}^{n}\{y_i x_i\} \\
&= \sum_{i=1}^{n}\{\hat{y}_i x_i - y_i x_i\} \\
&= \sum_{i=1}^{n}\{(\hat{y}_i - y_i)x_i\} \\
&= \sum_{i=1}^{n} g_i(\beta) \\
&= \nabla l(\beta)
\end{aligned}$$

(C) The idea here is that, instead of using the gradient calculated from all n data points to choose our step direction in gradient descent, we use the

gradient $g_i(\beta)$ calculated from a single data point, sampled randomly from the whole data set. Because this single-data-point gradient is an unbiased estimate of the full-data gradient, we move in the right direction toward the minimum, on average.

Code up stochastic gradient descent for logistic regression, in which each step takes the form

$$\beta^{(t+1)} = \beta^{(t)} - \gamma^{(t)} g_t(\beta^{(t)}) \,,$$

where $g_t(\beta)$ is the gradient contribution from single randomly sampled data point, evaluated at the current guess for β.

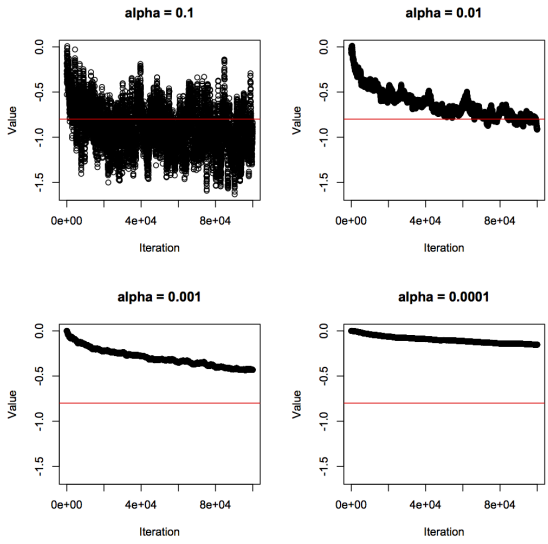

Figure 1: Trace of β_1 for Varying Step Size, α

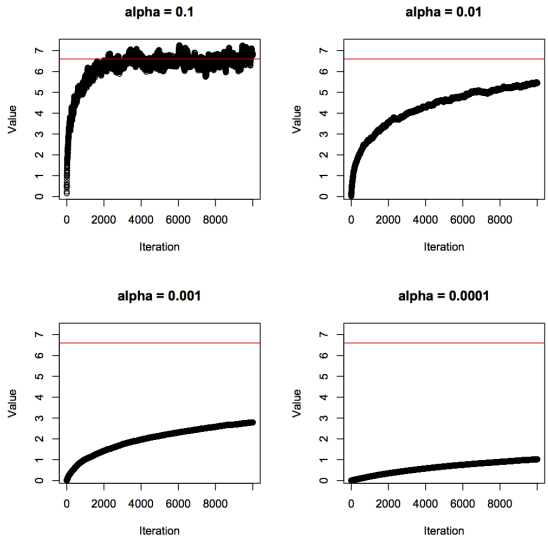

Figure 2: Trace of β_2 for Varying Step Size, α

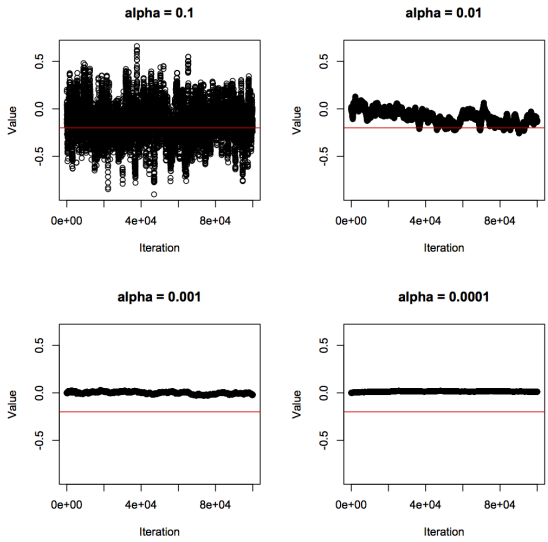

Figure 3: Trace of β_3 for Varying Step Size, α

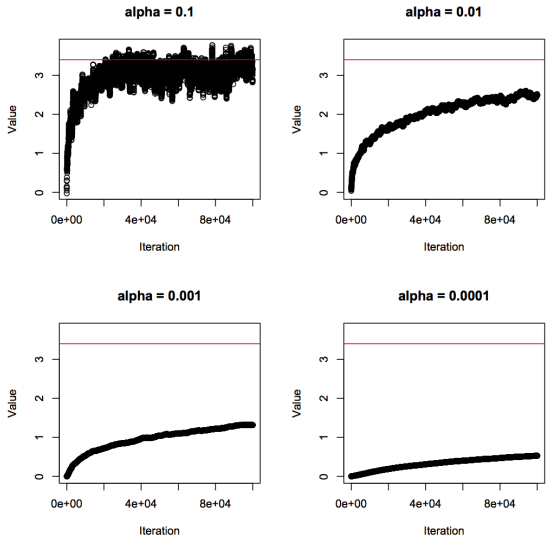

Figure 4: Trace of β_4 for Varying Step Size, α

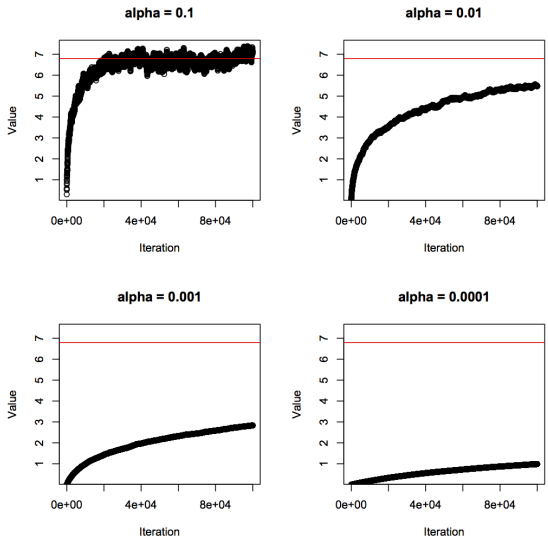

Figure 5: Trace of β_5 for Varying Step Size, α

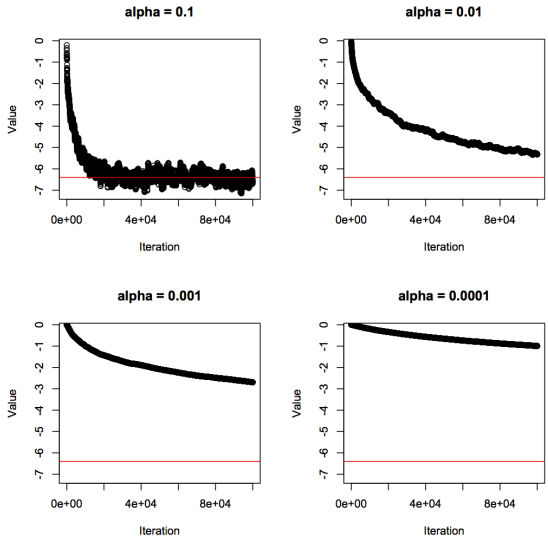

Figure 6: Trace of β_6 for Varying Step Size, αP

(D) Now try a decaying step size. Specifically, use the Robbins–Monro rule for step sizes:

$$\gamma^{(t)} = C(t + t_0)^{-\alpha},$$

where $C > 0$, $\alpha \in [0.5, 1]$, and t_0 (the "prior number of steps") are constants. The exponent α is usually called the learning rate. Clearly the closer α is to 1, the more rapidly the step sizes decay.

Implement the Robbins-Monro rule in your SGD code. Pick a smallish t_0 (1 or 2) and run with it. Fiddle around with C and α to see if you can get good performance.

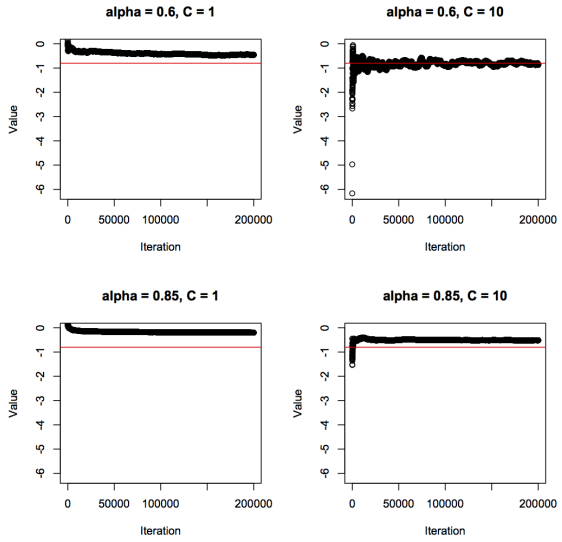

Figure 7: Trace of β_1 for Varying Step Size, α

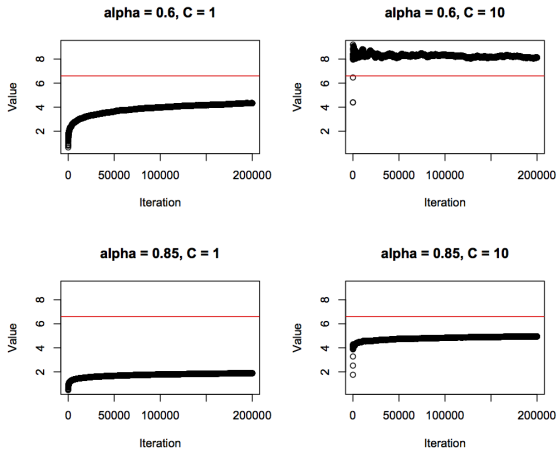

Figure 8: Trace of β_2 for Varying Step Size, α

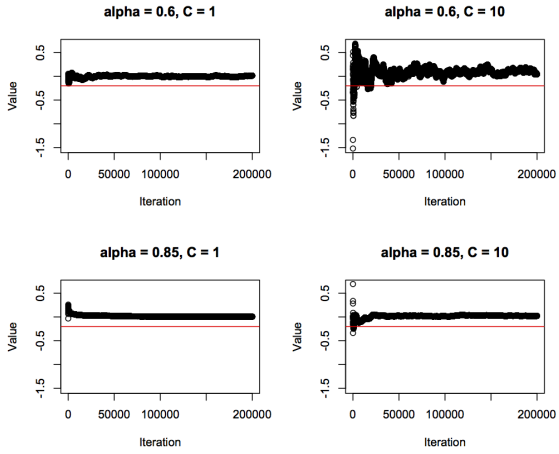

Figure 9: Trace of β_3 for Varying Step Size, α

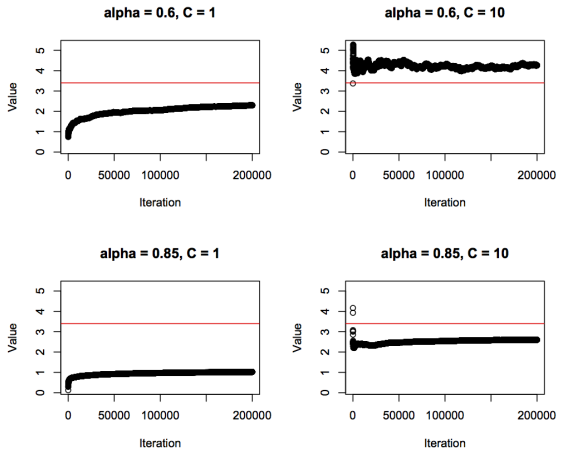

Figure 10: Trace of β_4 for Varying Step Size, α

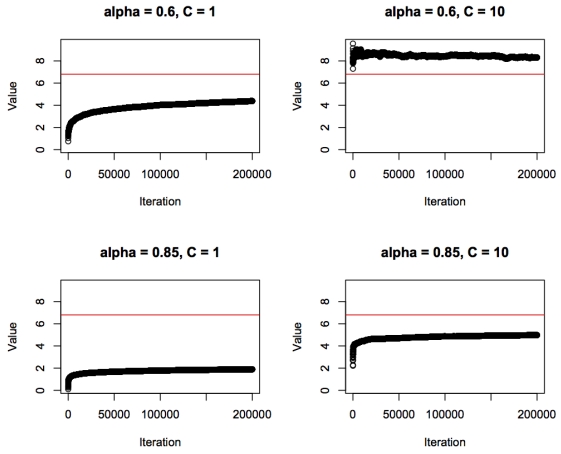

Figure 11: Trace of β_5 for Varying Step Size, α

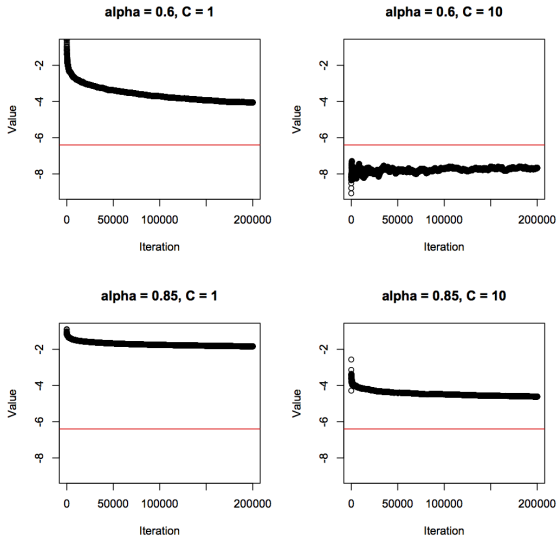

Figure 12: Trace of β_6 for Varying Step Size, αP