CSCI 5521 Homework 2

John Nguyen

November 14, 2019

For this assignment, I will use the notation $(x_i, y_i)_{i=1,\dots,n}$ instead of using the notation $(x^t, r^t)_{t=1,\dots,n}$.

When I use the log function, I mean the natural log, ln

Problem 1

(a). Projected gradient descent is gradient descent under some constraints. In this case, we apply gradient descent on f with a vector $\mathbf{w}$ under the constraint $||\mathbf{w}||_2 \leq c$.

We first computer the gradient of f, ∇f, since it is used in projected gradient descent. Since we are applying gradient descent on the $\mathbf{w}$ input, we compute $\nabla f = \frac{\partial f}{\partial \mathbf{w}}$:

$$\begin{aligned}\frac{\partial f}{\partial \mathbf{w}} &= \frac{\partial}{\partial \mathbf{w}}\left[\frac{1}{n}\sum_{i=1}^{n}\left(-y_i\mathbf{w}^T\mathbf{x}_i + \ln\left(1+\exp\left(w^T\mathbf{x}_i\right)\right)\right) + \frac{\lambda}{2}||\mathbf{w}||_2^2\right] \\ &= \frac{1}{n}\sum_{i=1}^{n}\left(-\frac{\partial}{\partial \mathbf{w}}\left[y_i\mathbf{w}^T\mathbf{x}_i\right] + \frac{\partial}{\partial \mathbf{w}}\left[\ln\left(1+\exp\left(\mathbf{w}^T\mathbf{x}_i\right)\right)\right]\right)\end{aligned}$$

First, we will evaluate $\frac{\partial}{\partial \mathbf{w}}\left[y_i\mathbf{w}^T\mathbf{x}_i\right]$. Notice that $y_i \in \{0,1\}$, so y_i is just a constant and not a vector. So, $\frac{\partial}{\partial \mathbf{w}}\left[y_i\mathbf{w}^T\mathbf{x}_i\right] = y_i\frac{\partial}{\partial \mathbf{w}}\left[\mathbf{w}^T\mathbf{x}_i\right]$. By Formula 69 from the Matrix Cookbook, $y_i\mathbf{w}^T\mathbf{x}_i = y_i\mathbf{x}_i$.
We now compute $\frac{\partial}{\partial \mathbf{w}}\left[\ln\left(1+\exp\left(\mathbf{w}^T\mathbf{x}_i\right)\right)\right]$. First, we apply the chain rule to find:

$$\frac{\partial}{\partial \mathbf{w}}\ln\left(1+\exp\left(\mathbf{w}^T\mathbf{x}_i\right)\right) = \frac{1}{1+\exp(\mathbf{w}^T\mathbf{x}_i)}\cdot\frac{\partial}{\partial \mathbf{w}}\left[1+\exp(\mathbf{w}^T\mathbf{x}_i)\right]$$

We will now evaluate $\frac{\partial}{\partial \mathbf{w}}\left[1+\exp(\mathbf{w}^T\mathbf{x}_i)\right]$. We use the the chain rule again to find:

$$\frac{\partial}{\partial \mathbf{w}}\left[1+\exp(\mathbf{w}^T\mathbf{x}_i)\right] = \left[\frac{\partial}{\partial \mathbf{w}}(1) + \frac{\partial}{\partial \mathbf{w}}\exp(\mathbf{w}^T\mathbf{x}_i)\right] = \left[(0) + \exp(\mathbf{w}^T\mathbf{x}_i)\mathbf{x}_i\right]$$

So,

$$\frac{\partial}{\partial \mathbf{w}}\left[\ln\left(1+\exp\left(\mathbf{w}^T\mathbf{x}_i\right)\right)\right] = \frac{1}{1+\exp(\mathbf{w}^T\mathbf{x}_i)}\cdot\exp(\mathbf{w}^T\mathbf{x}_i)\mathbf{x}_i = \frac{\exp(\mathbf{w}^T\mathbf{x}_i)\mathbf{x}_i}{1+\exp(\mathbf{w}^T\mathbf{x}_i)}$$

So:

$$\nabla f = \frac{\partial f}{\partial \mathbf{w}} = \frac{1}{n}\sum_{i=1}^{n}\left[-y_i\mathbf{x}_i + \frac{\exp(\mathbf{w}^T\mathbf{x}_i)\mathbf{x}_i}{1+\exp(\mathbf{w}^T\mathbf{x}_i)}\right]$$

Next, we discuss termination conditions for gradient descent. There are primarily 3 popular termination conditions: maximum iterations, loss threshold and improvement threshold. In the first termination condition, if gradient descent has iterated t times, the optimization stops and whatever value of $\mathbf{w}$ is left after t iterations will be the output of gradient descent. If we use loss threshold, then we iterate gradient descent until the loss function is sufficiently small, and then terminate. Lastly, improvement threshold is a similar idea, where when we begin to modify $\mathbf{w}$ by a sufficiently small amount, we terminate gradient descent. These are some termination conditions. Any of these termination conditions may be used in gradient descent, so below, when we mention termination condition, any one of these 3 solutions may be implemented.

Lastly, notice that we optimize $\mathbf{w}$ to find a direction for the input vector. The magnitude of the input vector does not matter, it is the direction that matters. Therefore if we have some $\mathbf{w}$, we can find an input vector with equivalently good results by calculating: $\mathbf{w}' = \frac{\mathbf{w}}{||\mathbf{w}||_2}$, where $||\mathbf{w}'|| = 1$. Then to keep with our constraint, $||\mathbf{w}||_2 \leq c$, it is sufficient to calculate: $\mathbf{w}' = c * \left(\frac{\mathbf{w}}{||\mathbf{w}||}\right)$,

Algorithm 1 *PROJECTED_GRADIENT_DESCENT*(X, y, η)

```
1: Randomly initialize $\mathbf{w}$.
2: while (Termination Condition is False) do
3:     gradient = $\nabla f(\mathbf{w})$
4:     $\mathbf{w} = \mathbf{w} - (\eta)(\text{gradient})$
5:     $\mathbf{w} = \frac{\mathbf{w}}{||\mathbf{w}||_2} * c$
6: return $\mathbf{w}$
```

Note that in the above algorithm, X and y are used to calculate $\nabla f(\mathbf{w})$. Furthermore, $f(\mathbf{w})$ and ∇f are defined above.

(b). f is strictly convex if:

$$f(\mathbf{x}) \geq f(\mathbf{y}) + (\mathbf{x} - \mathbf{y})^T \nabla f(\mathbf{y}) + \frac{\alpha}{2}||\mathbf{x} - \mathbf{y}||^2$$

for some value α. Notice that the above equation is similar to the second degree Taylor Expansion of f:

$$f(\mathbf{x}) \approx f(\mathbf{y}) + (x - y)^T \nabla f(\mathbf{y}) + (\mathbf{x} - \mathbf{y})^T H(\mathbf{x} - \mathbf{y})$$

where H is the Hessian matrix of f, i.e. the second derivative of f. Therefore, to prove the desired innequality, it is sufficient to show there exists some α such that:

$$\frac{1}{2}(\mathbf{x} - \mathbf{y})^T H(\mathbf{x} - \mathbf{y}) \geq \frac{\alpha}{2}||\mathbf{x} - \mathbf{y}||^2$$

An equivalent innequality is:

$$(\mathbf{x} - \mathbf{y})^T H(\mathbf{x} - \mathbf{y}) \geq \alpha||\mathbf{x} - \mathbf{y}||^2$$

If we apply eigenvalue decomposition of H, we obtain matrices: $H = Q\Lambda Q^{-1}$ where $Q, \Lambda \in \mathbb{R}^{d \times d}$, Λ is a diagonal matrix with the eigenvalues of H along the diagonal and Q is the matrix of eigenvectors corresponding to the eigenvalues in Λ. Without loss of generality, Λ is a diagonal matrix of the form:

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0... & 0 & \\ 0 & \lambda_2 & 0... & 0 & \\ 0 & 0 & \lambda_3... & 0 & \\ ... & ... & ... & ... & \\ 0 & 0 & 0 & ... & \lambda_d \end{bmatrix}$$

where λ_i is the eigenvalues of H and in descending order. So $\lambda_1 \geq lambda_i \geq lambda_d$ for all $1 \leq i \leq d$. Then by our decomposition, we find that H is positive definite, so:

$$(\mathbf{x}-\mathbf{y})^T H(\mathbf{x}-\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T Q\Lambda Q^{-1}(\mathbf{x}-\mathbf{y}) = \frac{\sum_{i=1}^d \lambda_i}{d}(\mathbf{x}-\mathbf{y})^T(\mathbf{x}-\mathbf{y}) = \frac{\sum_{i=1}^d \lambda_i}{d}||\mathbf{x}-\mathbf{y}||^2$$

Therefore, if we choose $\alpha = \lambda_d$, since $\lambda_d \leq \lambda_i$ for all $i = 1, 2, 3, ..., d$, we find that the desired innequality holds:

$$(\mathbf{x}-\mathbf{y})^T H(\mathbf{x}-\mathbf{y}) \geq \alpha||\mathbf{x}-\mathbf{y}||^2$$

Therefore f is λ_d-strictly convex.

(c). f is smooth if the following innequality is true:

$$f(\mathbf{x}) \leq f(\mathbf{y}) + (\mathbf{x}-\mathbf{y})^T\nabla f(\mathbf{y}) + \frac{\beta}{2}||\mathbf{x}-\mathbf{y}||^2$$

for some value of β. Notice that the right hand side is similar to the second order Taylor Series expansion of f:

$$f(\mathbf{x}) \approx f(\mathbf{y}) + (x-y)^T\nabla f(\mathbf{y}) + (\mathbf{x}-\mathbf{y})^T H(\mathbf{x}-\mathbf{y})$$

Thus the problem simplifies to the following innequality, by similar logic to Problem 1(b):

$$(\mathbf{x}-\mathbf{y})^T H(\mathbf{x}-\mathbf{y}) \leq \beta||\mathbf{x}-\mathbf{y}||^2$$

Again, since H is positive definite, we can choose $\beta = \lambda_1$, i.e. the first eigenvalue/greatest eigenvalue of H. Therefore, $\lambda_1 \geq \lambda_i$ for all $i = 1, 2, 3, ..., d$. Then:

$$(\mathbf{x}-\mathbf{y})^T H(\mathbf{x}-\mathbf{y}) = \frac{\sum_{i=1}^d \lambda_i}{d}||\mathbf{x}-\mathbf{y}||^2 \leq \lambda_1||\mathbf{x}-\mathbf{y}||^2$$

Thus f is λ_1-smooth.

(d). This solution is taken from the lecture slides. In Problem 1(b), we proved f is λ_d-strictly convex and in Problem 1(c), we proved f is λ_1-smooth. λ_1 is the first eigenvalue of the Hessian matrix of f and λ_d is the d^{th} (last) eigenvalue of the Hessian matrix f. We use the lecture slides on gradient descent (page 10) to argue that:

$$f(\mathbf{w}_T) - f(\mathbf{w}^*) \leq \frac{\lambda_1}{2}\exp\left(-\frac{4T}{\frac{\lambda_1}{\lambda_d}+1}\right)||\mathbf{w}_T - \mathbf{w}^*||^2$$

In the above formula, $\mathbf{w}_T$ is the value of $\mathbf{w}$ on the Tth iteration. $\mathbf{w}^*$ is the optimal $\mathbf{w}$ value. T is the current iteration number. Lastly, λ_1 and λ_d are defined above as the first and last eigenvalues of the hessian matrix of f.

Problem 2

(a). The Expectation Maximization (EM) Algorithm is used to study a dataset when there are unobservable latent variables. In our case, our data is $\chi = \{x^1, x^2, x^3, ..., x^n\}$, where $x^t \in \mathbb{R}^d$. We are told our data is sampled from a mixture of k multivariate Gaussians, so for any $1 \leq t \leq n$:

$$x^t \sim \sum_{h=1}^k \pi_h G^h$$

where,

$$G^h \sim N(\mu_h, \Sigma_h)$$

So in our problem, we are using EM to find the parameters $\phi = \{(\pi_h, \mu_h, \Sigma_h) | h = 1, 2, 3, ..., k\}$ such that the likelihood of sampling χ is maximized. Our latent/unobservable variables are z_i^h, for $h = 1, ..., k$. z_i^h is the membership weight of data point i in mixture k. Expectation Maximization iterates between 2 parts until convergence: the Expectation Step (E-step) and the Maximization step (M-step).

In the Expectation Step, some parameters ϕ_l are either initialized on the first iteration, or obtained from the M-step in later iterations. Let l be the current iteration. In this step, we calculate the expected value of z_i^h for $h = 1, ..., k$ and $i = 1, ..., n$ by assuming our parameters ϕ_t are correct. Using this, we find that:

$$E\left[z_i^h | x^t, \phi_l\right] = P(G_i | x^t, \phi^l)$$

After calculating these weights/soft-labels, we proceed to the M-step.

In the Maximization Step, we optimize π_h, μ_h and Σ_h for $h = 1, ..., k$ such that the probability of sampling χ is maximized. We use the calculations described in the solution to Problem 2(b) using the weights π_h calculated in the previous E-step.

(b). Let z_i^h be the soft-labels calculated in the E-Step and suppose we were on the l^{th} iteration. The following values are calculated, for all $h = 1, 2, 3, ..., k$:

$$\pi_h = \frac{\sum_{i=1}^n z_i^h}{n}$$

$$\mu_h^{l+1} = \frac{\sum_{i=1}^n z_i^h x^i}{\sum_{i=1}^n z_i^h}$$

$$\Sigma_h^i = \frac{\sum_{i=1}^n z_i^h (x^i - m_i^{l+1})(x^t - m_i^{l+1})^T}{\sum_{i=1}^n z_i^h}$$

(c). Suppose we have π_h, μ_h and Σ_h for all $h = 1, ..., k$ and let $\phi_l = \{(\phi_h, \mu_h, \Sigma_h) | h = 1, 2, ..., k\}$. We want to calculate the posterior probability $p(G_h | x^i)$ Notice that implicitly, we are assuming ϕ_l, so by our notation, $p(G_h | x^i) = p(G_h | x^i, \phi_l)$. We derive $p(G_h | x^i)$ using Baye's Rule:

$$p(G_h | x_i) = \frac{p(x^i | G_h) p(G_h)}{\sum_{j=1}^k p(x^i | G_j) p(G_j)}$$

Problem 3

Error Rates for MyLogisticReg2 on Boston50						
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	SD
0.41	0.27	0.67	0.22	0.41	0.39	0.15

Error Rates for MyLogisticReg2 on Boston75						
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	SD
0.35	0.29	0.54	0.26	0.54	0.40	0.12

Error Rates for Logistic Regression on Boston50						
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	SD
0.13	0.12	0.09	0.28	0.22	0.17	0.07

Error Rates for Logistic Regression on Boston75						
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	SD
0.10	0.08	0.13	0.11	0.05	0.09	0.03

Problem 4

(see the code submitted with this document)