Name:

COMS 4772 Homework Set 2

(1) Binary logistic regression can formulated as the following optimization problem:

$$\min_{\theta} \sum_{t=1}^{T} \log(1 + \exp(-y_t x_t^T \theta))$$

where $y_t \in \{-1, 1\}$ are class labels, x_t are feature vectors in $\mathbb{R}^n$, and $\theta \in \mathbb{R}^n$ is the vector of unknown weights. For mathematical convenience, we can define

$$\tilde{x}_t = y_t x_t,$$

multiplying the features by their corresponding labels to decrease the notational burden.

Consider a ridge regularized logistic regression problem, where we impose a 2-norm constraint on the weights vector:

$$\min_{\theta} \sum_{t=1}^{T} \log(1 + \exp(-\tilde{x}_t^T \theta)) \quad \text{s.t. } \|\theta\|_2 \leq \tau.$$

(a) Compute the dual of this problem.
Recall the Fenchel dual framework:

$$\min_x \phi(x) = \langle c, x \rangle + k(x) + h(b - Ax)$$
$$\max_z p^*(z) = \langle b, z \rangle - h^*(z) - k^*(A^* z - c).$$

We need only assign a correspondence between components of the constrained logistic regression problem, and the terms c, k, h, b, A. In particular, take

- $b = 0$
- $c = 0$
- $k(x) = \delta(x \mid \tau \mathbb{B}_2^n)$
- $A = \widetilde{X}^T$, where each column of $\widetilde{X}$ is feature $\tilde{x}_i$
- $h(y) = \sum_{t=1}^{T} \log(1 + \exp(y_i))$

A couple of comments. First, it is worthwhile to convince yourself that our objective is indeed given by $h(-A\theta)$. Second, notice how this decomposition simplifies the structure of the problem, by separating out the linear terms from the cross-entropy error. Now, we just need to compute two conjugates, and we are essentially done.

- $$k^*(z) = \sup_x \left\{ z^T x - \delta(x \mid \tau \mathbb{B}_2^n) \right\}$$
$$= \sup_{\|x\|_2 \leq \tau} \left\{ z^T x \right\} = \tau \|z\|_2.$$

- $$h^*(z) = \sup_y \left\{ z^T y - \sum_{t=1}^{T} \log(1+\exp(y_i)) \right\}$$
$$= \sum_{i=1}^{T} \sup_{y_i} \left\{ z_i y_i - \log(1+\exp(y_i)) \right\}.$$

  Now, we just compute the conjugate of the scalar function $\log(1+\exp(y_i))$. Taking the derivative of the expression

  $$zy - \log(1+\exp(y))$$

  with respect to y and setting it to 0, we find $z = \frac{\exp(y)}{1+\exp(y)} = \frac{1}{1+\exp(-y)}$, which is our old friend the sigmoid. This tells us that z must be in $[0,1]$, and if it is, the inverse is given by the logit function. Plugging this back in, we have

  $$\begin{aligned} f^*(z) &= z\text{logit}(z) - \log\left(1 + \frac{z}{1-z}\right) + \delta\left(z \mid [0,1]\right) \\ &= z\text{logit}(z) + \log\left(1-z\right) + \delta\left(z \mid [0,1]\right) \\ &= z\log(z) + (1-z)\log(1-z) + \delta\left(z \mid [0,1]\right). \end{aligned}$$

  Using the Fenchel formula now, the dual problem is given by

  $$\begin{aligned} &\max_z -\sum_{i=1}^{T} z\log(z) + (1-z)\log(1-z) - \tau\|X^T z\|_2 \\ &\text{s.t. } z \in [0,1]^T. \end{aligned}$$

  This problem is essentially smooth, except where $X^T z = 0$, and has a simple box constraint. Note that I included the constraint $\|x\|_2 \leq \tau$ in the primal to give you a nice function to play the role of $\kappa(x)$.

(b) What is the dimension of the dual variable? Briefly discuss the merits of the primal vs. dual formulations from the point of view of algorithmic development.
The dual variable has the dimension of the number of data points/labels. Both the primal and dual problems are smooth with simple constraints. If you have an underdetermined problem, you may prefer to solve the dual problem, since the variable dimension would be smaller.

(c) If instead of $\|\theta\|_2 \leq \tau$, we had decided to impose the constraint

$$-\mathbf{1} \leq \theta \leq \mathbf{1}$$

how does the dual change?
We replace the 2-norm constraint with an infinity norm constraint, so the only difference is in $k(x)$, and correspondingly, in $k^*(x)$. We computed the conjugate of the indicator of the infinity norm ball in class - it is the one norm. The dual becomes

$$\begin{aligned} &\max_z -\sum_{i=1}^{T} z\log(z) + (1-z)\log(1-z) - \|X^T z\|_1 \\ &\text{s.t. } z \in [0,1]^T. \end{aligned}$$

(2) Recall that the prox operator is defined by

$$\text{prox}_g(y) = \arg\min_x \frac{1}{2}\|x-y\|^2 + g(x).$$

(a) Show that

$$\text{prox}_{g^*}(y) = y - \text{prox}_g(y)$$

$$\begin{aligned}\text{prox}_{g(^*)}(y) &= \arg\min_x \frac{1}{2}\|x-y\|^2 + \sup_w \left\{w^T x - g(w)\right\} \\ &= \arg\min_x \sup_w \left\{\frac{1}{2}\|x-y\|^2 + w^T x - g(w)\right\}\end{aligned}$$

If we now minimize the expression inside the sup with respect to x, we find

$$\overline{x} - y + w = 0 \quad \Rightarrow \quad \overline{x} = y - w.$$

In words, the minimizing value $\overline{x}$ can be written as $y - w$ for any w. If we use the maximizing value $\overline{w}$, we will have

$$\overline{x} = \text{prox}_{g^*(y)} = y - \overline{w}.$$

But what is $\overline{w}$? To find out, we plug in $x(w) = y - w$ and maximize the expression inside the sup with respect to w:

$$\begin{aligned}\max_w \left\{\frac{1}{2}\|x-y\|^2 + w^T x - g(w)\right\} &= \max_w \left\{\frac{1}{2}\|(y-w)-y\|^2 + w^T(y-w) - g(w)\right\} \\ &= \max_w \left\{\frac{1}{2}\|w\|^2 + w^T(y-w) - g(w)\right\} \\ &= \max_w \left\{-\frac{1}{2}\|w\|^2 + w^T y - g(w)\right\} \\ &= \max_w \left\{-\frac{1}{2}\|w-y\|^2 - g(w)\right\} + \frac{1}{2}\|y\|^2\end{aligned}$$

The maximizing value of w is given by

$$\overline{w} = \arg\min_w \frac{1}{2}\|y-w\|^2 + g(w) = \text{prox}_g(y).$$

We just flipped the sign and ignored $\frac{1}{2}\|y\|^2$, which does not depend on w. Combining this with our characterization of $\overline{z}$ in terms of $\overline{w}$, we get the formula.

(b) Use part (a) to compute

$$\text{prox}_{\lambda\|\cdot\|_1}(y).$$

$$\begin{aligned}\text{prox}_{\lambda\|\cdot\|_1}(y) &= y - \text{prox}_{\lambda\mathbb{B}_\infty}(y) \\ &= \begin{cases} y_i - \lambda & \text{if} \quad y_i > \lambda \\ y_i - y_i = 0 & \text{if} \quad \|y_i\| \le \lambda \\ y_i - (-\lambda) = y_i + \lambda & \text{if} \quad y_i < -\lambda \end{cases}\end{aligned}$$

The prox formula make quick work of computing the soft thresholding operator!

(3) In class, we discussed iterative soft thresholding for solving the problem

$$\min_x \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1.$$

In this problem, you are going to apply this algorithm to sparse logistic regression models, and also try a famous acceleration technique of Beck & Teboulle to improve the algorithm. The problem is *sparse* binary logistic regression:

$$\min_\theta \sum_{t=1}^{T} \log(1 + \exp(-\tilde{x}_t^T\theta)) + \lambda\|\theta\|_1.$$

where as in the previous question, $\tilde{x}_t = y_t x_t$. Just as in sparse linear regression, we add the 1-norm penalty to drive many of the coefficients down to 0.

(a) Download the starting script file, and make sure you understand the problem setup.

(b) Implement a proximal splitting method for the above problem. You may use a constant step size. At every iteration, your algorithm should print a line listing the value and iteration.

To show that you implemented the algorithm, copy and paste a run over the first 10 and last 10 iterations into a verbatim environment, as shown below, say for 100 iterations:

```
iter 1
iter 2
...
iter 10
iter 91
iter 92
iter 100
```

(c) Solve the same problem with CVX, and show that your solution (as well as the value of your solution) agrees with the CVX solution, and its value.

(d) Skim the FISTA paper: `http://mechroom.technion.ac.il/~becka/papers/71654.pdf`
On page 11, find the FISTA algorithm (the fixed step size version). Implement it for the logistic regression problem, again pasting the first 10 and last 10 iterations:

```
iter 1
iter 2
...
iter 10
iter 91
iter 92
iter 100
```

(e) Make a plot, comparing per-iteration progress of the two algorithms on the same problem. The x-axis of your plot should be iteration number, and the y axis the value of the objective function. Did the acceleration... accelerate anything?

(4) Consider the problem of minimizing a smooth function subject to inequality constraints:

$$\min_x f(x) \quad \text{s.t.} \quad Cx \leq c.$$

For our purposes, it is convenient to introduce nonnegative slack variables $s \geq 0$, rewriting the problem

$$\min_{x,s} f(x) \quad \text{s.t.} \quad Cx + s = c, \quad s \geq 0.$$

The Lagrangian for this problem is given by

$$\mathcal{L}(x, s, \lambda) = f(x) + \lambda^T (Cx + s - c) + \delta(s|\mathbb{R}^n_+)$$

(a) Obtain the first-order necessary condition for a local minimum of $\mathcal{L}$ in x.
Differentiate the Lagrangian expression with respect to x to obtain

$$\nabla f(x) + C^T \lambda = 0.$$

(b) Obtain the necessary condition for a local maximum of $\mathcal{L}$ in λ.
Differentiate the Lagrangian expression with respect to λ to get

$$Cx + s - c = 0.$$

(c) Argue that at any saddle point of $\mathcal{L}$, $\bar{\lambda}_i \bar{s}_i = 0$.
A saddle point $(\bar{x}, \bar{s}, \bar{\lambda})$ means that

$$\mathcal{L}(\bar{x}, \bar{s}, \lambda) \leq \mathcal{L}(\bar{x}, \bar{s}, \bar{\lambda}) \leq \mathcal{L}(x, s, \bar{\lambda})$$

for any feasible x, s, λ. Suppose that $\bar{\lambda}_i \bar{s}_i > 0$, so that both are positive. Then in particular, if we modify s by cutting the ith slack in half, and call the resulting slack $\tilde{s}$, we have

$$C\bar{x} + \tilde{s} - c = -\bar{s}_i/2 < 0.$$

This gives us

$$\mathcal{L}(\bar{x}, \tilde{s}, \bar{\lambda}) = \mathcal{L}(\bar{x}, \bar{s}, \bar{\lambda}) - \bar{\lambda}_i \bar{s}_i/2 < \mathcal{L}(\bar{x}, \bar{s}, \bar{\lambda}).$$

which contradicts the saddle point inequality. λ should be also required to stay positive in the original formulation, so this completes the proof.

(d) Now consider a log-barrier modified primal problem:

$$\min_{x,s} f(x) - \mu \sum \log(s_i) \quad \text{s.t.} \quad Cx + s = c.$$

(e) Form the Lagrangian for this problem, and compute equations corresponding to first-order necessary conditions in all three variables x, s, λ. Compare these equations to the equations in parts (a-c).

The Lagrangian is given by

$$f(x) - \mu \sum \log(s_i) + \lambda^T (Cx + s - c).$$

The necessary conditions are

$$\begin{aligned}\nabla f(x) + C^T\lambda &= 0\\ Cx + s - c &= 0\\ -\mu + \lambda_i s_i &= 0.\end{aligned}$$

These are analogous to the previous equations, except that complementarity conditions are now relaxed to $\lambda_i s_i = \mu$ instead of 0.

(5) **Bonus**.

(a) Design a Newton method to directly solve the optimality conditions in part (e) of (4). You will be able to represent the higher order system as a 3×3 block matrix, with blocks for x, s, λ. Once you have the general form, please specify it to the case

$$\min_x \frac{1}{2}\|Ax - b\|^2 \quad \text{s.t.} -\mathbf{1} \leq x \leq \mathbf{1}.$$

We can compute the Jacobian associated to the equations

$$F(x,s,\lambda) = \begin{aligned}\nabla f(x) + C^T\lambda &= 0\\ Cx + s - c &= 0\\ -\mu + \lambda_i s_i &= 0.\end{aligned}$$

We get

$$J(x,s,\lambda) = \begin{bmatrix}\nabla^2 f(x) & 0 & C^T\\ C & I & 0\\ 0 & D(\lambda) & D(s)\end{bmatrix}$$

where $D(\lambda)$ is a diagonal matrix with elements of λ on the diagonal, and $D(s)$ is a diagonal matrix with elements of s on the diagonal. Newton's method now is just the iteration

$$\begin{bmatrix}\Delta x\\ \Delta s\\ \Delta\lambda\end{bmatrix} = -J^{-1}F.$$

followed by

$$\begin{bmatrix}x^{k+1}\\ s^{k+1}\\ \lambda^{k+1}\end{bmatrix} = \begin{bmatrix}x^k\\ s^k\\ \lambda^k\end{bmatrix} + \gamma\begin{bmatrix}\Delta x\\ \Delta s\\ \Delta\lambda\end{bmatrix}$$

where γ is chosen so neither of s or λ components go negative. For the particular problem above, we have

$$\nabla^2 f = A^T A, \quad C = \begin{bmatrix}I\\ -I\end{bmatrix}, \quad c = \mathbf{1}^{2n}.$$

(b) Implement your Newton method to solve the log-barrier regression problem for a fixed value of μ, and verify that your solution matches that of CVX. Be careful with the step length - don't let your updated s components go negative. To initialize, set all components of s and λ to 10.

To show you implemented the method, paste the iterations in a verbatim environment, and also show that you get the same value as CVX. At each iteration, output the iteration number, the value of the log-barrier objective, the value of μ, and/or the norm of the KKT system in part (e) of (4) that you are trying to drive to 0.

(c) Modify your algorithm to divide μ by 10 every other iteration. Again, paste your iterations into a verbatim environment in this document, and check that you got the same solution as CVX on the box-constrained regression problem. If so, you just implemented your first primal-dual interior point method.

(d) Write a proximal gradient method for the box-constrained regression problem, and make a plot of function value vs. iteration comparing this method to your interior point method.