CSE 512: Homework Due Wednesday, Sept. 23

1. Consider the problem

$$\underset{x}{\text{minimize}} \quad \overbrace{\underbrace{\frac{1}{2}\|Ax-b\|_2^2}_{=:f(x)}+\frac{\rho}{2}\|x\|_2^2}^{F(x)} \tag{1}$$

where $A \in \mathbb{R}^{m\times n}$ and $n > m$.

(a) Is the function $f(x) = \frac{1}{2}\|Ax-b\|_2^2$ L-smooth? Is it μ- strongly convex? If not, why? If so, what are the value of L and μ?

Ans. (0.5 pts) The Hessian is $\nabla^2 f(x) = A^TA$. Then $L = \lambda_{\max}(A^TA)$ and $\mu = \lambda_{\min}(A^TA)$. Since $n > m$, then $\mu = 0$, and the function is not strongly convex. However, since L is a finite number, the function is L -smooth.

(b) Is the function $F(x) = f(x) + \frac{\rho}{2}\|x\|_2^2$ L-smooth or μ-strongly convex? If not, why? If so, what are the values of L and μ?

Ans. (0.5 pts) The trick here is to see that

$$\nabla^2 F(x) = A^TA + \rho I$$

and furthermore

$$\lambda_{\max}(A^TA+\rho I) = \lambda_{\max}(A^TA)+\rho, \qquad \lambda_{\min}(A^TA+\rho I) = \lambda_{\min}(A^TA)+\rho.$$

Again, $L = \lambda_{\max}(A^TA+\rho I)$ is finite, so F is L-smooth, and now since $\mu = \rho > 0$, it is also strongly convex.

(c) **Exponential convergence.** Consider $b = 0$ and $\rho > 0$. Show that in this case, then gradient descent with step size $\alpha < 1/L$ on (1) converges with *exponential complexity*, e.g. for some constant c, the error $f(x^{(t)}) - f(x^*) = O(c^t)$.

Ans. (0.5 pts) It is sufficient here to cite previous responses and say that a strongly convex function has exponential convergence.

If you were to prove it, here is an example proof.

First, note that for this scenario, the optimal objective value is $F(x) = f(x) = 0$, and can be achieved by $x^* = 0$. If $b = 0$ then gradient descent repeats the iteration scheme

$$\begin{aligned} x^{(t+1)} &= x^{(t)} - \alpha A^TAx^{(t)} - \alpha\rho x^{(t)} \\ &= (I - \alpha A^TA - \alpha\rho I)x^{(t)} \\ &= (I - \alpha A^TA - \alpha\rho I)^t x^{(1)}. \end{aligned}$$

Take the eigenvalue decomposition $A^TA = U\mathbf{diag}(\lambda)U^T$, and do a change of variables $z^{(t)} := U^Tx^{(t)}$. Then the recurrence simplifies to

$$z^{(t)} = (I - \alpha\mathbf{diag}(\lambda) - \alpha\rho I)^t z^{(1)}$$

and in fact each step is requires an element-wise multiplication. So, we try to set up a contraction, since we know that $z^* = U^Tx^* = 0$ is a solution:

$$\|z^{(t)}\|_\infty \le (1 - \alpha\underbrace{\lambda_{\min}}_{=0} - \alpha\rho)^t \|z^{(1)}\|_\infty.$$

Taking in particular $\alpha < 1/L$ gives

$$\|z^{(t)}\|_\infty \le \left(1 - \frac{\rho}{L}\right)^t \|z^{(1)}\|_\infty.$$

Note also that in general, for any vector $z \in \mathbb{R}^n$, $\|z\|_2 \leq \sqrt{d}\|z\|_\infty$, and furthermore, since U is orthonormal,

$$\|z^{(t)}\|_2 = \|Ux^{(t)}\|_2 = \|x^{(t)}\|_2.$$

Finally,

$$\begin{aligned} f(x^{(t)}) - \underbrace{f^*}_{=0} &= \frac{1}{2}\|Ax^{(t)}\|_2^2 + \frac{\rho}{2}\|x^{(t)}\|_2^2 \\ &= \frac{1}{2}\|\Lambda z^{(t)}\|_2^2 + \frac{\rho}{2}\|z^{(t)}\|_2^2 \\ &\leq (\lambda_{\max} + \rho)\frac{1}{2}\|z^{(t)}\|_2^2 = O(\beta c^{(t)}) \end{aligned}$$

for

$$\beta = (\lambda_{\max} + \rho)n\|z^{(1)}\|_\infty, \qquad c = \left(1 - \frac{\rho}{L}\right).$$

(d) **Nullspace component. (2.5pts)** Now consider $A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$, $b = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, and $\rho = 0$.

i. Recall that for any vector x, from the linear decomposition theorem, we can uniquely write

$$x = u + v, \qquad u \in \mathbf{null}(A), \quad v \in \mathbf{range}(A^T).$$

We will denote the operation of pulling out the nullspace component of x as $u = \mathbf{proj}_{\mathbf{null}(A)}(x)$. Suppose that we run gradient descent for (1), starting at $x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. After t iterations, what is $\mathbf{proj}_{\mathbf{null}(A)}(x^{(t)})$? Why?

Ans. First, the term b is a red herring, and doesn't really matter. In particular, suppose that for any t, we form the decomposition $x^{(t)} = u^{(t)} + v^{(t)}$ where $u^{(t)} = \mathbf{proj}_{\mathbf{null}(A)}(x^{(t)})$. Then

$$x^{(t+1)} = x^{(t)} - \underbrace{\alpha A^T(Ax^{(t)} - b)}_{g}.$$

Then for *any* b, $\mathbf{proj}_{\mathbf{null}(A)}(g) = 0$, in particular because g is in the range of A^T. You can prove this in more detail by taking the SVD of $A = U\Sigma V^T$, and therefore

$$\mathbf{proj}_{\mathbf{null}(A)}(g) = (I - VV^T)g \overset{g=A^Tw}{=} (I - VV^T)A^Tw = (V - V)\Sigma^T U^T w = 0.$$

In other words, if $\mathbf{proj}_{\mathbf{null}(A)}(x^{(0)}) = 0$, then $\mathbf{proj}_{\mathbf{null}(A)}(x^{(t)}) = 0$ for *any* $t > 0$.

ii. Now suppose that $x^{(0)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$. After t iterations, what is $\mathbf{proj}_{\mathbf{null}(A)}(x^{(t)})$? Why?

Ans. Using the same logic as in the previous part, by linearity of projections,

$$\mathbf{proj}_{\mathbf{null}(A)}(x^{(t)}) = \mathbf{proj}_{\mathbf{null}(A)}(x^{(t-1)}) - \underbrace{\mathbf{proj}_{\mathbf{null}(A)}(\alpha A^T(Ax^{(t-1)} - b))}_{=0} = \mathbf{proj}_{\mathbf{null}(A)}(x^{(0)}).$$

Since $x^{(0)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \in \mathbf{null}(A)$, then $\mathbf{proj}_{\mathbf{null}(A)}(x^{(t)}) = x^{(0)}$ for all t.

iii. Now suppose that $\rho = 1$. For both cases of initial values, after t iterations, what is $\mathbf{proj}_{\mathbf{null}(A)}(x^{(t)})$? Why?

Ans. With regularization, the iteration scheme at each step now includes a multiplicative shrinking of $x^{(t)}$ at each step; namely,

$$x^{(t+1)} = x^{(t)} - \underbrace{\alpha A^T(Ax^{(t)} - b)}_{\in \mathbf{null}(A)} - \alpha x^{(t)}.$$

So, now, at each step,

$$\mathbf{proj}_{\mathbf{null}(A)}(x^{(t)}) = (1-\alpha)\mathbf{proj}_{\mathbf{null}(A)}(x^{(t-1)}) = (1-\alpha)^t \mathbf{proj}_{\mathbf{null}(A)}(x^{(0)}).$$

If $x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, then this nullspace component is still 0 for all t. If $x^{(0)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \in \mathbf{null}(A)$, then this component is $(1-\alpha)^t \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, which for $\alpha < 1$, shrinks at each step.

2. **Convex functions**

(a) Recall the objective function in logistic regression

$$f(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \log(\sigma(y_i x_i^T \theta)), \qquad \sigma(s) = \frac{1}{1+\exp(-s)}.$$

Show that $f(\theta)$ is convex.

Ans. (0.5 pts) Since f is twice differentiable, the easiest way to do this is to look at the Hessian and show that it is positive semidefinite.

$$\nabla^2 f(\theta) = Z^T D Z$$

where each row of Z is $y_i x_i$, and $D = \mathbf{diag}(d_1, ..., d_m)$ for $d_i = \sigma(y_i x_i^T \theta)(1 - \sigma(y_i x_i^T \theta)) \geq 0$. Then for any w, $w^T \nabla^2 f(\theta) w = \|D^{1/2} Z w\|_2^2 \geq 0$, so the Hessian is always positive semidefinite.

(b) In matrix factorization, we attempt to characterize a matrix $R \in \mathbb{R}^{m \times n}$ in terms of low-rank factors $R \approx UV^T$, where $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{n \times r}$. Take $r = 1$. (Note that $\mathbf{rank}(R) > 1$ in general.) Show that

$$f(u, v) = \frac{1}{2}\|R - uv^T\|_F^2$$

is nonconvex.

Ans. (0.5 pts) Gradient

$$\nabla_u f(u, v) = (uv^T - R)v, \qquad \nabla_v f(u, v) = (vu^T - R^T)u$$

Hessian

$$\nabla^2 f(u, v) = \begin{bmatrix} \nabla_{uu}^2 f(u, v) & \nabla_{uv}^2 f(u, v) \\ \nabla_{vu}^2 f(u, v) & \nabla_{vv}^2 f(u, v) \end{bmatrix} = \begin{bmatrix} v^T v I & 2uv^T - R \\ 2vu^T - R^T & u^T u I \end{bmatrix}$$

Without loss of generality, assume that the maximum singular value of R is 2. We can always renormalize all the terms here to make this true. Writing the singular value decomposition of R as

$$R = \sum_{i=1}^{\min\{m,n\}} \sigma_i u_i v_i^T, \qquad \sigma_1 = 2$$

we pick $u = u_2$ and $v = v_2$. Then

$$\nabla^2 f(u_2, v_2) = \begin{bmatrix} I & 2u_2 v_2^T - R \\ 2v_2 u_2^T - R^T & u_2^T u_2 I \end{bmatrix}.$$

To show that the Hessian is not positive semidefinite at this point, it suffices to find any w where $w^T \nabla^2 f(u_2, v_2) w < 0$. Pick $w = \begin{bmatrix} u_1 \\ v_1 \end{bmatrix}$. Then

$$w^T \nabla^2 f(u_2, v_2) w = u_1^T u_1 + v_1^T v_1 - 2u_1^T R v_1 = 1 + 1 - 2\sigma_1 = -2.$$

3. **Convex sets**

(a) **Norm balls.** Consider $\|x\|$ any norm. Show that the norm properties dictate that the *norm balls*, defined as

$$\mathcal{B}_r = \{x : \|x\| \leq r\}$$

are always convex sets. Show that for any $r > 0$, the complement set

$$\bar{\mathcal{B}}_r = \{x : \|x\| > r\}$$

is nonconvex.

Ans. (0.5 pts) Positive homogeneity and subadditivity give you the convexity of $\mathcal{B}_r$. Suppose that $\|x\| \leq r$ and $\|y\| \leq r$; this, $x, y \in \mathcal{B}_r$. Then for any $0 \leq \theta \leq 1$,

$$\|\theta x + (1-\theta)y\| \overset{\text{subadditivity}}{\leq} \|\theta x\| + \|(1-\theta)y\| \overset{\text{pos. homogeneity}}{=} \theta \underbrace{\|x\|}_{\leq r} + (1-\theta)\underbrace{\|y\|}_{\leq r} \leq r.$$

Thus, $\theta x + (1-\theta)y \in \mathcal{B}_r$.

To show the nonconvexity of $\bar{\mathcal{B}}_r$, it suffices to pick $x = -y$, for some $\|x\| > r$. Then $\|y\| > r$ by symmetry, but picking $\theta = 1/2$, then $\|\theta x + (1-\theta)y\| = 0 < r$ for any $r > 0$.

(b) **Level sets.** For a function $f : \mathbb{R}^d \to \mathbb{R}$, we define the *level sets* as

$$\mathcal{S}_r = \{x : f(x) \leq r\}.$$

Show that if f is convex, then all of its level sets $\mathcal{S}_r$ are convex. Is the opposite true? (If a function has only convex level sets, is it necessarily convex?)

Ans. (0.5 pts) The first part follows very similarly to the proof for convex norm balls. In particular, suppose that $x \in \mathcal{S}_r$ and $y \in \mathcal{S}_r$, which implies that $f(x) \leq r$ and $f(y) \leq r$. Then by convexity, for any $0 \leq \theta \leq 1$,

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y) \leq r$$

and therefore $\theta x + (1-\theta)y \in \mathcal{S}_r$.

The opposite is not necessarily true, as is shown by the existence of *quasi-convex functions*. In particular, consider the function $f(x) = \sqrt{|x|}$. This function has all convex level sets ($\mathcal{S}_r = [-\sqrt{r}, \sqrt{r}]$ convex intervals), but picking $x = 0$, $y = 1$, and $\theta = 1/2$, we see that

$$f(\theta x + (1-\theta)y) = f(1/2) = 1/\sqrt{2} > 1/2 = \theta f(x) + (1-\theta)f(y).$$

4. Consider the hard margin SVM

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\|\theta\|_2^2 \qquad \text{subject to} \quad y_i x_i^T \theta \geq 1, \quad i = 1, ..., m \tag{2}$$

We would like to solve (2) using projected gradient descent. However, projecting on this constraint set (a halfspace)

$$\mathcal{H} = \{\theta : y_i x_i^T \theta \geq 1, \quad \forall i\}$$

is extremely nontrivial. (One possible way to do it is to use Dykstra's projection on the intersection of convex sets, where you iteratively project on the halfspace determined by each i, and use proper scaling. However, this takes a long time to converge, so we won't go there.)

We do this by first introducing *slack variables* s_i, and rewriting (2) equivalently as

$$\begin{array}{ll} \underset{\theta \in \mathbb{R}^n, s \in \mathbb{R}^m}{\text{minimize}} & \frac{1}{2}\|\theta\|_2^2 \\ \text{subject to} & y_i x_i^T \theta = 1 + s_i, \quad i = 1, ..., m \\ & s \geq 0 \end{array} \tag{3}$$

This problem is still a bit hard to solve, because the simultaneous projection on the affine constraint $y_i x_i^T \theta = 1 + s_i$ and the halfspace constraint $s \geq 0$ is as hard as projecting on $\mathcal{H}$. So, we will use a trick by transforming a constraint

to a penalty. Specifically, we want to find a function $\phi(s)$ which is large when s violates the nonnegativity constraint, and 0 otherwise. We pick

$$\phi(s) = \frac{1}{2}\sum_{i=1}^{m}(\max\{-s_i, 0\})^2$$

and we consider the revised problem

$$\begin{array}{ll} \underset{\theta\in\mathbb{R}^n, s\in\mathbb{R}^m}{\text{minimize}} & \frac{1}{2}\|\theta\|_2^2 + \rho\phi(s) \\ \text{subject to} & y_i x_i^T\theta = 1 + s_i, \quad i = 1, ..., m. \end{array} \tag{4}$$

(a) For a fixed value of ρ, what is the gradient of the objective function of (4)? Is it L-smooth, and if so, what is the value of L?

Ans. (1 pts) The gradient of f can be written as

$$\nabla f(\theta, s) = \begin{bmatrix}\nabla_\theta f(\theta, s) \\ \nabla_s f(\theta, s)\end{bmatrix} = \begin{bmatrix}\theta \\ \rho\min\{s_i, 0\}\end{bmatrix}.$$

Since the gradient $\nabla_\theta f(\theta, s)$ doesn't depend on s, and $\nabla_s f(\theta, s)$ doesn't depend on θ, the problem is *separable*, and in fact we can consider the L-smoothness of each gradient block separately. First,

$$\|\nabla_\theta f(\theta_1, s_1) - \nabla_\theta f(\theta_2, s_2)\|_2^2 = \|\theta_1 - \theta_2\|_2^2 \le \|\theta_1 - \theta_2\|_2^2 + \|s_1 - s_2\|_2^2$$

so $\nabla_\theta f(\theta, s)$ is 1-Lipschitz continuous.

Similarly,

$$\|\nabla_s f(\theta_1, s_1) - \nabla_s f(\theta_2, s_2)\|_2^2 \le \rho^2\|s_1 - s_2\|_2^2 \le \rho^2(\|\theta_1 - \theta_2\|_2^2 + \|s_1 - s_2\|_2^2)$$

so $\nabla_s f(\theta, s)$ is ρ-Lipschitz continuous.

Overall, we get that ∇f is $\max\{1, \rho\}$-Lipschitz continuous, so f is $\max\{1, \rho\}$-smooth.

(b) The least squares solution given by most solvers is also the *least norm solution*. That is, if the linear system $A\theta = b$ is solvable (there exists at least one solution) then the command `solve(A,b)` gives the unique solution to the optimization problem

$$\underset{\theta}{\text{minimize}} \quad \|\theta\|_2 \qquad \text{subject to} \quad A\theta = b.$$

Use this information to describe the steps to computing the projection onto the feasible set in (4). That is, given any $\hat{\theta}$, $\hat{s}$, show how to return the solution to

$$\underset{\theta, s}{\text{minimize}} \quad \|\theta - \hat{\theta}\|_2^2 + \|s - \hat{s}\|_2^2 \qquad \text{subject to} \quad y_i x_i^T\theta = 1 + s_i,\ i = 1, ..., m. \tag{5}$$

Ans. (1 pts) We first rewrite some things to make this problem look more approachable. We concatenate all our variables together

$$\hat{w} = \begin{bmatrix}\hat{\theta} \\ \hat{s}\end{bmatrix}$$

and write out the linear system in a more familiar way:

$$A = \begin{bmatrix} Z & -I \end{bmatrix}, \quad b = \mathbf{1}.$$

Now (5) can be rewritten as

$$\underset{w}{\text{minimize}} \quad \|w - \hat{w}\|_2^2 \qquad \text{subject to} \quad Aw = b. \tag{6}$$

The next step is to simply use a change of variables. We define $\tilde{w} = w - \hat{w}$, and then rewrite (6) as

$$\underset{w}{\text{minimize}} \quad \|\tilde{w}\|_2^2 \qquad \text{subject to} \quad A\tilde{w} = b - A\hat{w}. \tag{7}$$

Now, (7) is just a min-norm least squares problem! We can solve this using simply `solve(A, b-np.dot(A, what))`, or `solve(A, b-A* what)`!

Then w can be recovered by taking $w = \tilde{w} + \hat{w}$, and the projected θ, s can be extracted from $w = \begin{bmatrix}\theta \\ s\end{bmatrix}$.

(c) **Coding (2pts)**

- Download `arrhythmia.zip` and take a look at the three files inside. The original data is loaded in `arrhythmia.data` which is a csv file, with 279 features followed by a class label. A description of each feature and label is given in `arrhythmia.names`. Take a look at it, and understand what it is saying.
- The file `arrhythmia.mat` contains the data from `arrhythmia.data` loaded and organized, so that you have $x_i \in \mathbb{R}^{279}$ containing the $n = 279$ features and $y_i \in \{1, 2, \cdots, 15\}$ indicating the patient's diognosis. The matrix X and vector y are packed so that

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_m^T \end{bmatrix} \in \mathbb{R}^{m \times n} \text{ and } y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^m.$$

  We want to convert this multiclass classification problem to a binary classification problem, so adjust y so that $y = 1$ means the patient is normal, and $y = -1$ means the patient is anything but normal. Comment on the class balancing; e.g., does it look like each class has enough "data representatives"?
- **Train/test split** To eliminate randomness, I have given you two index vectors: `idx_train` and `idx_test`, which randomly form a 60/40 train/test split. Later, when we discuss cross validation, we will see that you actually need to make these splits multiple times, to convince yourself that you are not overfitting. For now, we will just use these.
  Form a train and test data and label set using these indices. In python, your code should look like

```
Xtrain = X[idx_train,:]
ytrain = y[idx_train]
Xtest = X[idx_test,:]
ytest = y[idx_test]
```

  and in MATLAB,

```
Xtrain = X(idx_train,:);
ytrain = y(idx_train);
Xtest = X(idx_test,:);
ytest = y(idx_test);
```

- **Normalizing** Now we will use a more "standardized" normalization scheme, which is what is recommended if you don't really know anything special about your data. If we were medical experts, we might try something fancier, but for this exercise let's just do the simplest thing: de-meaning, followed by de-variancing.
  As before, first calculate the mean data vector over the training data

$$x_{\text{mean}} = \frac{1}{m_{\text{train}}} \sum_{i=1}^{m_{\text{train}}} x_i$$

  and remove this offset from the train and test data.
  Next, calculate the mean standard deviation over the training data:

$$x_{\text{std}}[k] = \frac{1}{m_{\text{train}}} \sqrt{\sum_{i=1}^{m_{\text{train}}} x_i[k]^2}$$

  This can be done with the built in `std` function in MATLAB or Python. Remove this multiplicatively from the train and test data, e.g.

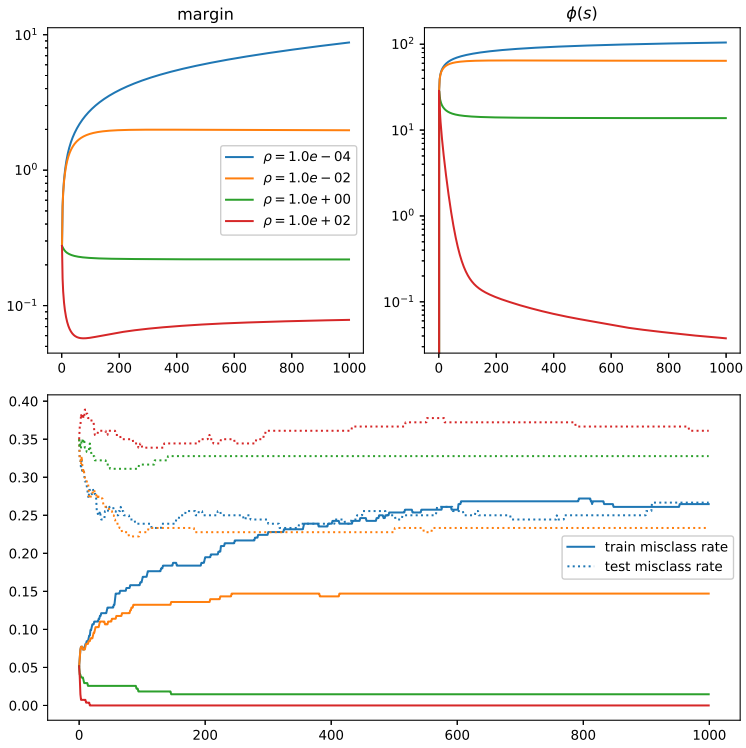
$$x_i[k] \leftarrow x_i[k]/x_{\text{std}}[k]$$

  Now we're ready to go!

i. Solve the hard margin SVM reformulation in (4), sweeping $\rho = 10^{-4}, 10^{-2}, 1., 10^2$. Use a zero initialization as before, and $1/L$ as your stepsize with L as calculated in part (a). Run for 1000 iterations.
Plot the margin value and the penalty $\phi(s)$ for the train set, and the train and test misclassificatition errors. Do this for $t = 1, ..., 1000$. Report also these final values.

Ans. Final values:

ρ	margin	$\phi(s)$	train misclass rate	test misclass rate
10^{-4}	8.761	104.703	0.265	0.267
10^{-2}	1.971	64.090	0.147	0.233
1	0.219	13.7833	0.015	0.328
10^{2}	0.078	0.038	0.0	0.361



ii. Is the data separable? Is this a reasonable approach for constrained optimization, given large enough ρ?
Ans. Yes, the data is separable. With a large enough ρ, we are able to achieve $\phi(s) = 0$, which shows that the feasible set in (7) is attainable.

iii. Comment a bit on the *test* misclassification rate for varying values of ρ.
Ans. While a large enough ρ ensures that (4) (the formulation we solve) is equivalent to (3) (the actual hard margin SVM), it seems to worsen test performance. There seems to be a sweet spot, at about $\rho = 0.01$, where both performance of train and test misclassification rate seem reasonable, but if we increase ρ too much then the test misclassification rate begins to degrade.
Actually, this is an example of overfitting. Hard margin SVM in itself is asking a bit too much of this dataset; even though the data *is* separable, by forcing it this way we are creating a hyperplane that is a bit too tuned to the train set, and not that effective on the test set. While we can use cross validation (discussed later) to pick the best value of ρ, the conclusion I would draw from this experiment is to simply use the soft margin SVM instead, which uses a *hinge penalty function* for ϕ.

Challenge!

This challenge will look at two interesting extensions of the SVM problem.

1. **Projected gradient descent** Consider the convex constrained optimization problem

$$\underset{\theta}{\text{minimize}} \quad f(\theta) \qquad \text{subject to} \quad \theta \in \mathcal{C} \tag{8}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a convex function and $\mathcal{C}$ is a convex set. The *normal cone* to $\mathcal{C}$ at a given point θ is defined as the set of points orthogonal to the tangent cone of $\mathcal{C}$ at θ; explicitly, it is

$$\mathcal{N}_{\mathcal{C}}(\theta) := \{g : g^T(\theta' - \theta) \leq 0, \quad \forall \theta' \in \mathcal{C}\}.$$

Recall that in *unconstrained* optimization, an optimality condition for convex optimization is that θ^* minimizes $f(\theta)$ if and only if $0 = \nabla f(\theta^*)$. In *constrained* optimization, the optimality condition is a bit more complicated:

$$\theta^* \text{ optimizes (8)} \iff -\nabla f(\theta^*) \in \mathcal{N}_{\mathcal{C}}(\theta^*).$$

(a) Show that if θ^* is in the interior of $\mathcal{C}$, then this reduces to the condition of $0 = \nabla f(\theta^*)$. (We say that θ is in the interior of $\mathcal{C}$ if for any vector v we can find $\epsilon > 0$ small enough such that $\theta + \epsilon v \in \mathcal{C}$.)

Ans. This follows from the observation that if θ is in the interior of $\mathcal{C}$ then $\mathcal{N}_{\mathcal{C}}(\theta) = \{0\}$. Specifically, take $\theta = \theta^* + \epsilon v$ for any $v : \|v\|_2 = 1$. Then $\theta^* \in \mathcal{N}_{\mathcal{C}}$ implies $\epsilon \nabla f(x^*)^T v = 0$ for all v. Therefore $\nabla f(x^*) = 0$.

(b) Recall the *projected gradient descent* method for solving (8) at each iteration computes

$$\theta^{(t+1)} = \mathbf{proj}_{\mathcal{C}}(\theta^{(t)} - \alpha \nabla f(\theta^{(t)}))$$

for some step size $\alpha > 0$. Show that when this method is stationary, then optimality is reached. That is, any $\alpha > 0$,

$$\theta^* = \mathbf{proj}_{\mathcal{C}}(\theta^* - \alpha \nabla f(\theta^*)) \quad \iff \quad -\nabla f(\theta^*) \in \mathcal{N}_{\mathcal{C}}(\theta^*).$$

Ans. Let's define condition (A) as $\theta^* = \mathbf{proj}_{\mathcal{C}}(\theta^* - \alpha \nabla f(\theta^*))$, and condition (B) as $-\nabla f(\theta^*) \in \mathcal{N}_{\mathcal{C}}(\theta^*)$. We wish to show that (A) $\iff$ (B).

$$\begin{aligned}
(A) \qquad &\iff && \theta^* = \underset{\theta \in \mathcal{C}}{\text{argmin}} \|\theta^* - \alpha \nabla f(\theta^*) - \theta\|_2^2 \\
&\overset{\text{argmin = it's a minimum}}{\iff} && \|\theta^* - \alpha \nabla f(\theta^*) - \theta^*\|_2^2 \leq \|\theta^* - \alpha \nabla f(\theta^*) - \theta\|_2^2, \quad \forall \theta \in \mathcal{C} \\
&\overset{\text{simplify}}{\iff} && \alpha^2 \|\nabla f(\theta^*)\|_2^2 \leq \alpha^2 \|\nabla f(\theta^*)\|_2^2 + \|\theta^* - \theta\|_2^2 - \alpha \nabla f(\theta^*)^T(\theta^* - \theta), \quad \forall \theta \in \mathcal{C} \\
&\overset{\text{simplify}}{\iff} && -\nabla f(\theta^*)^T(\theta - \theta^*) \leq \frac{\|\theta^* - \theta\|_2^2}{\alpha}, \quad \forall \theta \in \mathcal{C}.
\end{aligned}$$

Since this is true for all $\alpha > 0$, for all $\theta \in \mathcal{C}$ arbitrarily close to θ^*, then we can safety say that (A) $\iff$ (B).

2. **Kaczmarz method.** In question 4 we found a way to find a separation in our dataset by solving some projection problems at each iteration. But, what if the number of training samples were very large? Then computing the projection at each step is extremely burdensome.

Von Neumann in the 1949 figured out that a way to project on the intersection of hyperplanes is to just alternatingly project on each hyperplane. That is, if I want to find some θ where $A\theta = b$, then I could break down A and b as

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_K \end{bmatrix}, \qquad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ v_K \end{bmatrix}$$

and we simply need to find θ where

$$A_i \theta = b_i \qquad \text{for } i = 1, ..., K.$$

Then using von Neumann's approach, we would simply pick any $\theta^{(0)}$, and alternatingly do

$$\theta^{(t+1)} = \mathbf{proj}_{\mathcal{C}_i}(\theta^{(t)})$$

where $\mathcal{C}_i = \{\theta : A_i\theta = b_i\}$. Then, for i large enough, $\theta^{(i)}$ will converge to the true intersection of hyperplanes.

This principle, when applied to this data separation problem, is also sometimes known as Kaczmarz' method. Specifically, we are going to break down our data matrix so that we only operate on one sample at a time. Unlike von Neumann's approach, however, we are only going to make a few passes through the data. This doesn't ensure full feasibility, but the hypothesis is that we get something reasonable nonetheless.

(a) Describe the projection method on feasibility of a *single* data sample: e.g. given $\hat{\theta}$, how would we solve

$$\underset{\theta}{\text{minimize}} \quad \|\theta - \hat{\theta}\|_2^2 \qquad \text{subject to} \quad y_i x_i^T \theta = 1?$$

Use the linear decomposition theorem to give an *explicit* solution, e.g. using matrix-vector multiplications only, and no matrix inverses. (One scalar inverse should be used.)

Note that we are ignoring the slack variable in this approach, and forcing an equality rather than inequality.

Ans. From previous bits of the homework, we know that our goal is to solve the min-norm least squares problem

$$\underset{\tilde{\theta}}{\text{minimize}} \quad \|\tilde{\theta}\|_2^2 \qquad \text{subject to} \quad z^T\tilde{\theta} = 1 - z^T\hat{\theta}$$

for $z = \begin{bmatrix} y_i x_i \\ 1 \end{bmatrix}$.

From the linear decomposition theorem, we know that we can uniquely write $\tilde{\theta} = \tilde{u} + \tilde{v}$ where

$$\tilde{u} = \mathbf{proj}_{\mathbf{null}(z^T)}(\tilde{\theta}), \qquad \tilde{v} = \mathbf{proj}_{\mathbf{range}(z)}(\tilde{\theta}).$$

Explicitly, from previous notes, we know that in fact, if we had $\tilde{w}$, then

$$\tilde{v} = \frac{z^T\tilde{\theta}}{z^Tz}z, \qquad \tilde{u} = \tilde{w} - \tilde{v}.$$

We could plug this into our optimization, getting the equivalent reformulation:

$$\underset{\tilde{u},\,\tilde{v}}{\text{minimize}} \quad \|\tilde{u}\|_2^2 + \|\tilde{v}\|_2^2 \qquad \text{subject to} \quad \underbrace{z^T\tilde{u}}_{=0} + z^T\tilde{v} = 1 - z^T\hat{\theta}$$

which shows us that in fact, $\tilde{u}$ does not participate in the constraints at all; therefore $\tilde{u} = 0$ minimizes the objective and is attained. Now our problem reduces to

$$\underset{\tilde{v}}{\text{minimize}} \quad \|\tilde{v}\|_2^2 \qquad \text{subject to} \quad z^T\tilde{v} = 1 - z^T\hat{\theta}$$

where our desired $\tilde{\theta} = \tilde{v}$. But wait! the term $z^T\tilde{\theta}$ is constrained! Therefore

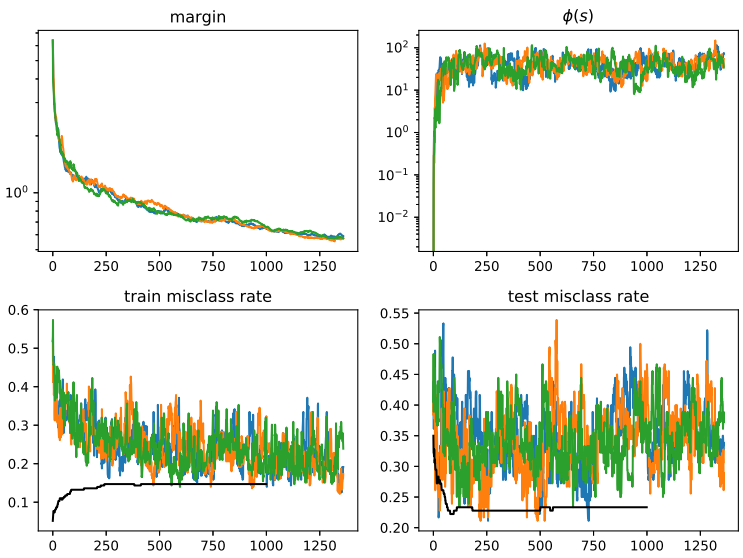
$$\tilde{\theta} = \tilde{v} = \frac{z^T\tilde{v}}{z^Tz}z = \frac{1 - z^T\hat{\theta}}{z^Tz}z.$$

Overall, this gets us

$$\theta = \frac{1 - z^T\hat{\theta}}{z^Tz}z + \hat{\theta}.$$

(b) Initialize at $\theta^{(0)} = 0$. Randomly permute the training data samples. Then, taking 5 passes through the training data (permuting the data at each pass) where for each data sample, you project the current iterate on the feasible set for each data sample. After the 5 passes, pass, plot the margin size, infeasibility ($\sum_i \max\{1 - y_i x_i^T\theta, 0\}$), and misclassification rate over the test and train set. Do this for 5 separate random permutations.

Ans.



In black is the comparison against the modified hard-margin SVM approach, using $\rho = 0.01$. (Best parameter from previous part.)

(c) Comment on what you see. Is this a viable method? How is its generalization behavior? Compare it with the behavior from the SVM approach.

Ans. The thing I want to point out here is that here, each iteration touches only *one data sample*, whereas in the previous approach, with the SVM, at each iteration we did a projection on the *entire data set*. Yet, we are clearly able to do quite a bit with the training misclassification rate, and the test misclassification rate does go down a bit, though not as well as the best case ρ in the modified hard margin SVM. Overall, I would rate this method as "not bad, but not as good as full on SVM, but much more scalable to big data."