CSE 512: Homework 3 — Due Wednesday October 16

1. **(2 pts)** *Logistic regression for Binary MNIST*

   (a) What is the gradient of the logistic loss function

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

   where $y_i \in \{-1, 1\}$? (Hint: Check out problem 1 again.)

   hw1_soln1
   cse 353 hw 5

   (b) **Coding gradient descent.** Do **not** use `scikit-learn` or other built in tools for this exercise. Please only use the packages that are already imported in the notebook. [1]

   - Open `mnist_logreg.ipynb`. We will use logistic regression to diffrentiate 4's from 9's, a notoriously tricky problem. Run the first box to see what the data looks like.
   - In the second box I have set up the problem for you by pulling out the train and test set, selecting out only the data samples related to 4's and 9's. I have not altered the data in any other way. While other normalization tricks will help you improve the accuracy, for the purposes of this exercise we will forgo them, so that it's easy to compare everyone's solutions.
   - Fill in the next box by providing code that will return the loss function value and the gradient. Make sure that everything is normalized, e.g. don't forget the $1/m$ term in the front of our loss function. Run the script. If done correctly, you should see

     `45.192, 12343.177`

   - Write a script that returns the classification accuracy given θ.
   - Use gradient descent to minimize the logistic loss for this classification problem. Use a step size of 10^{-6}.
   - **(1.5 pts)** Run for 1500 iterations. In your report, give the plot of the train loss and train/test misclassification rate, plotted as a function of *iterations*. Report the final train and test accuracy values.

   (c) **Coding stochastic gradient descent.** Do **not** use `scikit-learn` or other built in tools for this exercise. Please only use the packages that are already imported in the notebook. Now, fill in the next box a function that takes in θ and a minibatch $\mathcal{B}$ as either a list of numbers or as an np.array, and returns the *minibatch gradient*

   $$\nabla_{\mathcal{B}} f(\theta) = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \nabla f_i(\theta)$$

   where $f_i(\theta)$ is the contribution to the gradient from datapoint i:

   $$f_i = -\log(\sigma(y_i x_i^T \theta)).$$

   Run the script. If done correctly, you should see the number 5803.5 printed out.

   (d) Write a script to run stochastic gradient descent over logistic regression. When coding up the minibatching, make sure you cycle through an entire training set once before moving on to the next epoch. Additionally, use `time()` to record the runtime, and compare the performance of gradient descent and stochastic gradient descent, using a minibatch size of 50 data samples and running for 50000 iterations. Return a plot that compares the objective loss, train accuracy, and test accuracy between the two optimization methods, as a function of *runtime*. Comment on the pros and cons of the two methods.

   Important Remember that calculating the loss function and train/test accuracy requires making *full passes* through the data. If you do this at each iteration, you will not see any runtime benefit between stochastic gradient descent and gradient descent. Therefore I recommend you log these values every 10 iterations for gradient descent, and every 100 iterations for stochastic gradient descent.

[1] This is to test your understanding of the basic machine learning concepts, like calculating accuracy and logistic loss; in the future you can use whatever tools you'd like.

2. **(2 pts) Logistic regression and the Fisher Information Matrix.** Recall that the training of logistic regression models is equivalent to minimizing the following convex optimization problem

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad f(\theta) := \frac{1}{m} \sum_{i=1}^{m} \log(1 + \exp(-y_i x_i^T \theta)) \tag{LogReg}$$

for data features $x_1, ..., x_m \in \mathbb{R}^n$ and binary labels $y_1, ..., y_m \in \{-1, 1\}$.

(a) **(.5 pt)** Derive the gradient of f with respect to θ. Hint: It may help to use the sigmoid function $\sigma(s) = (1 + \exp(-s))^{-1}$.

Hint 2: It may help also to use linearity of gradients, and first just work with subproblems

$$f_i(\theta) = \log(1 + \exp(-y_i x_i^T \theta)).$$

(b) **(0.5 pt)** Derive the Hessian of f w.r.t. θ. Hint: Don't attempt this without first solving part (a) successfully.

(c) **(1 pt)** The Fisher information matrix is often used to describe the amount of information about a paramether θ that is carried by observables (x_i, y_i). Formally, it is written as

$$I(\theta) = \mathbb{E}_{x,y} \left[(\nabla_\theta \log(p_\theta(x, y))) (\nabla_\theta \log(p_\theta(x, y)))^T \, | \theta \right]$$

where $p_\theta(x, y)$ is the PDF distribution of x, y under parameter choice θ. Now consider the logistic model, where

$$p_\theta(x, y) = \sigma(y x^T \theta).$$

Show that the Fisher information matrix is equivalent to the Hessian of f in (LogReg), if x_i, y_i were drawn i.i.d. from this distribution, in the limit of infinite data ($m \to +\infty$).

Note that this implies that, for logistic models, *we can identify the Hessian using only gradient information,* which may be a significant computational benefit.

Hint: It helps to recall that $1 - \sigma(s) = \sigma(-s)$.

Hint 2: Recall the *exact* formula for an expectation of a continuous distribution. Don't try to guess this part.

3. **Generalization and linear regression. (2pts)** Consider a linear model

$$y_i = x_i^T \theta + z_i, \qquad z_i \sim \mathcal{N}(0, 1)$$

Here, $x_i \in \mathbb{R}^n$ where n is very large. After accruing m training pairs (x_i, y_i) for $i = 1, ..., m$, I form a data matrix and target vector

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_m^T \end{bmatrix}, \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}.$$

Now I will compose a *truncated* linear model by minimizing the following regression loss

$$\theta_t = \underset{\theta}{\operatorname{argmin}} \, \|X_t \theta - y\|_2^2, \qquad t \ll n$$

where X_t contains only the first t columns of X, and θ_t is a vector with length t. My new linear model will then perform inference on future data points as

$$\hat{y} = x_t^T \hat{\theta}_t$$

where $\hat{\theta}_t$ are the learned truncated parameters, and x_t contains only the first t elements of x.

Assume that the number of datapoints $m \gg n$ and each truncated datasample is linearly independent, even if t is very small. Also, assume $X_t^T X_t$ is invertible.

(a) What is the bias of a future prediction $x_t^T \theta_t$?

(b) In terms of big-O notation, what is the bias' dependence on t? (linear? quadratic? exponential?) Assume that every value in X has about the same magnitude.

(c) What is the variance of $x_t^T \theta_t$?

(d) In terms of big-O notation, what is the variance's dependence on t? (linear? quadratic? exponential?)

4. **Logistic regression fundamentals. (2pts)** Recall that in logistic regression, our goal is to minimize the following loss function

$$\underset{\theta}{\text{minimize}} \quad -\sum_{i=1}^{m} \log(\sigma(y_i x_i^T \theta)) \tag{LogLoss}$$

In the following cases, there exists an intuitive answer for what the best θ. In each case, find what that θ is using whatever method you choose (hint, draw a picture). Then, argue why it is optimal by showing that it minimizes (LogLoss) (or that it does in a limit).

(a)

i	x	y
1	$(1, 1)$	1
2	$(-1, -1)$	1

(b)

i	x	y
1	$(1, -1)$	1
2	$(1, 1)$	-1

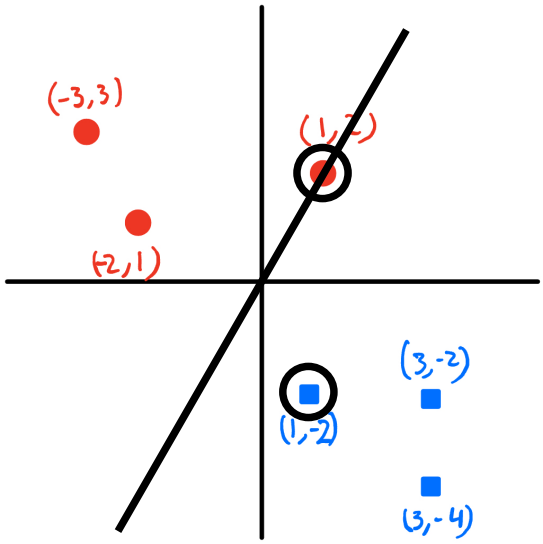
(c)

i	x	y
1	$(1, 1)$	1
2	$(1, 1)$	-1

(d)

i	x	y
1	$(1, 1)$	1
2	$(1, -1)$	-1
3	$(-1, 1)$	-1
4	$(-1, -1)$	1

5. **Margins and SVM (2pts)**



Consider the problem of separating the red circles and blue squares from above. We will use a support vector machine with no bias, e.g. the discriminator must go through the origin.

(a) Draw a discriminator that goes through the origin, perfectly separates the two sets, and visually maximizes the distance between the discriminator and any datapoint. Circle the two points closest to the discriminator, which form the support vectors.

(b) Deduce the exact values in θ, such that the red circles get labels $+1$ and the blue squares get labels -1

(c) Compute the margin distance of each point to the discriminator you proposed. You can leave your answer in radicals.

point	margin distance
(1,2)	
(-3,3)	
(-2,1)	
(3,-2)	
(1,-2)	
(3,-4)	