Large Scale Matrix Computation, Optimization and Learning (EE270) HW #3

Winter 2021 Due: **Friday, February 26, 11:59pm**

- After a genuine attempt to solve the homework problems by yourself, you are free to collaborate with your fellow students to find solutions to the homework problems. Regardless of whether you collaborate with other students, you are required to type up or write your own solutions. Copying homework solutions from another student or from existing solutions is a serious violation of the honor code. Please take advantage of the professor's and TA's office hours. We are here to help you learn, and it never hurts to ask!
- **The assignments should be submitted via Gradescope including your code attached as pdf**

1. **Logistic Regression (20 pts)**

   In this question, we will study logistic regression, which is a popular method in machine learning. We let $w \in \mathbb{R}^p$ be our decision variable. w represents weights for the columns of the $n \times p$ data matrix X, where the i^{th} row of X is x_i^T. Let

   $$\sigma(a) \triangleq \frac{1}{1+e^{-a}} = \frac{e^a}{1+e^a}$$

   be the sigmoid/logistic function. This function is non-convex. However $-\log(\sigma(a))$ is convex, which arises in maximum likelihood estimation as we describe next.

   We assume that we collect data x_i, and a binary response variable y_i, which is the label, where

   $$y_i = \begin{cases} +1 \text{ with probability } \sigma(w^T x_i) \\ -1 \text{ with probability } 1-\sigma(w^T x_i) \end{cases}. \tag{1}$$

   We can write the above in the compact form $p(y_i|w,x_i) = \sigma(y_i w^T x_i)$ since $\sigma(a) = 1 - \sigma(-a)$. If we collect the observations $\{y_i\}_{i=1}^n$, then the probability of observing this outcome is $p(y|w,X) = \prod_{i=1}^n p(y_i|w,x_i)$ and so the negative log-likelihood, which we will use as our objective function, is

   $$\ell(w) \triangleq -\log(p(y|w,X)) = \sum_{i=1}^n \log(1+e^{-y_i w^T x_i}).$$

   Note that minimizing negative log-likelihood is same as maximizing the likelihood. This corresponds to maximum likelihood estimation of the parameter w. Once w is identified, we can use (1) to infer the label of a test data point x.

   (a) Derive the gradient $\nabla \ell(w)$.

   (b) Derive the Hessian $\nabla^2 \ell(w)$.

   (c) Is the cost function $\ell(w)$ convex?

2. **Logistic Regression for Spam E-mail Classification (20 pts)**

Download the email spam data set , which is available at `https://github.com/probml/pmtk3/tree/master/data/spamData` in Matlab and plain text format. This set consists of 4601 email messages, from which 57 features have been extracted. These are as follows

- 48 features, in [0 100], giving the percentage of words in a given message which match a given word on the list. The list contains words such as "business", "free", "george", etc.
- 6 features, in [0 100], giving the percentage of characters in the email that match a given character on the list. The characters are ; ([! $ #
- Feature 55: The average length of an uninterrupted sequence of capital letters (max is 40.3, mean is 4.9)
- Feature 56: The length of the longest uninterrupted sequence of capital letters (max is 45, mean is 52.6)
- Feature 57: The sum of the lengths of uninterrupted sequence of capital letters (max is 25.6, mean is 282.2)

Load the data set using the provided link. In the Matlab version, there is a training set of size 3065 and a test set of size 1536. In the plain text version, there is no predefined testing/training set, you can randomly shuffle the data to pick a training set of size 3065 and use the rest for testing (or you can use the Matlab data along with scipy.io.loadmat).

There are different methods to pre-process the data, e.g., standardize the columns of X. For this problem, transform the features using $\log(x_{ij} + 0.1)$. One could also add some regularization to the loss function which can help generalization error but this is not necessary. Also note that you need to transform the labels from $\{0, 1\}$ to $\{1, -1\}$.

(a) Run gradient descent with a fixed step-size. Plot the value of the cost function at each iteration and find a reasonable step-size for fast convergence

(b) Repeat the previous part using gradient descent with momentum.

(c) Implement gradient descent with Armijo line search. This procedure is as follows: Assume that we are at the point x_k and have a search direction p_k (for gradient descent $p_k = -\nabla f(x_k)$). Then, the Armijo line search procedure is:

- Pick an initial step-size t
- Initialize the parameters $0 < \rho < 1$ and $0 < c < 1$ (typical values are $c = 10^{-4}$ and $\rho = 0.9$)
- While $f(x_k + tp_k) > f(x_k) + ct\nabla f(x_k)^T p_k$, do $t \leftarrow \rho t$
- Terminate the procedure if $f(x_k + tp_k) \leq f(x_k) + ct\nabla f(x_k)^T p_k$

The test statement in the while loop is the Armijo condition. If $p_k = -\nabla f(x_k)$, then the test is accepted when $f(x_k + tp_k) \leq f(x_k) - ct\|\nabla f(x_k)\|_2^2$. In general, the second term is negative as long as p_k is a descent direction. One can prove this linesearch procedure will terminate.

Find a good estimate for the initial step-size by trial and error. A simple idea is to use the final step-size from the previous step, but this can be unnecessarily small. You may want to do this, but increase the step-size by a factor of 2.

3. **Newton's Method is Affine Invariant (20 pts)**

In this question, we will prove the affine invariance of Newton's method. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function. Consider an affine transform $y \to Ay + b$, where $A \in \mathbb{R}^{n\times n}$ is invertible and $b \in \mathbb{R}^n$. Define the function $g : \mathbb{R}^n \to \mathbb{R}$ by $g(y) = f(Ay + b)$. Denote by $x^{(k)}$ the k^{th} iterate of Newton's method performed on f. Denote $y^{(k)}$ the k^{th} iterate of Newton's method performed on g.

(a) Show that if $x^{(k)} = Ay^{(k)} + b$, then $x^{(k+1)} = Ay^{(k+1)} + b$.

(b) Show that Newton's decrement does not depend on the coordinates, i.e., show that $\lambda(x^{(k)}) = \lambda(y^{(k)})$, where $\lambda(x) = (\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x))^{1/2}$.

Together, this implies that Newton's method is affine invariant. As an important consequence, Newton's method cannot be improved by a change of coordinates, unlike gradient descent.

4. **Newton's Method for Convex Optimization (20 pts)**

(a) Implement Newton's method for the logistic regression problem in Problem 1. Plot the value of the cost function at each iteration and find a reasonable step-size for fast convergence.

(b) Implement randomized Newton's method with uniform sampling sketch, i.e., sampling rows of $H^{1/2}$ uniformly at random where H and $H^{1/2}$ denote Hessian and its square-root respectively. Plot the value of the cost function at each iteration and find a reasonable step-size and sketch-size for fast convergence.

5. **Fast Johnson-Lindenstrauss Transform (FJLT) using Hadamard Matrices (20 pts)**

(a) Construct an 128×1024 FJLT matrix as follows

Set $m = 128$ and $n = 1024$

Define $H_1 := \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

Construct $H_{10} \in \mathbb{R}^{1024,1024}$ recursively via $H_{k+1} = \begin{bmatrix} H_k & H_k \\ H_k & -H_k \end{bmatrix}$

Generate D as an $n\times n$ diagonal matrix of uniformly random ± 1 variables (Rademacher distribution)

Generate an $m \times n$ uniform sub-sampling matrix P scaled with $\frac{\sqrt{n}}{\sqrt{m}}$ (uniform sampling sketch)

Form the FJLT matrix $S = \frac{1}{\sqrt{n}} PHD$.

(b) Verify that $S^T S$ is a multiple of identity. Scale S appropriately if needed to obtain $S^T S = I$.

(c) Generate a data matrix A of size 1024×10 using i.i.d. standard Gaussian variables. Plot the singular values of A and singular values of SA.

(d) In part (c), SA is a Johnson-Lindenstrauss embedding of 10 vectors (1024 dimensional column vectors of A) to dimension 128. Verify that the pairwise distances are approximately preserved, i.e., there exists an $\epsilon > 0$

$$(1-\epsilon)\|A_i - A_j\|_2^2 \leq \|S(A_i - A_j)\|_2^2 \leq (1+\epsilon)\|A_i - A_j\|_2^2 \ \forall i, j\,, \tag{2}$$

where A_i is the i-th column of A. Find ϵ^*, the smallest value of ϵ that satisfy (2) for a single realization of the random construction. Note that the matrices D and P are

constructed randomly, while H is deterministic. What is the minimum, maximum, and mean value of ϵ^* in 100 random realizations of the construction?
Hint: The JL embedding property in (2) specifies $2d^2$ linear inequalities of the form $\epsilon \geq c_{ij}$, hence the smallest ϵ is $\epsilon^* = \max_{ij} c_{ij}$.

(e) Generate a data matrix A of size 1024×10, and a vector b of size 1024×1 using i.i.d. standard Gaussian variables. Solve the least squares problem

$$x^* = \arg\min_x \|Ax - b\|_2^2 .$$

Apply the FJLT to A and b as SA and Sb. Solve the sketched least squares problem

$$\tilde{x} := \arg\min_x \|SAx - Sb\|_2^2 .$$

Find the Euclidean distance between the solutions, i.e., $\|\tilde{x} - x^*\|_2$, between their predictions, i.e., $\|A\tilde{x} - Ax^*\|_2$ and the approximation ratio $(\|A\tilde{x} - b\|_2^2)/(\|Ax^* - b\|_2^2)$ of the objective value.