ECE 50024 / STAT 59800: Machine Learning I
Spring 2024
Instructor: Prof. Qi Guo, Developer: Prof. Stanley H Chan

Homework 4

Spring 2024
(Due: Thursday, Mar 7, 2023, 11:59 pm Eastern Time)

Please submit your homework through **gradescope**. You can write, scan, type, etc. But for the convenience of grading, please merge everything into a **single PDF**.

Objective

There are three things you will learn in this homework:

(a) Understand some theoretical properties about logistic regression.
(b) Implement a logistic regression in CVX, and visualize the decision boundary.
(c) Apply kernel trick to logistic regression.

You will be asked some of these questions in Quiz 4.

Exercise 1. Logistic regression + gradient descent
We analyze the convergence behavior of the logistic regression when the data is **linearly separable**.

Recall that the logistic regression tries to minimize the cross-entropy loss:

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\operatorname{argmin}} \; -\sum_{n=1}^{N} \Big\{ y_n \log h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) + (1-y_n)\log(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \Big\}, \tag{1}$$

where $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1+\exp\{-\boldsymbol{\theta}^T\boldsymbol{x}\}}$ is the sigmoid function. As usual, we assume that $\boldsymbol{\theta} = [\boldsymbol{w},\ w_0]$. We consider the gradient descent algorithm. We know that the objective function is convex, and so we consider the gradient descent algorithm. The iterations are (if you take the derivative of $J(\boldsymbol{\theta})$ and rearrange the terms):

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha_k \left(\sum_{n=1}^{N} (h_{\boldsymbol{\theta}^{(k)}}(\boldsymbol{x}_n) - y_n)\boldsymbol{x}_n \right) \tag{2}$$

for some choices of the step size α_k.

(i) Prove that if two classes of data in $\mathbb{R}^d$ are linearly separable, then the magnitude of the slope $\|\boldsymbol{w}\|_2$ and the magnitude of the intercept $|w_0|$ would tend to ∞.

(ii) What happens if we restrict $\|\boldsymbol{w}\|_2 \leq c_1$ and $|w_0| < c_2$ for some $c_1, c_2 > 0$? What other ways can you come up with to counter the nonconvergence issue?

(iii) Does linear separability of data cause nonconvergence for the other linear classifiers that we have studied? Why?

Exercise 2. Implement logistic regression
Download `hw4_data.zip` from Brightspace. There are two classes with class labels $y_n = 1$ and $y_n = 0$.

(a) Show that the logistic regression loss is given by

$$J(\boldsymbol{\theta}) = -\frac{1}{N}\left\{\left(\sum_{n=1}^{N} y_n \boldsymbol{x}_n\right)^T \boldsymbol{\theta} - \sum_{n=1}^{N} \log\left(1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}\right)\right\}.$$

(b) Introduce a regularization term $\lambda\|\boldsymbol{\theta}\|^2$ and normalize the data fidelity term so that the loss becomes

$$J(\boldsymbol{\theta}) = -\frac{1}{N}\left\{\left(\sum_{n=1}^{N} y_n \boldsymbol{x}_n\right)^T \boldsymbol{\theta} - \sum_{n=1}^{N} \log\left(1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}\right)\right\} + \lambda\|\boldsymbol{\theta}\|^2. \tag{3}$$

Use CVXPY to minimize this loss function for the dataset I provided. Your $\boldsymbol{\theta}$ should be $\boldsymbol{\theta} = [\theta_2, \theta_1, \theta_0]^T$. Please use $\lambda = 0.0001$.

(c) Scatter plot the data points by marking the two classes in two colors. Then plot the decision boundary.

(d) Repeat (c), but this time using the Bayeisn decision rule. Note that since the covariance matrices are not identical, the decision boundary is not a straight line. To plot the decision boundary, you can create a grid of testing sites in the range of $[-5, 10] \times [-5, 10]$ (with 100 points along each dimension). Evaluate the decision on these testing sites. And then plot the decision using `plt.contour`.

Exercise 3. Kernel trick

Let us continue to use the dataset in Exercise 2. Our goal here is to implement the kernel trick.

(a) In Python, construct the kernel matrix $\boldsymbol{K}$, where

$$[\boldsymbol{K}]_{m,n} = \exp\left\{-\|\boldsymbol{x}_m - \boldsymbol{x}_n\|^2/h\right\}, \tag{4}$$

where $h = 1$. Print `K[47:52,47:52]`.

(b) Let us assume that that $\boldsymbol{\theta} = \sum_{n=1}^{N} \alpha_n \boldsymbol{x}_n$ for some α_n's. Then

$$\boldsymbol{\theta}^T \boldsymbol{x} = \sum_{n=1}^{N} \alpha_n \langle \boldsymbol{x}_n, \boldsymbol{x} \rangle.$$

The kernel trick says that we can replace $\langle \boldsymbol{x}_n, \boldsymbol{x} \rangle$ by a kernel $K(\boldsymbol{x}_n, \boldsymbol{x})$. Apply the kernel trick to the loss function in (3). Show that the new loss is

$$J(\boldsymbol{\alpha}) = -\frac{1}{N}\left\{\boldsymbol{y}^T \boldsymbol{K}\boldsymbol{\alpha} - \boldsymbol{1}^T \log(e^{\boldsymbol{0}} + e^{\boldsymbol{K}\boldsymbol{\alpha}})\right\} + \lambda \boldsymbol{\alpha}^T \boldsymbol{K}\boldsymbol{\alpha}.$$

(c) Implement the kernel logistic regression training in CVXPY. Report the first two elements of the regression coefficients $\boldsymbol{\alpha}$.

(d) Scatter plot the data points and plot the decision boundary, just like what you did in Exercise 2(d).

Exercise 4. Project Check Point 3

On Gradescope, there is an assignment called Project Check Point 3. Upload a manuscript that demonstrates solving a toy problem using your reimplementation. A toy problem can be a very simple dataset, e.g., the MNIST, or a hypothetical problem that you create just to demonstrate that your reimplementation works. The manuscript needs to include every details in a well-organized way, so that the teaching staff can easily understand what you are doing. It needs to show results that demonstrate your reimplementation is working on the toy problem. The grade of this checkpoint will be based on quality of the report and whether your results show you successfully reimplement the paper for the toy problem. You also need to include in your report a link to a Github repository of your code, so that the teaching staff can check if you reused other people's code for your project. Apart from typical Python packages, e.g., Numpy, Scipy, Scikit-learn, TensorFlow, PyTorch, etc., you cannot use other people's code in your re-implementation.

The format is the same as the Check Point 1. Each team only needs to upload one copy, but make sure to list all group members of the team. The manuscript needs to be written using Latex following the ICML template listed on the course website.

The Check Point 3 will be graded separately from your HW4 submission, and is worth 15% of the final project grade. The grading will be based on the following criteria:

- Did the team describe clearly the problem they aim to solve with all necessary details well organized?
- Did the results demonstrate the re-implementation is working on the toy problem?
- Is the code well organized and commented for teaching staff to understand?
- Is the report well written? Did the team copy or closely resemble the original paper or other people's reimplementation? Heavy penalty will be added to the manuscripts that copies or closely resembles the original paper or other people's reimplementation.