Tsinghua-Berkeley Shenzhen Institute
Learning From Data
Fall 2020

Programming Assignment 1

Issued: Friday 25th September, 2020 **Due:** Friday 9th October, 2020

POLICIES

- **Acknowledgments:** We expect you to make an honest effort to solve the problems individually. As we sometimes reuse problem set questions from previous years, covered by papers and web pages, we expect the students **NOT** to copy, refer to, or look at the solutions in preparing their answers (relating to an unauthorized material is considered a violation of the honor principle). Similarly, we expect you not to google directly for answers (though you are free to google for knowledge about the topic). If you do happen to use other material, it must be acknowledged in `README.md`, with a citation on the submitted solution.
- **Required homework submission format:** You should submit the assignment by the invitation link `https://classroom.github.com/a/vUmE9ElL`. This link will create a private GitHub repository from the code template. Push your modification to the master branch and view the auto-grading results afterwards. The teaching assistant will grade your assignment mainly based on the rightness of your programming implementation. Optionally you can write your ideas in `README.md` or in code comments.
- **Collaborators:** If you collaborated with others, in `README.md`, list their names and GitHub ID and for which question(s).

1.1. (5 points) *Linear regression with regularization.* In class, we have learned linear regression using the linear model

$$\boldsymbol{y} = X\boldsymbol{\theta} + \boldsymbol{\epsilon}$$

where $\boldsymbol{X}$ is the observation matrix, $\boldsymbol{\theta}$ is the weight vector to be estimated and $\boldsymbol{\epsilon}$ is the Gaussian noise. To minimize the noise effect in the model, we do a minimization problem

$$\min_{\theta} ||\boldsymbol{y} - X\boldsymbol{\theta}||^2$$

and get the best weight estimator $\arg\min_{\theta} ||\boldsymbol{y} - X\boldsymbol{\theta}||^2 = (X^T X)^{-1} X^T \boldsymbol{y}$. This process may be problematic when matrix $X^T X$ is nearly singular, which means it has very small singular values. To deal with this problem, we add a regularization in linear model and we have

$$\min_{\theta} ||\boldsymbol{y} - X\boldsymbol{\theta}||^2 + \alpha||\boldsymbol{\theta}||^2$$

where $\alpha \geq 0$ is a hyper-parameter. The method with extra α is also known as *ridge regression.*

Please implement ridge regression by completing the code in **ridge_regression.py**. Your implementation should handle the case when $\alpha = 0$.

Hint: Minimizing the loss $||\boldsymbol{y} - X\boldsymbol{\theta}||^2 + \alpha||\boldsymbol{\theta}||^2$ is equivalent to solving the equation $(X^T X + \alpha I)\boldsymbol{\theta} = X^T\boldsymbol{y}$. You can use `numpy.linalg.lstsq` to solve $\boldsymbol{\theta}$.

1.2. (5 points) *Logistic regression with Newton's method.* You have learned in class that using maximal likelihood to estimate the parameters of the logistic regression model is equivalent to maximizing:

$$l(\theta) = \sum_{i=1}^{m} y^{(i)} \log h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) + (1 - y^{(i)}) \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})) \text{ where } h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \boldsymbol{x})}$$

We can use Newton's method to find the optimal $\boldsymbol{\theta}$, which has the following update scheme for $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - H^{-1}\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta})|_{\boldsymbol{\theta}_t}$$

where H is the Hessian matrix for the likelihood function $\mathcal{L}$. The scheme can be written in compact form as:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + (X^T R X)^{-1} X^T(\boldsymbol{y} - \boldsymbol{\mu}), \text{ where } \mu_i = h_{\boldsymbol{\theta}_t}(\boldsymbol{x}^{(i)}) \text{ and } R_{ii} = \mu_i(1 - \mu_i)$$ [1]

Using Newton's method to fit the logistic model is also called *iterative reweighted least sqaures* (IRLS).

Please implement IRLS by completing the code in **logistic_regression.py**.

Notice:

1.1. Use matrix operations other than loops for efficiency. If the running time of Auto-Grading steps exceeds 5 minutes, you will get point deductions.

1.2. You are expected to only use `numpy` packages to implement the algorithms.

1.3. All questions assume that the data are centered around zero. Therefore, you do not need to train the extra bias parameter in your code.

[1] R is a diagonal matrix