COMS W4721 Spring 2020 Homework 1: Maximum Likelihood, Linear Regression, Bias-Variance Tradeoffs

My name (my UNI@columbia.edu)

January 26, 2020

For this assignment I collaborated with the following people. Blank entries in this table means that I have worked on the corresponding parts on my own.

Problem	Collaborators with their UNIs	Part
Problem 2	Collaborator 1 (uni1@columbia.edu)	Part (a), (b)
	Collaborator 2 (uni2@columbia.edu)	Part (b)
Problem 3		
Problem 4	Collaborator 3 (uni3@columbia.edu)	Part (c)

Problem 1

Introduction: Please briefly describe your academic/career goals and your expectations about the course.

Answer:

Problem 2

In this problem we will review the principle of *maximum likelihood estimation.*

(a) We are given a coin which falls its heads up with probability $0 < \theta < 1$. Each throw is a Bernoulli random variable $x = \begin{cases} 1, \text{if falls heads up} \\ 0, \text{if falls tails up} \end{cases}$.

For a Bernoulli random variable x: the probability mass function of x is given by:

$$\Pr(x; \theta) = \theta^x (1-\theta)^{(1-x)}$$

Suppose we repeat the coin toss N times to collect the data $\{x^{(i)}\}_{i=1}^N$. Write the log likelihood function $\ln \mathcal{L}(\theta; \{x^{(i)}\}_{i=1}^N)$ and the maximum likelihood estimation of θ, $\hat{\theta}_{\mathsf{MLE}}$.
Solution:

(b) Suppose instead we are given a die with K sides with each side falling its heads up with probability $0 < \theta_k < 1$. While we can represent the result of a throw using a categorical variable $x \in \{1, \cdots, K\}$, we can use 1-of-K encoding:

$$x = k \quad \Leftrightarrow \quad \mathbf{x} = [0, \cdots, \underbrace{1}_{\substack{\text{k-th} \\ \text{position} \\ := x_k}}, \cdots, 0]$$

Each throw is a categorical random variable $\mathbf{x} \sim \text{Categorical}(\theta_1, \cdots, \theta_K)$ such that $\Pr(x_k = 1; \theta) = \theta_k$ and $\sum\limits_{k=1}^{K} \theta_k = 1$. The probability mass function at $\mathbf{x}$ is given by:

$$\Pr(\mathbf{x}; \theta_1, \cdots, \theta_K) = \prod_{k=1}^{K} \theta_k^{x_k}$$

Suppose we throw the die N times and obtain the data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$. Write the log likelihood function $\ln \mathcal{L}(\theta; \{\mathbf{x}^{(i)}\}_{i=1}^{N})$ and the maximum likelihood estimation of θ, $\hat{\theta}_{\mathsf{MLE}}$.

Hint: Once you obtain the log likelihood function $\mathcal{L}(\theta; \{\mathbf{x}^{(i)}\}_{i=1}^{N})$, you will need to add the Lagrangian multiplier part that takes the probability sum constraint $\sum\limits_{k=1}^{K} \theta_k = 1$. For $\lambda \in \mathbb{R}$, the Lagrangian is:

$$\ln \mathcal{L}_\lambda(\theta; \{\mathbf{x}^{(i)}\}_{i=1}^{N}) = \ln \mathcal{L}(\theta; \{\mathbf{x}^{(i)}\}_{i=1}^{N}) + \lambda \left(1 - \sum_{k=1}^{K} \theta_k\right)$$

Take the partial derivative of $\ln \mathcal{L}(\theta; \{\mathbf{x}^{(i)}\}_{i=1}^{N})$ with respect to each θ_k and λ, set them to zero, and solve for each variable of θ_k. Appendix E of Bishop's book may be helpful for this problem.

Solution:

(c) In class we derived a MLE estimator for the univariate Gaussian assumption. For $\{x^{(i)} \in \mathbb{R}\}_{i=1}^{N}$ i.i.d, we chose $p(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$ and solved for $\hat{\mu}_{\textsf{MLE}}$ and $\hat{\sigma}^2_{\textsf{MLE}}$. Repeat this exercise for the multivariate case: now assume $\{\mathbf{x}^{(i)} \in \mathbb{R}^d\}_{i=1}^{N}$ and choose $p(\mathbf{x}|\mu, \mathbf{\Sigma}) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{\mathsf{det}(\Sigma)}} \exp\left(\frac{-(\mathbf{x}-\mu)^T \mathbf{\Sigma}^{-1} (\mathbf{x}-\mu)}{2}\right)$. Write the log likelihood function $\ln \mathcal{L}(\{\mu, \Sigma\}; \{x^{(i)}\}_{i=1}^{N})$ and solve for $\hat{\mu}_{\textsf{MLE}}$ and $\hat{\Sigma}_{\textsf{MLE}}$.

Solution:

Problem 3

In this problem we will use a numerical optimization routine to obtain maximum likelihood estimate of parameters.

Suppose $\{x^{(i)} \in \mathbb{R}\}_{i=1}^{N}$ with $x^{(i)} \sim p(x; x_0, \gamma)$ defined as:

$$p(x; x_0, \gamma) = \frac{1}{\pi \exp(\gamma) \left[1 + \left(\frac{x - x_0}{\exp(\gamma)}\right)^2\right]}$$

(a) Prove that $p(x; x_0, \gamma)$ is a probability density function.

Solution:

(b) Prove that the mean $\mathbb{E}_{x \sim p(x; x_0, \gamma)}[x]$ is undefined.

Solution:

(c) Write the log likelihood function $\ln \mathcal{L}(\{x_0, \gamma\}; \{x^{(i)}\}_{i=1}^{N})$ and the expression for $\frac{\partial \ln \mathcal{L}(\{x_0,\gamma\};\{x^{(i)}\}_{i=1}^{N})}{\partial x_0}$ and $\frac{\partial \ln \mathcal{L}(\{x_0,\gamma\};\{x^{(i)}\}_{i=1}^{N})}{\partial \gamma}$. Plot the log likelihood value as a 3D surface plot: x-axis should run over x_0 and y-axis should run over γ. z-axis should correspond to the log likelihood value at the corresponding (x_0, γ) pair. Include the plot in your writeup. Do the stationary points (solutions to the maximum likelihood equations) have closed form solutions?

Solution:

(d) Write a program to obtain an estimate of $\theta = \{x_0, \gamma\}$ using the dataset **problem3.csv**. If you are using Python, it is helpful to utilize scipy.optimize.minimize function. Choose gradient descent optimizer or a quasi-Newton optimizer such as BFGS. List the optimizer that was chosen for this problem with the initial iterate. Tabulate the coordinates of the iterates of the optmization process and the final converged solution.

Solution:

Problem 4

In this problem you will implement ridge regression estimator using gradient descent. Although the ridge regression does have a closed form solution, gradient descent form lets you avoid explicit matrix inversion and scale to larger data. We will see this in our subsequent lectures.

For this problem, we assume we are given the labeled data pair:
$\{(\mathbf{x}^{(i)} \in \mathbb{R}^d, y^{(i)} \in \mathbb{R})\}_{i=1}^N$. The regularized objective function in this case is given by:

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} L(b, \mathbf{w}; \{(\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N) := \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - (b + \mathbf{w}^T \cdot \mathbf{x}^{(i)}))^2 + \lambda \cdot ||\mathbf{w}||_2^2$$

The pseudocode for performing gradient descent is given by the following algorithm. The main structure consists of a loop which continues for a given number of epochs T. η is the learning rate that controls the amount you want to step into the direction of the negative gradient, and λ is the regularization parameter.

```
function GDRidge($S_{\text{train}} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$, $T$, $\eta$, $\lambda$)
    Initialize the bias term $b \leftarrow 0$ and the slope $\mathbf{w} \leftarrow \mathbf{0}$
    for $t = 1, \cdots, T$ do
        $b_{\text{new}} \leftarrow b - \eta \cdot \frac{\partial L}{\partial b}$,    $\mathbf{w}_{\text{new}} \leftarrow \mathbf{w} - \eta \cdot \frac{\partial L}{\partial \mathbf{w}}$
        $b \leftarrow b_{\text{new}}$,    $\mathbf{w} \leftarrow \mathbf{w}_{\text{new}}$
    end for
end function
```

(a) Write the expression for the gradient update for b and $\mathbf{w}$.
Solution:

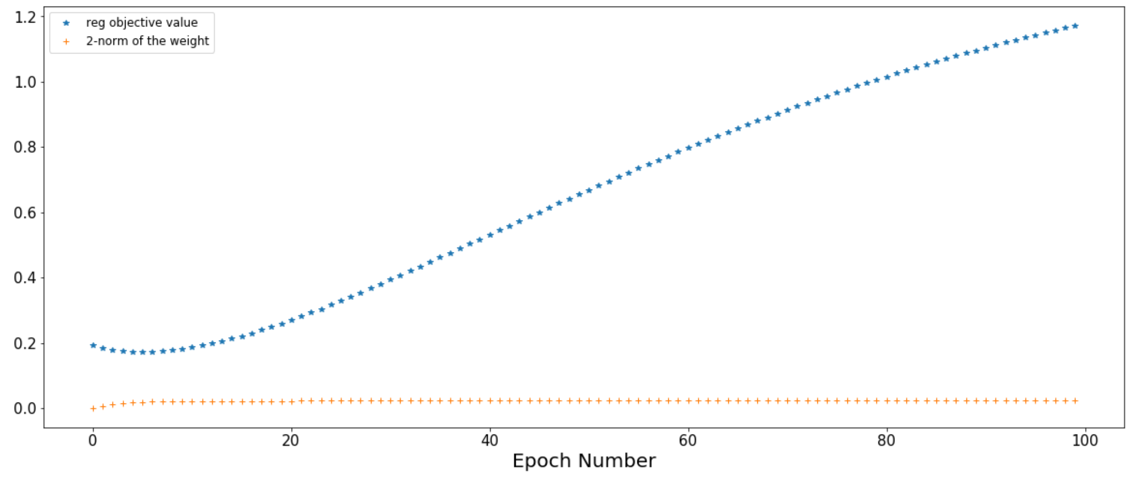
(b) Implement the function GDRidge described above. Please include your source code in the writeup.
Solution:

(c) Use the Boston housing data (`https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html`) and scale the data appropriately: standardize the feature matrix and [0, 1] scale the y values. Choose $T = 100$ and $\eta = 0.01$.

For $\lambda = 0.1$, provide a x-y plot with the epoch number as x-axis and plot the following quantities on the y-axis:

- The value of regularized objective L at the start of each epoch.
- The 2-norm of the weight vector $\mathbf{w}$ at the start of each epoch.

Here is an example plot:



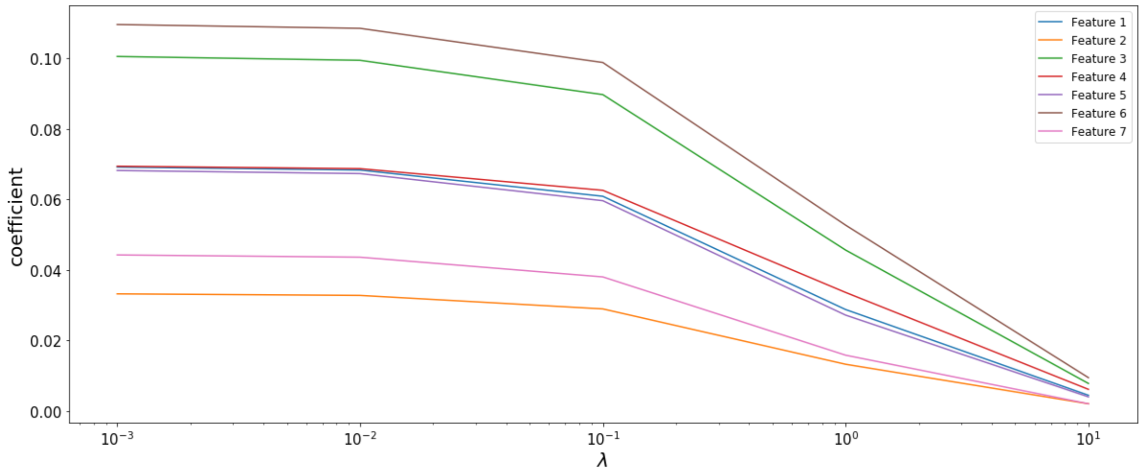
This will require you to modify the function written in Part (b) to compute the required quantities.

Solution:

(d) The next plot will examine how the coefficients for each of the features change as the regularization parameter is varied.

Provide a coefficient path plot for the Boston housing data for $\lambda = 10, 1, 0.1, 0.01, 0.001$. x-axis is for the regularization value and y-axis for the coefficient of the final converged iterate of your gradient descent algorithm. Make sure to scale the data appropriately. Choose $T = 100$ and $\eta = 0.01$.

Here is an example plot for a dataset with 7 features: there is a connected path for each of the 7 features as the regularization parameter is varied.



What do you notice about the behavior of the coefficients as the regularization is varied? Include the coefficient path plot and your analysis.

Solution: