EECS289A Introduction to Machine Learning, Project T Final - Quiz Questions

Han Liu, Peng Tan, Dilu Xu, Jinyan Zhao
Department of Civil Engineering
Universitu of California, Berkeley
Berkeley, CA 94704
{han_liu, tanpeng, diluxu, jinyan_zhao}@berkeley.edu

November 30, 2020

1 Elastic Net

In class we discussed two types of regularization, l_1 and l_2. Both are useful, and sometimes it is helpful combine them, giving the objective function below (**in this problem we have excluded w_0 for simplicity**):

$$F(\boldsymbol{w}) = \frac{1}{2}\sum_{j=1}^{n}(y^{(j)} - \sum_{i=1}^{d}\boldsymbol{w}_i\boldsymbol{x}_i^{(j)})^2 + \alpha\sum_{i=1}^{d}|\boldsymbol{w}_i| + \frac{\lambda}{2} + \sum_{i=1}^{d}\boldsymbol{w}_i^2 \tag{1}$$

Here, $(\boldsymbol{x}^{(j)}, y^{(j)})$ is j-th example in the training data, w is a d dimensional weight vector, λ is a regularization parameter for the l_2 norm of w, and α is a regularization parameter for the l_1 norm of w. This approach is called the Elastic Net, and you can see that it is a generalization of Ridge and Lasso regression: It reverts to Lasso when $\lambda = 0$, and it reverts to Ridge when $\alpha = 0$. In this question, we are going to derive the coordinate descent (CD) update rule for this objective.

Let g, h, c be real constants, and consider the function of x

$$f_1(x) = c + gx + \frac{1}{2}hx^2 \tag{2}$$

1. [4 points] What is the x^* that minimizes $f_1(x)$? (i.e. calculate $x^* = \arg\min f_1(x)$)
   <u>**Answer:**</u>
   Take the gradient of $f_1(x)$ and set it to 0.

$$g + hx = 0 \tag{3}$$

$$x^* = -\frac{g}{h} \tag{4}$$

Let α be an additional real constant, and consider another function of x

$$f_2(x) = c + gx + \frac{1}{2}hx^2 + \alpha|x|(h > 0, \alpha > 0) \tag{5}$$

This is a piecewise function, composed of two quadratic functions:

$$f_2^-(x) = c + gx + \frac{1}{2}hx^2 - \alpha x \tag{6}$$

and

$$f_2^+(x) = c + gx + \frac{1}{2}hx^2 + \alpha x \tag{7}$$

Let $\tilde{x}^- = \arg\min_{x\in\mathbb{R}} f_2^-(x)$ and $\tilde{x}^+ = \arg\min_{x\in\mathbb{R}} f_2^+(x)$.
(**Note:** The argmin is taken over $(-\infty, +\infty)$ here.

2. [6 points] What are $\tilde{x}^+$ and $\tilde{x}^+$? Show that $\tilde{x}^- \geq \tilde{x}^+$.
   Answer:
   Using the answer from part 1), we get $\tilde{x}^+ = -\frac{g+\alpha}{h}$ and $\tilde{x}^- = -\frac{g-\alpha}{h}$. Since $\tilde{x}^- - \tilde{x}^+ = \frac{2\alpha}{h} \geq 0$, we have $\tilde{x}^- \geq \tilde{x}^+$.

3. [12 points] Draw a picture of $f_2(x)$ in each of the three cases below:

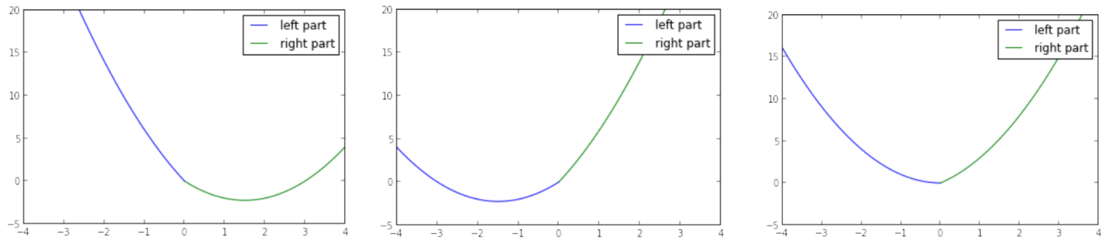
   (a) $\tilde{x}^- > 0$, $\tilde{x}^+ > 0$

   (b) $\tilde{x}^- < 0$, $\tilde{x}^+ < 0$

   (c) $\tilde{x}^- > 0$, $\tilde{x}^+ < 0$

   For each case, mark the minimum as either 0, $\tilde{x}^-$, or $\tilde{x}^+$. (You do not need to draw perfect curves, just get the rough shape and the relative locations of the minima to the x-axis)
   Answer:
   Fig. 1 gives the example picture of three cases. To understand the answer, note the following:

   - $f_2^+(0) = f_2^-(0)$, this means the two piece of f_2 match each other at x = 0.
   - When $\tilde{x}^- \geq 0$, the minimum of the left part is at 0, i.e. $\arg\min_{x\in(-\infty,0]} f_2^-(x) = 0$
   - When $\tilde{x}^- \leq 0$, the minimum of the left part is at $\tilde{x}^-$, i.e. $\arg\min_{x\in(-\infty,0]} f_2^-(x) = \tilde{x}^-$
   - Similar rules holds for the right part of the curve.



(a) $\tilde{x}^- > 0$, $\tilde{x}^+ > 0$, the minima is at $\tilde{x}^+$.

(b) $\tilde{x}^- < 0$, $\tilde{x}^+ < 0$, the minima is at $\tilde{x}^-$.

(c) $\tilde{x}^- > 0$, $\tilde{x}^+ < 0$, the minima is at 0.

Fig. 1. Example picture of three cases.

4. [8 points] Write x^*, the minimum of $f_2(x)$, as a piecewise function of g. (Hint: what is g in the cases above?)
Answer:
Summarizing the result from the previous question, we have

$$x^* = \begin{cases} \tilde{x}^+ & \text{if } \tilde{x}^+ > 0 \\ \tilde{x}^- & \text{if } \tilde{x}^- < 0 \\ 0 & \text{if } \tilde{x}^+ < 0, \tilde{x}^- > 0 \end{cases} \tag{8}$$

This is equivalent to the soft-threshold function

$$x^* = \begin{cases} -\frac{g+\alpha}{h} & \text{if } g < -\alpha \\ 0 & \text{if } g \in [-\alpha, \alpha] \\ -\frac{g-\alpha}{h} & \text{if } g > \alpha \end{cases} \tag{9}$$

5. [8 points] Now let's derive the update rule for w_k. Fixing all parameters but w_k, express the objective function in the form of Eq. (5) (here w_k is our x). You do not need to explicity write constant factors, you can put them all together as c. What are g and h? Then putting it all together, write down the coordinate descent algorithm for the Elastic Net (**again excluding** w_0). You can write the update for w_k in terms of the g and h you calculated in the previous question.
Answer:

$$g = \sum_{j=1}^{n} x_k^{(j)} \Big(\sum_{i \neq k} \boldsymbol{w}_i \boldsymbol{x}_i^{(j)} - y_i\Big) \tag{10}$$

$$h = \lambda + \sum_{j=1}^{n} (x^{(j)})^2 \tag{11}$$

The algorithm is shown in Algorithm. 1. You can compare it to the CD algorithm for Lasso, the only difference is adding λ to h.

Algorithm 1: Coordinate Descent for Elastic Net

```
while not converged do
    for $k \in \{1, 2, \cdots, d\}$ do
        $g = \sum_{j=1}^{n} x_k^{(j)} (\sum_{i \neq k} \boldsymbol{w}_i \boldsymbol{x}_i^{(j)} - y_i)$;
        $h = \lambda + \sum_{j=1}^{n} (x^{(j)})^2$;
        $w_k = \begin{cases} -\frac{g+\alpha}{h} & \text{if } g < -\alpha \\ 0 & \text{if } g \in [-\alpha, \alpha] \\ -\frac{g-\alpha}{h} & \text{if } g > \alpha \end{cases}$;
    end
end
```

2 True or False questions

Answer True or False. Justify your answer very briefly in 1-2 sentences.

1. Increasing the regularization parameter λ in lasso regression leads to sparser regression coefficients.
   <u>**Answer:**</u>
   True. Larger regularization parameter penalizes non-zero coefficients more, leading to sparser solution.

2. In Lasso regression, can the regulariser increase the sparsity of the resulting solutions?
   <u>**Answer:**</u>
   True, Lasso introduces the L1 norm penalty, so increasing the penalty will cause more and more of the parameters to be driven to zero. As seen in Fig. 2, where the red contours are representing the error function and the solid blue areas the constraint regions for the parameters, in the Lasso case, the function is likely to hit the constraint region on one of the four corners. This will make a parameter equal with zero.

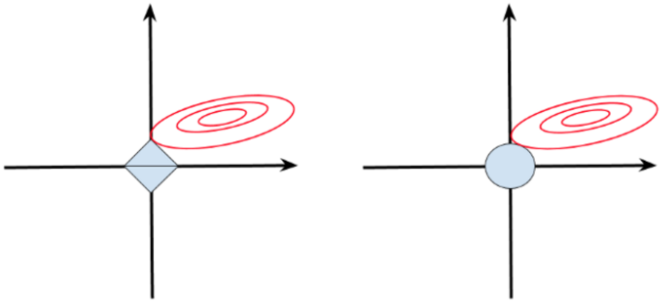


Fig. 2. Spam-Estimation picture for the LASSO (left) and Ridge regression (right).

3. Is the objective of Lasso regression differentiable and convex?
   <u>**Answer:**</u>
   False, it is not differentiable. As Lasso uses L1 norm penalty, this L1 loss function is defined based on the absolute value of the difference between two values. Since the loss function is a modulus function, there will exist an inflection point where the slope cannot be computed and the function is not differentiable. In contrast, for the case of Ridge regression, where we use L2 norm penalty, we can get the slope at any point and the objective is differentiable.

4. Are the objectives of Ridge and Lasso regression both convex and have closed-form solutions?
   <u>**Answer:**</u>
   False, they are both convex, but Lasso does not always have a closed-form solution. LASSO regression must be solved numerically by using quadratic programming or more general convex optimisation methods.

5. Is it true that the L1 term in Lasso has the following purposes: performing feature selection, compensating for overfitting and smoothing?
   <u>**Answer:**</u>

False. The first two purposes are true, but not the last one. As it is mentioned early, both L1 and L2 terms are introduced to compensate for overfitting. By making the parameters equal with zero, the L1 norm discards some features and, thus, performs feature selection. In terms of smoothing, we cannot say L1 has this purpose, as the L1 loss function does not have continuous derivatives.

3 Quick questions

Explain **in one or two sentences** why the statements are true (or false).

1. L2 loss is more robust to outliers than L1 loss.
   Answer:
   False
   The gradient of L2 loss can grow without bound whereas the L1 loss gradient is bounded, hence the influence of an outlier is limited.

2. Logistic loss is better than L2 loss in classification tasks.
   Answer:
   True
   With logistic loss, correctly classified points that are far away from the decision boundary have much less impact on the decision boundary.

3. In terms of feature selection, L2 regularization is preferred since it comes up with sparse solutions.
   Answer:
   False
   L1 regularization (LASSO) comes up with sparse solutions due to nonvanishing gradient at 0.

4 Gradient descent

Denote by $\boldsymbol{x} \in \mathbb{R}^d$ data, by $\boldsymbol{w} \in \mathbb{R}^d$ the weight vector, by $y \in \mathbb{R}y$ lables, by $\lambda > 0$ a regularization constant, and by m the total number of data points. Let R(w) be the regularized risk function:

$$R(w) := \frac{1}{m}\sum_{i=1}^{m} l(y_i - \langle w, x_i\rangle) + \frac{\lambda}{2}||\boldsymbol{w}||^2 \quad \text{where} \quad l(\xi) = \begin{cases} \frac{1}{2}\xi^2 & \text{if } |\xi| < 1 \\ |\xi| - \frac{1}{2} & \text{otherwise} \end{cases} \tag{12}$$

1. Calculate the batch gradient of R(w) with respect to w.
   Answer:
   Define g_i as the gradient of ith data point.

$$g_i = \begin{cases} (\langle w, x_i\rangle - y)x_i & \text{if} |\langle w, x_i\rangle - y| < 1 \\ \text{sgn}(\langle w, x_i\rangle - y)x_i & \text{otherwise} \end{cases} \tag{13}$$

$$\frac{\partial R}{\partial w} = \frac{1}{m}\sum_{i=1}^{m} g_i + \lambda w \tag{14}$$

2. Write out a stochastic gradient descent learning algorithm for minimizing R(w).
   Answer:
   Randomly select a training instance x_i, update w as follows:

$$\boldsymbol{w} \leftarrow (1 - \lambda\eta_t)\boldsymbol{w} - \eta_t g_i \tag{15}$$

3. Name two common choices for choosing the learning rate and briefly explain their properties.
   Answer:
   $O(\frac{1}{\sqrt{t}}), O(\frac{1}{t})$ for strong convexity, Polynomial decay, AdaGrad, etc.

5 Multiple choice

1. You run gradient descent for 15 iterations with $\alpha = 0.3$ and compute after each iteration. You find that the value of $J(\theta)$ decreases slowly and is still decreasing after 15 iterations. Based on this, which of the following conclusions seems most plausible?
   a. Rather than use the current value of α, it'd be more promising to try a larger value of α (say $\alpha = 1.0$).
   b. Rather than use the current value of α, it'd be more promising to try a smaller value of α (say $\alpha = 0.1$).
   c. $\alpha = 0.3$ is an effective choice of learning rate.
   Answer:
   a

2. You run gradient descent for 15 iterations with $\alpha = 0.3$ and compute after each iteration. You find that the value of $J(\theta)$ decreases quickly then levels off. Based on this, which of the following conclusions seems most plausible?
   a. Rather than use the current value of α, it'd be more promising to try a larger value of α (say $\alpha = 1.0$).
   b. Rather than use the current value of α, it'd be more promising to try a smaller value of α (say $\alpha = 0.1$).
   c. $\alpha = 0.3$ is an effective choice of learning rate.
   Answer:
   c

3. Suppose you have $m = 23$ training examples with $n = 5$ features (excluding the additional all-ones feature for the intercept term, which you should add). The normal equation is $\theta = (X^T X)^{-1} X^T y$. For the given values of m and n, what are the dimensions of θ, X, and y in this equation?
   a. X is 23 × 5, y is 23 × 1, θ is 5 × 5
   b. X is 23 × 6, y is 23 × 6, θ is 6 × 6
   c. X is 23 × 6, y is 23 × 1, θ is 6 × 1
   d. X is 23 × 5, y is 23 × 1, θ is 5 × 1
   Answer:
   c
   X has m rows and n+1 columns (+1 because of the $x_0 = 1$ term). y is m-vector. θ is an $(n+1)$-vector

4. Suppose you have a dataset with $m = 1000000$ examples and $n = 200000$ features for each example. You want to use multivariate linear regression to fit the parameters θ to our data. Should you prefer gradient descent or the normal equation?
   a. Gradient descent, since it will always converge to the optimal θ.
   b. Gradient descent, since $(X^T X)^{-1}$will be very slow to compute in the normal equation.
   c. The normal equation, since it provides an efficient way to directly find the solution.
   d. The normal equation, since gradient descent might be unable to find the optimal θ.
   Answer:
   b
   With $n = 200000$ features, you will have to invert a $200001x200001$ matrix to compute the normal equation. Inverting such a large matrix is computationally expensive, so gradient descent is a good choice.

5. Which of the following are reasons for using feature scaling?

a. It is necessary to prevent gradient descent from getting stuck in local optima.
b. It speeds up solving for θ using the normal equation.
c. It prevents the matrix $X^T X$ (used in the normal equation) from being non-invertable (singular/degenerate).
d. It speeds up gradient descent by making it require fewer iterations to get to a good solution.

<u>**Answer:**</u>

d

For a, the cost function $J(\theta)$ for linear regression has no local optima. For b, the magnitute of the feature values are significant in terms of computational cost. For c, feature scaling has nothing to do with matrix inversion. For d, feature scaling speeds up gradient descent by avoiding many extra iterations that are required when one or more features takes on much larger values than he rest.