Regularization and Logistic Regression

Steve Avsec

Illinois Institute of Technology

January 29, 2024

Overview

1. Where We Left Off
2. Regularization
3. Logistic Regression and the Like

Condition Numbers

$\text{Cond}(A) = \sigma_1(A)/\sigma_r(A)$ where $\sigma_j(A)$ denotes the jth largest signular value of A and $\text{rank}(A) = r$.

Condition Numbers

$\mathrm{Cond}(A) = \sigma_1(A)/\sigma_r(A)$ where $\sigma_j(A)$ denotes the jth largest signular value of A and $\mathrm{rank}(A) = r$.

Alternatively, if $A = U\Sigma V^t$ is the singular value decomposition of A and $A^\dagger = V\Sigma^\dagger U^t$ be the pseudoinverse. Then $\mathrm{Cond}(A) = \sigma_1(A)\sigma_1(A^\dagger)$.

Condition Numbers

$\mathrm{Cond}(A) = \sigma_1(A)/\sigma_r(A)$ where $\sigma_j(A)$ denotes the jth largest signular value of A and $\mathrm{rank}(A) = r$.

Alternatively, if $A = U\Sigma V^t$ is the singular value decomposition of A and $A^\dagger = V\Sigma^\dagger U^t$ be the pseudoinverse. Then $\mathrm{Cond}(A) = \sigma_1(A)\sigma_1(A^\dagger)$.

A matrix A is *ill-conditioned* or *poorly-conditioned* if $\mathrm{Cond}(A)$ is "large" (greater than the precision of the machine).

Least-Squares Regression

Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ be some data ($\mathbf{x}_j \in \mathbb{R}^p$, $y_j \in \mathbb{R}$). Let $f_1, \ldots, f_K : \mathbb{R}^p \to \mathbb{R}$ be a set of “suitable” functions.

Least-Squares Regression

Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ be some data ($\mathbf{x}_j \in \mathbb{R}^p$, $y_j \in \mathbb{R}$). Let $f_1, \ldots, f_K : \mathbb{R}^p \to \mathbb{R}$ be a set of “suitable” functions.
Then the linear combination $f = c_1 f_1 + \ldots + c_K f_K$ that minimizes

$$\min_{f \in \mathrm{span}(f_1,\ldots,f_K)} \|\mathbf{y} - f(X)\|_2^2 = \sum_{j=1}^{N} |y_j - f(\mathbf{x}_j)|^2$$

can be calculated by $\mathbf{c} = A^\dagger \mathbf{y}$ where

$$A = \begin{bmatrix} f_1(\mathbf{x}_1) & \cdots & f_K(\mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ f_N(\mathbf{x}_N) & \cdots & f_K(\mathbf{x}_N) \end{bmatrix}$$

Least-Squares Regression

Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ be some data ($\mathbf{x}_j \in \mathbb{R}^p$, $y_j \in \mathbb{R}$). Let $f_1, \ldots, f_K : \mathbb{R}^p \to \mathbb{R}$ be a set of "suitable" functions.
Then the linear combination $f = c_1 f_1 + \ldots + c_K f_K$ that minimizes

$$\min_{f \in \operatorname{span}(f_1, \ldots, f_K)} \|\mathbf{y} - f(X)\|_2^2 = \sum_{j=1}^{N} |y_j - f(\mathbf{x}_j)|^2$$

can be calculated by $\mathbf{c} = A^\dagger \mathbf{y}$ where

$$A = \begin{bmatrix} f_1(\mathbf{x}_1) & \cdots & f_K(\mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ f_N(\mathbf{x}_N) & \cdots & f_K(\mathbf{x}_N) \end{bmatrix}$$

This works great if A is well-conditioned.

Principal Component Regression

What if A is ill-conditioned?

Principal Component Regression

What if A is ill-conditioned?

The most na ive solution is to replace A with A_m, the best rank-m approximation to A. This is given by

$$A_m = U\Sigma_m V^t$$

where Σ_m has the same first m values as Σ and sets the other $r - m$ singular values to 0.

Principal Component Regression

What if A is ill-conditioned?

The most na ive solution is to replace A with A_m, the best rank-m approximation to A. This is given by

$$A_m = U\Sigma_m V^t$$

where Σ_m has the same first m values as Σ and sets the other $r - m$ singular values to 0.

We can choose m such that $\text{Cond}(A_m)$ is less than a fixed value. For instance, if $\sigma_1(A) = 10^5$ and we wish $\text{Cond}(A_m) < 10^7$, then choose m such that all $\sigma_j(A) < 10^{-2}$ are set to 0.

Ridge Regression

Well we could add a term to our minimizer:

$$\min_{f \in \text{span}(f_1, \ldots, f_K)} \|y - \sum_j c_j f_j(X)\|_2 + \lambda \sum |c_j|^2$$

Here λ is a fixed parameter that can be tuned.

Ridge Regression

Well we could add a term to our minimizer:

$$\min_{f \in \text{span}(f_1,\ldots,f_K)} \|y - \sum_j c_j f_j(X)\|_2 + \lambda \sum |c_j|^2$$

Here λ is a fixed parameter that can be tuned.
Another way to look at this:

$$\mathbf{c} = (A^t A + \lambda I_K)^{-1} A^t \mathbf{y} = A_\lambda^\dagger$$

where $A_\lambda^\dagger = V \sigma_\lambda^\dagger U^t$ and where $\sigma_\lambda^\dagger$ is the following function applied component-wise to Σ.

$$g_\lambda(z) = \frac{z}{z^2 + \lambda}$$

Some Notation

For a vector $\mathbf{x} \in \mathbb{R}^n$, define the p-norm by

$$\|\mathbf{x}\|_p = \left(\sum_{j=1}^{n} |x_j|^p \right)^{\frac{1}{p}}$$

for $1 \leq p < \infty$. For the ∞ case,

$$\|\mathbf{x}\|_\infty = \max_j |x_j|$$

Similarly for Matrices

For an $N \times K$ matrix A, we can define

$$\|A\|_p = \left(\mathrm{Tr}\left((A^t A)^{\frac{p}{2}}\right)\right)^{\frac{1}{p}}$$

This is equivalent to the vector p-norm of the singular values of A.

Similarly for Matrices

For an $N \times K$ matrix A, we can define

$$\|A\|_p = \left(\mathrm{Tr}\left((A^t A)^{\frac{p}{2}}\right)\right)^{\frac{1}{p}}$$

This is equivalent to the vector p-norm of the singular values of A.

For $p = 2$, this is the *Fr obenius* norm of A.

Similarly for Matrices

For an $N \times K$ matrix A, we can define

$$\|A\|_p = \left(\operatorname{Tr}\left((A^t A)\frac{p}{2}\right)\right)^{\frac{1}{p}}$$

This is equivalent to the vector p-norm of the singular values of A.

For $p = 2$, this is the *Fr obenius* norm of A.

For $p = \infty$, we recover the ”operator norm” of A:

$$\|A\|_\infty = \max_{\mathbf{x} \neq 0} \frac{\|A\mathbf{x}\|_2}{\|\mathbf{x}\|_2}$$

LASSO

Least Absolute Shrinkage and Selection Operator uses the minimizer:

$$\begin{aligned}\min_{f \in \operatorname{span}(f_1,\ldots,f_K)} & \|\mathbf{y} - \sum_j c_j f_j(X)\|_2 + \lambda \sum_j |c_j| \\ &= \|\mathbf{y} - \sum_j c_j f_j(X)\|_2 + \lambda \|\mathbf{c}\|_1\end{aligned}$$

LASSO

Least Absolute Shrinkage and Selection Operator uses the minimizer:

$$\begin{aligned}\min_{f \in \mathrm{span}(f_1,\ldots,f_K)} & \|\mathbf{y} - \sum_j c_j f_j(X)\|_2 + \lambda \sum_j |c_j| \\ &= \|\mathbf{y} - \sum_j c_j f_j(X)\|_2 + \lambda \|\mathbf{c}\|_1\end{aligned}$$

Because the 1-norm is not differentiable, there is no slick matrix magic we can apply here.

LASSO

Least Absolute Shrinkage and Selection Operator uses the minimizer:

$$\begin{aligned}\min_{f \in \text{span}(f_1, \ldots, f_K)} \|\mathbf{y} - \sum_j c_j f_j(X)\|_2 + \lambda \sum_j |c_j| \\ = \|\mathbf{y} - \sum_j c_j f_j(X)\|_2 + \lambda \|\mathbf{c}\|_1\end{aligned}$$

Because the 1-norm is not differentiable, there is no slick matrix magic we can apply here.

LASSO solutions are sparse (most of the entries of **c** are sparse.

The Picture

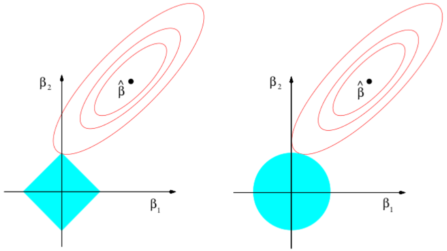

FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the least squares error function.*

Connections to Compressive Sensing

Look for solutions to $A\mathbf{x} = \mathbf{y}$ where A is $k \times N$ for $N >> k$.

Connections to Compressive Sensing

Look for solutions to $A\mathbf{x} = \mathbf{y}$ where A is $k \times N$ for $N >> k$.

There are ”many” solutions, but ”real world” ones tend to be sparse.

Connections to Compressive Sensing

Look for solutions to $A\mathbf{x} = \mathbf{y}$ where A is $k \times N$ for $N >> k$.

There are "many" solutions, but "real world" ones tend to be sparse.

Reference: *A Mathematical Introduction to Compressive Sensing*, S. Foucart, H. Rauhut

Setup

Recall we could express linear regression by

$$\mathbf{y} = X\mathbf{c} + \varepsilon$$

Setup

Recall we could express linear regression by

$$\mathbf{y} = X\mathbf{c} + \varepsilon$$

Another way this is commonly expressed is

$$E[\mathbf{y}|X] = X\mathbf{c}$$

Setup

Recall we could express linear regression by

$$\mathbf{y} = X\mathbf{c} + \varepsilon$$

Another way this is commonly expressed is

$$E[\mathbf{y}|X] = X\mathbf{c}$$

This can be modified to:

$$E[\mathbf{y}|X] = g^{-1}(X\mathbf{c})$$

here g is called a link function.

Common Link Functions

- Logit function (Logistic regression)

$$g(z) = \ln\left(\frac{z}{1-z}\right)$$

- Log function (Poisson regression)

$$g(z) = \ln(z)$$

- Negative inverse (Exponential)

$$g(z) = -\frac{1}{z}$$

- Inverse squared

$$g(z) = \frac{1}{z^2}$$

Gradient Descent: A Detour

Let $F : \mathbb{R}^N \to \mathbb{R}$ be a (differentiable) function.

Gradient Descent: A Detour

Let $F : \mathbb{R}^N \to \mathbb{R}$ be a (differentiable) function.

Then at a point $\mathbf{a} \in \mathbb{R}^N$, the function F changes most rapidly in the direction of

$$\pm \nabla F(\mathbf{a})$$

Gradient Descent: A Detour

Let $F : \mathbb{R}^N \to \mathbb{R}$ be a (differentiable) function.
Then at a point $\mathbf{a} \in \mathbb{R}^N$, the function F changes most rapidly in the direction of

$$\pm \nabla F(\mathbf{a})$$

So if you want to find a local minimum, start at a point $\mathbf{a}_0$ and then compute the next point by

$$\mathbf{a}_{n+1} = \mathbf{a}_n + \lambda_n \nabla F(\mathbf{a}_n).$$

There are many choices for λ_n, but a common one is

$$\lambda_n = \frac{\left|(\mathbf{a}_n - \mathbf{a}_{n-1}) \cdot \nabla\left(F(\mathbf{a}_n) - F(\mathbf{a}_{n-1})\right)\right|}{\left\|\nabla\left(F(\mathbf{a}_n) - F(\mathbf{a}_{n-1})\right)\right\|_2^2}$$

Back to Logistic Regression

Suppose we have $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ where $\mathbf{x}_j \in \mathbb{R}^k$ and $y_j \in \{0, 1\}$.

Back to Logistic Regression

Suppose we have $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ where $\mathbf{x}_j \in \mathbb{R}^k$ and $y_j \in \{0, 1\}$.

Objective: Find the $k - 1$-dimensional hyperplane that best separates the $\mathbf{x}_j$ labeled 0 from those labeled 1.

Back to Logistic Regression

Suppose we have $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ where $\mathbf{x}_j \in \mathbb{R}^k$ and $y_j \in \{0, 1\}$.

Objective: Find the $k - 1$-dimensional hyperplane that best separates the $\mathbf{x}_j$ labeled 0 from those labeled 1.

If g is the logit function, then

$$g^{-1}(y) = \frac{1}{1 - e^{-X\mathbf{c}}}$$

Back to Logistic Regression

Suppose we have $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ where $\mathbf{x}_j \in \mathbb{R}^k$ and $y_j \in \{0, 1\}$.

Objective: Find the $k - 1$-dimensional hyperplane that best separates the $\mathbf{x}_j$ labeled 0 from those labeled 1.

If g is the logit function, then

$$g^{-1}(y) = \frac{1}{1 - e^{-X\mathbf{c}}}$$

Logistic Regression Finale

So we can express

$$P(y = n|X) = \left(\frac{1}{1 - e^{X\mathbf{c}}}\right)^n \left(1 - \left(\frac{1}{1 - e^{X\mathbf{c}}}\right)\right)^{1-n}$$

Logistic Regression Finale

So we can express

$$P(y = n|X) = \left(\frac{1}{1 - e^{X\mathbf{c}}}\right)^n \left(1 - \left(\frac{1}{1 - e^{X\mathbf{c}}}\right)\right)^{1-n}$$

Taking logs and averaging over samples, we get the function

$$F(\mathbf{c}) = \frac{1}{N}\sum_{j=1}^{N} y_j \ln(g^{-1}(\mathbf{x}_j\mathbf{c}) + (1 - y_j)\ln(1 - g^{-1}(\mathbf{x}_j\mathbf{c}))$$

Logistic Regression Finale

So we can express

$$P(y = n|X) = \left(\frac{1}{1 - e^{X\mathbf{c}}}\right)^{n}\left(1 - \left(\frac{1}{1 - e^{X\mathbf{c}}}\right)\right)^{1-n}$$

Taking logs and averaging over samples, we get the function

$$F(\mathbf{c}) = \frac{1}{N}\sum_{j=1}^{N} y_j \ln(g^{-1}(\mathbf{x}_j\mathbf{c}) + (1 - y_j)\ln(1 - g^{-1}(\mathbf{x}_j\mathbf{c}))$$

From here, compute the gradient of F and use gradient descent!