S&DS 365 / 665

Intermediate Machine Learning

Sparse Regression

Friday, January 28

Yale

Topics

- *Regression*
- High dimensional regression
- Sparsity and the lasso

Regression

We observe pairs $(X_1, Y_1), \ldots, (X_n, Y_n)$.

$\mathcal{D} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ is called the *training data*

$Y_i \in \mathbb{R}$ is the *response*; $X_i \in \mathbb{R}^p$ is the *covariate* (or feature) vector

For example, suppose we have n subjects and Y_i is the blood pressure of subject i and $X_i = (X_{i1}, \ldots, X_{ip})$ is a vector of $p = 5{,}000$ gene expression levels for subject i

Remember: $Y_i \in \mathbb{R}$ and $X_i \in \mathbb{R}^p$

$\boxed{\text{Given a new pair } (X, Y)\text{, we want to predict } Y \text{ from } X.}$

Regression

Let $\widehat{Y}$ be a prediction of Y. The *prediction error* or *risk* is

$$R = \mathbb{E}(Y - \widehat{Y})^2$$

where $\mathbb{E}$ is the expected value (mean).

The best predictor is the *regression function*

$$m(x) = \mathbb{E}(Y|X = x) = \int y\, p(y \,|\, x) dy.$$

However, the true regression function $m(x)$ is not known. We need to estimate $m(x)$.

Regression

Given the training data $\mathcal{D} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ we want to construct $\widehat{m}$ to make

$$\text{prediction risk} = R(\widehat{m}) = \mathbb{E}(Y - \widehat{m}(X))^2$$

small. Here, (X, Y) is a new pair.

Bias-variance decomposition

$$R(\widehat{m}) = \int \text{bias}^2(x) p(x) dx + \int \text{var}(x) p(x) + \sigma^2$$

where

$$\begin{aligned} \text{bias}(x) &= \mathbb{E}(\widehat{m}(x)) - m(x) \\ \text{var}(x) &= \text{Variance}(\widehat{m}(x)) \\ \sigma^2 &= \mathbb{E}(Y - m(X))^2 \end{aligned}$$

Bias-Variance Tradeoff

Prediction Risk = Bias^2 + Variance

Prediction methods with low bias tend to have high variance.

Prediction methods with low variance tend to have high bias.

For example, the predictor $\widehat{m}(x) \equiv 0$ has 0 variance but will be terribly biased.

To predict well, we need to balance the bias and the variance.

Bias-Variance Tradeoff

More generally, we need to tradeoff approximation error against estimation error:

$$R(\widehat{f}) - R^* = \underbrace{R(\widehat{f}) - \inf_{f \in \mathcal{F}} R(f)}_{\text{estimation error}} + \underbrace{\inf_{f \in \mathcal{F}} R(f) - R^*}_{\text{approximation error}}$$

where R^* is the smallest possible risk and $\inf_{f \in \mathcal{F}} R(f)$ is smallest possible risk using class of estimators $\mathcal{F}$.

- Approximation error is a generalization of squared bias
- Estimation error is a generalization of variance
- Decomposition holds more generally, even for classification

Linear Regression

Try to find the best linear predictor, that is, a predictor of the form:

$$m(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p.$$

Important: We do *not* assume the true regression function is linear.

Can always define $x_1 = 1$; then the intercept is β_1 and we can write

$$m(x) = \beta_1 x_1 + \cdots + \beta_p x_p = \beta^T x$$

where $\beta = (\beta_1, \ldots, \beta_p)$ and $x = (x_1, \ldots, x_p)$.

Low Dimensional Linear Regression

Assume for now that p (= length of each X_i) is small. To find a good linear predictor we choose β to minimize the *training error*:

$$\text{training error} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \beta^T X_i)^2$$

The minimizer $\widehat{\beta} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_p)$ is called the *least squares estimator*.

Low Dimensional Linear Regression

The least squares estimator is:

$$\widehat{\beta} = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbb{Y}$$

where

$$\mathbb{X}_{n \times p} = \begin{pmatrix} X_{11} & X_{12} & \cdots & X_{1p} \\ X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{np} \end{pmatrix}$$

and

$$\mathbb{Y} = (Y_1, \ldots, Y_n)^T.$$

Exercise: Show this

Low Dimensional Linear Regression

Summary: the least squares estimator is $m(x) = \widehat{\beta}^T x = \sum_j \widehat{\beta}_j x_j$ where

$$\widehat{\beta} = (\mathbb{X}^T\mathbb{X})^{-1}\mathbb{X}^T\mathbb{Y}.$$

When we observe a new X, we predict Y to be

$$\widehat{Y} = \widehat{m}(X) = \widehat{\beta}^T X.$$

We can try to improve this by:

(i) dealing with high dimensions

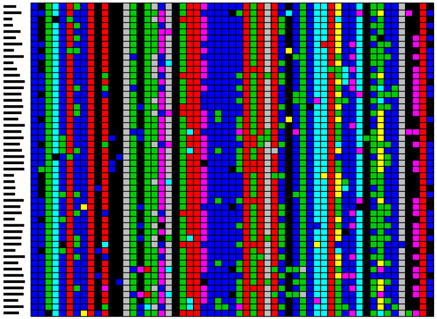
(ii) using something more flexible than linear predictors.

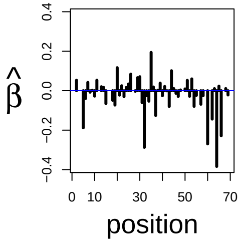
Example

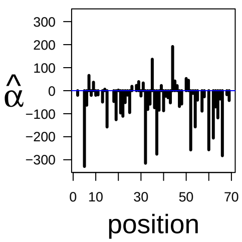
Y = HIV resistance

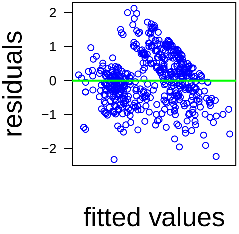
X_j = amino acid in position j of the virus.

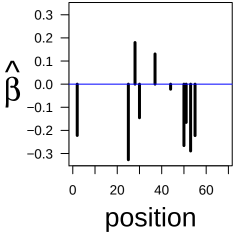
$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_{100} X_{100} + \epsilon$$











Top left: $\widehat{\beta}$

Top right: marginal regression coefficients (one-at-a-time)

Bottom left: $\widehat{Y}_i - Y_i$ versus $\widehat{Y}_i$

Bottom right: a sparse regression (coming up soon)

Topics

- Regression
- *High dimensional regression*
- Sparsity and the lasso

High Dimensional Linear Regression

Now suppose p is large. We even might have $p > n$ (more predictors than data points).

The least squares estimator is not defined since $\mathbb{X}^T\mathbb{X}$ is not invertible. The variance of the least squares prediction is huge.

Recall the bias-variance tradeoff:

Prediction Error = Bias^2 + Variance

We need to increase the bias so that we can decrease the variance.

Ridge Regression

Recall that the least squares estimator minimizes the training error $\frac{1}{n}\sum_{i=1}^{n}(Y_i - \beta^T X_i)^2$.

Instead, we can minimize the *penalized training error*:

Ridge regression

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}\sum_{i=1}^{n}(Y_i - \beta^T X_i)^2 + \lambda\|\beta\|_2^2$$

where $\|\beta\|_2 = \sqrt{\sum_j \beta_j^2}$.

The solution is (exercise!):

$$\widehat{\beta} = (\mathbb{X}^T\mathbb{X} + \lambda I)^{-1}\mathbb{X}^T\mathbb{Y}$$

Ridge Regression

The tuning parameter λ controls the bias-variance tradeoff:

$$\begin{aligned} \lambda = 0 &\quad \Longrightarrow \quad \text{least squares.} \\ \lambda = \infty &\quad \Longrightarrow \quad \widehat{\beta} = 0. \end{aligned}$$

We choose λ to minimize $\widehat{R}(\lambda)$ where $\widehat{R}(\lambda)$ is an estimate of the prediction risk.

Ridge Regression

To estimate the prediction risk, do *not* use training error:

$$R_{\text{training}} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2, \qquad \widehat{Y}_i = X_i^T\widehat{\beta}$$

because it is biased: $\mathbb{E}(R_{\text{training}}) < R(\widehat{\beta})$

Instead, we use *leave-one-out cross-validation*:

1. leave out (X_i, Y_i)
2. find $\widehat{\beta}$
3. predict Y_i: $\widehat{Y}_{(-i)} = \widehat{\beta}^T X_i$
4. repeat for each i

Leave-one-out cross-validation

Can be shown that

$$
\begin{aligned}
\widehat{R}(\lambda) &= \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{(Y_i - \widehat{Y}_i)^2}{(1 - H_{ii})^2} \\
&\approx \frac{R_{\text{training}}}{\left(1 - \frac{p}{n}\right)^2} \\
&\approx R_{\text{training}} + \frac{2\,p\,\widehat{\sigma}^2}{n}
\end{aligned}
$$

where

$$
\begin{aligned}
H &= \mathbb{X}(\mathbb{X}^T\mathbb{X} + \lambda I)^{-1}\mathbb{X}^T \\
p &= \text{trace}(H)
\end{aligned}
$$

Example

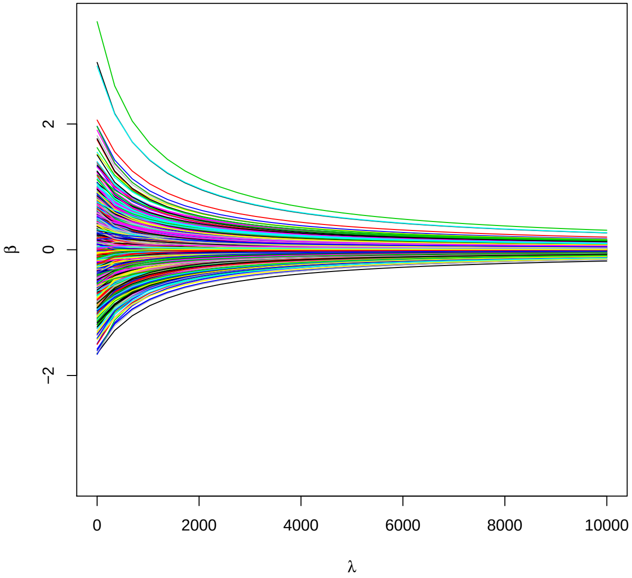
$$Y = 3X_1 + \cdots + 3X_5 + 0X_6 + \cdots + 0X_{1000} + \epsilon$$

$n = 100,\ p = 1{,}000$.

So there are 1000 covariates but only 5 are relevant.

What does ridge regression do in this case?

Ridge Regularization Paths



Sparse Linear Regression

Ridge regression does not take advantage of sparsity.

Maybe only a small number of covariates are important predictors. How do we find them?

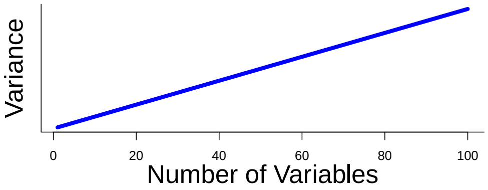
We could fit many submodels (with a small number of covariates) and choose the best one. This is called *model selection*.

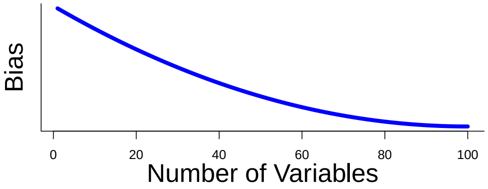
Now the inaccuracy is

prediction error = bias^2 + variance

The bias is the errors due to omitting important variables. The variance is the error due to having to estimate many parameters.

The Bias-Variance Tradeoff





The Bias-Variance Tradeoff

This is a Goldilocks problem: Can't use too few or too many variables.

Have to choose just the right variables.

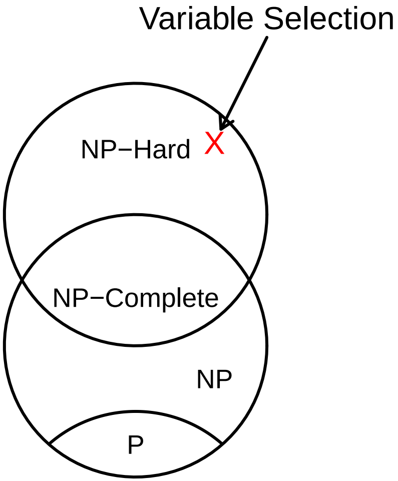
Have to try all models with one variable, two variables,...

If there are p variables then there are 2^p models.

Suppose we have 30,000 genes. We have to search through $2^{30,000}$ models. But $2^{30,000} >$ number of atoms in the universe.

This problem is NP-hard. This was a major bottleneck in statistics for many years.

You are Here



Two things that save us

Two key ideas to make this feasible are sparsity and convex relaxation.

Sparsity: probably only a few genes are needed to predict some disease Y. In other words, of $\beta_1, \ldots, \beta_{30,000}$ most $\beta_j \approx 0$.

But which ones? (Needle in a haystack.)

Convex Relaxation: Replace model search with something easier.

It is the marriage of these two concepts that makes it all work.

Topics

- Regression
- High dimensional regression
- *Sparsity and the lasso*

Sparsity

Consider:

$$\beta = (5, 5, 5, 0, 0, 0, \ldots, 0).$$

This vector is high-dimensional but it is sparse.

Here is a less obvious example:

$$\beta = (50, 12, 6, 3, 2, 1.4, 1, 0.8., 0.6, 0.5, \ldots)$$

It turns out that, if the β_j's die off fairly quickly, then β behaves a like a sparse vector.

Sparsity

We measure the (lack of) sparsity of $\beta = (\beta_1, \dots, \beta_p)$ with the q-norm

$$\|\beta\|_q = \left(|\beta_1|^q + \cdots + |\beta_p|^q\right)^{1/q} = \left(\sum_j |\beta_j|^q\right)^{1/q}.$$

Which values of q measure (lack of) sparsity?

sparse:	$a =$	1	0	0	$\cdots$	0
not sparse:	$b =$	$1/\sqrt{p}$	$1/\sqrt{p}$	$1/\sqrt{p}$	$\cdots$	$1/\sqrt{p}$

	✓ $q = 0$	✓ $q = 1$	× $q = 2$
$\|a\|_q$	1	1	1
$\|b\|_q$	p	$\sqrt{p}$	1

Lesson: Need to use $q \leq 1$ to measure sparsity.

Sparsity

So we estimate $\beta = (\beta_1, \ldots, \beta_p)$ by minimizing

$$\sum_{i=1}^{n} \Big(Y_i - (\beta_0 + \beta_1 X_{i1} + \cdots + \beta_p X_{ip}) \Big)^2$$

subject to the constraint that β is sparse i.e. $\|\beta\|_q \leq$ small.

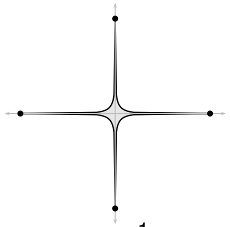
Can we do this minimization?

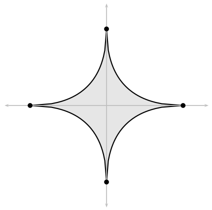
If we use $q = 0$ this is same as searching through all 2^p models.

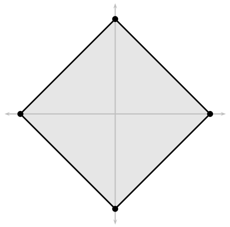
What about other values of q?

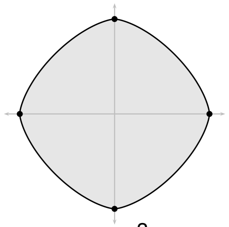
What does the set $\{\beta : \ \|\beta\|_q \leq \text{small}\}$ look like?

The set $\|\beta\|_q \leq 1$ when $p = 2$


$q = \frac{1}{4}$


$q = \frac{1}{2}$


$q = 1$


$q = \frac{3}{2}$

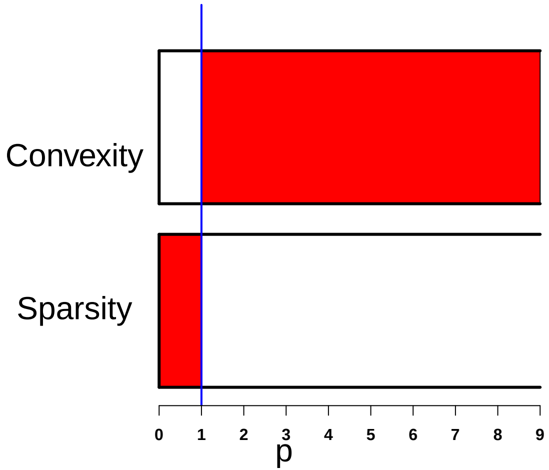
Sparsity Meets Convexity

We need these sets to have a nice shape (convex). If so, the minimization is no longer NP-hard. In fact, it is easy.

Sensitivity to sparsity:	$q \leq 1$ (actually, $q < 2$ suffices)
Convexity (niceness):	$q \geq 1$

This means we should use $q = 1$.

Where Sparsity and Convexity Meet



Sparsity Meets Convexity

Lasso regression

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}\sum_{i=1}^{n}(Y_i - \beta^T X_i)^2 + \lambda\|\beta\|_1$$

where $\|\beta\|_1 = \sum_j |\beta_j|$.

Invented by Rob Tibshirani in 1996. (Related work by Donoho and others around the same time).

Lasso

The result is an estimated vector

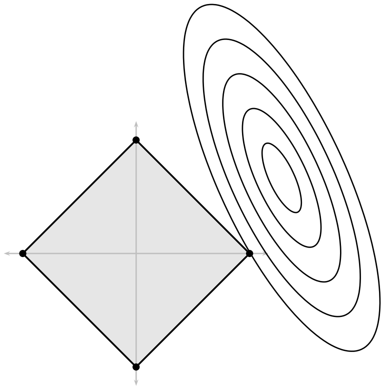
$$\widehat{\beta}_1, \ldots, \widehat{\beta}_p$$

Most are 0!

Magically, we have done model selection without searching (thanks to sparsity plus convexity).

The next picture explains why some $\widehat{\beta}_j = 0$.

Sparsity: How corners create sparse estimators



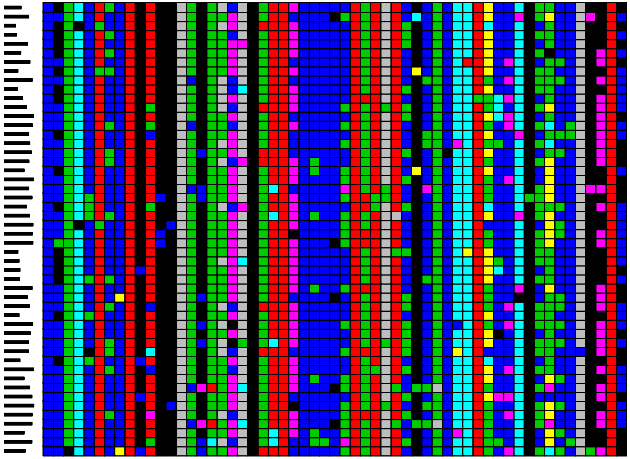
Standardization

Note that, for both lasso and ridge regression, it’s important to standarize the features — scale so that the standard deviation of each feature (column) is a constant.

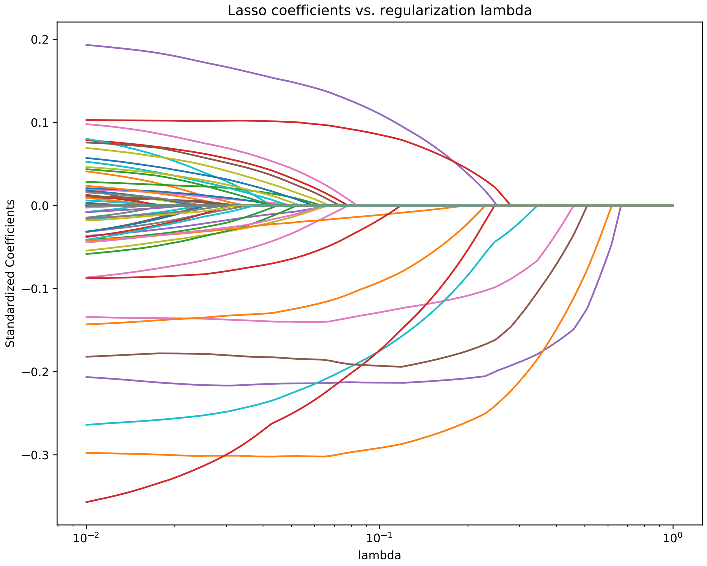
Let’s go to the notebook!

The lasso: HIV example again

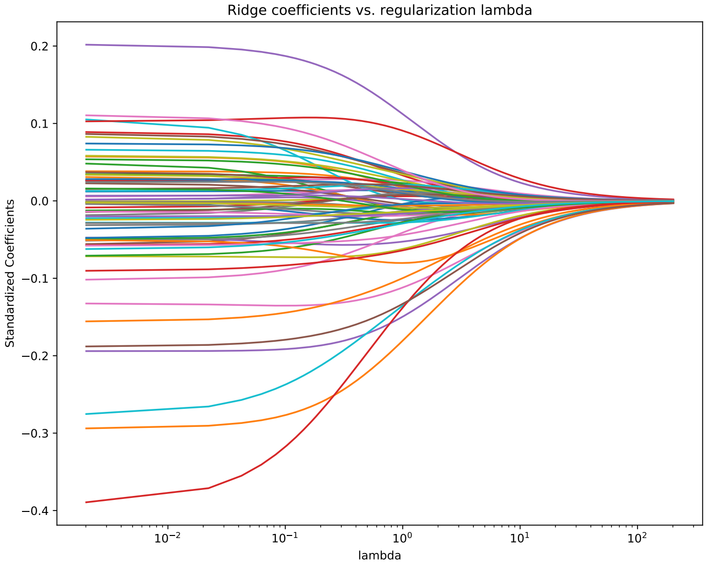
- Y is resistance to HIV drug.
- X_j = amino acid in position j of the virus.
- $p = 99$, $n \approx 100$.



The lasso: HIV example

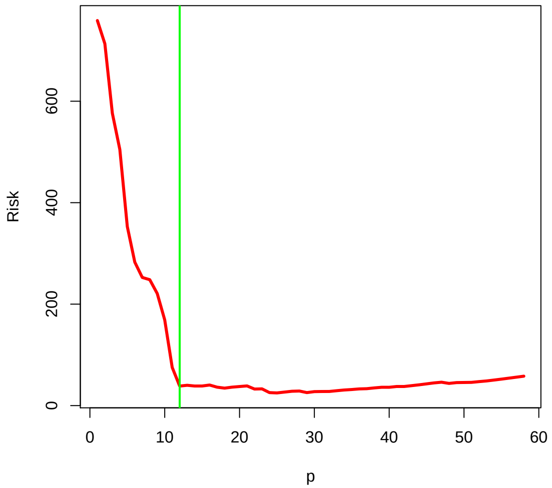


Contrast with ridge regression

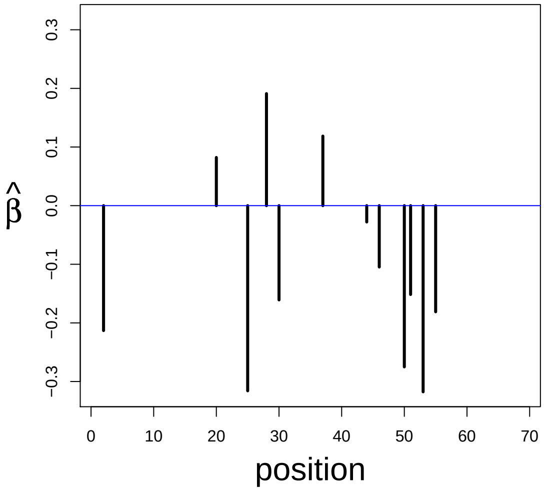


Selecting λ

We choose the sparsity level by estimating prediction error.



The lasso: An example



Sparsity and convexity

To summarize: we penalize the sums of squares with

$$\|\beta\|_q = \left(\sum_j |\beta_j|^q \right)^{1/q}.$$

To get a sparse answer: $q < 2$.

To get a convex problem: $q \geq 1$.

So $q = 1$ works.

The marriage of sparsity and convexity is one of the biggest developments in statistics and machine learning.

The lasso

- $\widehat{\beta}(\lambda)$ is called the lasso estimator. Then define

$$\widehat{S}(\lambda) = \left\{ j :\ \widehat{\beta}_j(\lambda) \neq 0 \right\}.$$

- After you find $\widehat{S}(\lambda)$, you should re-fit the model by doing least squares on the sub-model $\widehat{S}(\lambda)$.

The lasso

Choose λ by risk estimation.

Re-fit the model with the non-zero coefficients. Then apply leave-one-out cross-validation:

$$\widehat{R}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{(Y_i - \widehat{Y}_i)^2}{(1 - H_{ii})^2} \approx \frac{1}{n}\frac{RSS}{\left(1 - \frac{s}{n}\right)^2}$$

where H is the hat matrix and $s = \#\{j : \widehat{\beta}_j \neq 0\}$.

Choose $\widehat{\lambda}$ to minimize $\widehat{R}(\lambda)$.

The lasso

The complete steps are:

1. Find $\widehat{\beta}(\lambda)$ and $\widehat{S}(\lambda)$ for each λ.
2. Choose $\widehat{\lambda}$ to minimize estimated risk.
3. Let $\widehat{S}$ be the selected variables.
4. Let $\widehat{\beta}$ be the least squares estimator using only $\widehat{S}$.
5. Prediction: $\widehat{Y} = X^T\widehat{\beta}$.

Some convexity theory for the lasso

Consider a simpler model than regression: Suppose $Y \sim N(\mu, 1)$. Let $\widehat{\mu}$ minimize

$$A(\mu) = \frac{1}{2}(Y - \mu)^2 + \lambda|\mu|.$$

How do we minimize $A(\mu)$?

- Since A is convex, we set the subderivative = 0. Recall that c is a *subderivative* of $f(x)$ at x_0 if

$$f(x) - f(x_0) \geq c(x - x_0).$$

- The *subdifferential* $\partial f(x_0)$ is the set of subderivatives. Also, x_0 minimizes f if and only if $0 \in \partial f$.

ℓ_1 and soft thresholding

- If $f(\mu) = |\mu|$ then

$$\partial f = \begin{cases} \{-1\} & \mu < 0 \\ [\,-1\,,\,1\,] & \mu = 0 \\ \{+1\} & \mu > 0. \end{cases}$$

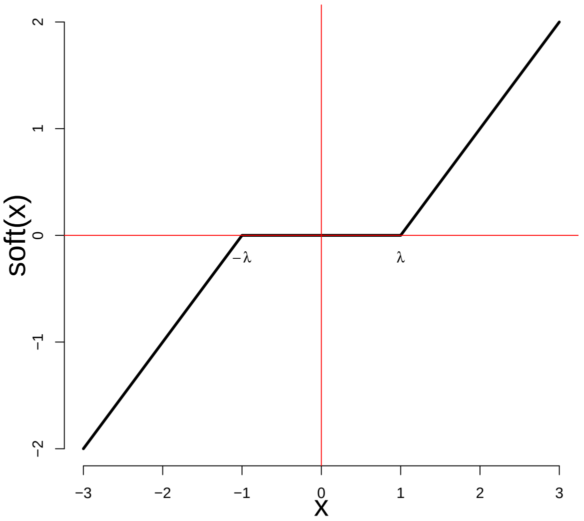
- Hence,

$$\partial A = \begin{cases} \{\mu - Y - \lambda\} & \mu < 0 \\ \{\mu - Y + \lambda z :\ -1 \le z \le 1\} & \mu = 0 \\ \{\mu - Y + \lambda\} & \mu > 0. \end{cases}$$

ℓ_1 and soft thresholding

- $\widehat{\mu}$ minimizes $A(\mu)$ if and only if $0 \in \partial A$.
- So
$$\widehat{\mu} = \begin{cases} Y + \lambda & Y < -\lambda \\ 0 & -\lambda \leq Y \leq \lambda \\ Y - \lambda & Y > \lambda. \end{cases}$$
- This can be written as
$$\widehat{\mu} = \text{soft}(Y, \lambda) \equiv \text{sign}(Y)\ (|Y| - \lambda)_+.$$

ℓ_1 and soft thresholding



The lasso: Computing $\widehat{\beta}$

To minimize $\sum_i (Y_i - \beta^T X_i)^2 + \lambda \|\beta\|_1$ by *coordinate descent*:

- Set $\widehat{\beta} = (0, \ldots, 0)$ then iterate the following
- for $j = 1, \ldots, p$:
  - set $R_i = Y_i - \sum_{s \neq j} \widehat{\beta}_s X_{si}$
  - Set $\widehat{\beta}_j$ to be least squares fit of R_i's on X_j.
  - $\widehat{\beta}_j \leftarrow \text{soft}(\widehat{\beta}_j, \lambda / \sum_i X_{ij}^2)$
- Then use least squares $\widehat{\beta}$ on selected subset S.

Variations on the lasso

- "Elastic net": minimize

$$\sum_{i=1}^{n}(Y_i - \beta^T X_i)^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2$$

- Group lasso:

$$\beta = (\underbrace{\beta_1, \ldots, \beta_k}_{v_1}, \ldots, \underbrace{\beta_t, \ldots, \beta_p}_{v_m})$$

minimize:

$$\sum_{i=1}^{n}(Y_i - \beta^T X_i)^2 + \lambda \sum_{j=1}^{m} \|v_j\|$$

Summary

- For low dimensional (linear) prediction, we can use least squares.
- For high dimensional linear regression, we face a bias-variance tradeoff: omitting too many variables causes bias while including too many variables causes high variance.
- The key is to select a good subset of variables.
- The *lasso* (ℓ_1-regularized least squares) is a fast way to select variables.
- If there are good, sparse linear predictors, lasso will work well.