Mathematical techniques in data science

Lecture 15: Computing the lasso estimator

March 22th, 2019

Schedule

Week	Chapter
1	Chapter 2: Intro to statistical learning
3	Chapter 4: Classification
4	Chapter 9: Support vector machine and kernels
5, 6	Chapter 3: Linear regression
7	Chapter 8: Tree-based methods + Random forest
8	
9	Neural network
12	PCA $\rightarrow$ Manifold learning
11	Clustering: K-means $\rightarrow$ Spectral Clustering
10	Bootstrap + Bayesian methods + UQ
13	Reinforcement learning/Online learning/Active learning
14	Project presentation

Settings

- Linear model

$$Y = \beta^{(0)} + \beta^{(1)}X^{(1)} + \beta^{(2)}X^{(2)} + \dots \beta^{(p)}X^{(p)} + \epsilon$$

- $\mathbf{Y} \in \mathbb{R}^{n\times 1}, \quad \mathbf{X} \in \mathbb{R}^{n\times(p+1)}$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} 1 & | & | & \dots & | \\ \dots & x^{(1)} & x^{(2)} & \dots & x^{(p)} \\ 1 & | & | & \dots & | \end{bmatrix}$$

where $\left(x_j^{(i)}\right)_{j=1}^{n}$ are the observations of $X^{(i)}$.

Lasso

- The Lasso (Least Absolute Shrinkage and Selection Operator)

$$\hat{\beta}^{lasso} = \min_{\beta} \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\beta\|_2^2 + \lambda \sum_{j=1}^{p} |\beta^{(j)}|$$

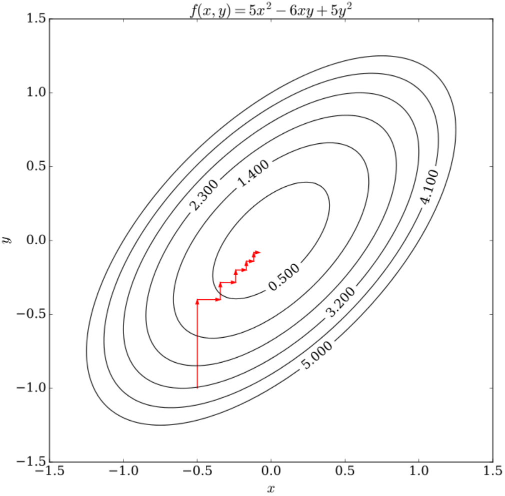
- lasso is often used in high dimensions
- cross-validation involves solving many lasso problems
- how can we compute the lasso estimator efficiently?

Coordinate descent

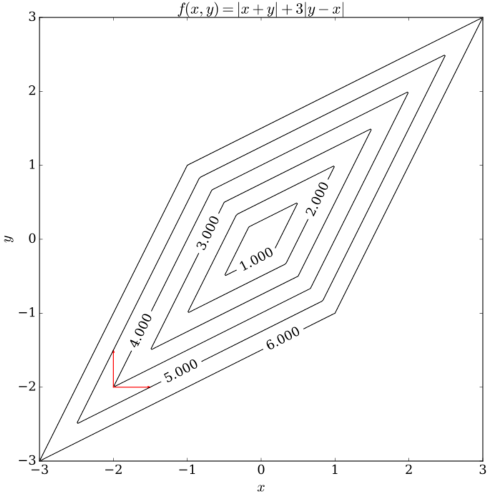
Objective: Minimize a function $f : \mathbb{R}^n \to \mathbb{R}$.
Strategy: Minimize each coordinate separately while cycling through the coordinates.

$$\begin{aligned}
x_1^{(k+1)} &= \operatorname*{argmin}_x f(x, x_2^{(k)}, x_3^{(k)}, \dots, x_p^{(k)}) \\
x_2^{(k+1)} &= \operatorname*{argmin}_x f(x_1^{(k+1)}, x, x_3^{(k)}, \dots, x_p^{(k)}) \\
x_3^{(k+1)} &= \operatorname*{argmin}_x f(x_1^{(k+1)}, x_2^{(k+1)}, x, x_4^{(k)}, \dots, x_p^{(k)}) \\
&\vdots \\
x_p^{(k+1)} &= \operatorname*{argmin}_x f(x_1^{(k+1)}, x_2^{(k+1)}, \dots, x_{p-1}^{(k+1)}, x).
\end{aligned}$$

Coordinate descent



Coordinate descent: may not converge



Coordinate descent

- in general, may not converge to the optimum
- works for lasso

Theorem

Suppose

$$f(x_1, \ldots, x_p) = f_0(x_1, \ldots, x_p) + \sum_{i=1}^{p} f_i(x_i)$$

where

- f_0 *is convex and twice differentiable*
- f_i *is convex* $(i = 1, \ldots, p)$
- f *is continuous and the set* $X_0 = \{x : f(x) \leq f(x^0)\}$ *is compact*

the the coordinate descent starting at x^0 *converges to the optimum*

Lasso

- The Lasso (Least Absolute Shrinkage and Selection Operator)

$$\hat{\beta}^{lasso} = \min_{\beta} \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\beta\|_2^2 + \lambda \sum_{j=1}^{p} |\beta^{(j)}|$$

- lasso is often used in high dimensions
- cross-validation involves solving many lasso problems
- how can we compute the lasso estimator efficiently?

Taking derivative: the differentiable part

- Minimize the residual sum of squares

$$L(\beta) = = \frac{1}{2}\sum_{i=1}^{n}(y_i - f(x_i))^2$$
$$= \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \beta^{(0)} - \beta^{(1)}x_i^{(1)} - \beta^{(2)}x_i^{(2)} - \ldots - \beta^{(p)}x_i^{(p)}\right)^2$$

- Taking derivative

$$\frac{\partial P_1}{\partial \beta^{(j)}}(\beta) = \sum_{i=1}^{n}(x_i\beta - y_i)x_i^{(j)}$$
$$= [x^{(j)}]^T\left(\beta^{(0)} + \beta^{(1)}x_i^{(1)} + \beta^{(2)}x_i^{(2)} + \ldots + \beta^{(p)}x_i^{(p)} - y_i\right)$$

Taking derivative: the non-differentiable part

$$\frac{\partial P_2}{\partial \beta^{(j)}}(\beta) = \begin{cases} 1 \quad \text{if} \quad \beta^{(j)} > 0 \\ s \in [-1, 1] \quad \text{if} \quad \beta^{(j)} = 0 \\ -1 \quad \text{if} \quad \beta^{(j)} < 0 \end{cases}$$

Critical point

$$\frac{\partial P}{\partial \beta^{(j)}}(\beta) = 0$$

Case 1: If $\beta^{(j)} > 0$ and

$$[x^{(j)}]^T \left(\beta^{(0)} + \beta^{(1)} x_i^{(1)} + \beta^{(2)} x_i^{(2)} + \ldots + \beta^{(p)} x_i^{(p)} - y_i\right) + \alpha = 0$$

This is equivalent to

$$\begin{aligned} \beta^{(j)} &= \frac{[x^{(j)}]^T \left(y_i - [X^{(-j)}]^T[\beta^{(-j)}]\right) - \alpha}{[x^{(j)}]^T[x^{(j)}]} \\ &= A^* - \frac{\alpha}{\|x^{(j)}\|^2} \end{aligned}$$

Critical point

$$\frac{\partial P}{\partial \beta^{(j)}}(\beta) = 0$$

Case 2: If $\beta^{(j)} < 0$ and

$$[x^{(j)}]^T \left(y_i - \beta^{(0)} - \beta^{(1)} x_i^{(1)} - \beta^{(2)} x_i^{(2)} - \ldots - \beta^{(p)} x_i^{(p)} \right) - \alpha = 0$$

This is equivalent to

$$\begin{aligned} \beta^{(j)} &= \frac{[x^{(j)}]^T \left(y_i - [X^{(-j)}]^T [\beta^{(-j)}] \right) + \alpha}{[x^{(j)}]^T [x^{(j)}]} \\ &= A^* + \frac{\alpha}{\|x^{(j)}\|^2} \end{aligned}$$

Critical point

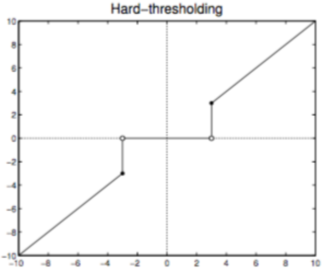
Thus

$$\beta_*^{(j)} = \begin{cases} A^* - \frac{\alpha}{\|x^{(j)}\|^2} & \text{if } A^* - \frac{\alpha}{\|x^{(j)}\|^2} > 0 \\ 0 & \text{otherwise} \\ A^* + \frac{\alpha}{\|x^{(j)}\|^2} & \text{if } A^* + \frac{\alpha}{\|x^{(j)}\|^2} < 0 \end{cases}$$

Coordinate descent: may not converge

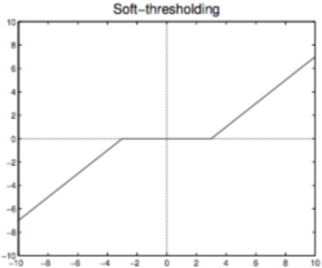
Hard-thresholding:

$$\eta^H_\epsilon(x) = x\mathbf{1}_{|x|>\epsilon}.$$



Soft-thresholding:

$$\eta^S_\epsilon(x) = \mathrm{sgn}(x)(|x|-\epsilon)_+$$



Note: soft-thresholding shrinks the value until it hits zero (and then leaves it at zero).

$$\eta^S_\epsilon(x) = \begin{cases} x-\epsilon & \text{if } x>\epsilon \\ x+\epsilon & \text{if } x<-\epsilon \\ 0 & \text{if } -\epsilon \le x \le \epsilon \end{cases}.$$