



Center for  
Data Science

# DS-GA 3001.007

# Introduction to Machine Learning

## Lecture 5

## Regularization - Ridge and Lasso

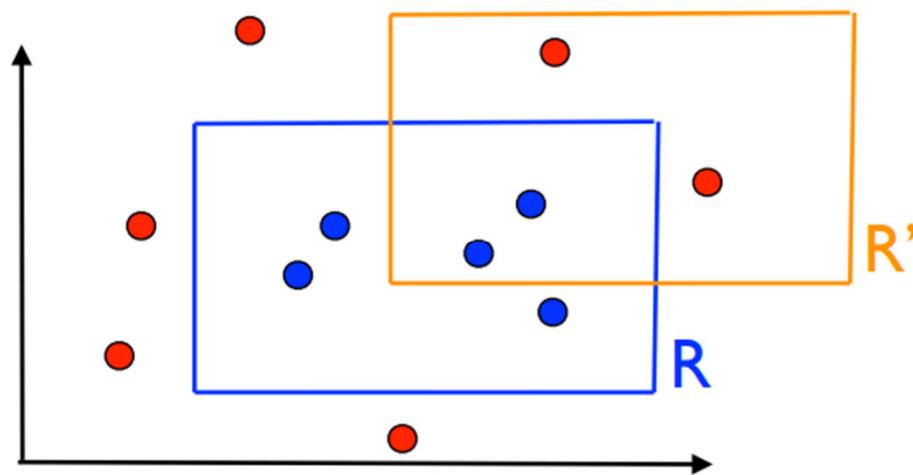
# Reminders

- ▶ Homework 3
  - ▶ Due on Sunday October 20
- ▶ Midterm
  - ▶ In class on Wednesday October 23
- ▶ Project
  - ▶ Proposal due October 31
  - ▶ Milestone due November 28
  - ▶ Report due December 15

advanced  
apply  
efficient  
knowledge  
science  
build  
good  
implement  
basics  
solye  
understanding  
gain  
model  
make  
expect  
work  
**machine**  
**learning**  
project  
deep  
hope  
idea  
datam  
understand  
create  
great  
learn  
mathematics  
problem  
technique  
research  
class  
basic  
theoretical  
concept  
Common  
experience  
design  
better  
fundamental  
academic

## Bound Difference In Sample and Out of Sample

- **Problem:** learn unknown axis-aligned rectangle  $R$  using as small a labeled sample as possible.



- **Hypothesis:** rectangle  $R'$ . In general, there may be false positive and false negative points.

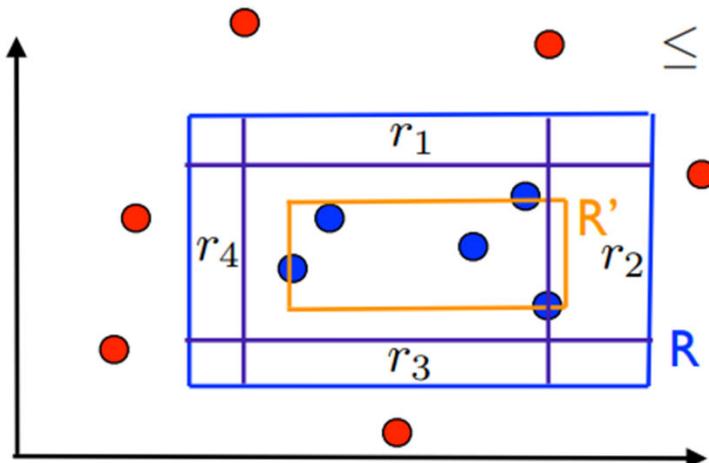
## Bound Difference In Sample and Out of Sample

- Errors can only occur in  $R - R'$ . Thus (geometry),

$R(R') > \epsilon \Rightarrow R' \text{ misses at least one region } r_i.$

- Therefore,  $\Pr[R(R') > \epsilon] \leq \Pr[\bigcup_{i=1}^4 \{R' \text{ misses } r_i\}]$

$$\leq \sum_{i=1}^4 \Pr[\{R' \text{ misses } r_i\}]$$
$$\leq 4(1 - \frac{\epsilon}{4})^m \leq 4e^{-\frac{m\epsilon}{4}}.$$



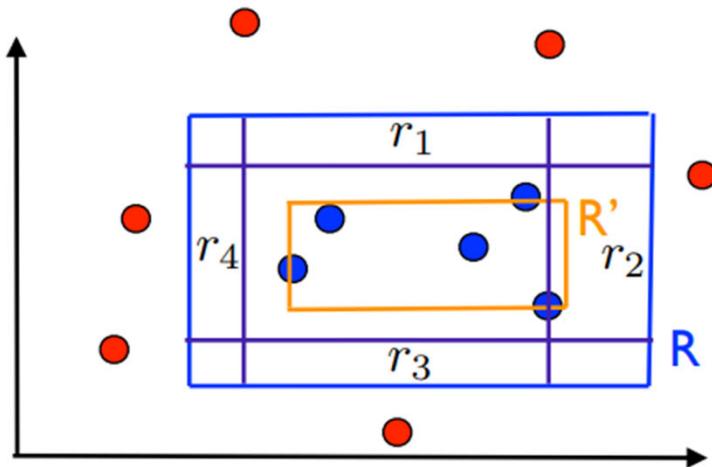
## Bound Difference In Sample and Out of Sample

- Set  $\delta > 0$  to match the upper bound:

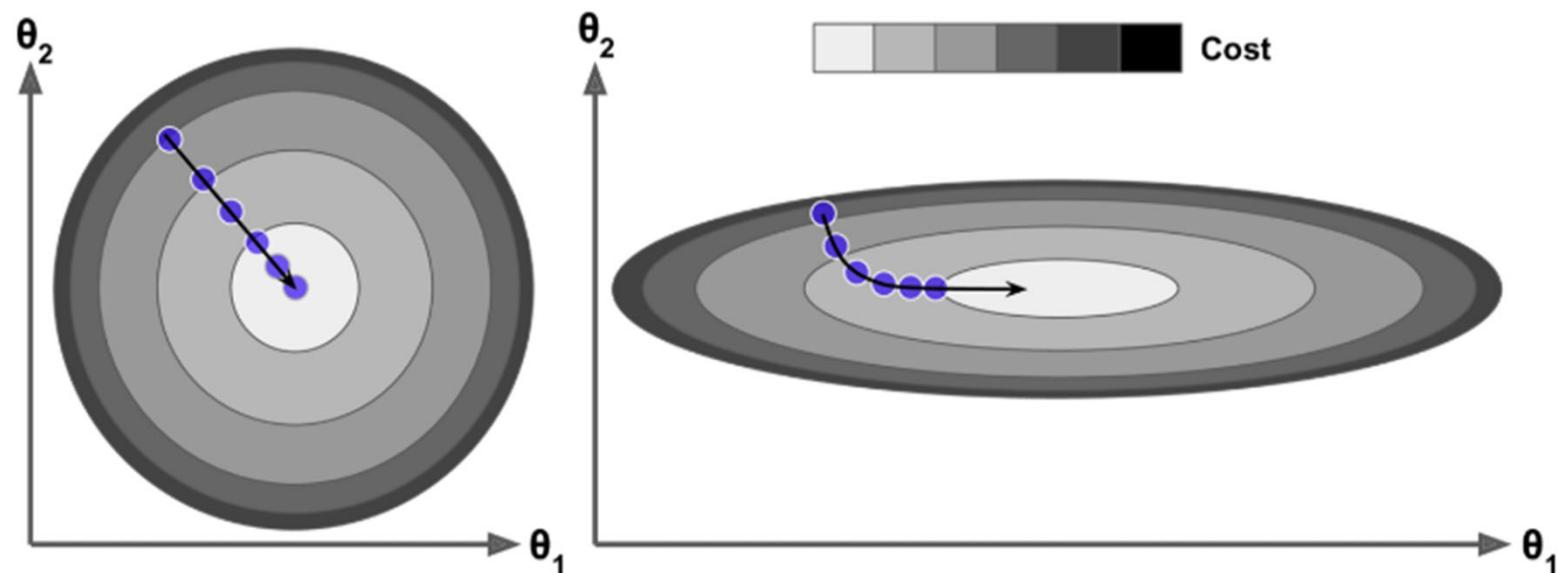
$$4e^{-\frac{m\epsilon}{4}} \leq \delta \Leftrightarrow m \geq \frac{4}{\epsilon} \log \frac{4}{\delta}.$$

- Then, for  $m \geq \frac{4}{\epsilon} \log \frac{4}{\delta}$ , with probability at least  $1 - \delta$ ,

$$R(R') \leq \epsilon.$$



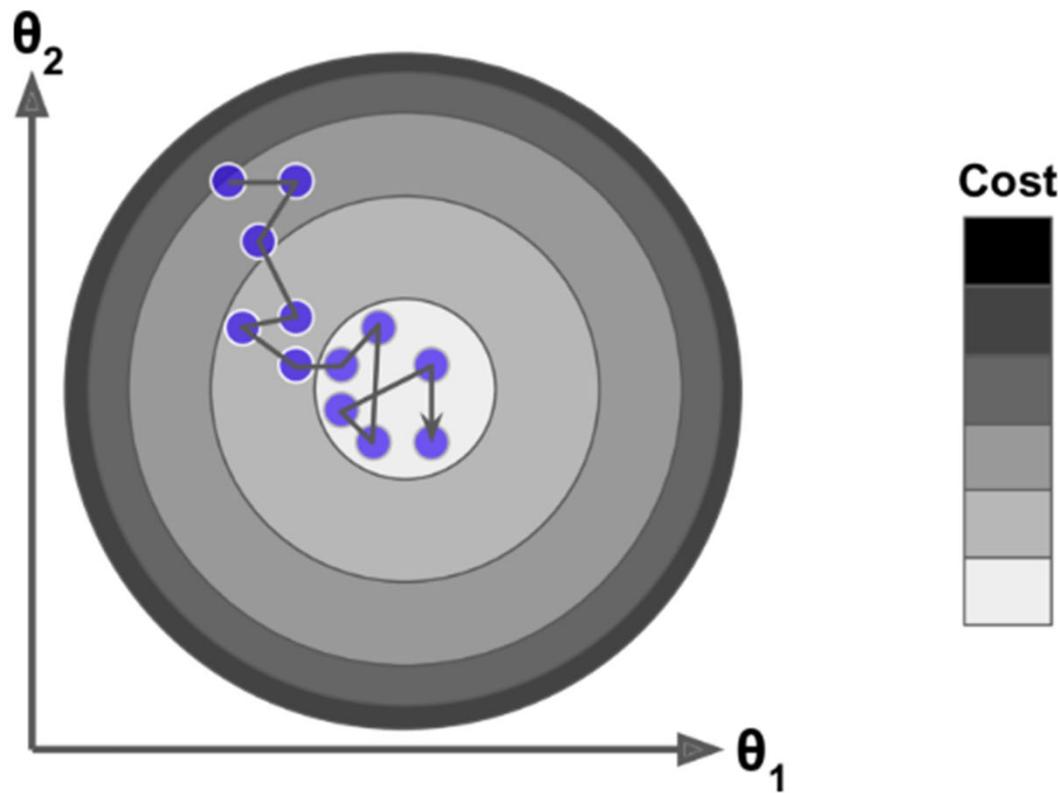
# Gradient Descent



## Gradient Descent

- Initialize  $x = 0$
- repeat
  - $x \leftarrow x - \underbrace{\eta}_{\text{step size}} \nabla f(x)$
- until stopping criterion satisfied

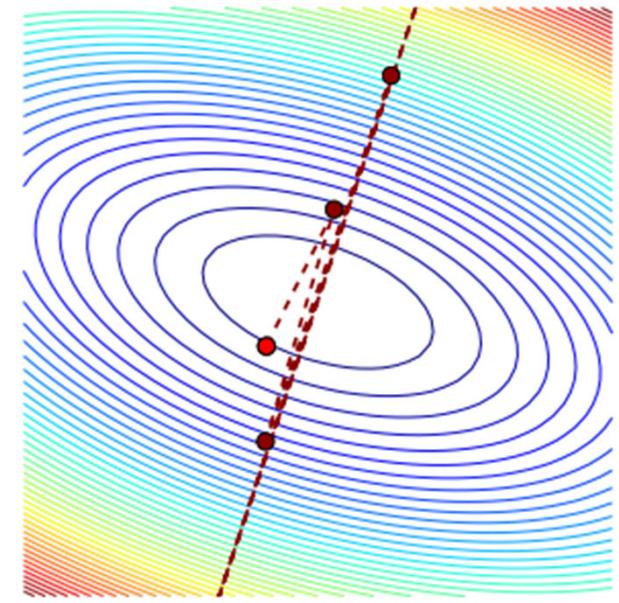
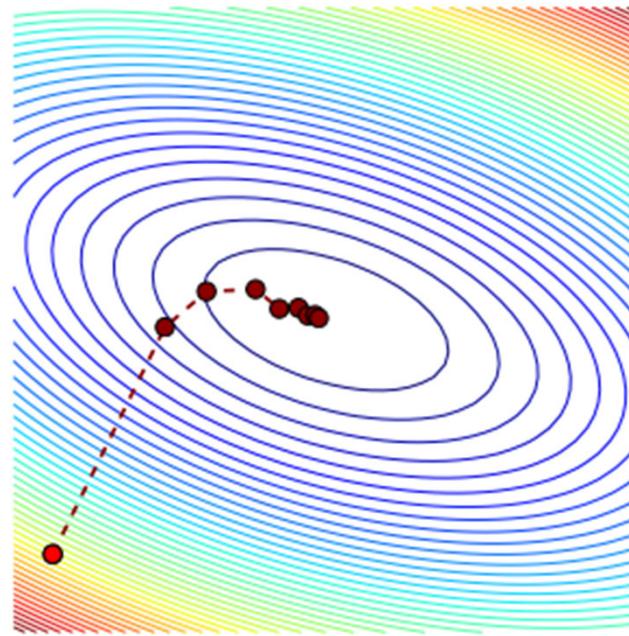
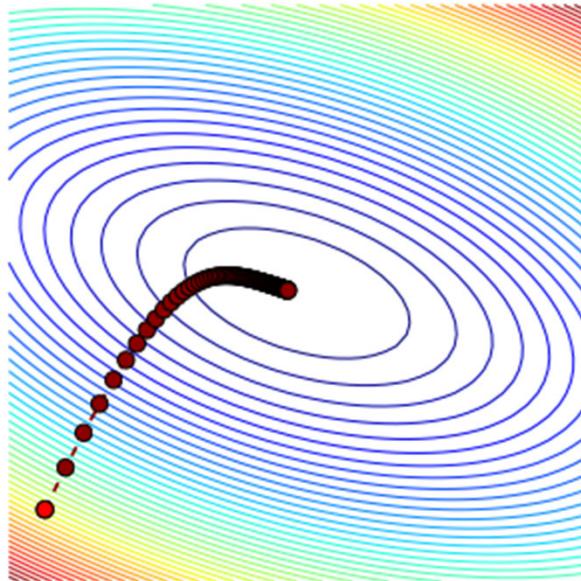
# Stochastic Gradient Descent



## Stochastic Gradient Descent

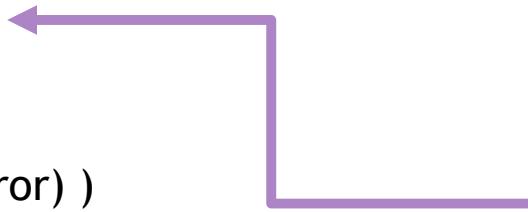
- initialize  $w = 0$
- repeat
  - randomly choose training point  $(x_i, y_i) \in \mathcal{D}_n$
  - $w \leftarrow w - \eta \underbrace{\nabla_w \ell(f_w(x_i), y_i)}_{\text{Grad(Loss on i'th example)}}$

# Convergence Analysis?



# Convergence Analysis?

- ▶ Fixed Step Size
  - ▶ Learning rate  $1/(\max \text{ of derivative})$
  - ▶ Number Iterations  $O(1/\text{error})$
- ▶ Varying Step Size
  - ▶ Back-tracking Line Search
    - ▶ Number Iterations  $O(1/\text{error})$  with better constant
  - ▶ Strongly Convex Functions
    - ▶ Learning rate  $1/(\text{iteration})$
    - ▶ Number Iterations  $O(\log(1/\text{error}))$



LECTURE 7!

# Perceptron Algorithm as SGD

PERCEPTRON( $\mathbf{w}_0$ )

```
1    $\mathbf{w}_1 \leftarrow \mathbf{w}_0$        $\triangleright$  typically  $\mathbf{w}_0 = \mathbf{0}$ 
2   for  $t \leftarrow 1$  to  $T$  do
3       RECEIVE( $\mathbf{x}_t$ )
4        $\hat{y}_t \leftarrow \text{sgn}(\mathbf{w}_t \cdot \mathbf{x}_t)$ 
5       RECEIVE( $y_t$ )
6       if ( $\hat{y}_t \neq y_t$ ) then
7            $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \mathbf{x}_t$      $\triangleright$  more generally  $\eta y_t \mathbf{x}_t$ ,  $\eta > 0$ .
8       else  $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t$ 
9   return  $\mathbf{w}_{T+1}$ 
```

# Perceptron Algorithm as SGD

- Take

$$F(\mathbf{w}) = \frac{1}{T} \sum_{t=1}^T \max(0, -y_t(\mathbf{w} \cdot \mathbf{x}_t))$$

## Perceptron Algorithm as SGD

► Take

$$F(\mathbf{w}) = \frac{1}{T} \sum_{t=1}^T \max(0, -y_t(\mathbf{w} \cdot \mathbf{x}_t))$$

► Set

---

$$\tilde{F}(\mathbf{w}, \mathbf{x}) = \max(0, -f(\mathbf{x})(\mathbf{w} \cdot \mathbf{x}))$$

# Perceptron Algorithm as SGD

► Take

$$F(\mathbf{w}) = \frac{1}{T} \sum_{t=1}^T \max(0, -y_t(\mathbf{w} \cdot \mathbf{x}_t))$$

► Set

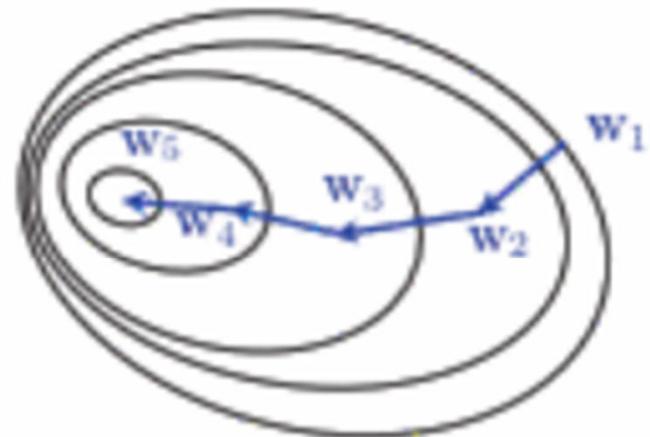
$$\tilde{F}(\mathbf{w}, \mathbf{x}) = \max(0, -f(\mathbf{x})(\mathbf{w} \cdot \mathbf{x}))$$

► Update Guess

$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t - \eta \nabla_{\mathbf{w}} \tilde{F}(\mathbf{w}_t, \mathbf{x}_t) & \text{if } \mathbf{w} \mapsto \tilde{F}(\mathbf{w}, \mathbf{x}_t) \text{ differentiable at } \mathbf{w}_t \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$

## Perceptron Algorithm as SGD

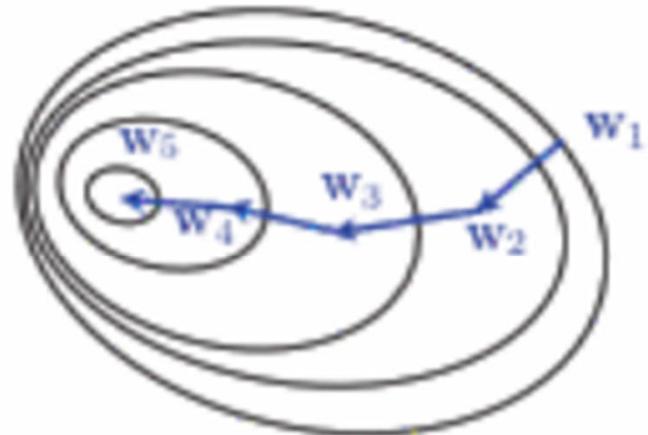
$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t + \eta y_t \mathbf{x}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0; \\ \mathbf{w}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0; \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$



## Perceptron Algorithm as SGD

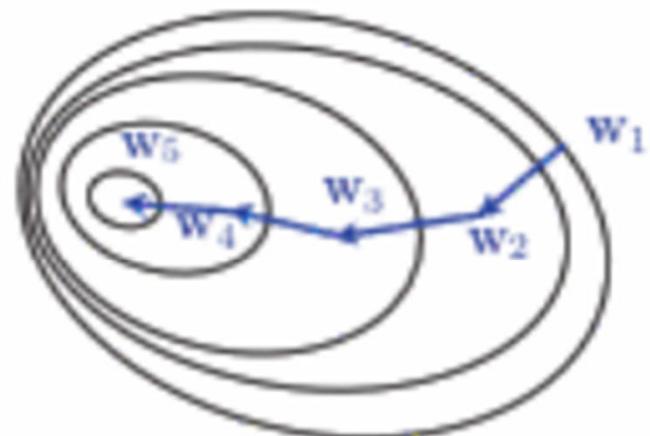
$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t + \eta y_t \mathbf{x}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0; \\ \mathbf{w}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0; \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$

$$\nabla_{\mathbf{w}} \tilde{F}(\mathbf{w}, \mathbf{x}_t) = -y \mathbf{x}_t \quad \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0$$



## Perceptron Algorithm as SGD

$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t + \eta y_t \mathbf{x}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0; \\ \mathbf{w}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0; \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$



$$\nabla_{\mathbf{w}} \tilde{F}(\mathbf{w}, \mathbf{x}_t) = -y \mathbf{x}_t \quad \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0$$

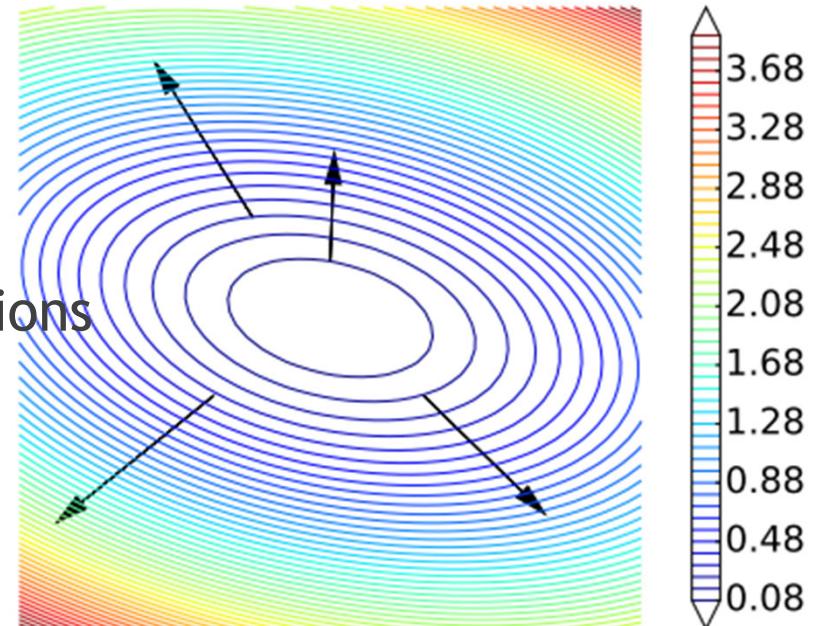
$$\nabla_{\mathbf{w}} \tilde{F}(\mathbf{w}, \mathbf{x}_t) = 0 \quad \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0.$$

# Objectives

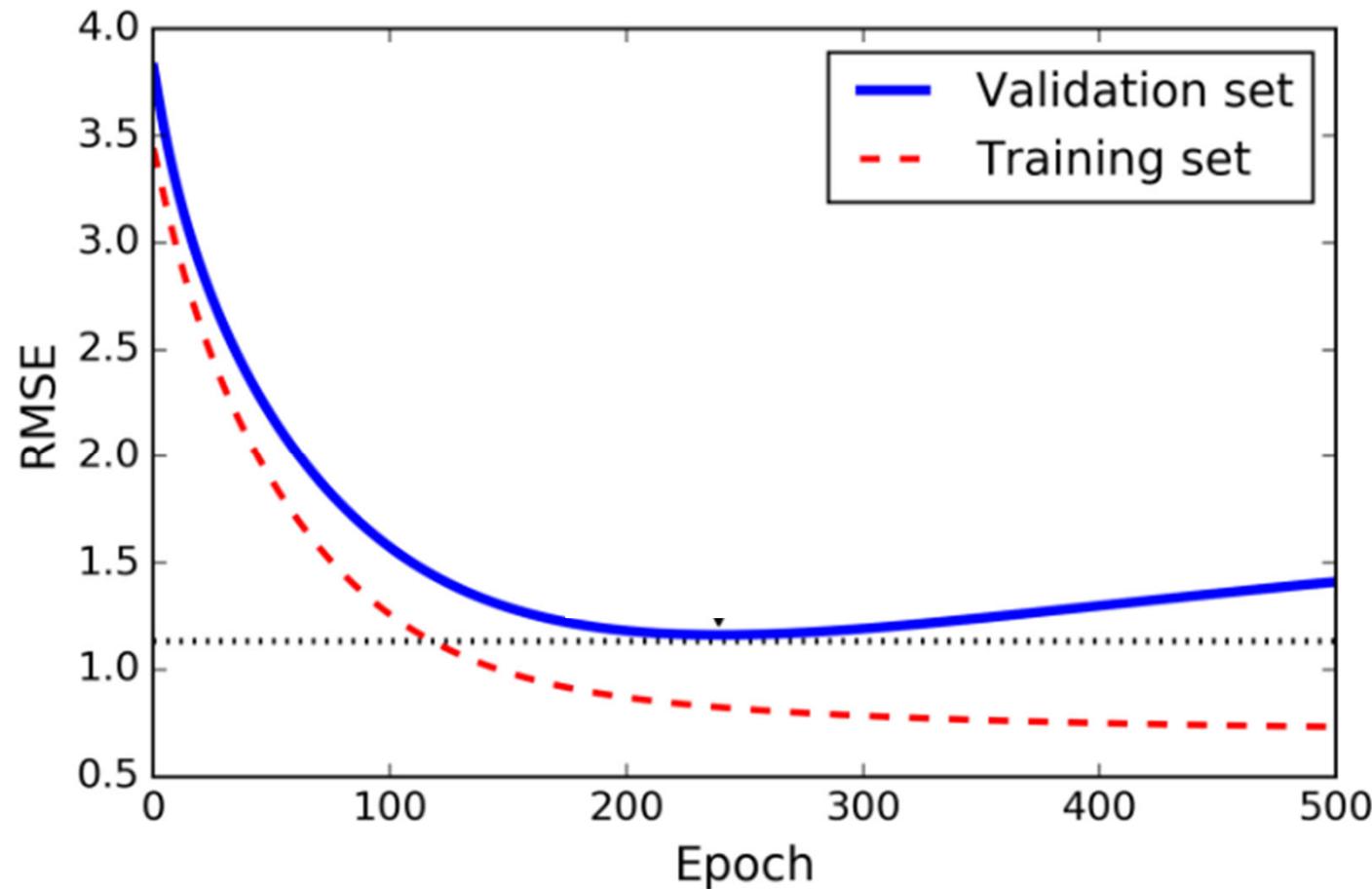
- ▶ Overfitting
  - ▶ How to avoid overfitting by using regularization?
- ▶ Regularization
  - ▶ Formulate a linear estimation problem with a regularization
  - ▶ Compute an L1-regularized or L2-regularized estimate
  - ▶ Compute optimal regularization level using cross validation
- ▶ Compare and Contrast

# Agenda

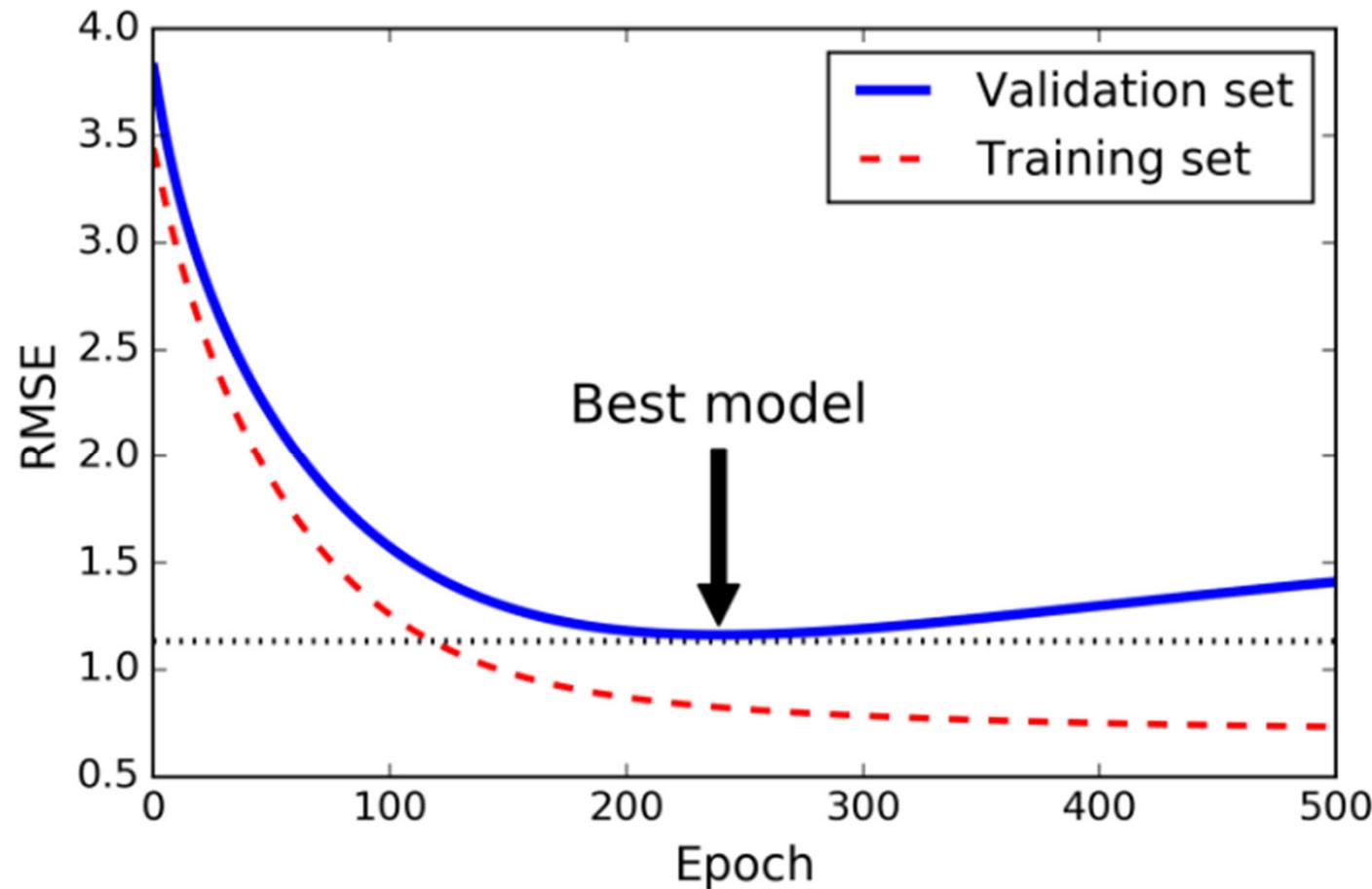
- ▶ Lesson
  - ▶ Ridge Regression and Normal Equations
  - ▶ Lasso Regression and Sparsity
  - ▶ Similarities and Differences
- ▶ Demo
  - ▶ Implementing Ridge, Lasso and Elastic Net



# Why Regularize Loss Functions?



# Why Regularize Loss Functions?



# Ridge Regression and Normal Equations

$$\frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}) = \frac{1}{2}\mathbf{w}^T(\mathbf{X}^T\mathbf{X})\mathbf{w} - \mathbf{w}^T(\mathbf{X}^T\mathbf{y})$$

$$\mathbf{X}^T\mathbf{X} = \sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^T = \sum_{i=1}^N \begin{pmatrix} x_{i,1}^2 & \cdots & x_{i,1}x_{i,D} \\ & \ddots & \\ x_{i,D}x_{i,1} & \cdots & x_{i,D}^2 \end{pmatrix}$$

$$\mathbf{X}^T\mathbf{y} = \sum_{i=1}^N \mathbf{x}_i y_i.$$

# Ridge Regression and Normal Equations

$$\frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}) = \frac{1}{2}\mathbf{w}^T(\mathbf{X}^T\mathbf{X})\mathbf{w} - \mathbf{w}^T(\mathbf{X}^T\mathbf{y})$$

$$[\mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{X}^T\mathbf{y}] = \sum_{i=1}^N \mathbf{x}_i(\mathbf{w}^T\mathbf{x}_i - y_i)$$

$$\mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^T\mathbf{y}$$

## Ridge Regression and Normal Equations

$$\frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}) = \frac{1}{2}\mathbf{w}^T(\mathbf{X}^T\mathbf{X})\mathbf{w} - \mathbf{w}^T(\mathbf{X}^T\mathbf{y})$$

$$[\mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{X}^T\mathbf{y}] = \sum_{i=1}^N \mathbf{x}_i(\mathbf{w}^T\mathbf{x}_i - y_i)$$

$$\hat{\mathbf{w}}_{OLS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

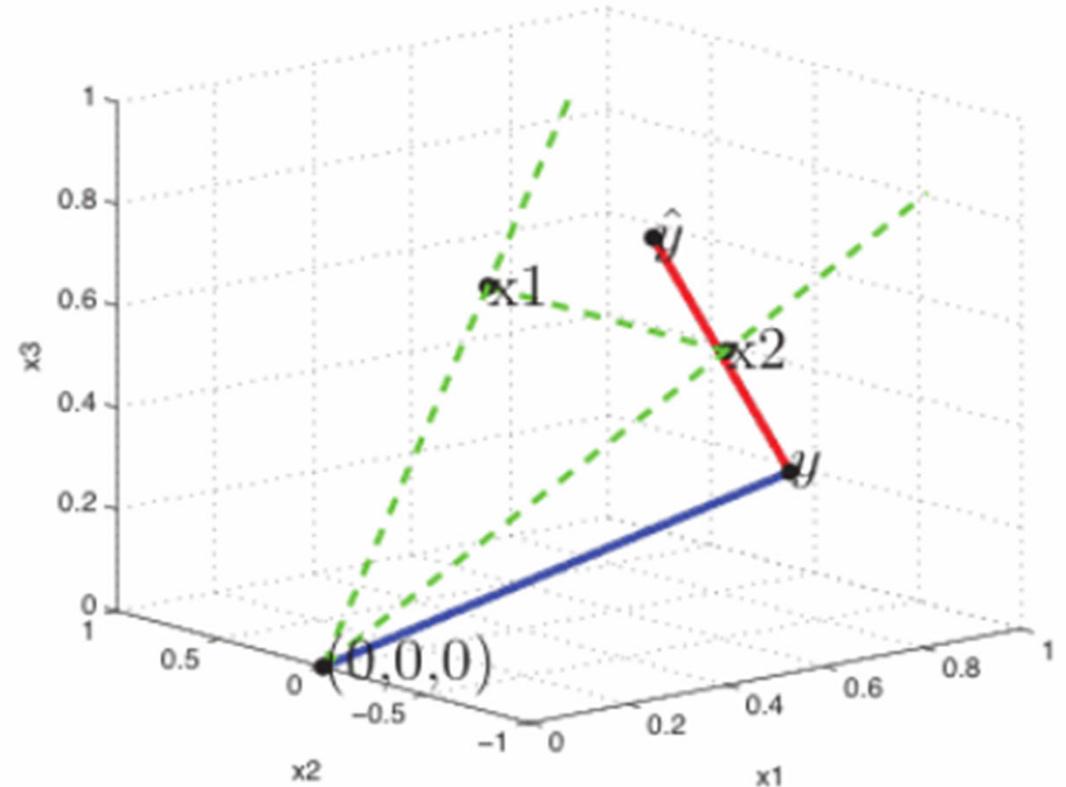
## How to Understand the Normal Equations?

$$\mathbf{X} = \begin{pmatrix} 1 & 2 \\ 1 & -2 \\ 1 & 2 \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} 8.8957 \\ 0.6130 \\ 1.7761 \end{pmatrix}$$

$$\hat{\mathbf{y}} = w_1 \tilde{\mathbf{x}}_1 + \cdots + w_D \tilde{\mathbf{x}}_D = \mathbf{X}\mathbf{w}$$

$$\operatorname*{argmin}_{\hat{\mathbf{y}} \in \text{span}(\{\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_D\})} \|\mathbf{y} - \hat{\mathbf{y}}\|_2.$$

# How to Understand the Normal Equations?



$$\tilde{\mathbf{x}}_j^T(\mathbf{y} - \hat{\mathbf{y}}) = 0 \Rightarrow \mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{w}) = \mathbf{0} \Rightarrow \mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

## Why does Ridge Regression shrink weights?

$$J(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N (y_i - (w_0 + \mathbf{w}^T \mathbf{x}_i))^2 + \lambda \|\mathbf{w}\|_2^2$$

$$\hat{\mathbf{w}}_{ridge} = (\lambda \mathbf{I}_D + \mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

## Why does Ridge Regression shrink weights?

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T \quad \xrightarrow{\hspace{1cm}} \quad \hat{\mathbf{w}}_{ridge} = \mathbf{V}(\mathbf{S}^2 + \lambda\mathbf{I})^{-1}\mathbf{S}\mathbf{U}^T\mathbf{y}$$

## Why does Ridge Regression shrink weights?

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T \quad \longrightarrow \quad \hat{\mathbf{w}}_{ridge} = \mathbf{V}(\mathbf{S}^2 + \lambda\mathbf{I})^{-1}\mathbf{S}\mathbf{U}^T\mathbf{y}$$

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\mathbf{w}}_{ridge} = \mathbf{U}\mathbf{S}\mathbf{V}^T\mathbf{V}(\mathbf{S}^2 + \lambda\mathbf{I})^{-1}\mathbf{S}\mathbf{U}^T\mathbf{y}$$

$$= \mathbf{U}\tilde{\mathbf{S}}\mathbf{U}^T\mathbf{y} = \sum_{j=1}^D \mathbf{u}_j \tilde{S}_{jj} \mathbf{u}_j^T \mathbf{y}$$

$$\tilde{S}_{jj} \triangleq [\mathbf{S}(\mathbf{S}^2 + \lambda I)^{-1}\mathbf{S}]_{jj} = \frac{\sigma_j^2}{\sigma_j^2 + \lambda}$$

# Why does Ridge Regression shrink weights?

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\mathbf{w}}_{ridge} = \sum_{j=1}^D \mathbf{u}_j \frac{\sigma_j^2}{\sigma_j^2 + \lambda} \mathbf{u}_j^T \mathbf{y}$$

vs

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\mathbf{w}}_{ls} = (\mathbf{U}\mathbf{S}\mathbf{V}^T)(\mathbf{V}\mathbf{S}^{-1}\mathbf{U}^T \mathbf{y}) = \mathbf{U}\mathbf{U}^T \mathbf{y} = \sum_{j=1}^D \mathbf{u}_j \mathbf{u}_j^T \mathbf{y}$$

$$\text{dof}(\lambda) = \sum_{j=1}^D \frac{\sigma_j^2}{\sigma_j^2 + \lambda}$$

## Question

- ▶ Why can't we just solve for the minimum in line search?
- ▶ Actually we can...sort of...it's the Levenberg-Marquardt algorithm
- ▶ While it's an iterative algorithm, the important step in the algorithm involves solving the normal equations.

$$f(x_i, \beta + \delta) \approx f(x_i, \beta) + \mathbf{J}_i \delta,$$

where

$$\mathbf{J}_i = \frac{\partial f(x_i, \beta)}{\partial \beta}$$

## Question

- ▶ Why can't we just solve for the minimum in line search?
- ▶ Actually we can...sort of...it's the Levenberg-Marquardt algorithm
- ▶ While it's an iterative algorithm, the important step in the algorithm involves solving the normal equations.

## Question

$$S(\boldsymbol{\beta} + \boldsymbol{\delta}) \approx \sum_{i=1}^m [y_i - f(x_i, \boldsymbol{\beta}) - \mathbf{J}_i \boldsymbol{\delta}]^2,$$

$$\begin{aligned} S(\boldsymbol{\beta} + \boldsymbol{\delta}) &\approx \|\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}\|^2 \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}] \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] - [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T \mathbf{J}\boldsymbol{\delta} - (\mathbf{J}\boldsymbol{\delta})^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] + \boldsymbol{\delta}^T \mathbf{J}^T \mathbf{J}\boldsymbol{\delta} \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] - 2[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T \mathbf{J}\boldsymbol{\delta} + \boldsymbol{\delta}^T \mathbf{J}^T \mathbf{J}\boldsymbol{\delta}. \end{aligned}$$

## Question

$$\begin{aligned} S(\boldsymbol{\beta} + \boldsymbol{\delta}) &\approx \| \mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta} \|^2 \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}] \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] - [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T \mathbf{J}\boldsymbol{\delta} - (\mathbf{J}\boldsymbol{\delta})^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] + \boldsymbol{\delta}^T \mathbf{J}^T \mathbf{J}\boldsymbol{\delta} \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] - 2[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T \mathbf{J}\boldsymbol{\delta} + \boldsymbol{\delta}^T \mathbf{J}^T \mathbf{J}\boldsymbol{\delta}. \end{aligned}$$

$$(\mathbf{J}^T \mathbf{J}) \boldsymbol{\delta} = \mathbf{J}^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})],$$

## Question

$$\begin{aligned} S(\boldsymbol{\beta} + \boldsymbol{\delta}) &\approx \| \mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta} \|^2 \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}] \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] - [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T \mathbf{J}\boldsymbol{\delta} - (\mathbf{J}\boldsymbol{\delta})^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] + \boldsymbol{\delta}^T \mathbf{J}^T \mathbf{J}\boldsymbol{\delta} \\ &= [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})] - 2[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]^T \mathbf{J}\boldsymbol{\delta} + \boldsymbol{\delta}^T \mathbf{J}^T \mathbf{J}\boldsymbol{\delta}. \end{aligned}$$

$$(\mathbf{J}^T \mathbf{J} + \lambda \mathbf{I}) \boldsymbol{\delta} = \mathbf{J}^T [\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})]$$

# Regularization gives Stability

## Ridge Regression (Tikhonov Form)

The ridge regression solution for regularization parameter  $\lambda \geq 0$  is

$$\hat{w} = \arg \min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^n \{w^T x_i - y_i\}^2 + \lambda \|w\|_2^2,$$

where  $\|w\|_2^2 = w_1^2 + \dots + w_d^2$  is the square of the  $\ell_2$ -norm.



## Ridge Regression (Ivanov Form)

The ridge regression solution for complexity parameter  $r \geq 0$  is

$$\hat{w} = \arg \min_{\|w\|_2^2 \leq r^2} \frac{1}{n} \sum_{i=1}^n \{w^T x_i - y_i\}^2.$$

## Regularization gives Stability

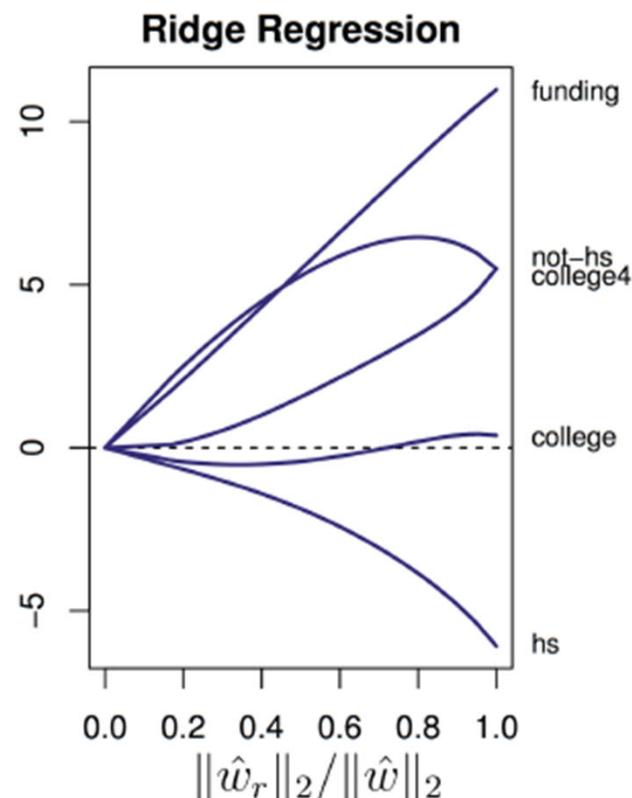
$$\hat{f}(x) = \hat{w}^T x,$$

$$\begin{aligned} |\hat{f}(x+h) - \hat{f}(x)| &= |\hat{w}^T (x+h) - \hat{w}^T x| = |\hat{w}^T h| \\ &\leq \|\hat{w}\|_2 \|h\| \end{aligned}$$

Question

$$\|\hat{w}\|_1 \geq \|\hat{w}\|_2,$$

# Regularization Path



$$\begin{aligned}\hat{w}_r &= \arg \min_{\|w\|_2^2 \leq r^2} \frac{1}{n} \sum_{i=1}^n (w^T x_i - y_i)^2 \\ \hat{w} &= \hat{w}_\infty = \text{Unconstrained ERM}\end{aligned}$$

- For  $r = 0$ ,  $\|\hat{w}_r\|_2 / \|\hat{w}\|_2 = 0$ .
- For  $r = \infty$ ,  $\|\hat{w}_r\|_2 / \|\hat{w}\|_2 = 1$

# Lasso Regression

## Lasso Regression (Tikhonov Form)

The lasso regression solution for regularization parameter  $\lambda \geq 0$  is

$$\hat{w} = \arg \min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^n \{w^T x_i - y_i\}^2 + \lambda \|w\|_1,$$

where  $\|w\|_1 = |w_1| + \dots + |w_d|$  is the  $\ell_1$ -norm.

---

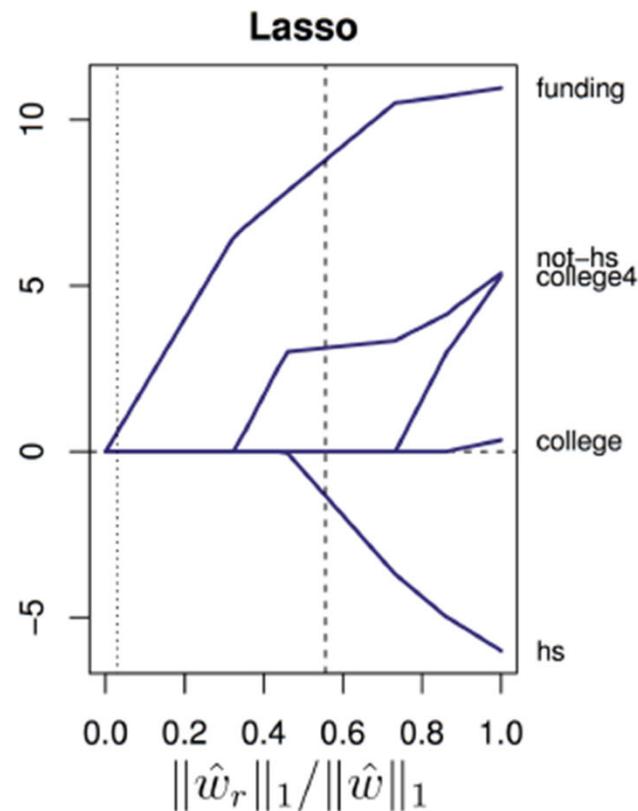


## Lasso Regression (Ivanov Form)

The lasso regression solution for complexity parameter  $r \geq 0$  is

$$\hat{w} = \arg \min_{\|w\|_1 \leq r} \frac{1}{n} \sum_{i=1}^n \{w^T x_i - y_i\}^2.$$

# Lasso Regression



$$\begin{aligned}\hat{w}_r &= \arg \min_{\|w\|_1 \leq r} \frac{1}{n} \sum_{i=1}^n (w^T x_i - y_i)^2 \\ \hat{w} &= \hat{w}_\infty = \text{Unconstrained ERM}\end{aligned}$$

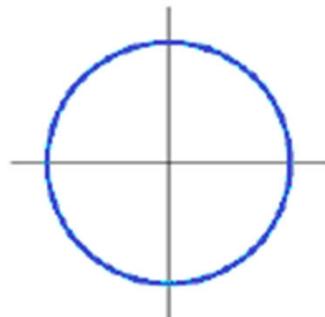
- For  $r = 0$ ,  $\|\hat{w}_r\|_1 / \|\hat{w}\|_1 = 0$ .
- For  $r = \infty$ ,  $\|\hat{w}_r\|_1 / \|\hat{w}\|_1 = 1$

## Lasso Regression and Sparsity

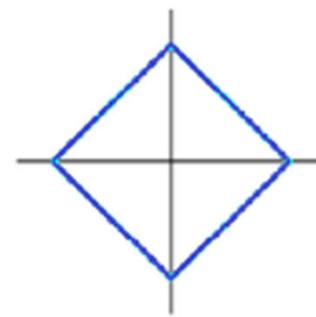
- ▶ Coefficient are 0 => don't need those features.  
What's the gain?
  - ▶ Time/expense to compute/buy features
  - ▶ Memory to store features (e.g. real-time deployment)
  - ▶ Identifies the important features
  - ▶ Better prediction? sometimes
  - ▶ As a feature-selection step for training a slower non-linear model

## Why Sparse Solutions

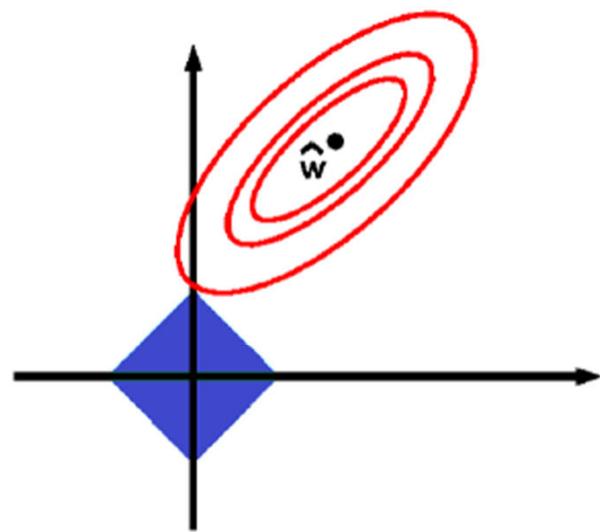
- $\ell_2$  contour:  
 $w_1^2 + w_2^2 = r$



- $\ell_1$  contour:  
 $|w_1| + |w_2| = r$



# Why Sparse Solutions



- Blue region: Area satisfying complexity constraint:  $|w_1| + |w_2| \leq r$
- Red lines: contours of  $\hat{R}_n(w) = \sum_{i=1}^n (w^T x_i - y_i)^2$ .

# Why Sparse Solutions

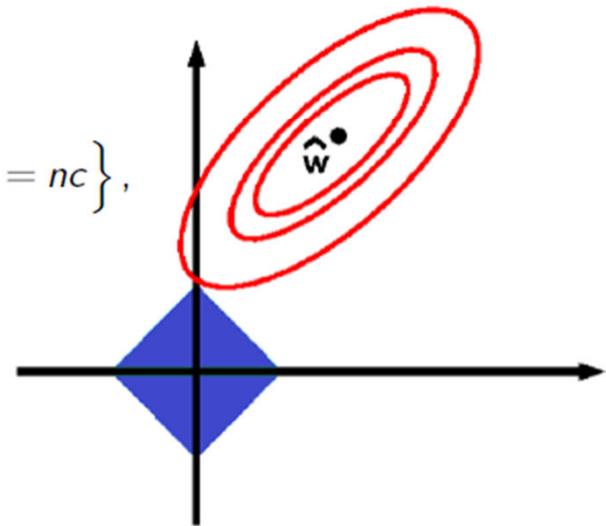
- By “completing the square”, we can show for any  $w \in \mathbb{R}^d$ :

$$\hat{R}_n(w) = \frac{1}{n} (w - \hat{w})^T X^T X (w - \hat{w}) + \hat{R}_n(\hat{w})$$

- Set of  $w$  with  $\hat{R}_n(w)$  exceeding  $\hat{R}_n(\hat{w})$  by  $c > 0$  is

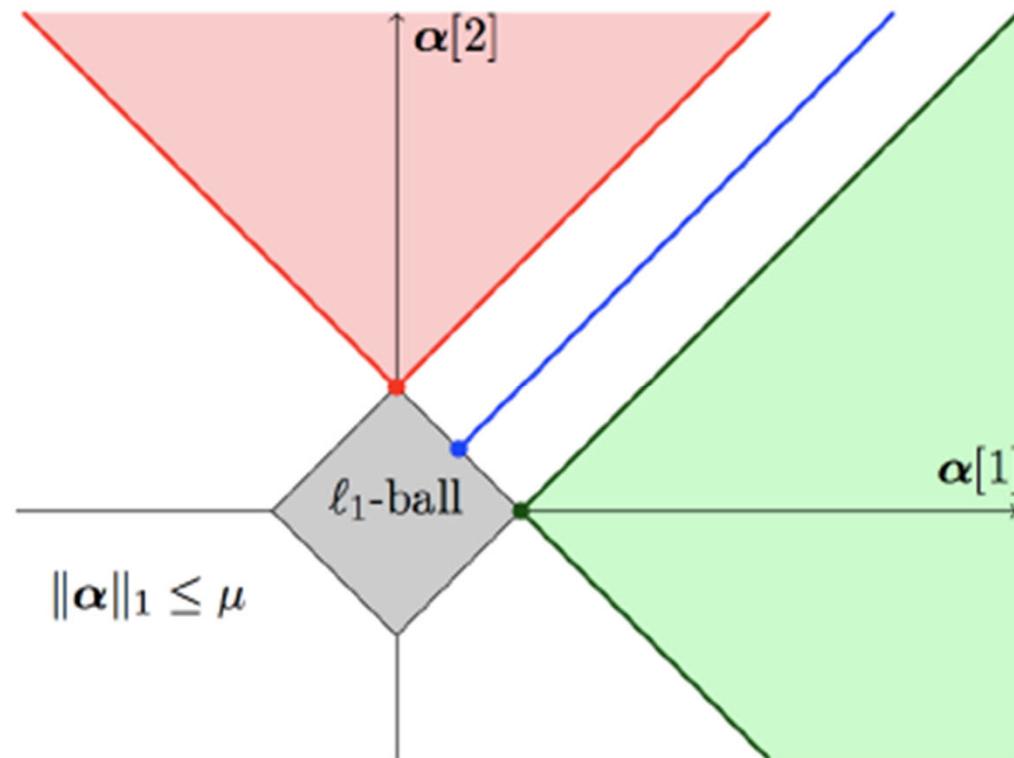
$$\left\{ w \mid \hat{R}_n(w) = c + \hat{R}_n(\hat{w}) \right\} = \left\{ w \mid (w - \hat{w})^T X^T X (w - \hat{w}) = nc \right\},$$

which is an **ellipsoid centered at  $\hat{w}$** .

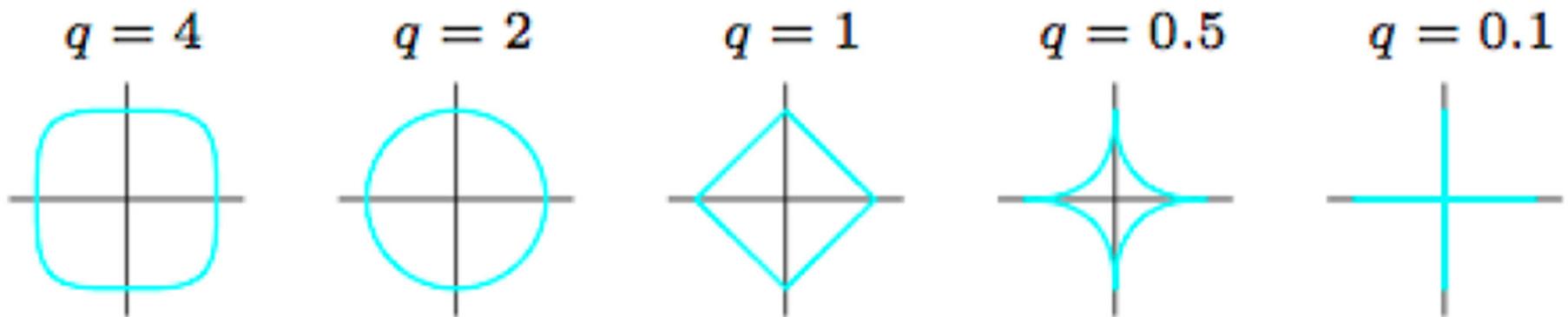


- Blue region: Area satisfying complexity constraint:  $|w_1| + |w_2| \leq r$
- Red lines: contours of  $\hat{R}_n(w) = \sum_{i=1}^n (w^T x_i - y_i)^2$ .

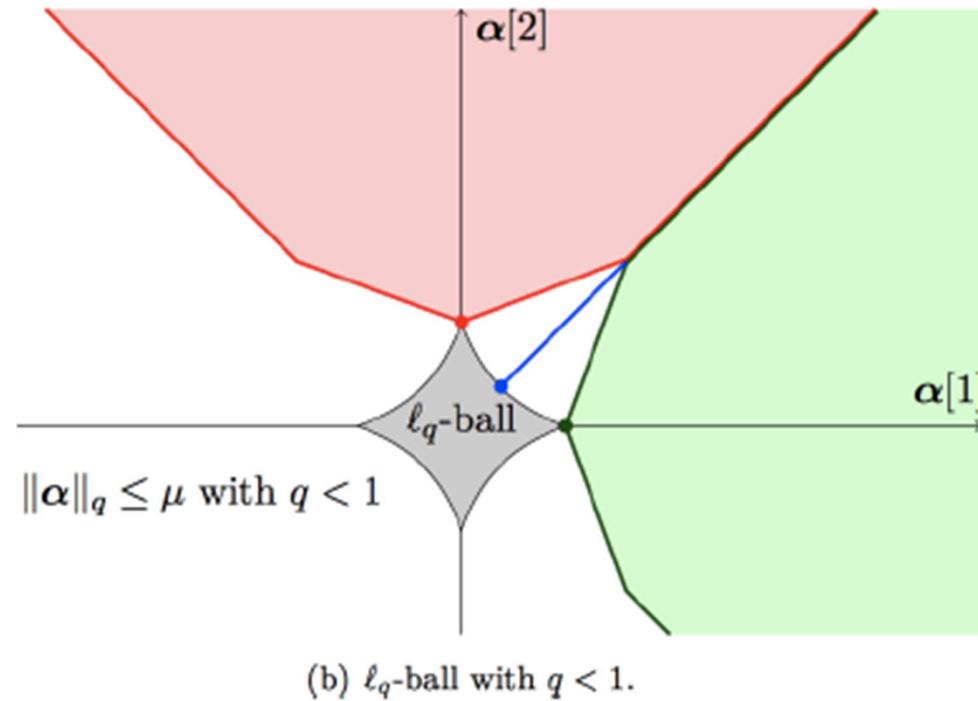
## Why Sparse Solutions



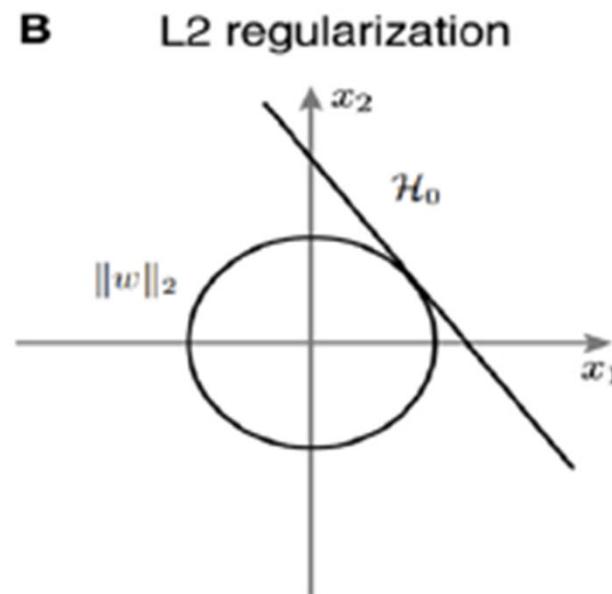
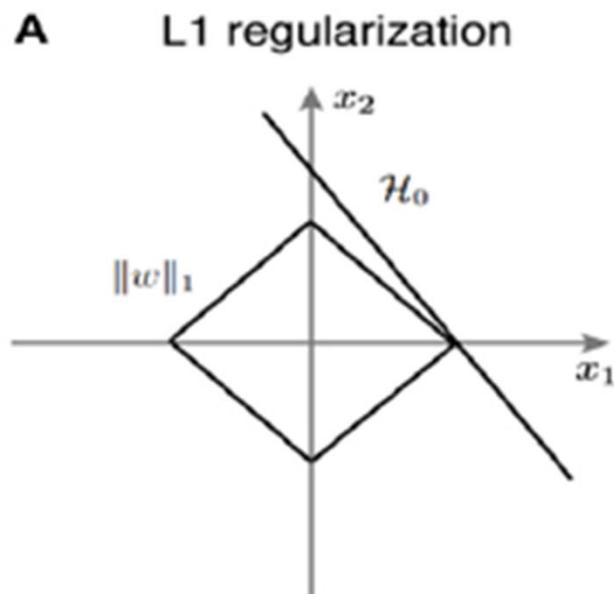
## Why Sparse Solutions



# Why Sparse Solutions



## Question



## Take-Aways

- ▶ Ridge
  - ▶ Explain the normal equations with a picture
  - ▶ How can you “invert” a matrix that has a different number of rows and columns
- ▶ Lasso
  - ▶ Explain what coordinate descent is, and why it is of particular interest for the Lasso.
  - ▶ Lasso optimization problem does not have a differentiable objective function. Give an equivalent formulation that has a differentiable objective function by dividing the weight vector into positive and negative parts.
  - ▶ Give reasons why we might want the sparsity that L1 regularization often provides.

## Take-Aways

- ▶ Ridge versus Lasso
  - ▶ Explain what happens when we do linear, lasso, and ridge regression with 2 identical features.
  - ▶ Explain what happens when  $x_1$  and  $x_2$  are highly correlated, but not exactly linearly related.
  - ▶ Why does Elastic Net provide us benefits of both L1 and L2 regularization?