CS60050: Machine Learning

Autumn 2024

Sudeshna Sarkar

Regularization in Linear Regression

2 -7 August 2024

Many Slides from Pat Virtue, CMU 10-315 Introduction to ML

Empirical Risk, Overfitting

Performance Measure: Loss Function

- We need a guiding mechanism to tell us how good our predictions are given an input.
- The loss for a given example (x, y) is given by

$$\text{loss}(Y, f(X))$$

- We want to perform well on any test data:

$$(X, Y) \sim P_{XY}$$

- Given an X drawn randomly from a distribution, how well does the predictor perform on average?

$$Risk\ R(f) = \mathbb{E}_{XY}[loss(Y, f(X))]$$

Learning as an Optimization

Objective

Given a loss function L, find f such that

Optimal Predictor:

$$f^* = arg\min_f \mathbb{E}[L(Y, f(X))]$$

Empirical Risk Minimizer:

$$\hat{f}_m = \arg\min_{f \in \mathcal{H}} \frac{1}{m} \sum_{i=1m} L\big(Y_i, f(X_i)\big)$$

$$\frac{1}{m} \sum_{i=1m} L\big(Y_i, f(X_i)\big) \xrightarrow{\text{law of large numbers}} \mathbb{E}_{XY}[L(Y, f(X))]$$

Loss and Risk in Regression

- **Square Loss**

$$loss(X, Y) = (f_\theta(X) - Y)^2$$

We found $\boldsymbol{\theta}$ which minimize the squared loss on data ***we already have***. This averaged loss is called **empirical risk.**

- **Risk $\boldsymbol{R(f)}$**: What we really want to do is predict the y values for points x ***we haven't seen yet***. i.e. minimize the expected loss on some new data.

$$\mathbb{E}[(f(X) - Y)^2]$$

Machine learning approximates risk-minimizing models with empirical-risk minimizing ones.

Risk Minimization

Generally minimizing empirical risk (loss on the data) instead of true risk works fine, but it can fail if:

- The **data sample is biased**. e.g. you cant build a (good) classifier with observations of only one class.
- There is **not enough data** to accurately estimate the parameters of the model. Depends on the complexity (number of parameters, variation in gradients, complexity of the loss function, generative vs. discriminative etc.).

Regularization

Poll 1

- Which is model do you prefer, assuming both have zero training error?
- Model structure (for both models):
- $h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \theta_5 x^5 + \theta_6 x^6 + \theta_7 x^7 + \theta_8 x^8$

- Model parameters:
- $\boldsymbol{\theta} = [\theta_0,\ \theta_1,\ \theta_2,\ \theta_3,\ \theta_4,\ \theta_5,\ \theta_6,\ \theta_7,\ \theta_8]^T$

A. $\boldsymbol{\theta}_A = [-190.0,\ -135.0,\ 310.0,\ 45.0,\ -62.0,\ 90.0,\ -82.0,\ -40.0,\ 29.0]^T$

B. $\boldsymbol{\theta}_B = [\ 25.5,\ -6.4,\ -0.8,\ 0.0,\ 6.6,\ -4.4,\ 0.2,\ -2.9,\ 0.1]^T$

Poll

Which is model do you prefer, assuming both have zero training error?

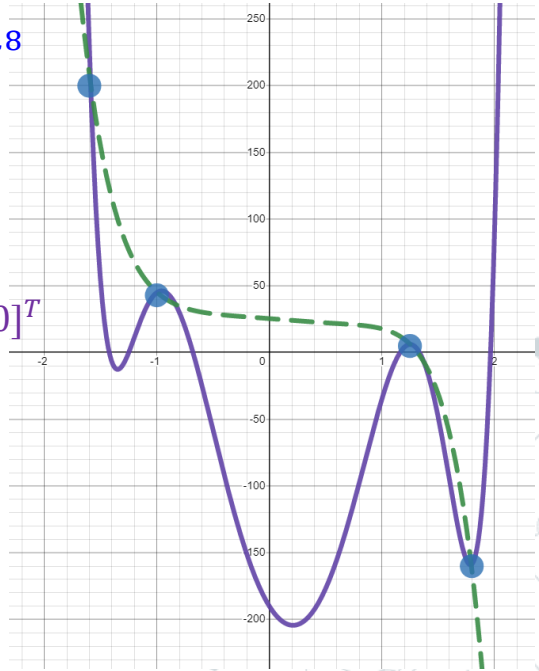
- Model structure (for both models):

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \theta_5 x^5 + \theta_6 x^6 + \theta_7 x^7 + \theta_8 x^8$$

- Model parameters: $\boldsymbol{\theta} = [\theta_0,\ \theta_1,\ \theta_2,\ \theta_3,\ \theta_4,\ \theta_5,\ \theta_6,\ \theta_7,\ \theta_8]^T$

A. $\boldsymbol{\theta}_A = [-190.0,\ -135.0,\ 310.0,\ 45.0,\ -62.0,\ 90.0,\ -82.0,\ -40.0,\ 29.0]^T$

B. $\boldsymbol{\theta}_B = [\ \ 25.5,\ \ -6.4,\ \ -0.8,\ 0.0,\ \ 6.6,\ -4.4,\ \ 0.2,\ \ -2.9,\ \ 0.1]^T$



Number of features used

	$M=0$	$M=1$	$M=3$	$M=9$
θ_0	0.19	0.82	0.31	0.35
θ_1		-1.27	7.99	232.37
θ_2			-25.43	-5321.83
θ_3			17.37	48568.31
θ_4				-231639.30
θ_5				640042.26
θ_6				-1061800.52
θ_7				1042400.18
θ_8				-557682.99
θ_9				125201.43

Slide credit: CMU MLD William Cohen

Too many features?

Overfitting

Definition: The problem of **overfitting** is when the model captures the noise in the training data instead of the underlying structure

Slide credit: CMU MLD Matt Gormley

Motivation: Regularization

Occam's Razor: prefer the simplest hypothesis

What does it mean for a hypothesis (or model) to be simple?

1. small number of features (model selection)
2. small number of "important" features (feature reduction)
3. small values for associated parameters

Case 1: we want "$\boldsymbol{\theta}$" such that most of its elements are small

Case 2: we want "$\boldsymbol{\theta}$" such that most of its elements are 0

Slide credit: CMU MLD Matt Gormley

Regularization

Key idea:

- Define regularizer $\Omega(\boldsymbol{\theta})$ that we will add to our minimization objective to keep the model simple.

$\Omega(\boldsymbol{\theta})$ should be:

- Small for a simple model
- Large for a complex model

Regularized Loss $= L(y, \hat{y}) + \lambda\Omega(\theta)$

Key idea: Define regularizer $\Omega(\boldsymbol{\theta})$ s.t. we tradeoff between fitting the data and keeping the model simple

L2 norm: square-root of sum of squares

L1 norm: sum of absolute values

L0 norm: count of non-zero values

Regularization is very important when training set is small and the number of features is very large.

Generalization capacity and Regularization

- How well will the learned function work on the unseen data?
- Occam's principle: A simple f can generalize better
- Regularization to keep the function from overfitting the training data. (More on overfitting later in the class)

$$\mathrm{f} = \arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} L\big(Y_i, f(X_i)\big) + \lambda \Omega(\theta)$$

- Ω is a measure of model complexity
- λ controls the amount of regularization.

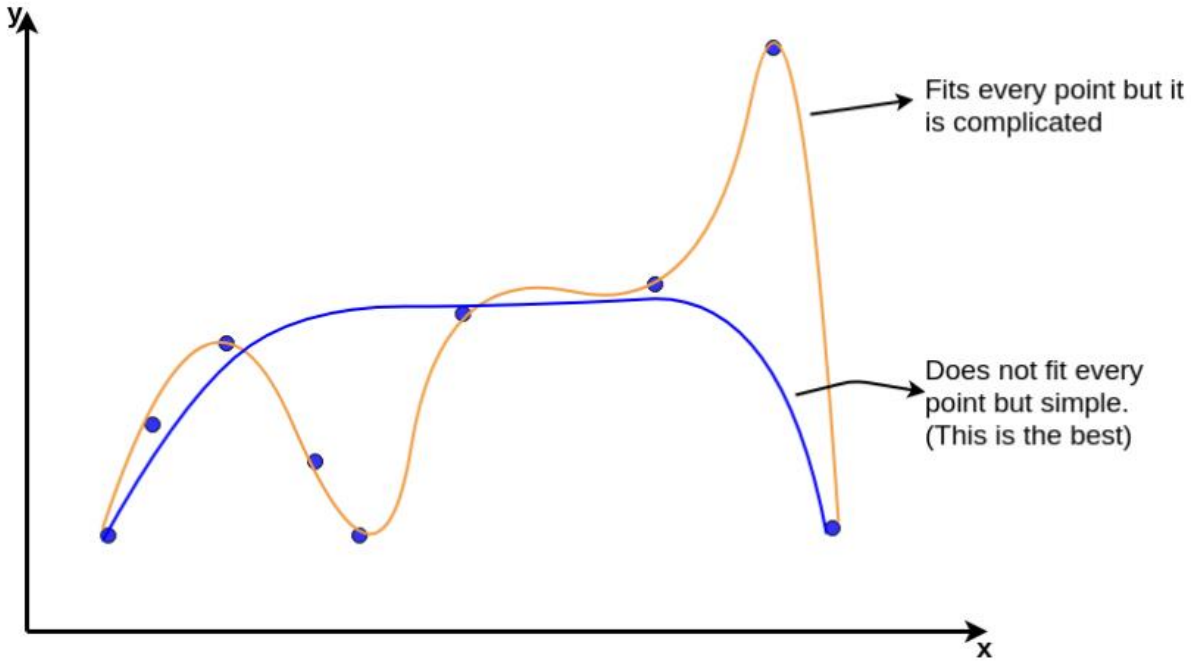
$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta}) + \lambda \Omega(\boldsymbol{\theta})$$

Generalization capacity and Regularization

- One way of regularization is to put a bias on the model forcing the learning to prefer certain types of weights over others.
- What makes for a “simpler” model for a linear model? Two ideas.
  1. If weights are large, a small change in a feature can result in a large change in the prediction.
  2. Might also prefer weights of 0 for features that aren’t useful

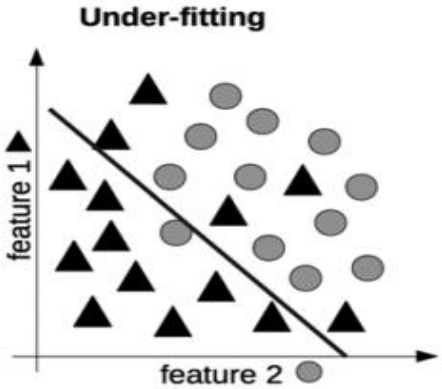
$$\mathrm{f} = \arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} l\big(Y_i, f(X_i)\big) + \lambda\, \Omega(\theta)$$

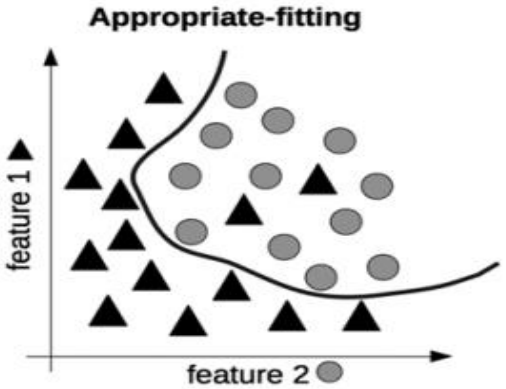
Generalizing Capacity

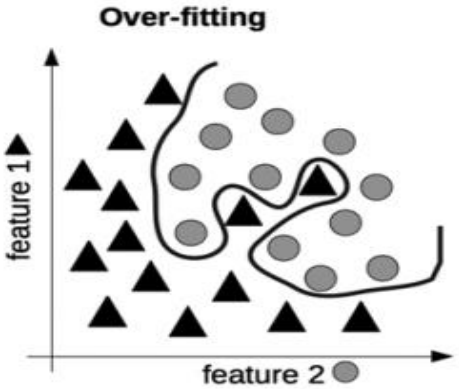


The blue curve has better generalization capacity. The orange curve overfits the data

Generalizing Capacity (contd..)







What is a good Regularizer?

Case 1: If weights are large, a small change in a feature can result in a large change in the prediction, Also gives too much weight to any one feature.

$$L2 \text{ Regularizer} = \sqrt{\sum_{\theta_j} {\theta_j}^2}$$

Case 2: Might also prefer weights of 0 for features that aren't useful.

$$L1 \text{ Regularizer} = \sum_{\theta_j} |\theta_j|$$

L2 norm: square-root of sum of squares

L1 norm: sum of absolute values

L0 norm: count of non-zero values

Squared weights penalizes large values more

Sum of weights will penalize small values more

On Regularization

Claim: Small weights $\theta = (\theta_1, \dots, \theta_d)$ ensure function $y = f(x) = \theta^\top x$ is *smooth (why smoothness?)*

Justification:

- Let $x_n, x_m \in \mathbb{R}^d$ such that

$$x_{n_j} = x_{m_j},\ j = 1,2, \dots, d-1, \text{ but } |x_{n_d} - x_{m_d}| = \epsilon$$

- Then $|y_n - y_m| = \epsilon w_d$
- If w_d is large, the difference would be large.

=> $f(x)$ is not smooth.

Predicting say Student grades to admissions (0/1).
- 2 students, all but 1 is same.

On Regularization

Claim: Small weights $\theta = (\theta_1, \dots, \theta_d)$ ensure function $y = f(x) = \theta^\top x$ is *smooth (why smoothness?)*

Justification:

- Let $x_n, x_m \in \mathbb{R}^d$ such that

$$x_{n_j} = x_{m_j},\ j = 1,2, \dots, d-1, \text{ but } |x_{n_d} - x_{m_d}| = \epsilon$$

- Then $|y_n - y_m| = \epsilon w_d$
- If w_d is large, the difference would be large.

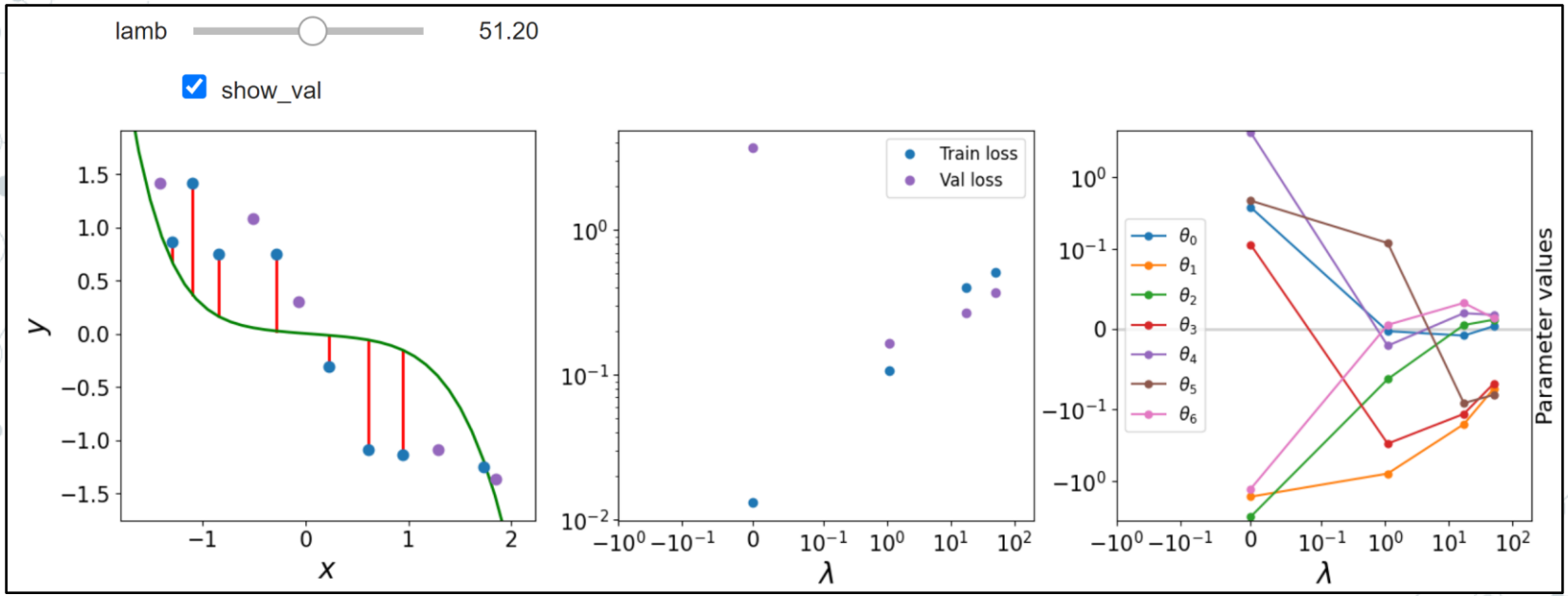
=> $f(x)$ is not smooth.

Predicting say Student grades to admissions (0/1).
- 2 students, all but 1 is same.

Best of both worlds

How can we keep the expressive power of a complex model while still avoiding overfitting?

Notebook demo: [regression_regularization.ipynb]



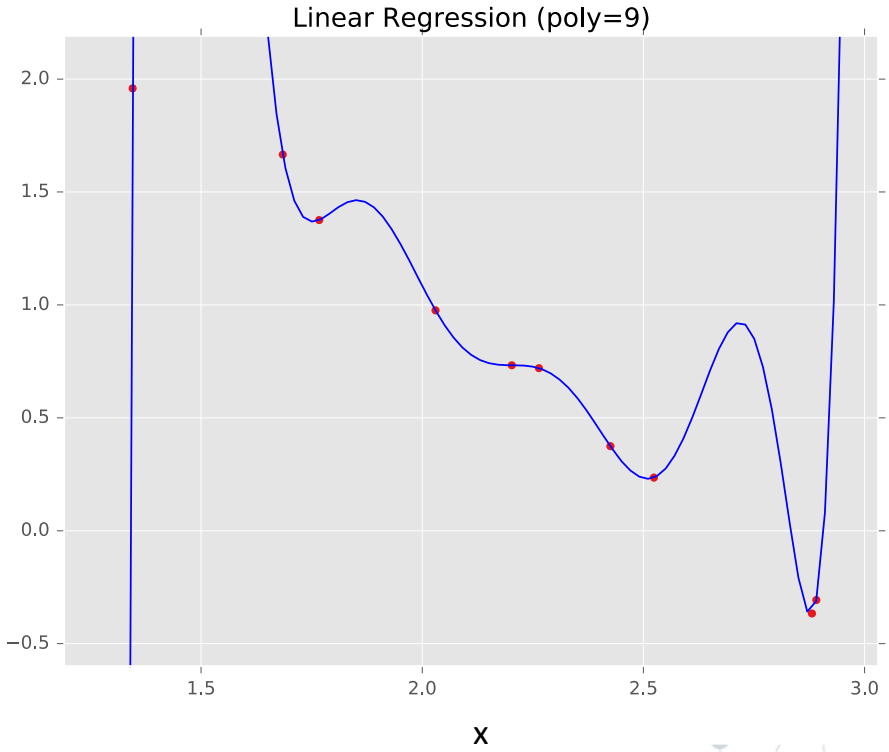
Example: Linear Regression

Goal: Learn $y = \mathbf{w}^T f(\mathbf{x}) + b$ where f(.) is a polynomial basis function

y	x	x^2	…	x^9
2.0	1.2	$(1.2)^2$	…	$(1.2)^9$
1.3	1.7	$(1.7)^2$	…	$(1.7)^9$
0.1	2.7	$(2.7)^2$	…	$(2.7)^9$
1.1	1.9	$(1.9)^2$	…	$(1.9)^9$

y

true "unknown" target function is linear with negative slope and gaussian noise



Slide credit: CMU MLD Matt Gormley

Symptoms of Overfitting

	$M = 0$	$M = 1$	$M = 3$	$M = 9$
θ_0	0.19	0.82	0.31	0.35
θ_1		-1.27	7.99	232.37
θ_2			-25.43	-5321.83
θ_3			17.37	48568.31
θ_4				-231639.30
θ_5				640042.26
θ_6				-1061800.52
θ_7				1042400.18
θ_8				-557682.99
θ_9				125201.43

Slide credit: CMU MLD William Cohen

Motivation: Regularization

Occam's Razor: prefer the simplest hypothesis

What does it mean for a hypothesis (or model) to be simple?

1. small number of features (model selection)
2. small number of "important" features (feature reduction)
3. small values for associated parameters

Slide credit: CMU MLD Matt Gormley

Regularization

Key idea:

- Define $Regularizer(\boldsymbol{\theta})$ that we will add to our minimization objective to keep the model simple.

$Regularizer(\boldsymbol{\theta})$ should be:

- Small for a simple model
- Large for a complex model

- L2 norm: square-root of sum of squares
- L1 norm: sum of absolute values
- L0 norm: count of non-zero values

Regularization

- What if we have too many features?
- Can linear regression itself solve the feature selection problem?

1. Case 1: we want "$\boldsymbol{\theta}$" such that most of its elements are small
2. Case 2: we want "$\boldsymbol{\theta}$" such that most of its elements are 0

Regularized Loss $= L(y, \hat{y}) + \lambda \, \text{Regularizer}(\theta)$

Regularization is very important when training set is small and the number of features is very large.

Generalization capacity and Regularization

- How well will the learned function work on the unseen data?
- Occam's principle: A simple f can generalize better
- Regularization to keep the function from overfitting the training data. (More on overfitting later in the class)

$$\mathrm{f} = \arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} l\big(Y_i, f(X_i)\big) + \lambda\, \mathrm{Complexity}(f)$$

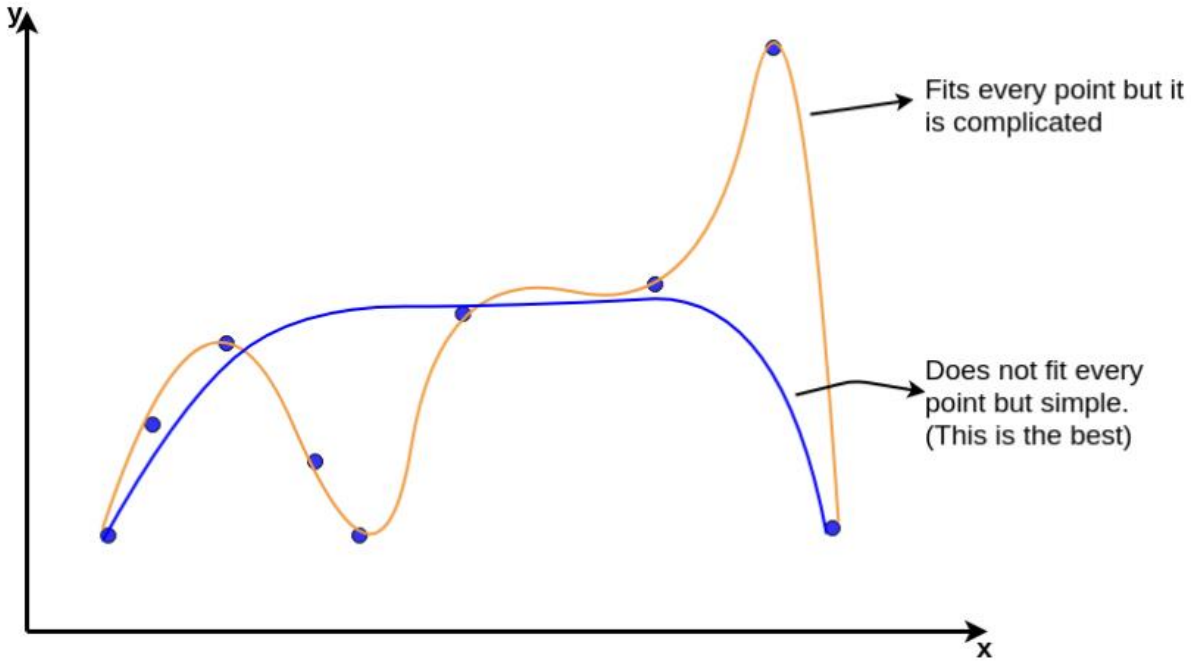
- λ controls the amount of regularization.

Generalization capacity and Regularization

- One way of regularization is to put a bias on the model forcing the learning to prefer certain types of weights over others.
- What makes for a “simpler” model for a linear model? Two ideas.
  1. If weights are large, a small change in a feature can result in a large change in the prediction.
  2. Might also prefer weights of 0 for features that aren’t useful

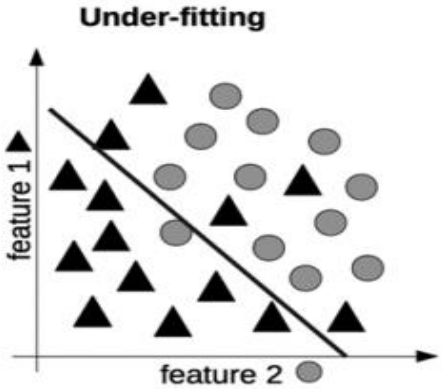
$$\mathrm{f} = \arg\min_{f \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{n} L\big(Y_i, f(X_i)\big) + \lambda\, \Omega(\theta)$$

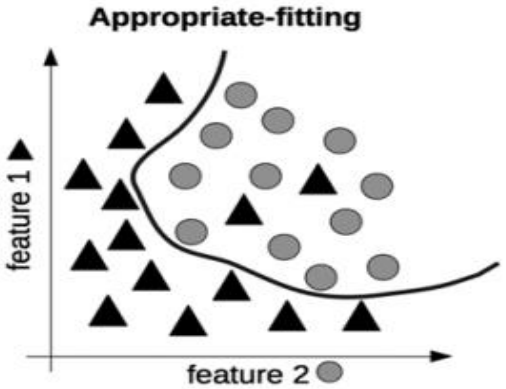
Generalizing Capacity

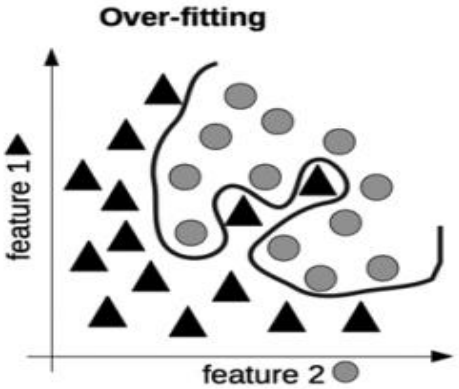


The blue curve has better generalization capacity. The orange curve overfits the data

Generalizing Capacity (contd..)







What is a good Regularizer?

- Case 1: If weights are large, a small change in a feature can result in a large change in the prediction, Also gives too much weight to any one feature.

$$L2 \text{ Regularizer} = \sqrt{\sum_{\theta_j} {\theta_j}^2}$$

- Case 2: Might also prefer weights of 0 for features that aren't useful.

$$L1 \text{ Regularizer} = \sum_{\theta_j} |\theta_j|$$

Squared weights penalizes large values more

Sum of weights will penalize small values more

On Regularization

Claim: Small weights $\theta = (\theta_1, \dots, \theta_d)$ ensure function $y = f(x) = \theta^\top x$ is *smooth (why smoothness?)*

Justification:

- Let $x_n, x_m \in \mathbb{R}^d$ such that

$$x_{n_j} = x_{m_j},\ j = 1,2, \dots, d-1, \text{ but } |x_{n_d} - x_{m_d}| = \epsilon$$

- Then $|y_n - y_m| = \epsilon w_d$
- If w_d is large, the difference would be large.

=> $f(x)$ is not smooth.

Predicting say Student grades to admissions (0/1).
- 2 students, all but 1 is same.

Linear Regression with Regularizers (Ridge Regression)

L2

Ridge Regression

- Modified Objective: Given $\{(x_n, y_n)\}_{n=1}^{m}$, find w such that

$$L_{emp}(f) = \frac{1}{m}\sum_{n=1}^{m}(y_n - \theta^T x_n)^2 + \lambda||\theta||^2$$

  - $||\theta||^2 = \theta^T\theta$
  - λ is a hyperparameter, controls the amount of regularization.

- Solution:

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_{n=1}^{m} 2(y_n - \theta^T x_n)(-x_n) + 2\lambda\theta = 0$$

Ridge Regression

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_{n=1}^{N} 2(y_n - \theta^T x_n)(-x_n) + 2\lambda\theta = 0$$

$$\Rightarrow \gamma(\theta) = \sum_{n=1}^{N} x_n(y_n - x_n^T\theta)$$

$$\Rightarrow \gamma(\theta) = \sum_{n=1}^{N} x_n y_n - \sum_{n=1}^{N} x_n x_n^T\theta$$

$$\Rightarrow \gamma\theta = X^T Y - X^T X W$$

$$\Rightarrow \gamma\theta + X^T X W = X^T Y$$

$$\Rightarrow \theta = (X^T X + \gamma I)^{-1} X^T Y$$

Ordinary Regression

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_{n=1}^{N} 2(y_n - \theta^T x_n)\frac{\partial L(\theta)}{\partial \theta} = 0$$

$$\Rightarrow \sum_{n=1}^{N} 2(y_n - \theta^T x_n)(-x_n) = 0$$

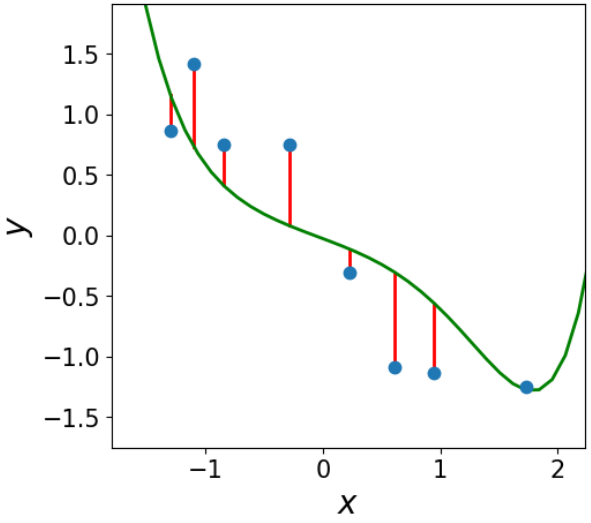
$$\Rightarrow \sum_{n=1}^{N} x_n y_n - \sum_{n=1}^{N} x_n x_n^\top\theta = 0$$

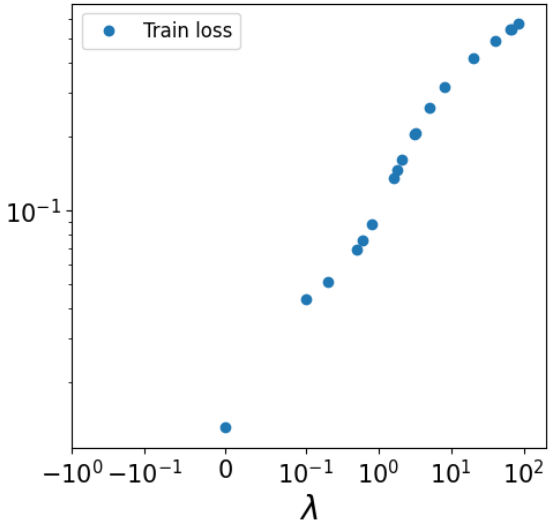
$$\Rightarrow \textstyle\sum_{n=1}^{N} x_n x_n^\top\theta = \sum_{n=1}^{N} x_n y_n$$

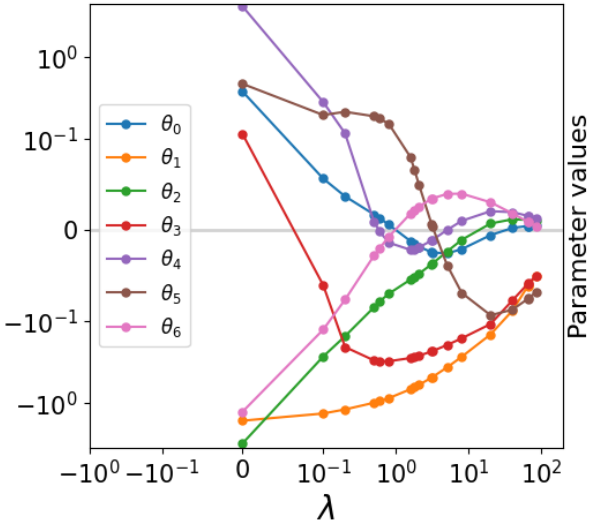
$$\Rightarrow \theta = (X^\top X)^{-1} X^\top Y$$

What is the best value for lambda?

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta}) + \lambda r(\boldsymbol{\theta})$$

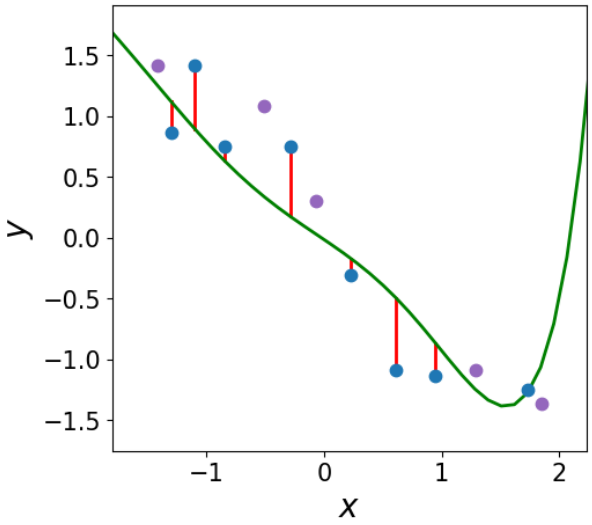


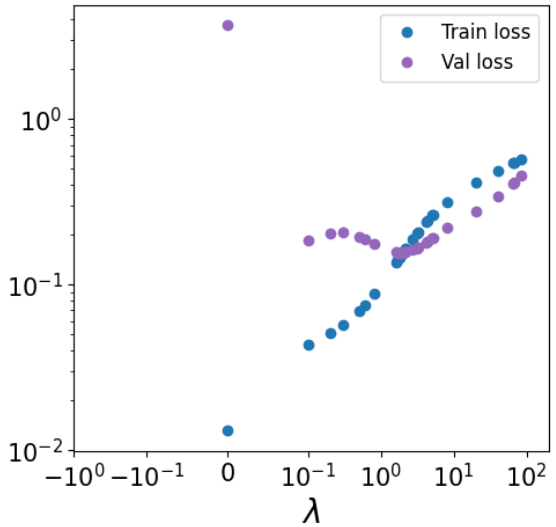


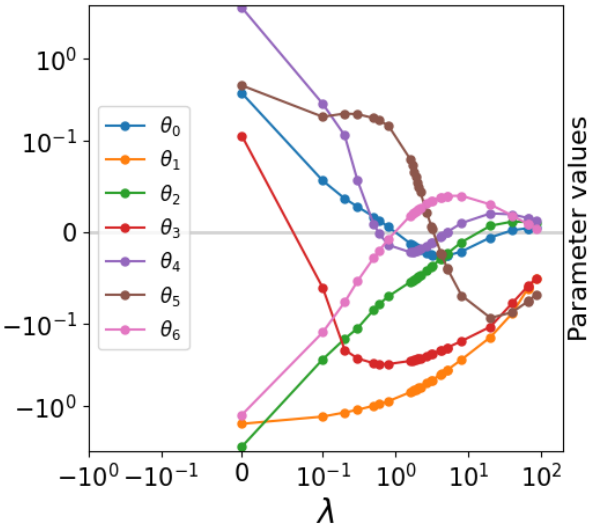


What is the best value for lambda?

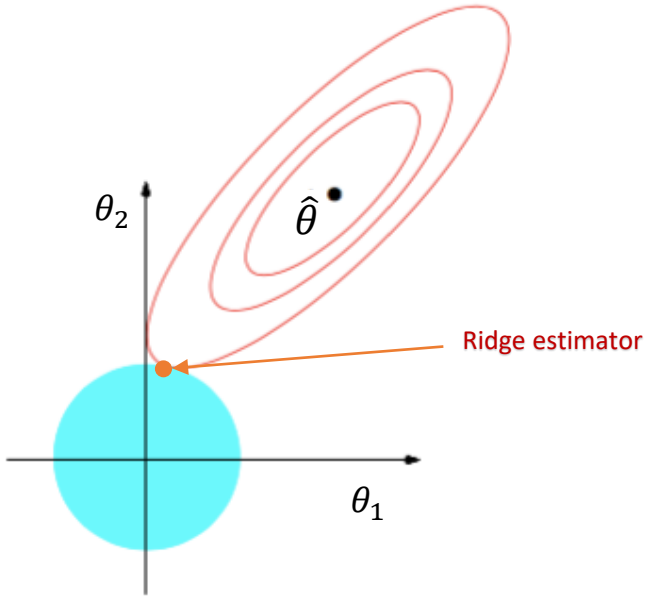
$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta}) + \lambda r(\boldsymbol{\theta})$$



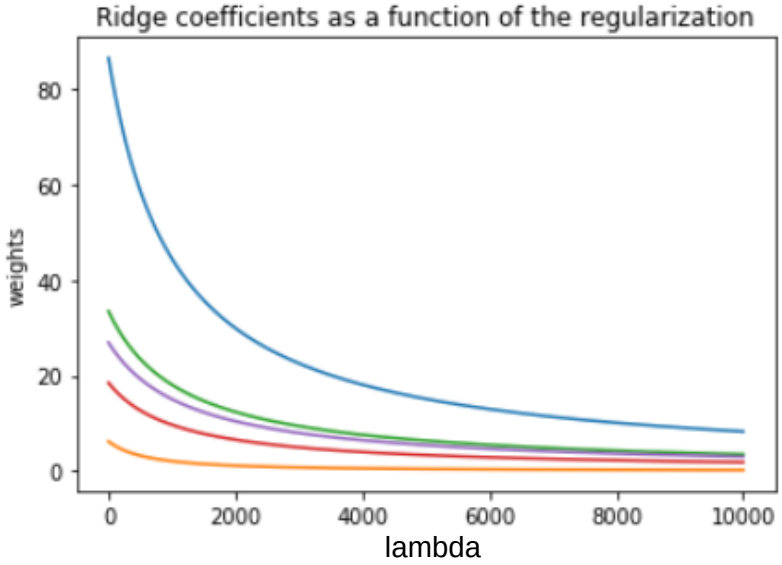




Ridge Regularization visualized



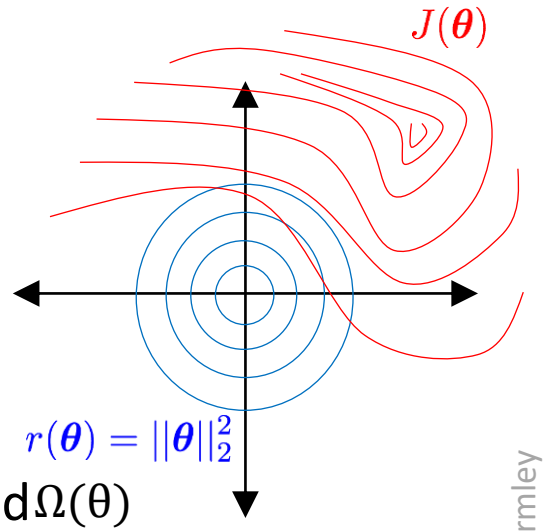
The ridge estimator is where the constraint and the loss intersect.



The values of the coefficients decrease as lambda increases, but they are not nullified.

Suppose we are minimizing J'(θ) where

$$J'(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) + \lambda\Omega(\boldsymbol{\theta})$$



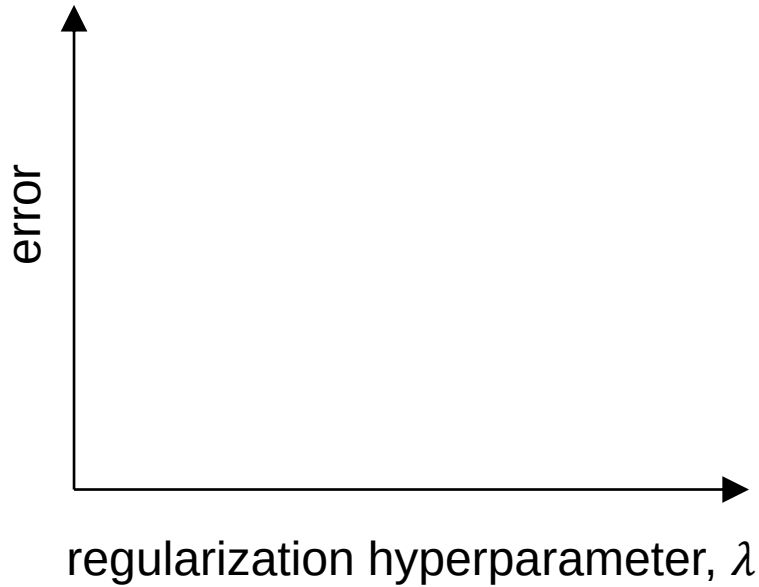
As λ increases, the minimum of J'(θ) will…

A. …move towards the midpoint between J'(θ) and $\Omega(\theta)$
B. …move towards the minimum of J(θ)
C. …move towards the minimum of $\Omega(\theta)$
D. …move towards a theta vector of positive infinities
E. …move towards a theta vector of negative infinities
F. …stay the same

Slide credit: CMU MLD Matt Gormley

Regularization Exercise

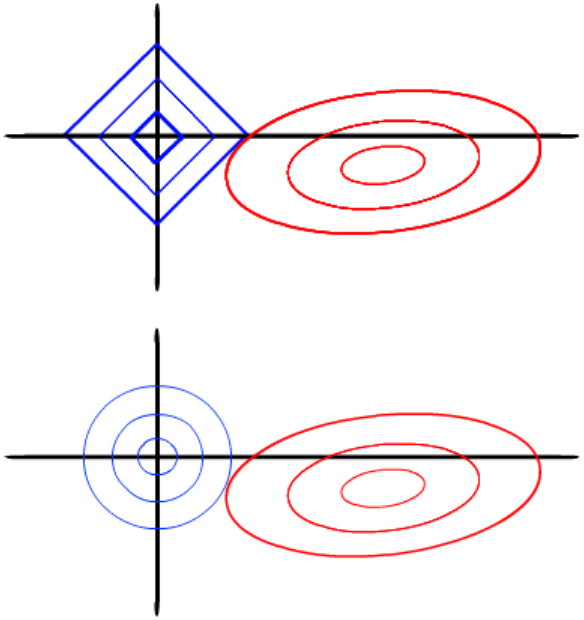
1. Plot train error vs. regularization hyperparameter
2. Plot validation error vs . regularization hyperparameter

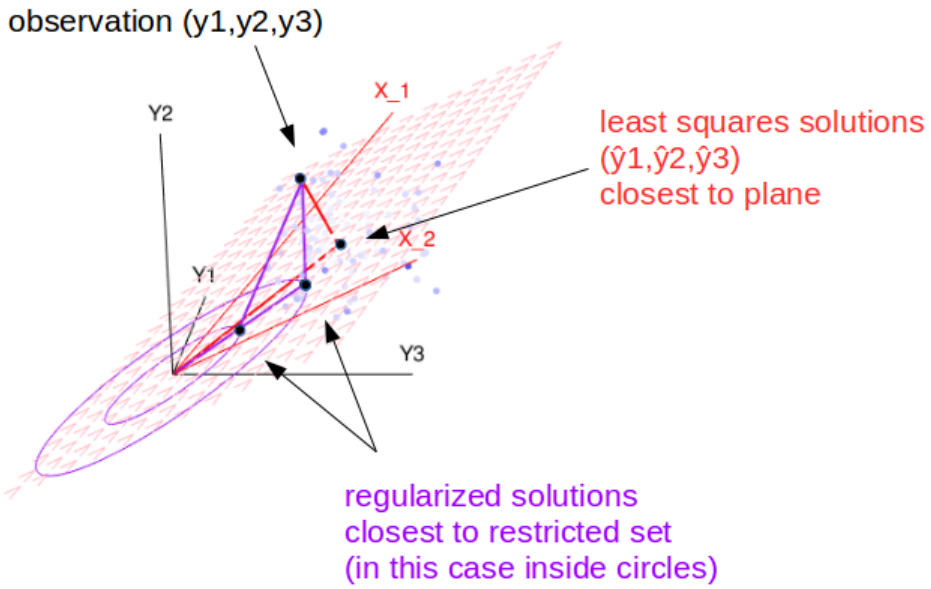


$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta}) + \lambda\Omega(\boldsymbol{\theta})$$

Slide credit: CMU MLD Matt Gormley

Geometric Explanation





On Convexity

- The squared loss function in linear regression is convex.
  - L2 regularizer it is strictly convex.

Gradient Descent Solution for Least Squares

- Ridge Regression has an analytical solution.

$$\theta^* = (X^T X + \lambda I)^{-1} X^T Y$$

- Involves inverting a $d \times d$ matrix.
- Difficult for large d
- Gradient Descent solution may be used.

L_1 Regularizer LASSO

- l_1 Regularizer $R(\theta) = \left|\left|\theta\right|\right|_1 = \Sigma_{j=1}^{d}|\theta_j|$

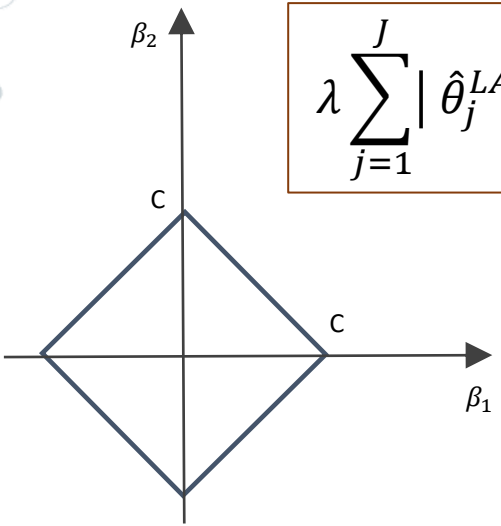
- Promotes θ to have very few non zero components.
- Since LASSO regression tend to produce zero estimates for a number of model parameters - we say that LASSO solutions are sparse - we consider LASSO to be a method for variable selection.

- LASSO has no conventional analytical solution, as the L1 norm has no derivative at 0.

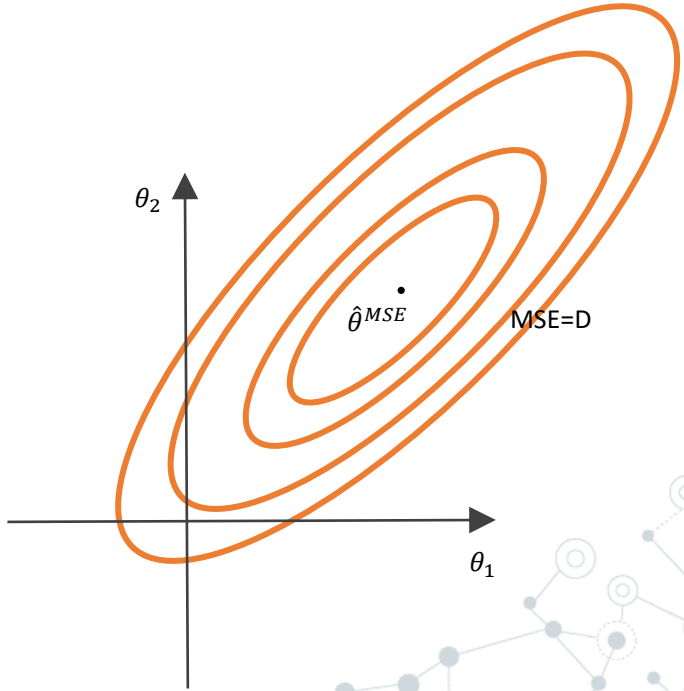
The Geometry of Regularization (LASSO)

$$L_{LASSO}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \boldsymbol{\theta}^T \boldsymbol{x}|^2 + \lambda \sum_{j=1}^{J} |\theta_j|$$
$$\hat{\boldsymbol{\theta}}^{LASSO} = \operatorname{argmin} L_{LASSO}(\boldsymbol{\theta})$$

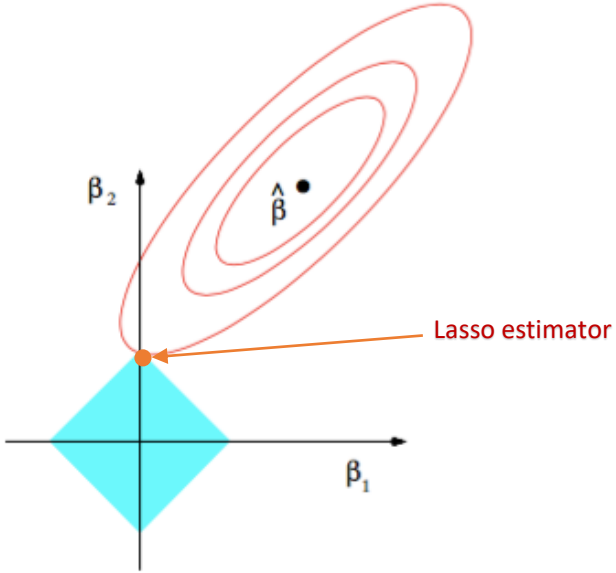
$$\frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{\boldsymbol{\theta}}^{LASSO^T} \boldsymbol{x}|^2 = D$$



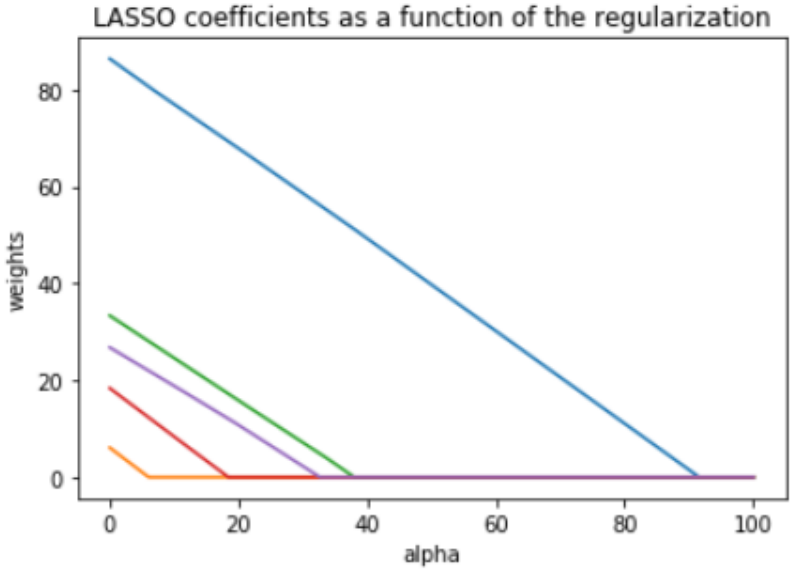
$$\lambda \sum_{j=1}^{J} |\hat{\theta}_j^{LASSO}| = C$$



LASSO visualized



The Lasso estimator tends to zero out parameters



The values of the coefficients decrease as lambda increases, and are nullified fast.

Lasso vs Ridge

Lasso tends to generate sparser solutions than a quadratic regularizer.