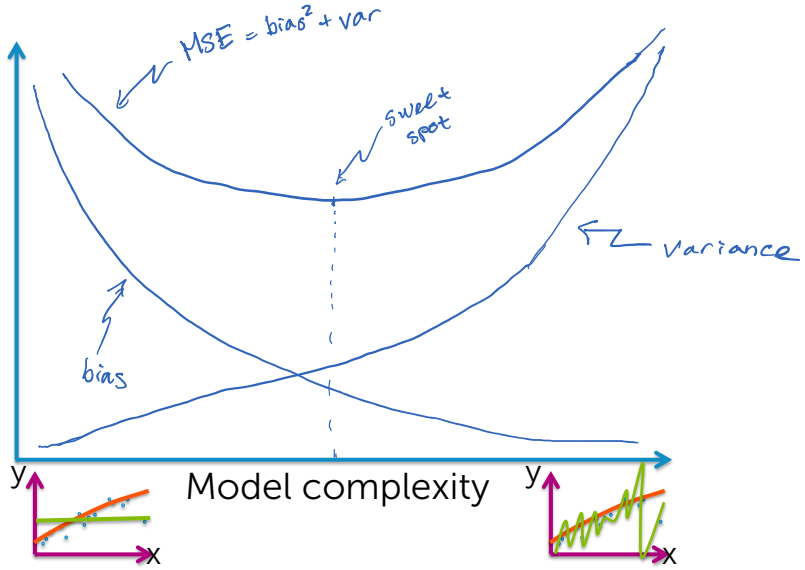
Ridge Regression Recap

CS 229: Machine Learning
Emily Fox
Stanford University
January 22, 2024

1

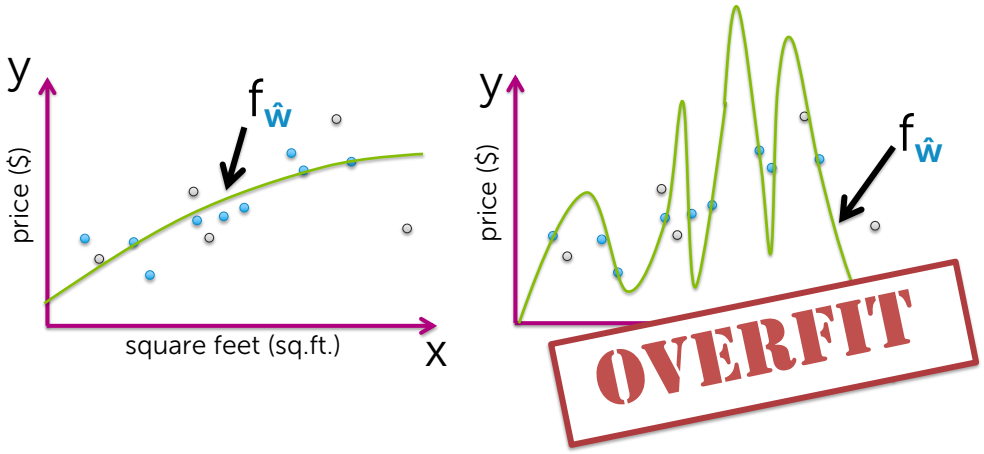
Bias-variance tradeoff



2

Flexibility of complex models (large D for linear regression)

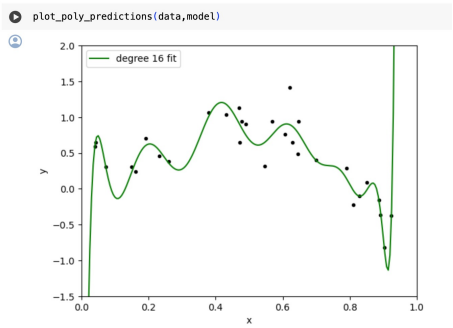
$$y_i = \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i) + \varepsilon_i$$



3

Symptom of overfitting

Often, overfitting associated with very large estimated $\hat{\mathbf{w}}$



Fit a degree-16 polynomial

```
model = polynomial_regression(data, deg=16)
print_coefficients(model)
```

```
Learned polynomial for degree 16:
             16             15             14             13
3.337e+08 x  - 2.226e+09 x  + 6.621e+09 x  - 1.156e+10 x
              12             11             10             9             8
 + 1.309e+10 x  - 1e+10 x  + 5.141e+09 x  - 1.658e+09 x + 2.26e+08 x
              7             6             5             4             3
 + 6.687e+07 x - 4.733e+07 x + 1.393e+07 x - 2.547e+06 x + 3.018e+05 x
              2
 - 2.188e+04 x + 839.3 x - 12.01
```

Woah!!!! Those coefficients are *crazy*! On the order of 10^9.

4

Consider specific total cost

Total cost =

measure of fit + measure of magnitude of coefficients

RSS($\mathbf{w}$) $\quad$ $||\mathbf{w}||_2^2$

5

Consider resulting objective

What if $\hat{\mathbf{w}}$ selected to minimize

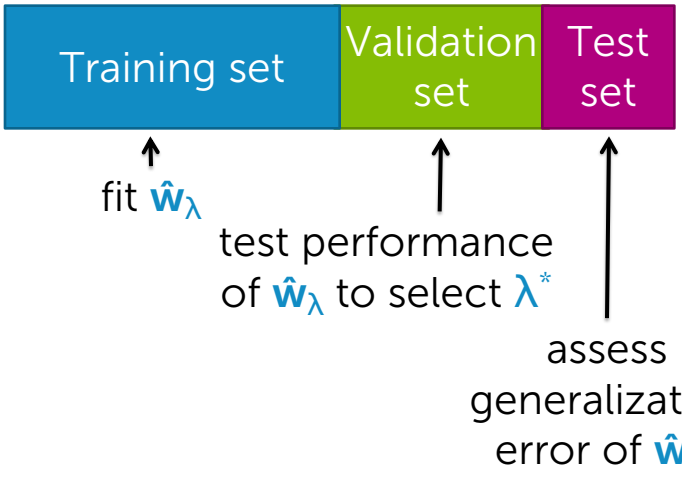
$$\text{RSS}(\mathbf{w}) + \lambda ||\mathbf{w}||_2^2$$

tuning parameter = balance of fit and magnitude

Ridge regression
(a.k.a L_2 regularization)

6

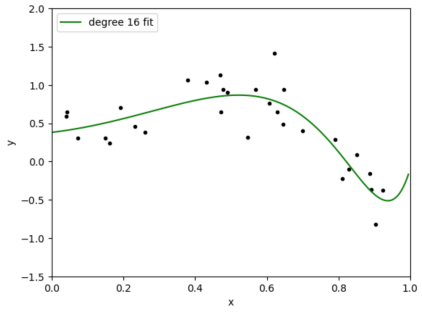
How to choose tuning parameter



7

Ridge penalty for polynomial regression (D=16)

```
plot_poly_predictions(data,model)
```




```
[ ] best_l2_penalty
```

```
0.003593813663804626
```

```
[ ] model = polynomial_ridge_regression(data, deg=16, l2_penalty=best_l2_penalty)
    print_coefficients(model)
```

```
Learned polynomial for degree 16:
          16          15          14          13          12          11
0.4539 x  + 0.4839 x  + 0.5053 x  + 0.5103 x  + 0.4885 x  + 0.4255 x
          10          9           8           7           6           5
 + 0.3026 x  + 0.09718 x - 0.2146 x - 0.6473 x - 1.179 x - 1.68 x
          4           3           2
 - 1.782 x - 0.727 x + 1.932 x + 0.5584 x + 0.3803
```


8

Lasso Regression:
Regularization for feature selection

CS 229: Machine Learning
Emily Fox
Stanford University
January 22, 2024

9

Feature selection task

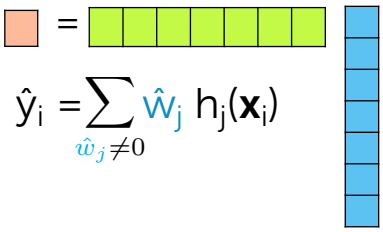
10

Why might you want to perform feature selection?

Efficiency:

- If size($\mathbf{w}$) = 100B, each prediction is expensive
- If $\hat{\mathbf{w}}$ sparse, computation only depends on # of non-zeros

many zeros



$$\hat{y}_i = \sum_{\hat{w}_j \neq 0} \hat{w}_j \, h_j(\mathbf{x}_i)$$

Interpretability:

- Which features are relevant for prediction?

11

Sparsity: Housing application

Lot size
Single Family
Year built
Last sold price
Last sale price/sqft
Finished sqft
Unfinished sqft
Finished basement sqft
floors
Flooring types
Parking type
Parking amount
Cooling
Heating
Exterior materials
Roof type
Structure style

Dishwasher
Garbage disposal
Microwave
Range / Oven
Refrigerator
Washer
Dryer
Laundry location
Heating type
Jetted Tub
Deck
Fenced Yard
Lawn
Garden
Sprinkler System
⋮

12

Option 1: All subsets or greedy variants

13

Exhaustive approach: "all subsets"

Consider all possible models, each using a subset of features

How many models were evaluated? each indexed by features included

$y_i = \varepsilon_i$ [0 0 0 ... 0 0 0]

$y_i = w_0 h_0(\mathbf{x}_i) + \varepsilon_i$ [1 0 0 ... 0 0 0]

$y_i = w_1 h_1(\mathbf{x}_i) + \varepsilon_i$ [0 1 0 ... 0 0 0]

⋮ ⋮

$y_i = w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \varepsilon_i$ [1 1 0 ... 0 0 0]

⋮ ⋮

$y_i = w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \ldots + w_D h_D(\mathbf{x}_i) + \varepsilon_i$ [1 1 1 ... 1 1 1]

feat 0, feat 1, ..., feat D

2 2 2 ... 2

2^{D+1}

$2^8 = 256$

$2^{30} = 1{,}073{,}741{,}824$

$2^{1000} = 1.071509 \times 10^{301}$

2^{100B} = HUGE!!!!!!

Typically, computationally infeasible

14

Choosing model complexity?

Option 1: Assess on validation set

Option 2: Cross validation will see soon

Option 3+: Other metrics for penalizing model complexity like BIC…

15

Greedy algorithms

Forward stepwise:
Starting from simple model and iteratively add features most useful to fit

Backward stepwise:
Start with full model and iteratively remove features least useful to fit

Combining forward and backward steps:
In forward algorithm, insert steps to remove features no longer as important

Lots of other variants, too.

16

Option 2: Regularize

17

Using regularization for feature selection

Instead of searching over a **discrete** set of solutions, can we use regularization?

- Start with full model (all possible features)
- "Shrink" some coefficients *exactly* to 0
  - i.e., knock out certain features
- Non-zero coefficients indicate "selected" features

18

Ridge regression: L_2 regularized regression

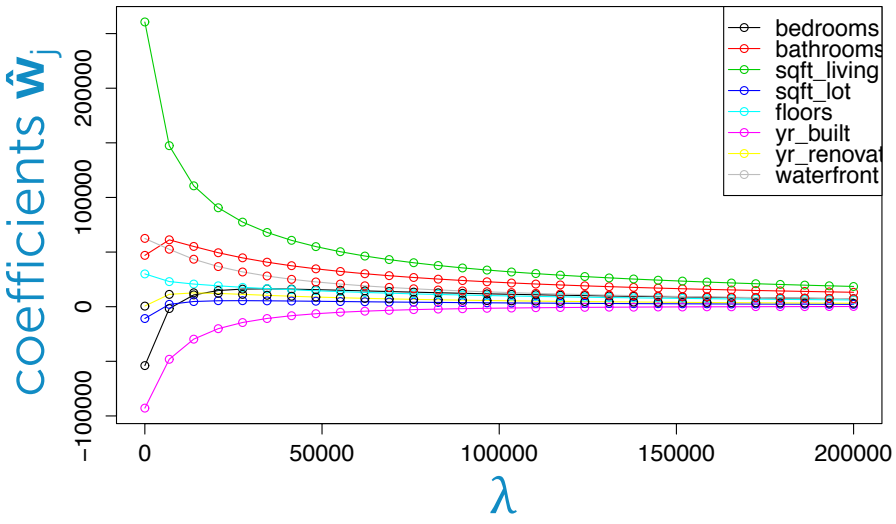
Total cost =
measure of fit + λ measure of magnitude of coefficients

RSS(**w**)

$||\mathbf{w}||_2^2 = w_0^2 + \ldots + w_D^2$

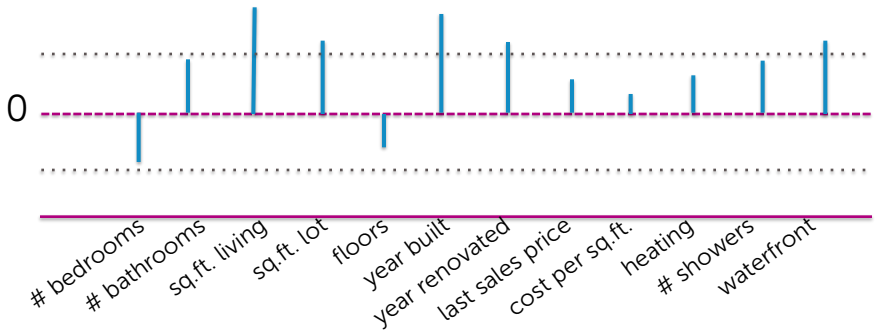
Encourages **small weights**
but not exactly 0

Coefficient path – **ridge**



Thresholding ridge coefficients?

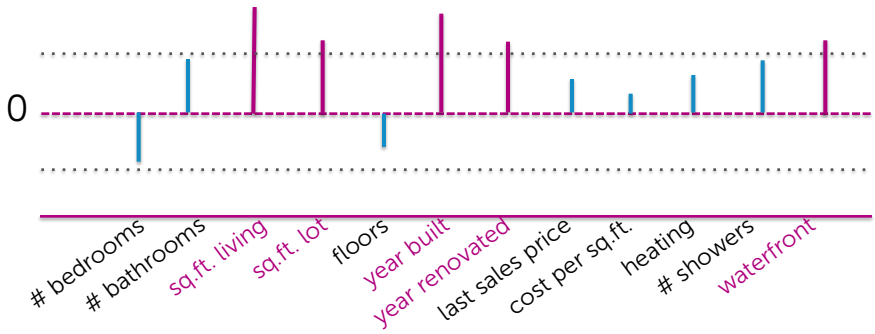
Why don't we just set small ridge coefficients to 0?



21

Thresholding ridge coefficients?

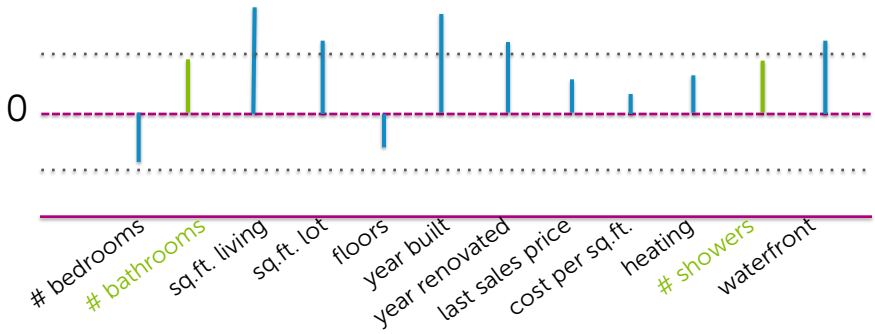
Selected features for a given threshold value



22

Thresholding ridge coefficients?

Let's look at two related features…

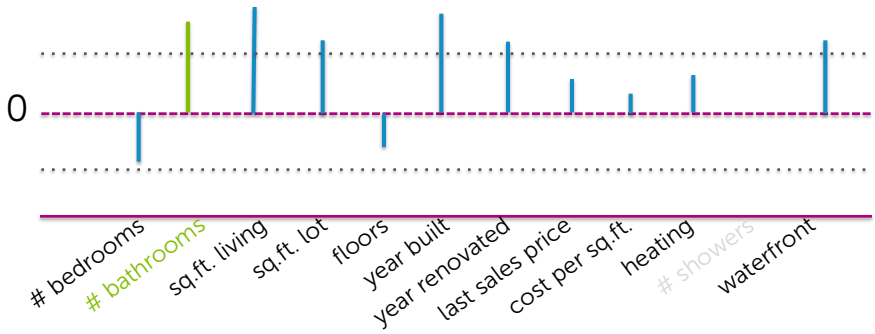


Nothing measuring bathrooms was included!

23

Thresholding ridge coefficients?

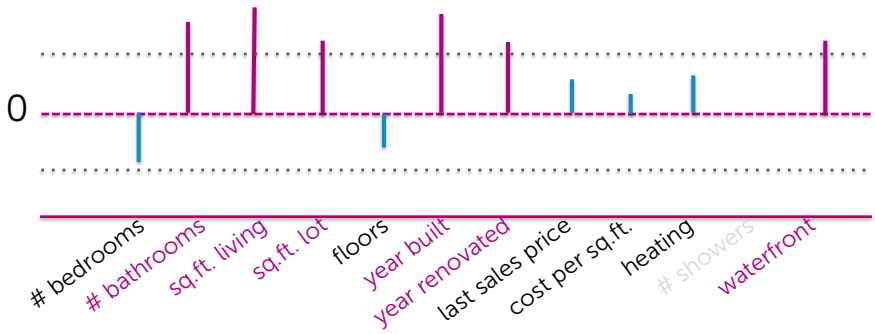
If only one of the features had been included…



24

Thresholding ridge coefficients?

Would have included bathrooms in selected model



Can regularization lead directly to sparsity?

25

Try this cost instead of ridge…

Total cost =

measure of fit + λ measure of magnitude of coefficients

RSS(**w**)

$||\mathbf{w}||_1 = |w_0| + \ldots + |w_D|$

Leads to **sparse** solutions!

Lasso regression
(a.k.a. L_1 regularized regression)

26

Lasso regression: L_1 regularized regression

Just like ridge regression, solution is governed by a continuous parameter λ

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1$$

tuning parameter = balance of fit and **sparsity**

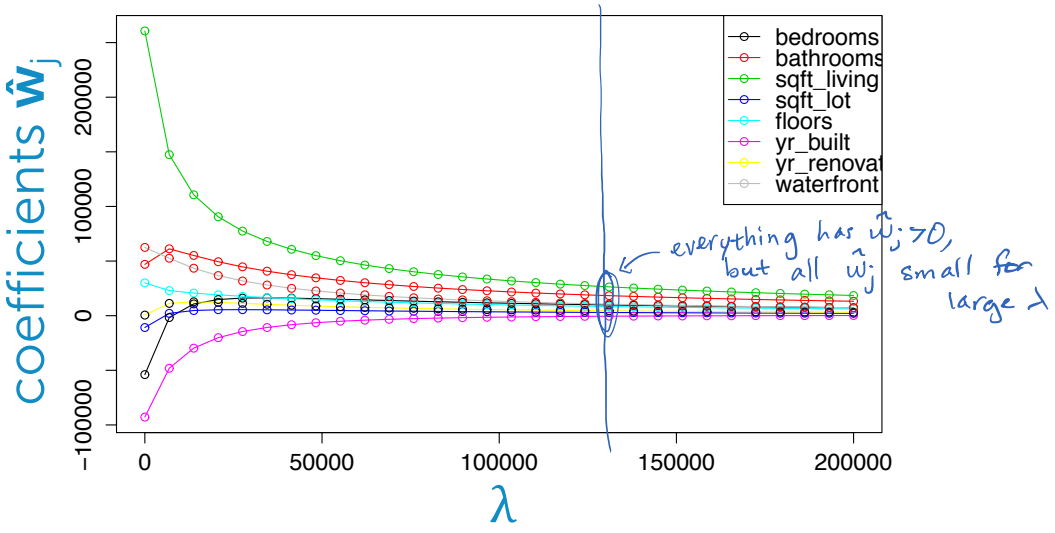
If λ=0: $\hat{w}^{lasso} = \hat{w}^{LS}$ (unreg. soln)

If λ=∞: $\hat{w}^{lasso} = 0$

If λ in between: $0 \le ||\hat{w}^{lasso}||_1 \le ||\hat{w}^{LS}||_1$

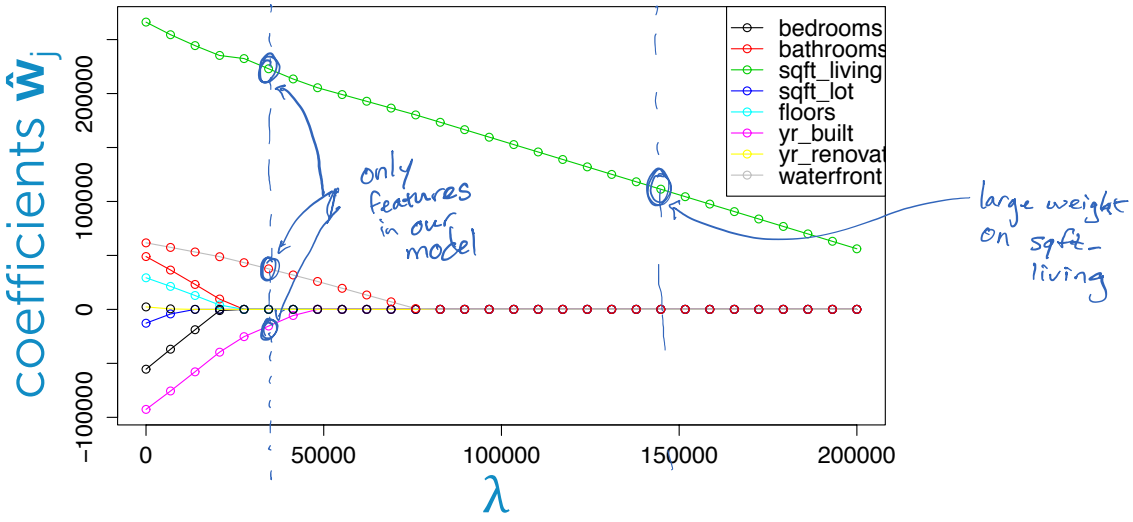
27

Coefficient path – **ridge**



28

Coefficient path – **lasso**



29

Revisit polynomial fit demo

What happens if we refit our high-order polynomial, but now using **lasso regression**?

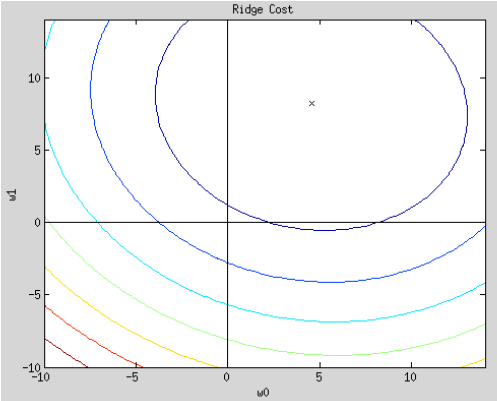
Will consider a few settings of λ …

30

Geometric intuition for sparsity of lasso solution

31

Visualizing the ridge cost in 2D



$RSS(w) + \lambda \|w\|_2^2$

ellipses ↑ circles

weighting

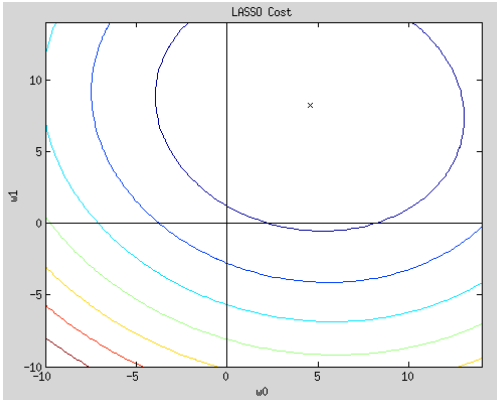
Movie: fcn of increasing λ

x = opt soln for a specific λ

$$\text{RSS}(\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2 = \sum_{i=1}^{N}(y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda\,(w_0^2 + w_1^2)$$

32

Visualizing the lasso cost in 2D



RSS(w) + λ||w||,
ellipses diamonds

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda\,(|w_0| + |w_1|)$$

33

Fitting the lasso regression model (for given λ value)

34

How we optimized past objectives

To solve for $\hat{\mathbf{w}}$, previously took gradient of total cost objective and either:

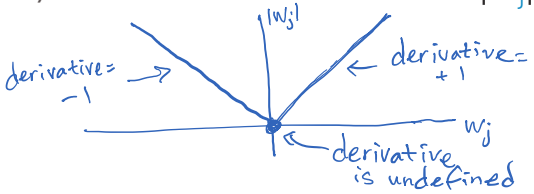
1) Derived closed-form solution
2) Used in gradient descent algorithm

Optimizing the lasso objective

Lasso total cost: $\text{RSS}(\mathbf{w}) + \lambda ||\mathbf{w}||_1$

Issues:

1) What's the derivative of $|w_j|$?



gradients → subgradients

2) Even if we could compute derivative, no closed-form solution

can use subgradient descent

Aside 1: Coordinate descent

37

Coordinate descent

Goal: Minimize some function g

$\min_{\mathbf{w}} g(\mathbf{w})$

$g(\mathbf{w}) = g(w_0, w_1, \ldots, w_D)$

when keeping others fixed

Often, hard to find minimum for all coordinates, but easy for each coordinate

Coordinate descent:

Initialize $\hat{\mathbf{w}} = 0$ (or smartly…)

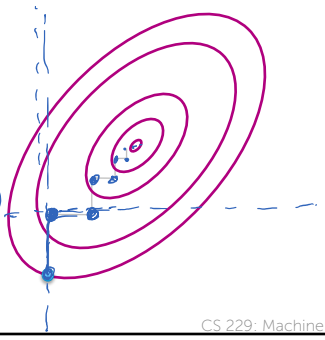
while not converged

pick a coordinate j

$\hat{w}_j \leftarrow \min_{w} g(\hat{w}_0, \hat{w}_1, \ldots, \hat{w}_{j-1}, w, w_{j+1}, \ldots, \hat{w}_D)$

values from prev. iter

just min jth coord



38

Comments on coordinate descent

How do we pick next coordinate?

- At random ("random" or "stochastic" coordinate descent), round robin, …

No stepsize to choose!

Super useful approach for *many* problems

- Converges to optimum in some cases (e.g., "strongly convex")
- Converges for lasso objective

39

Aside 2: Normalizing features

40

Normalizing features

Scale training **columns** (not rows!) as:

$$\underline{h}_j(\mathbf{x}_k) = \frac{h_j(\mathbf{x}_k)}{\sqrt{\sum_{i=1}^{N} h_j(\mathbf{x}_i)^2}}$$

Normalizer: Z_j

Apply same training scale factors to test data:

$$\underline{h}_j(\mathbf{x}_k) = \frac{h_j(\mathbf{x}_k)}{\sqrt{\sum_{i=1}^{N} h_j(\mathbf{x}_i)^2}}$$

Normalizer: Z_j

apply to test point

summing over training points

41

Aside 3: Coordinate descent for unregularized regression (for normalized features)

42

Optimizing least squares objective one coordinate at a time

$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))^2$$

normalized feature

Fix all coordinates $\mathbf{w}_{-j}$ and take partial w.r.t. w_j

all w_k except w_j

$$\frac{\partial}{\partial w_j} \text{RSS}(\mathbf{w}) = -2 \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))$$

$\sum_{k \neq j} w_k h_k - w_j h_j$

$$= -2 \sum h_j (y_i - \sum_{k \neq j} w_k h_k) + 2 w_j \sum_{i=1}^{N} h_j^2$$

by defn of normalized feat. $= 1$

$= \rho_j$

$$= -2\rho_j + 2 w_j$$

43

Optimizing least squares objective one coordinate at a time

$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))^2$$

Set partial = 0 and solve

$$\frac{\partial}{\partial w_j} \text{RSS}(\mathbf{w}) = -2\rho_j + 2 w_j = 0$$

$$\hat{w}_j = \rho_j$$

44

Coordinate descent for least squares regression

Initialize $\hat{\mathbf{w}} = 0$ (or smartly…)

while not converged

for j=0,1,…,D

compute: $\rho_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

residual *without* feature j

prediction *without* feature j

set: $\hat{w}_j = \rho_j$

Coordinate descent for lasso (for normalized features)

Coordinate descent for least squares regression

Initialize $\hat{\mathbf{w}} = 0$ (or smartly…)

while not converged

for j=0,1,…,D

compute: $\rho_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

residual *without* feature j

prediction *without* feature j

set: $\hat{w}_j = \rho_j$

Coordinate descent for lasso

Initialize $\hat{\mathbf{w}} = 0$ (or smartly…)

while not converged

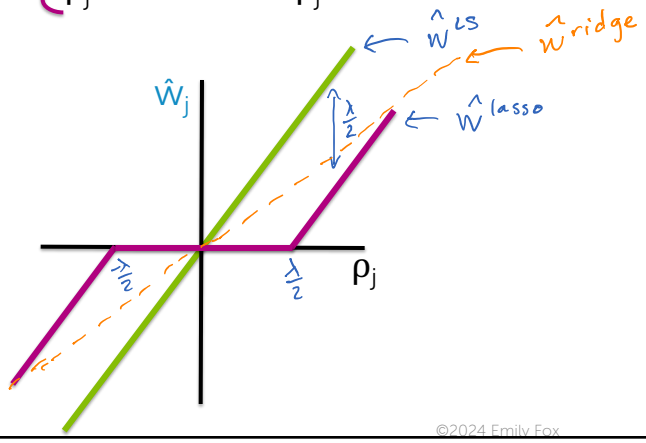
for j=0,1,…,D

compute: $\rho_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

set: $\hat{w}_j = \begin{cases} \rho_j + \lambda/2 & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\ \rho_j - \lambda/2 & \text{if } \rho_j > \lambda/2 \end{cases}$

Soft thresholding

$$\hat{w}_j = \begin{cases} \rho_j + \lambda/2 & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\ \rho_j - \lambda/2 & \text{if } \rho_j > \lambda/2 \end{cases}$$



How to assess convergence?

Initialize $\hat{\mathbf{w}} = 0$ (or smartly…)

while not converged

for j=0,1,…,D

compute: $\rho_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

set: $\hat{w}_j = \begin{cases} \rho_j + \lambda/2 & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\ \rho_j - \lambda/2 & \text{if } \rho_j > \lambda/2 \end{cases}$

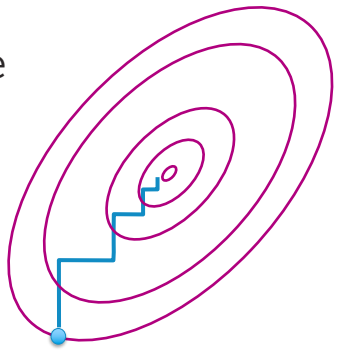
Convergence criteria

When to stop?

For convex problems, will start to take smaller and smaller steps

Measure size of steps taken in a full loop over all features

– stop when max step < ε



51

Other lasso solvers

Classically: Least angle regression (LARS) [Efron et al. '04]

Then: Coordinate descent algorithm [Fu '98, Friedman, Hastie, & Tibshirani '08]

Now:

- Parallel CD (e.g., Shotgun, [Bradley et al. '11])
- Other parallel learning approaches for linear models
  - Parallel stochastic gradient descent (SGD) (e.g., Hogwild! [Niu et al. '11])
  - Parallel independent solutions then averaging [Zhang et al. '12]
- Alternating directions method of multipliers (ADMM) [Boyd et al. '11]

52

Coordinate descent for lasso (for unnormalized features)

53

Coordinate descent for lasso with unnormalized features

Precompute: $z_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)^2$

Initialize $\hat{\mathbf{w}} = 0$ (or smartly…)

while not converged

for j=0,1,…,D

compute: $\rho_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

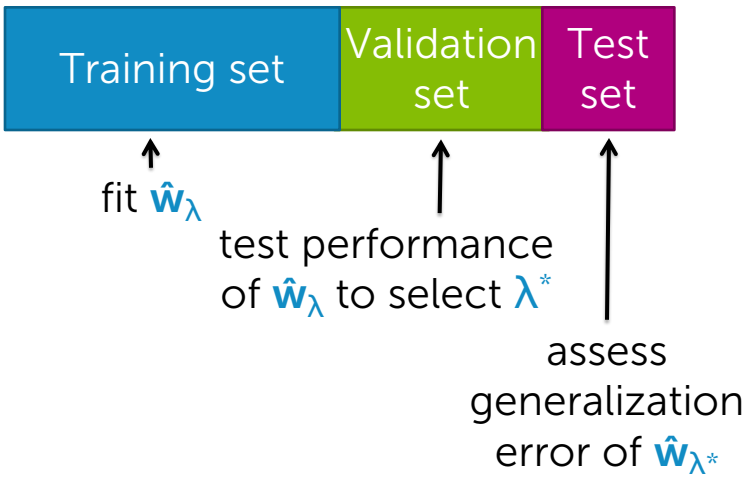
set: $\hat{w}_j = \begin{cases} (\rho_j + \lambda/2)/z_j & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\ (\rho_j - \lambda/2)/z_j & \text{if } \rho_j > \lambda/2 \end{cases}$

54

How to choose λ

55

If sufficient amount of data…



56

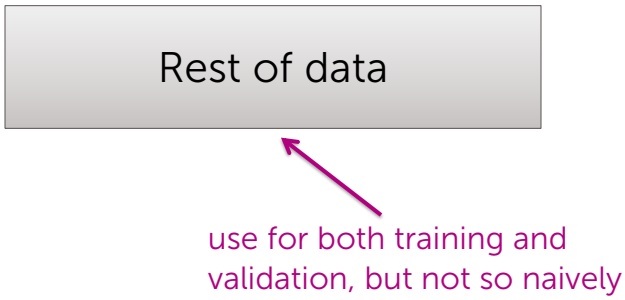
Start with smallish dataset

57

Still form test set and hold out

58

How do we use the other data?



59

Recall naïve approach

Is validation set enough to compare performance of $\hat{\mathbf{w}}_\lambda$ across λ values?

No

60

Choosing the validation set

Didn't have to use the last data points tabulated to form validation set

Can use any data subset

61

Choosing the validation set

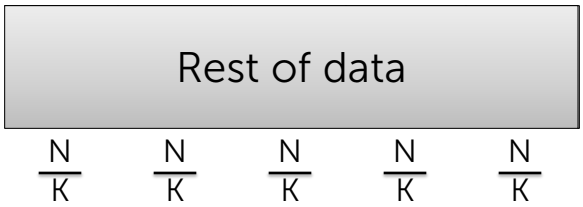
Which subset should I use?

ALL!

average performance over all choices

62

K-fold cross validation

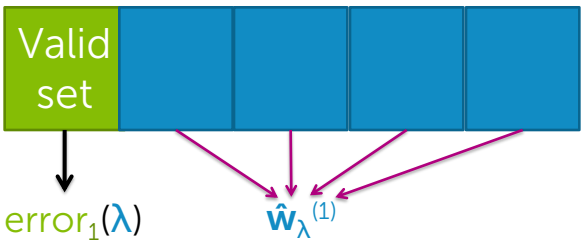


Preprocessing: Randomly assign data to K groups

(use same split of data for all other steps)

63

K-fold cross validation

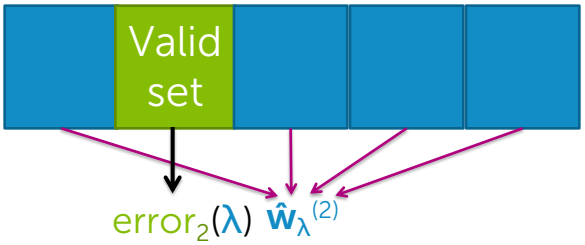


For k=1,…,K

1. Estimate $\hat{w}_\lambda^{(k)}$ on the training blocks
2. Compute error on validation block: $error_k(\lambda)$

64

K-fold cross validation

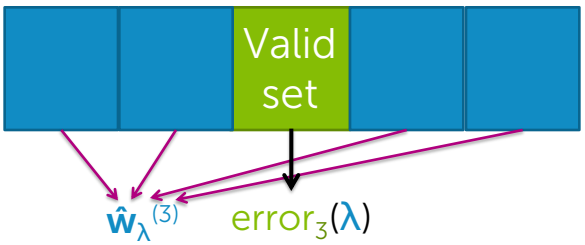


For k=1,…,K

1. Estimate $\hat{w}_\lambda^{(k)}$ on the training blocks
2. Compute error on validation block: $error_k(\lambda)$

65

K-fold cross validation

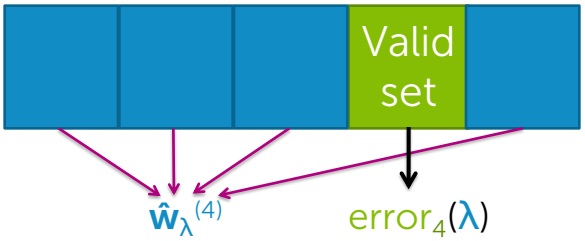


For k=1,…,K

1. Estimate $\hat{w}_\lambda^{(k)}$ on the training blocks
2. Compute error on validation block: $error_k(\lambda)$

66

K-fold cross validation

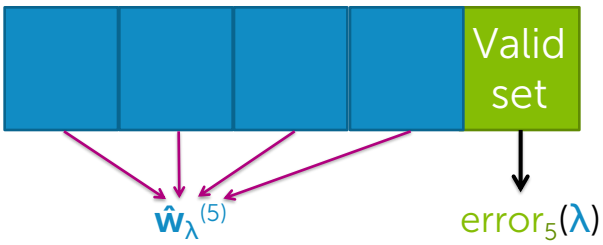


For k=1,…,K

1. Estimate $\hat{w}_\lambda^{(k)}$ on the training blocks
2. Compute error on validation block: $error_k(\lambda)$

67

K-fold cross validation



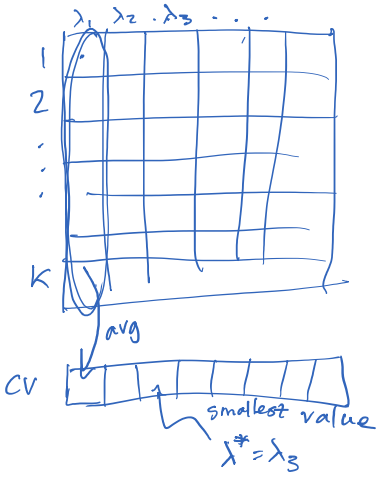
For k=1,…,K

1. Estimate $\hat{w}_\lambda^{(k)}$ on the training blocks
2. Compute error on validation block: $error_k(\lambda)$

Compute average error: $CV(\lambda) = \frac{1}{K}\sum_{k=1}^{K} error_k(\lambda)$

68

K-fold cross validation



Repeat procedure for each choice of λ

Choose λ* to minimize CV(λ)

69

What value of K?

Formally, the best approximation occurs for validation sets of size 1 (K=N)

leave-one-out cross validation

Computationally intensive

- requires computing N fits of model per λ

Typically, K=5 or 10

5-fold CV **10-fold CV**

70

Choosing λ via cross validation for lasso

Cross validation is choosing the λ that provides best predictive accuracy

Tends to favor less sparse solutions, and thus smaller λ, than optimal choice for feature selection

extensions developed that are selection consistent

c.f., "Probabilistic Machine Learning: An Introduction", Murphy, 2022 for further discussion

71

Practical concerns with lasso

72

Debiasing lasso

Lasso shrinks coefficients relative to LS solution
→ more bias, less variance

Can reduce bias as follows:

1. Run lasso to select features
2. Run least squares regression with only selected features

"Relevant" features no longer shrunk relative to LS fit of same reduced model

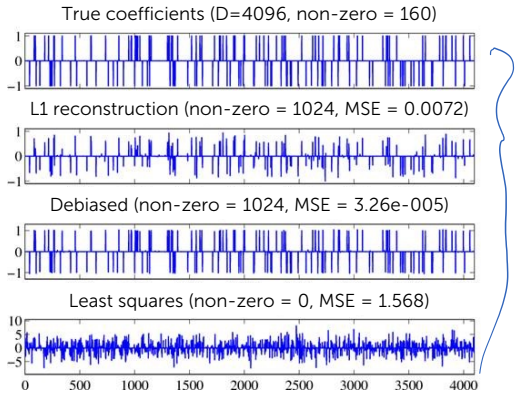


Figure used with permission of Mario Figueiredo
(*captions modified to fit course*)

Issues with standard lasso objective

1. With group of highly correlated features, lasso tends to select amongst them arbitrarily
   - Often prefer to select all together
2. Often, empirically ridge has better predictive performance than lasso, but lasso leads to sparser solution

Elastic net aims to address these issues

- hybrid between lasso and ridge regression
- uses L_1 and L_2 penalties

See Zou & Hastie '05 for further discussion

Summary for feature selection and lasso regression

75

What you can do now…

- Describe "all subsets" and greedy variants for feature selection
- Analyze computational costs of these algorithms
- Formulate lasso objective
- Describe what happens to estimated lasso coefficients as tuning parameter λ is varied
- Interpret lasso coefficient path plot
- Contrast ridge and lasso regression
- Estimate lasso regression parameters using an iterative coordinate descent algorithm
- Implement K-fold cross validation to select lasso tuning parameter λ

76

Deriving the lasso coordinate descent update

77

Optimizing lasso objective one coordinate at a time

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}\Big(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i)\Big)^2 + \lambda \sum_{j=0}^{D}|w_j|$$

Fix all coordinates $\mathbf{w}_{-j}$ and take partial w.r.t. w_j

derive *without* normalizing features

78

Part 1: Partial of RSS term

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))^2 + \lambda \sum_{j=0}^{D}|w_j|$$

$$\frac{\partial}{\partial w_j}\text{RSS}(\mathbf{w}) = -2\sum_{i=1}^{N} h_j(\mathbf{x}_i)\left(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i)\right)$$

$$= -2\sum_{i=1}^{N} h_j(x_i)\left(y_i - \sum_{k\neq j} w_k h_k(x_i) - w_j h_j(x_i)\right)$$

$$= \underbrace{-2\sum_{i=1}^{N} h_j(x_i)\left(y_i - \sum_{k\neq j} w_k h_k(x_i)\right)}_{\triangleq \rho_j} + 2w_j \underbrace{\sum_{i=1}^{N} h_j(x_i)^2}_{\triangleq z_j}$$

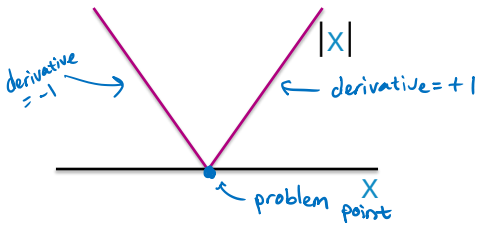
$$= -2\rho_j + 2w_j z_j$$

79

Part 2: Partial of L_1 penalty term

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))^2 + \lambda \sum_{j=0}^{D}|w_j|$$

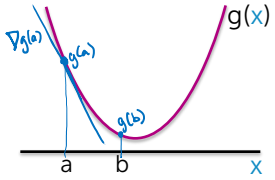
$$\lambda\frac{\partial}{\partial w_j}|w_j| = ???$$



80

Subgradients of convex functions

Gradients lower bound convex functions:

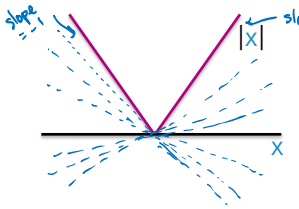


$g(b) \geq g(a) + \nabla g(a)(b-a)$

unique at x if function differentiable at x

Subgradients: Generalize gradients to non-differentiable points:

- Any plane that lower bounds function



For $|x|$ $\forall \in [-1,1]$

$\forall \in \partial g(x)$ subgradient of g at x if

$g(b) \geq g(a) + \forall (b-a)$

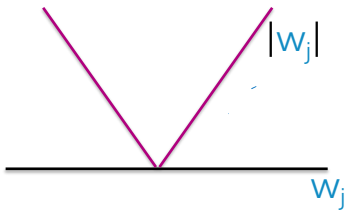
81

Part 2: *Subgradient* of L_1 term

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))^2 + \lambda \sum_{j=0}^{D} |w_j|$$

$$\lambda \partial_{w_j} |w_j| = \begin{cases} -\lambda & \text{when } w_j < 0 \\ [-\lambda, \lambda] & \text{when } w_j = 0 \\ \lambda & \text{when } w_j > 0 \end{cases}$$

$\lambda \cdot -1$ ← $-\lambda$; $\lambda \cdot [-1,1]$ ← $[-\lambda,\lambda]$; $\lambda \cdot 1$ ← λ



82

Putting it all together…

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))^2 + \lambda\sum_{j=0}^{D}|w_j|$$

$$\partial_{w_j}[\text{lasso cost}] = \overbrace{2z_j w_j - 2\rho_j}^{\text{from RSS}} + \overbrace{\begin{cases} -\lambda & \text{when } w_j < 0 \\ [-\lambda, \lambda] & \text{when } w_j = 0 \\ \lambda & \text{when } w_j > 0 \end{cases}}^{\text{from L1 penalty}}$$

$$= \begin{cases} 2z_j w_j - 2\rho_j - \lambda & \text{when } w_j < 0 \\ [-2\rho_j - \lambda, -2\rho_j + \lambda] & \text{when } w_j = 0 \\ 2z_j w_j - 2\rho_j + \lambda & \text{when } w_j > 0 \end{cases}$$

83

Optimal solution: Set subgradient = 0

$$\partial_{w_j}[\text{lasso cost}] = \begin{cases} 2z_j w_j - 2\rho_j - \lambda & \text{when } w_j < 0 \\ [-2\rho_j - \lambda, -2\rho_j + \lambda] & \text{when } w_j = 0 \\ 2z_j w_j - 2\rho_j + \lambda & \text{when } w_j > 0 \end{cases} = 0$$

Case 1 ($w_j < 0$):	$2z_j\hat{w}_j - 2\rho_j - \lambda = 0$ $\hat{w}_j = \frac{2\rho_j + \lambda}{2z_j} = \frac{\rho_j + \frac{\lambda}{2}}{z_j}$	For $\hat{w}_j < 0$, need $\rho_j < -\frac{\lambda}{2}$
Case 2 ($w_j = 0$):	$\hat{w}_j = 0$	For $\hat{w}_j = 0$, need $[-2\rho_j - \lambda, -2\rho_j + \lambda]$ to contain 0: $-2\rho_j + \lambda \geq 0 \rightarrow \rho_j \leq \frac{\lambda}{2}$ $\quad -2\rho_j - \lambda \leq 0 \rightarrow \rho_j \geq -\frac{\lambda}{2}$ $\quad -\frac{\lambda}{2} \leq \rho_j \leq \frac{\lambda}{2}$ so that $\hat{w}_j = 0$ is an optimum
Case 3 ($w_j > 0$):	$2z_j\hat{w}_j - 2\rho_j + \lambda = 0$ $\hat{w}_j = \frac{\rho_j - \frac{\lambda}{2}}{z_j}$	For $\hat{w}_j > 0$, need $\rho_j > \frac{\lambda}{2}$

84

Optimal solution: Set subgradient = 0

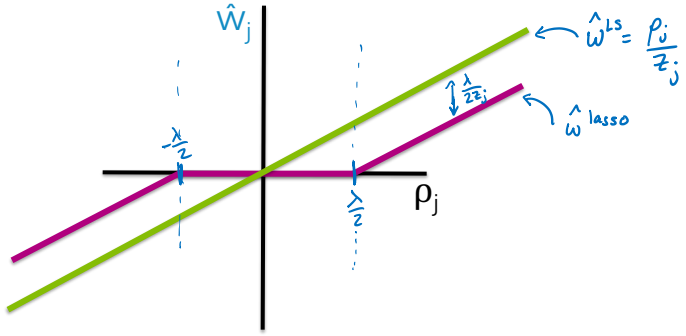
$$\partial_{w_j}[\text{lasso cost}] = \begin{cases} 2z_j w_j - 2\rho_j - \lambda & \text{when } w_j < 0 \\ [-2\rho_j - \lambda,\ -2\rho_j + \lambda] & \text{when } w_j = 0 \\ 2z_j w_j - 2\rho_j + \lambda & \text{when } w_j > 0 \end{cases}$$

$$= 0$$

$$\hat{w}_j = \begin{cases} (\rho_j + \lambda/2)/z_j & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2,\ \lambda/2] \\ (\rho_j - \lambda/2)/z_j & \text{if } \rho_j > \lambda/2 \end{cases}$$

Soft thresholding

$$\hat{w}_j = \begin{cases} (\rho_j + \lambda/2)/z_j & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2,\ \lambda/2] \\ (\rho_j - \lambda/2)/z_j & \text{if } \rho_j > \lambda/2 \end{cases}$$



Coordinate descent for lasso

Precompute: $z_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)^2$

Initialize $\hat{\mathbf{w}} = 0$ (or smartly…)

while not converged

for j=0,1,…,D

compute: $\rho_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

set: $\hat{w}_j = \begin{cases} (\rho_j + \lambda/2)/z_j & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\ (\rho_j - \lambda/2)/z_j & \text{if } \rho_j > \lambda/2 \end{cases}$

87

Geometric intuition for sparsity of lasso solution in more detail

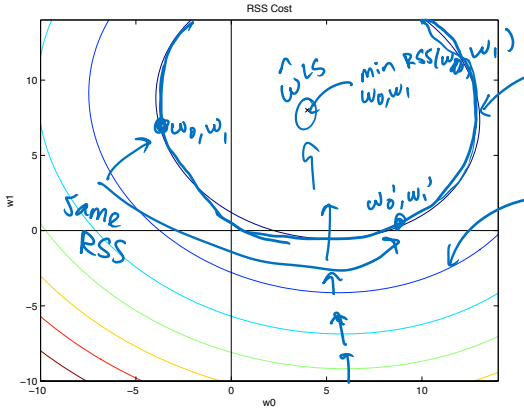
OPTIONAL

88

Geometric intuition for ridge regression

89

Visualizing the ridge cost in 2D



$RSS(w_0, w_1) = const_1$

$RSS(w_0, w_1) = const_2 > const_1$

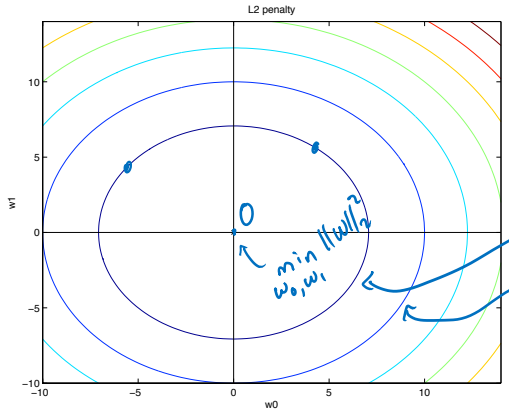
$\sum y_i^2 + w_0^2 \sum h_0^2 + w_1^2 \sum h_1^2 + \text{cross terms} = const$ (ellipse)

2 features for visualization

$$RSS(\mathbf{w}) + \lambda||\mathbf{w}||_2^2 = \sum_{i=1}^{N} (y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda (w_0^2 + w_1^2)$$

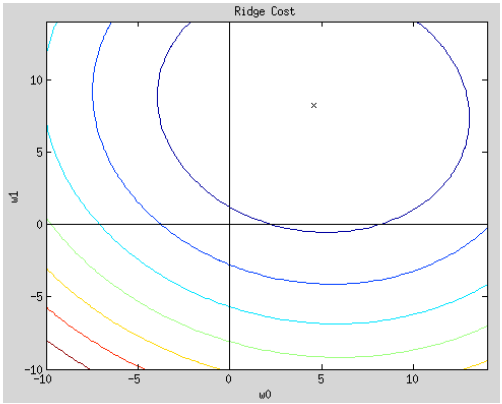
90

Visualizing the ridge cost in 2D



$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_2^2 = \sum_{i=1}^{N} (y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda (w_0^2 + w_1^2)$$

Visualizing the ridge cost in 2D



Add contour plots together

$\text{RSS}(w) + \lambda \|w\|_2^2$

ellipses ↑ circles

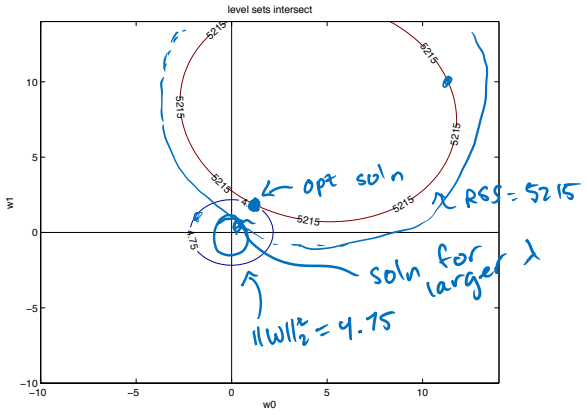
weighting

Movie: fcn of increasing λ

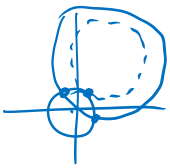
X = opt soln for specific λ

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_2^2 = \sum_{i=1}^{N} (y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda (w_0^2 + w_1^2)$$

Visualizing the ridge solution



For specific λ value, some balance between RSS and $\|w\|_2^2$



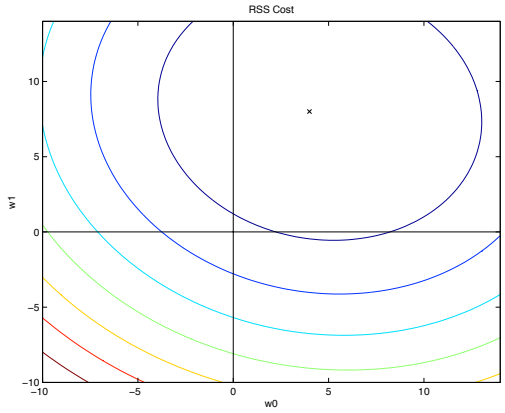
$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_2^2 = \sum_{i=1}^{N} (y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda (w_0^2 + w_1^2)$$

93

Geometric intuition for lasso

94

Visualizing the lasso cost in 2D

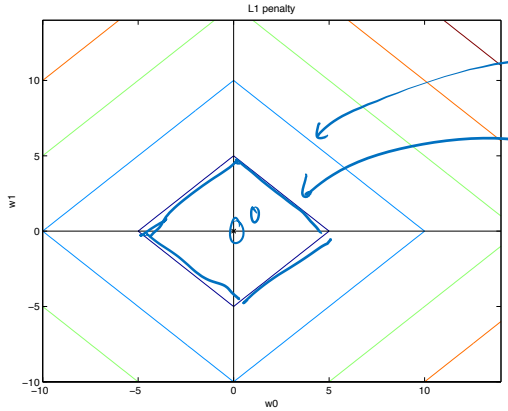


RSS contours for ridge + lasso are the same

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda\,(|w_0| + |w_1|)$$

95

Visualizing the lasso cost in 2D



$||w||_1 = \text{const}_2 > \text{const}_1$

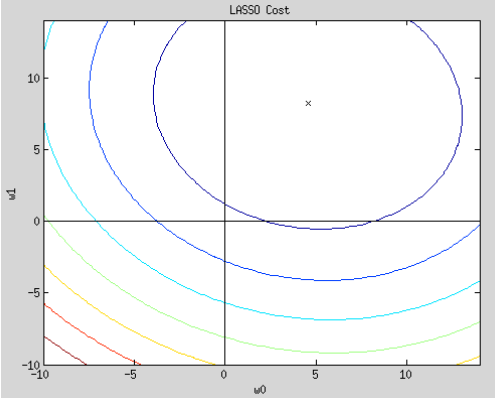
$||w||_1 = \text{const}_1$

$|w_0| + |w_1| = \text{const}$ diamonds

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda\,(|w_0| + |w_1|)$$

96

Visualizing the lasso cost in 2D



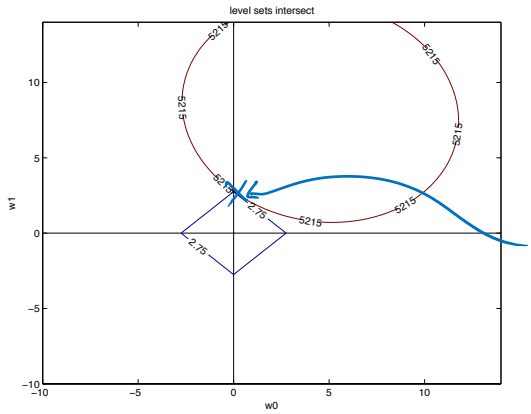
RSS(w) + λ ||w||, ellipses diamonds

Movie: as before

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda\,(|w_0| + |w_1|)$$

97

Visualizing the lasso solution



solution for specific value of λ

$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - w_0 h_0(\mathbf{x}_i) - w_1 h_1(\mathbf{x}_i))^2 + \lambda\,(|w_0| + |w_1|)$$

98