Lasso Regression:

Regularization for feature selection

CSE 446: Machine Learning
Emily Fox
University of Washington
January 18, 2017

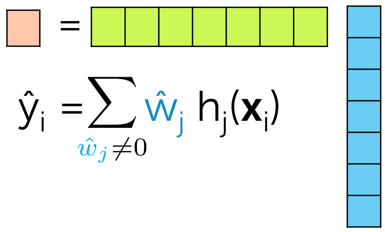
Feature selection task

Why might you want to perform feature selection?

Efficiency:

- If size($\mathbf{w}$) = 100B, each prediction is expensive
- If $\hat{\mathbf{w}}$ $\boxed{\text{sparse}}$, 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?

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
⋮

Option 1: All subsets or greedy variants

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]
$\vdots$	$\vdots$
$y_i = w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \varepsilon_i$	[1 1 0 … 0 0 0]
$\vdots$	$\vdots$
$y_i = w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \ldots + w_D h_D(\mathbf{x}_i) + \varepsilon_i$	[1 1 1 … 1 1 1]

2^D

$2^8 = 256$
$2^{30} = 1{,}073{,}741{,}824$
$2^{1000} = 1.071509 \times 10^{301}$
2^{100B} = HUGE!!!!!!

Typically, computationally infeasible

Choosing model complexity?

Option 1: Assess on validation set

Option 2: Cross validation

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

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.

Option 2: Regularize

Ridge regression: L_2 regularized regression

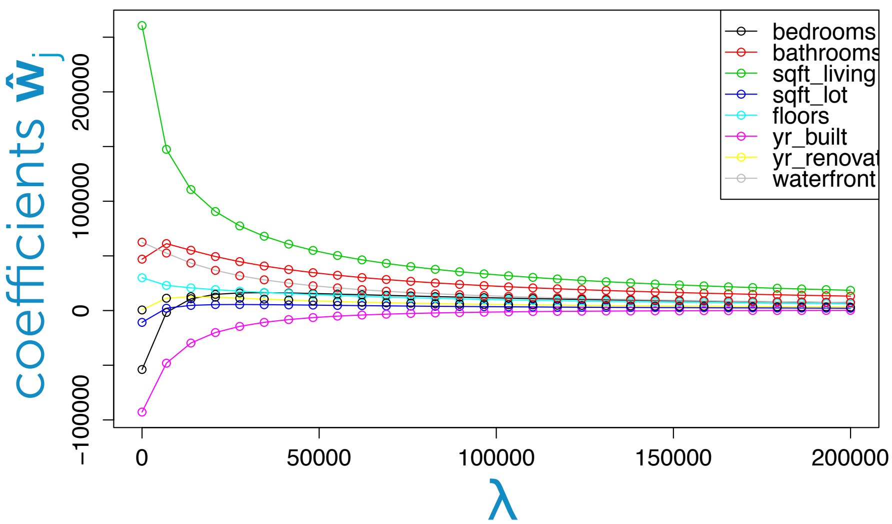
Total cost =

$$\underbrace{\text{measure of fit}}_{\text{RSS}(\mathbf{w})} + \lambda \underbrace{\text{measure of magnitude of coefficients}}_{||\mathbf{w}||_2^2 = w_0^2 + \ldots + w_D^2}$$

Encourages **small weights**

but not exactly 0

Coefficient path – ridge



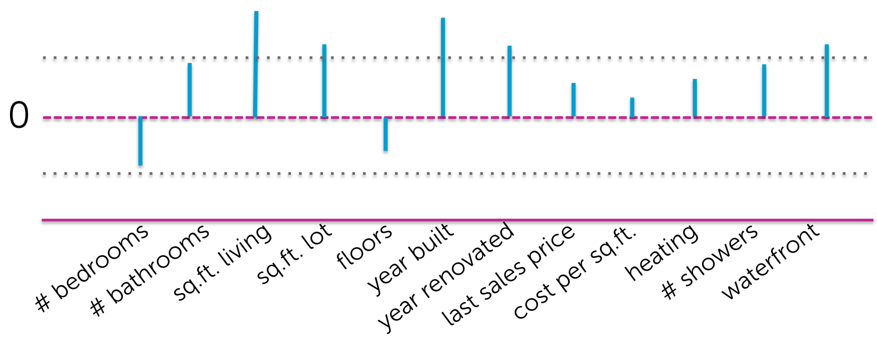
Using regularization for feature selection

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

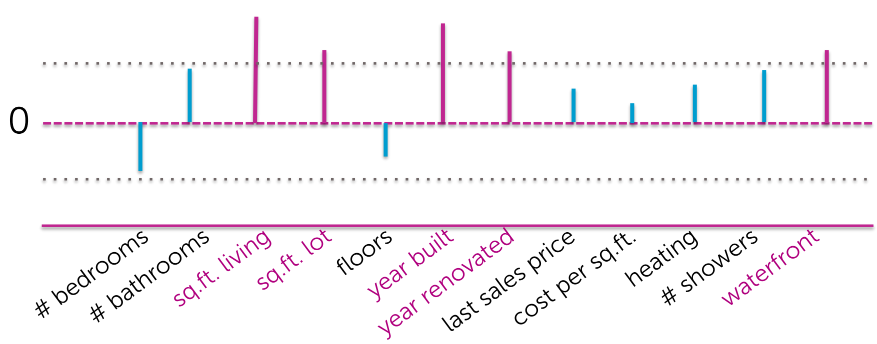
Thresholding ridge coefficients?

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



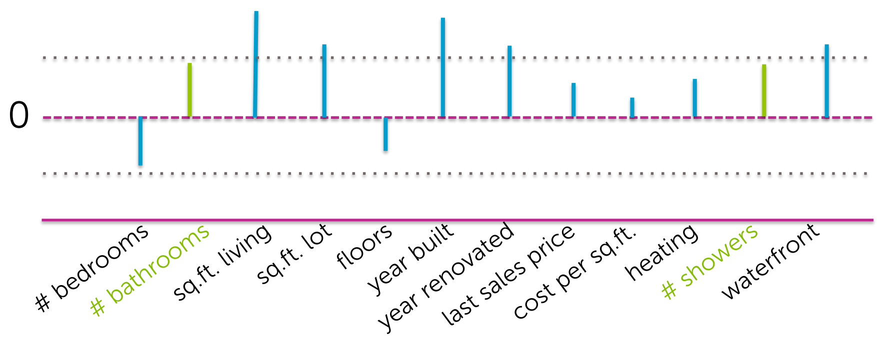
Thresholding ridge coefficients?

Selected features for a given threshold value



Thresholding ridge coefficients?

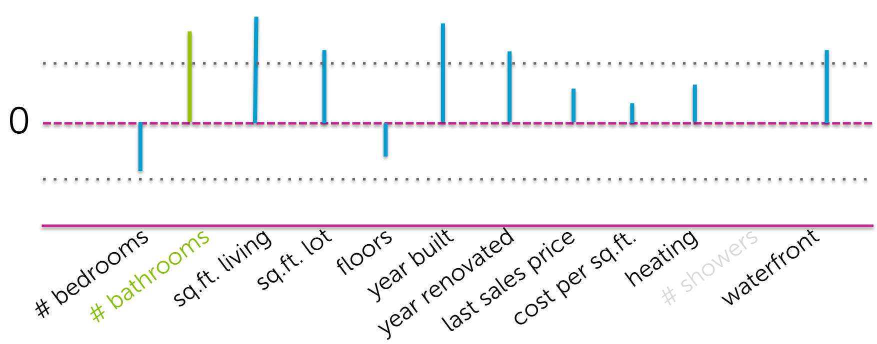
Let's look at two related features…



Nothing measuring bathrooms was included!

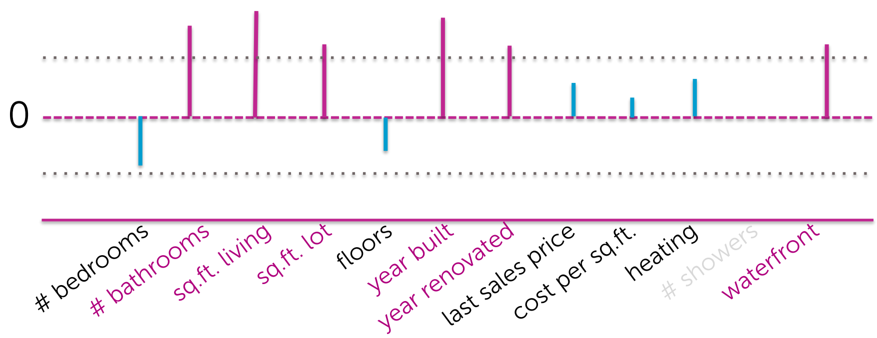
Thresholding ridge coefficients?

If only one of the features had been included…



Thresholding ridge coefficients?

Would have included bathrooms in selected model



Can regularization lead directly to sparsity?

Try this cost instead of ridge…

Total cost =

$$\underbrace{\text{measure of fit}}_{\text{RSS}(\mathbf{w})} + \lambda \underbrace{\text{measure of magnitude of coefficients}}_{||\mathbf{w}||_1 = |w_0| + \ldots + |w_D|}$$

Leads to **sparse** solutions!

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

Lasso regression: L_1 regularized regression

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

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

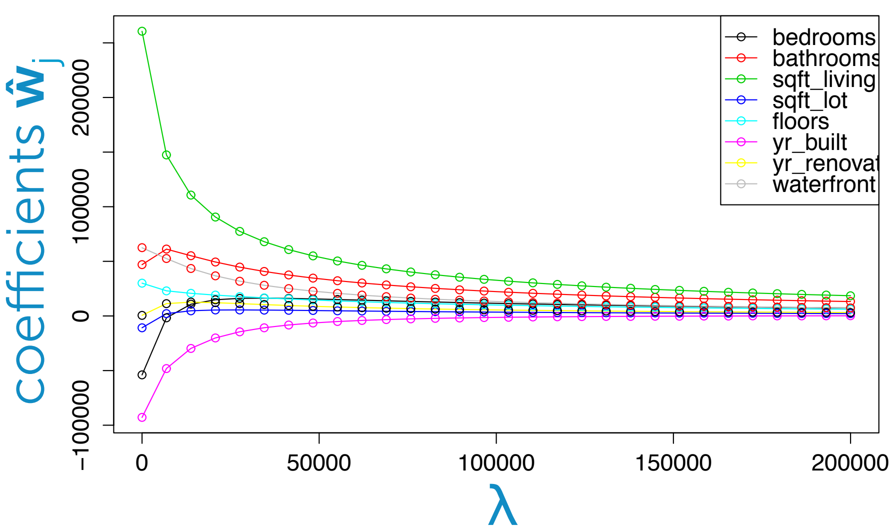
tuning parameter = balance of fit and **sparsity**

If λ=0:

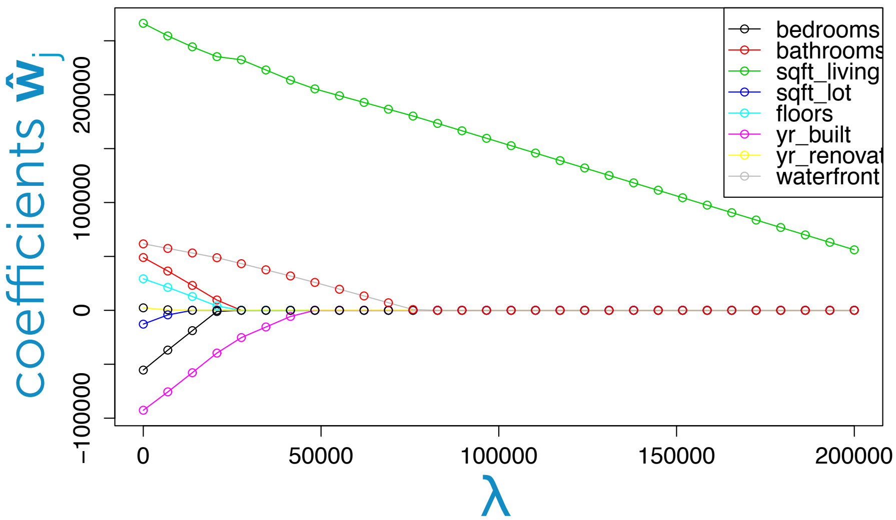
If λ=∞:

If λ in between:

Coefficient path – **ridge**



Coefficient path – **lasso**



Fitting the lasso regression model (for given λ value)

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

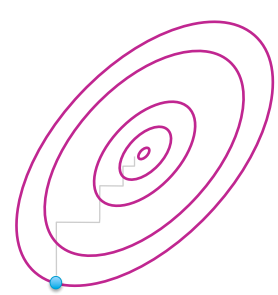
Coordinate descent

Goal: Minimize some function g

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
$\hat{w}_j \leftarrow$



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

Aside 2: Normalizing features

Normalizing features

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

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

Normalizer:
z_j

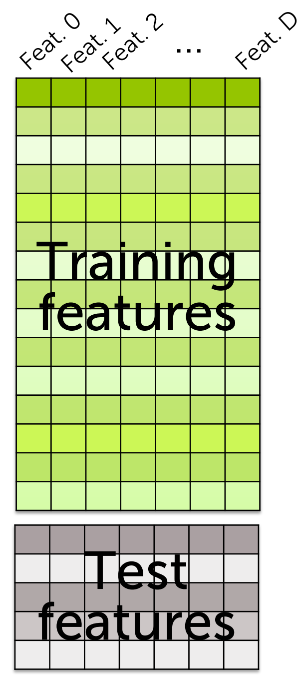
Apply same training scale factors to test data:

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

apply to test point

Normalizer:
z_j

summing over training points



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

Optimizing least squares objective one coordinate at a time

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

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

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

Optimizing least squares objective one coordinate at a time

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

Set partial = 0 and solve

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

Coordinate descent for least squares regression

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

while not converged

for j=0,1,...,D

residual *without* feature j

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

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

prediction *without* feature 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: $\rho_j = \sum_{i=1}^{N} \underline{h}_j(\mathbf{x}_i)\underbrace{(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))}_{\text{residual } \textit{without} \text{ feature j}}$

prediction *without* feature j

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

Coordinate descent for lasso

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

while not converged

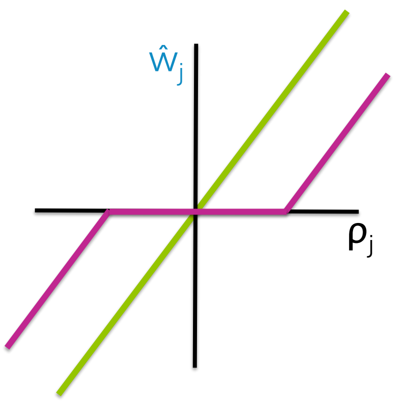
for j=0,1,...,D

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

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

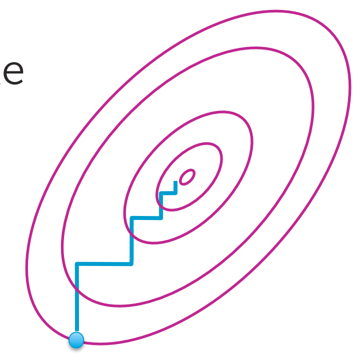
compute: $\rho_j = \sum_{i=1}^{N} \underline{h}_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

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

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]

Coordinate descent for lasso (for unnormalized features)

Coordinate descent for lasso with normalized features

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

while not converged

for j=0,1,…,D

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

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

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

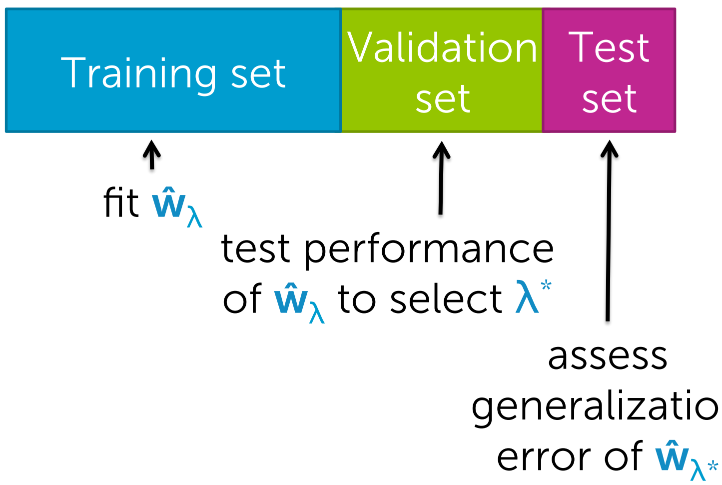
for j=0,1,…,D

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

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

How to choose λ

If sufficient amount of data…



Summary for feature selection and lasso regression

©2017 Emily Fox
CSE 446: Machine Learning

Impact of feature selection and lasso

Lasso has changed machine learning, statistics, & electrical engineering

But, for feature selection in general, be careful about interpreting selected features

- selection only considers features included
- sensitive to correlations between features
- result depends on algorithm used
- there are theoretical guarantees for lasso under certain conditions

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

Deriving the lasso coordinate descent update

OPTIONAL

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

Part 1: Partial of RSS term

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

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

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

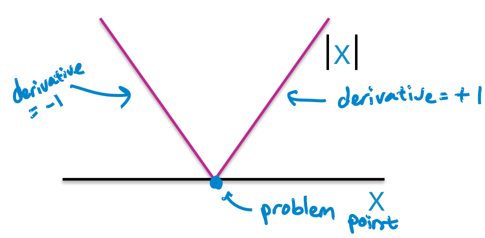
$$= \underbrace{-2\sum_{i=1}^{N} h_j(x_i)\Big(y_i - \sum_{k\neq j} w_k h_k(x_i)\Big)}_{\triangleq \rho_j} + 2w_j \underbrace{\sum_{i=1}^{N} h_j(x_i)^2}_{\triangleq z_j}$$

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

Part 2: Partial of L_1 penalty term

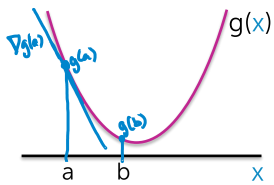
$$\text{RSS}(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i))^2 + \boxed{\lambda \sum_{j=0}^{D}|w_j|}$$

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



Subgradients of convex functions

Gradients lower bound convex functions:

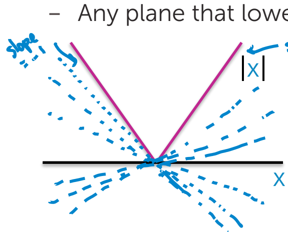


$g(b) \geq g(a) + \nabla g(a)(b-a)$

unique at x if function differentiable at x

Subgradients: Generalize gradients to non-differentiable points:

- Any plane that lower bounds function



For $|x|$

$V \in [-1,1]$

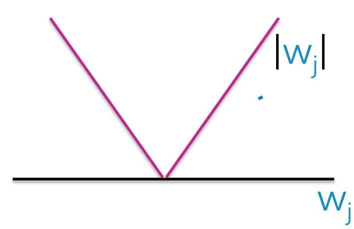
$V \in \partial g(x)$ subgradient of g at x if

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

Part 2: *Subgradient* of L_1 term

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

$$\lambda \partial_{w_j}|w_j| = \begin{cases} -\lambda \leftarrow \lambda \cdot -1 & \text{when } w_j < 0 \\ [-\lambda,\lambda] \leftarrow \lambda \cdot [-1,1] & \text{when } w_j = 0 \\ \lambda \leftarrow \lambda \cdot 1 & \text{when } w_j > 0 \end{cases}$$



Putting it all together...

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

$$\partial_{w_j}[\text{lasso cost}] = \overbrace{2z_j w_j - 2\rho_j}^{\text{from RSS}} + \overbrace{\begin{cases} -\lambda & \text{when } w_j < 0 \\ [-\lambda, \lambda] & \text{when } w_j = 0 \\ \lambda & \text{when } w_j > 0 \end{cases}}^{\text{from } \lambda\|\cdot\|_1 \text{ penalty}}$$

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

Optimal solution: Set subgradient = 0

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

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 so that $\hat{w}_j = 0$ is an optimum $-2\rho_j+\lambda \geq 0 \rightarrow \rho_j \leq \frac{\lambda}{2}$ $\quad -2\rho_j-\lambda \leq 0 \rightarrow \rho_j \geq -\frac{\lambda}{2}$ $\quad -\frac{\lambda}{2} \leq \rho_j \leq \frac{\lambda}{2}$
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 $\rho_j > \frac{\lambda}{2}$

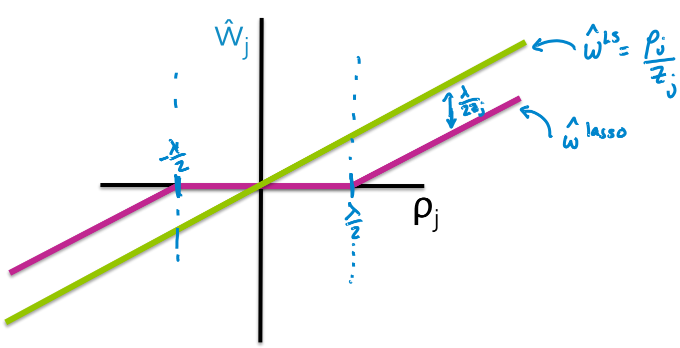
Optimal solution:
Set subgradient = 0

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

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

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