Ask Questions or say "Hi" before/during/after class!!!

CSE/STAT 416

Regularization – LASSO Regression

Hunter Schafer
University of Washington
April 7, 2021

Music: Ricky Montgomery

Pre-Lecture Video 1

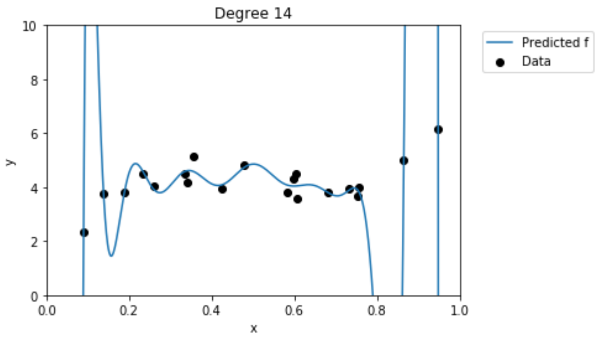
Recap Ridge

Recap: Number of Features

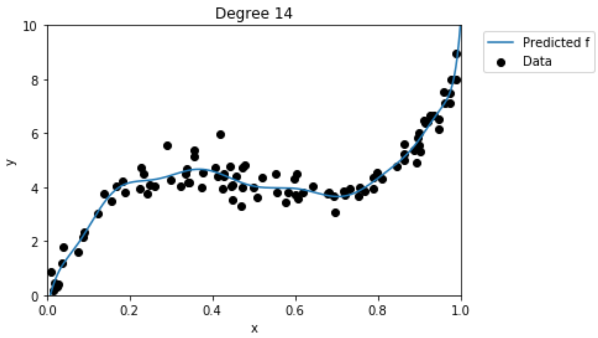
Overfitting is not limited to polynomial regression of large degree. It can also happen if you use a large number of features!

Why? Overfitting depends on how much data you have and if there is enough to get a representative sample for the complexity of the model.

$|w_j| \gg 0$



$|w_j| \approx$ reasonable



Recap: Ridge Regression

L2 norm $||w||_2^2 = \sum_{j=1}^{D} w_j^2$

Change quality metric to minimize

$$\hat{w} = \min_w RSS(W) + \underline{\lambda ||w||_2^2}$$

λ is tuning parameter that changes how much the model cares about the regularization term.

What if $\lambda = 0$?

$\hat{w} = \min_w RSS(w)$ exactly old problem!

$\rightarrow \hat{w}_{LS}$ This is called the <u>least squares</u> solution

What if $\lambda = \infty$?

If any $w_j \neq 0$, then $RSS(w) + \lambda ||w||_2^2 = \infty$

If $w = \vec{0}$ (all $w_j = 0$), then $RSS(w) + \lambda ||w||_2^2 = RSS(w)$ $<\infty$

Therefore, $\hat{w} = \vec{0}$ if $\lambda = \infty$

λ in between? $0 \leq ||\hat{w}||_2^2 \leq ||\hat{w}_{LS}||_2^2$

Poll Everywhere

Think

2 min

pollev.com/cs416

How should we choose the best value of λ?

- Pick the λ that has the smallest $RSS(\hat{w})$ on the **training set**
- Pick the λ that has the smallest $RSS(\hat{w})$ on the **test set**
- Pick the λ that has the smallest $RSS(\hat{w})$ on the **validation set**
- Pick the λ that has the smallest $RSS(\hat{w}) + \lambda||\hat{w}||_2^2$ on the **training set**
- Pick the λ that has the smallest $RSS(\hat{w}) + \lambda||\hat{w}||_2^2$ on the **test set**
- Pick the λ that has the smallest $RSS(\hat{w}) + \lambda||\hat{w}||_2^2$ on the **validation set**
- Pick the λ that results in the smallest coefficients
- Pick the λ that results in the largest coefficients
- None of the above

Choosing λ

For any particular setting of λ, use Ridge Regression objective

$$\hat{w}_{ridge} = \min_{w} RSS(w) + \lambda \left|\left|w_{1:D}\right|\right|_2^2$$

If λ is too small, will overfit to **training set**. Too large, $\hat{w}_{ridge} = 0$.

How do we choose the right value of λ? We want the one that will do best on **future data.** This means we want to minimize error on the validation set.

Don't need to minimize $RSS(w) + \lambda \left|\left|w_{1:D}\right|\right|_2^2$ on validation because you can't overfit to the validation data (you never train on it).

Another argument is that it doesn't make sense to compare those values for different settings of λ. They are in different "units" in some sense.

Choosing λ

Hyperparameter tuning

The process for selecting λ is exactly the same as we saw with using a validation set or using cross validation.

for λ in λs:

  Train a model using using Gradient Descent

$$\hat{w}_{ridge(\lambda)} = \min_{w} RSS_{train}(w) + \lambda\left|\left|w_{1:D}\right|\right|_2^2$$

  Compute validation error

$$validation\_error = RSS_{val}(\hat{w}_{ridge(\lambda)})$$

  Track λ with smallest $validation\_error$

Return λ^* & estimated future error $RSS_{test}(\hat{w}_{ridge(\lambda^*)})$

There is no fear of overfitting to validation set since you never trained on it! You can just worry about error when you aren't worried about overfitting to the data.

Pre-Lecture Video 2

Feature Selection and All Subsets

Benefits

Why do we care about selecting features? Why not use them all?

Complexity

Models with too many features are more complex. Might overfit!

Interpretability

Can help us identify which features carry more information.

Efficiency

Imagine if we had MANY features (e.g. DNA). $\widehat{w}$ could have 10^{11} coefficients. Evaluating $\hat{y} = \hat{w}^T h(x)$ would be very slow!

If $\hat{w}$ is **sparse**, only need to look at the non-zero coefficients

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

Sparsity: Housing

Might have many features to potentially use. Which are useful?

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

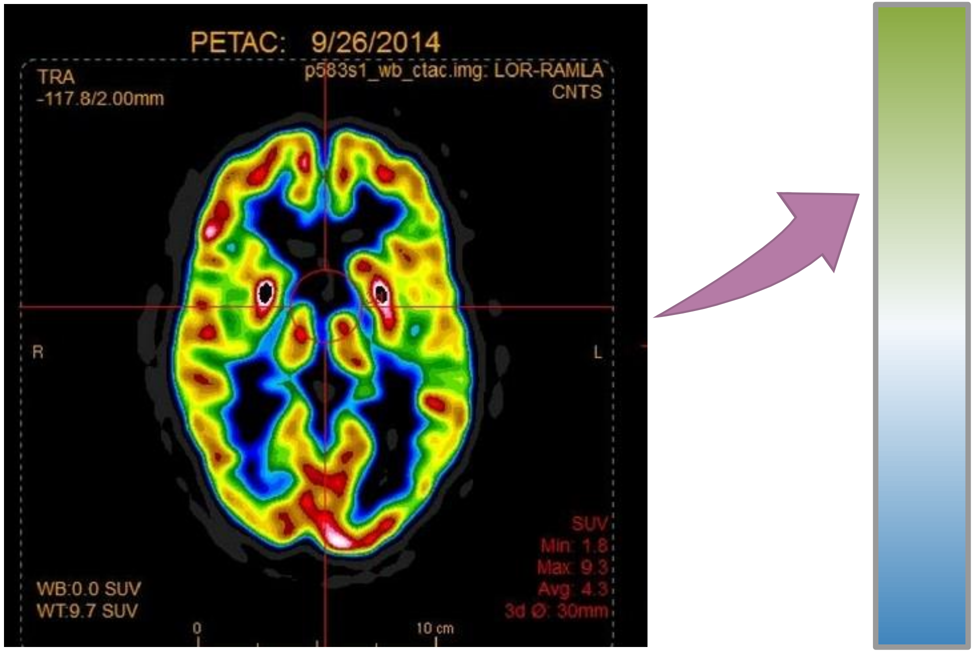
Dishwasher
Garbage disposal
Microwave
Range / Oven
Refrigerator
Washer
Dryer
Laundry location
Heating type
Jetted Tub
Deck
Fenced Yard
Lawn
Garden
Sprinkler System

…

Sparsity: Reading Minds

How happy are you? What part of the brain controls happiness?

x y

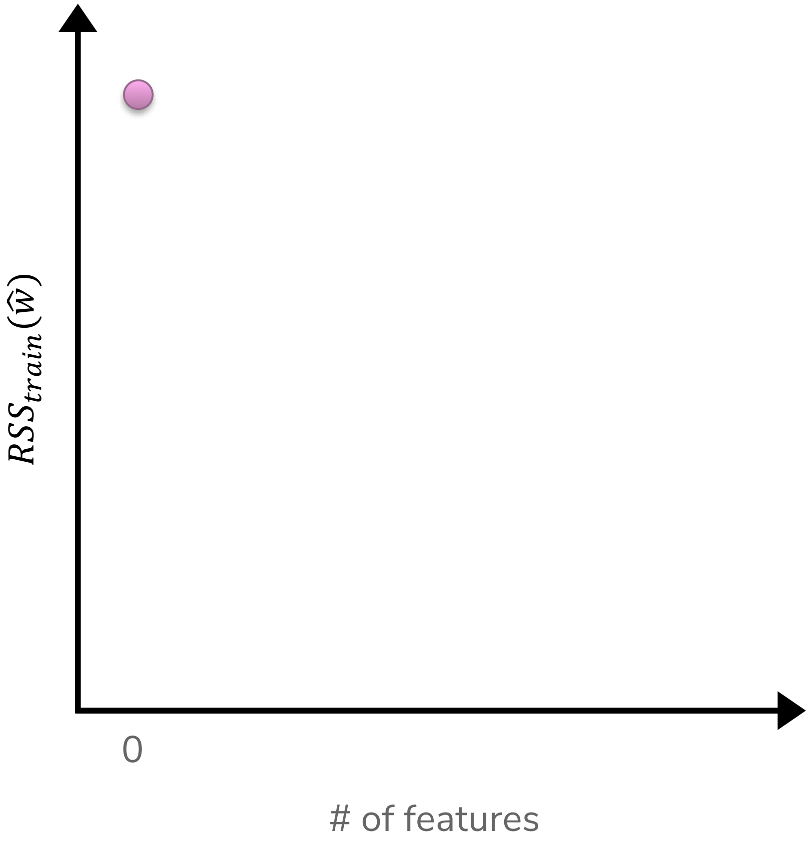


10 – Happy ☺

0 – Sad ☹

Best Model Size 0

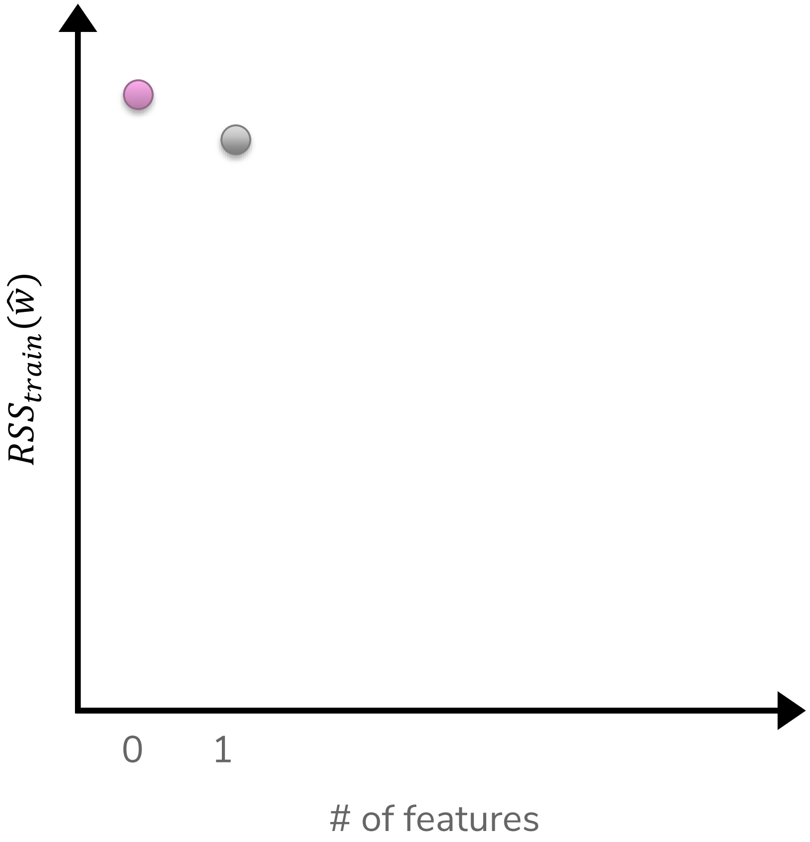
All Subsets



Noise only:
$y_i = \varepsilon_i$

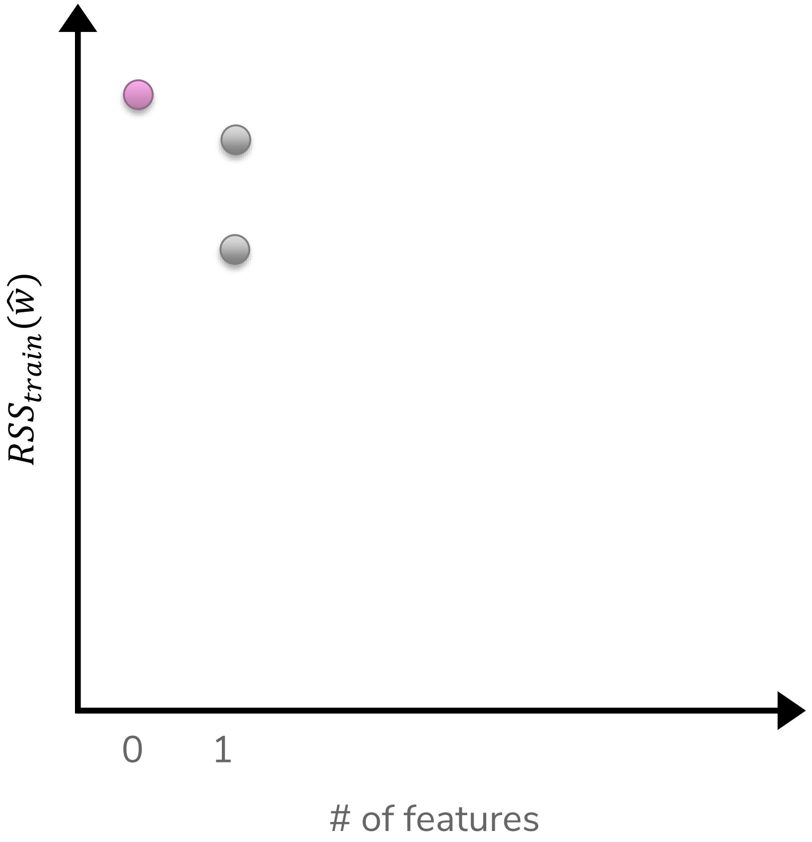
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



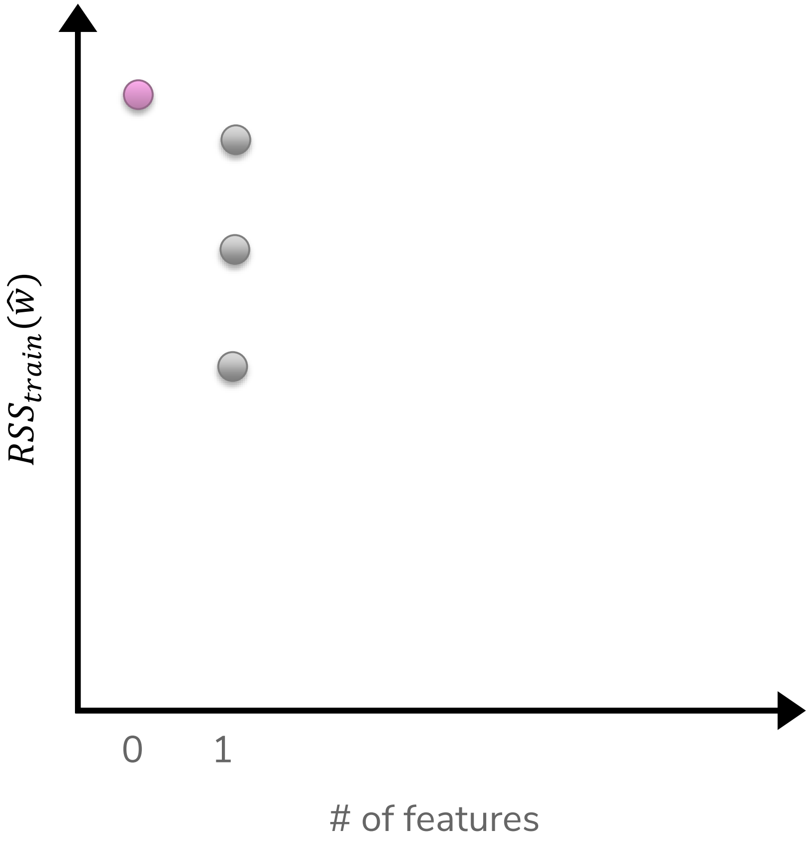
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



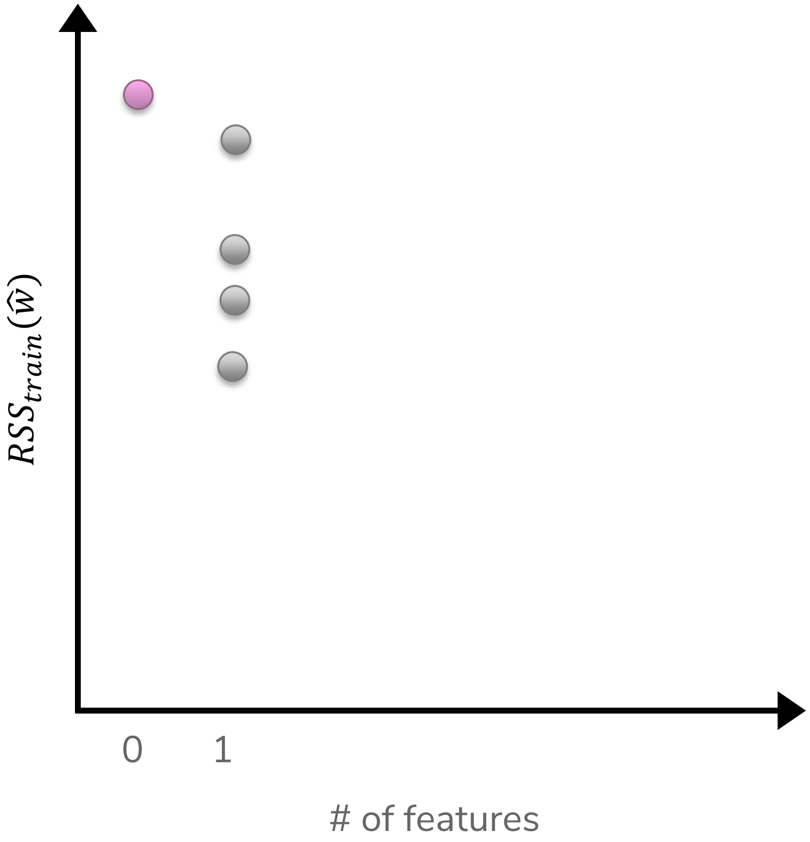
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



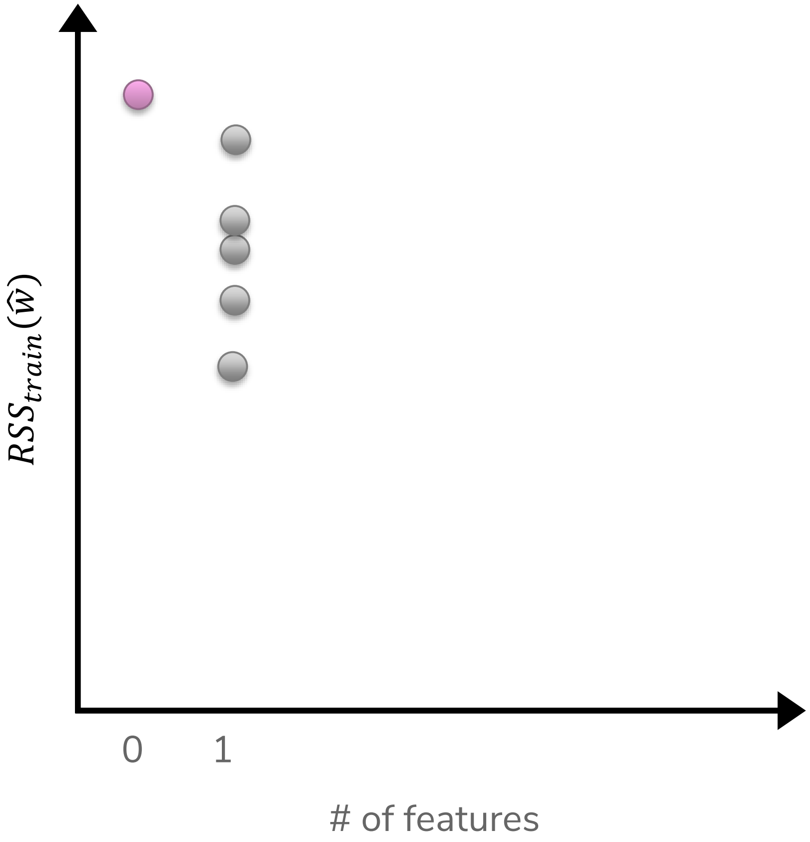
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



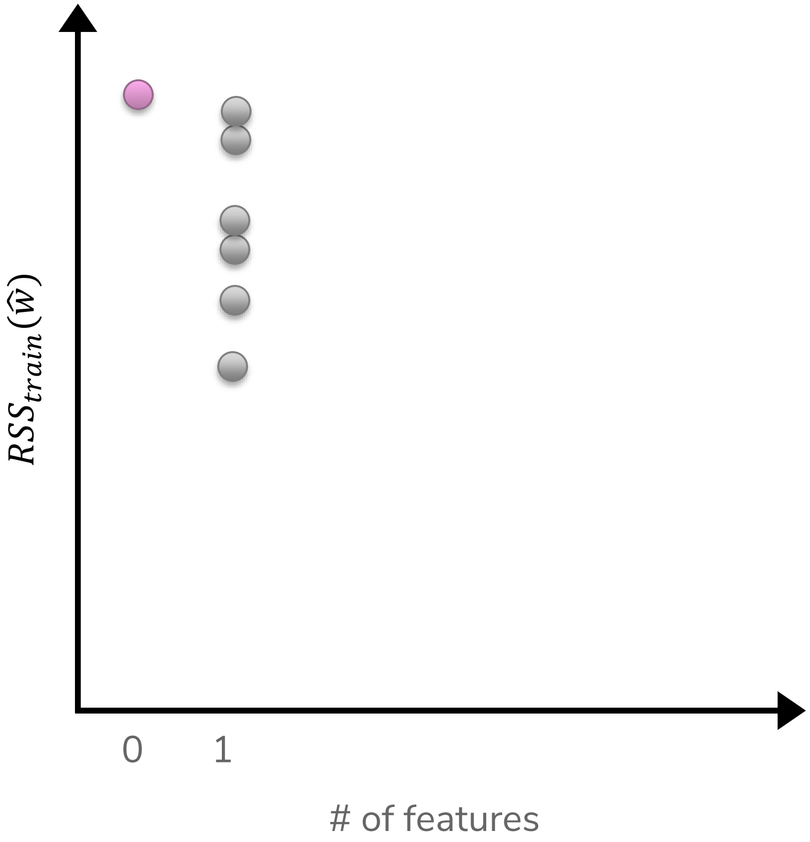
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

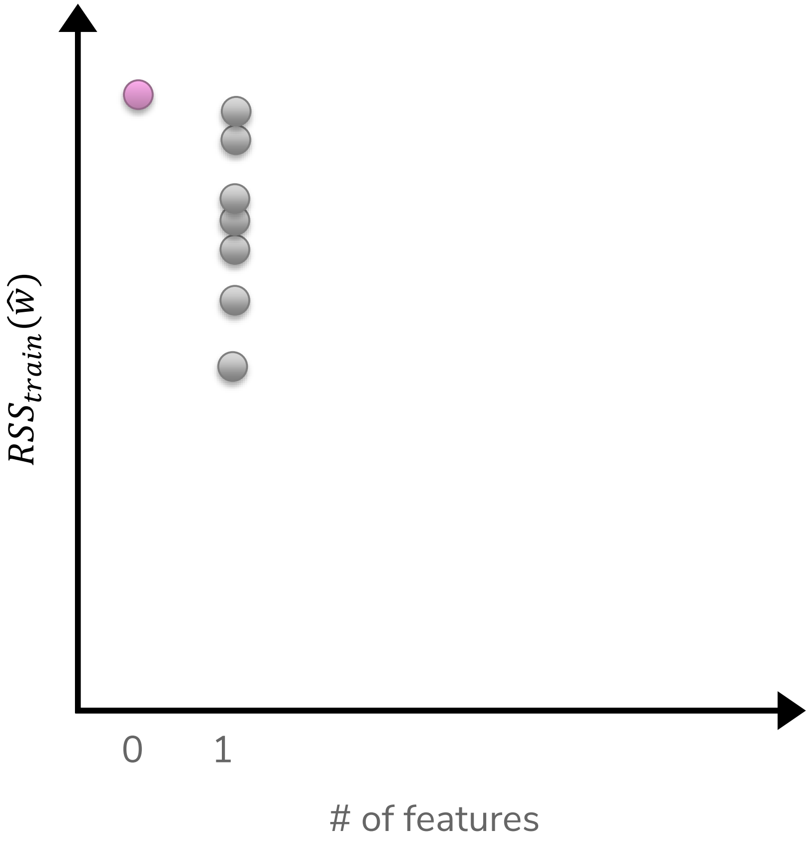
Best Model Size 1



Features

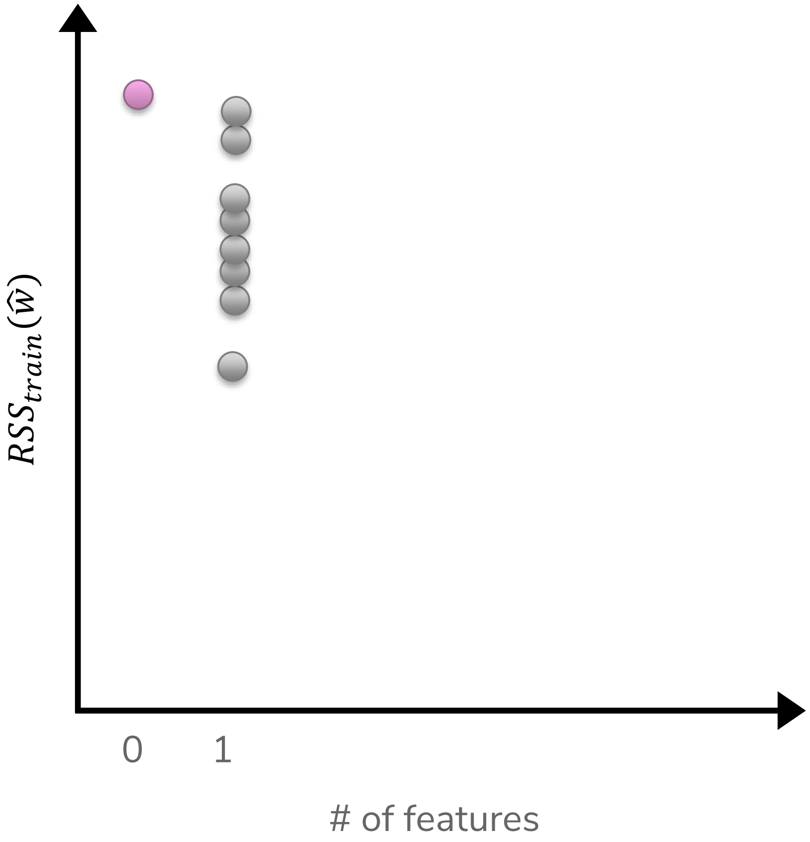
- # bathrooms
- # bedrooms
- sq.ft. living
- sq.ft lot
- floors
- year built
- year renovated
- waterfront

Best Model Size 1



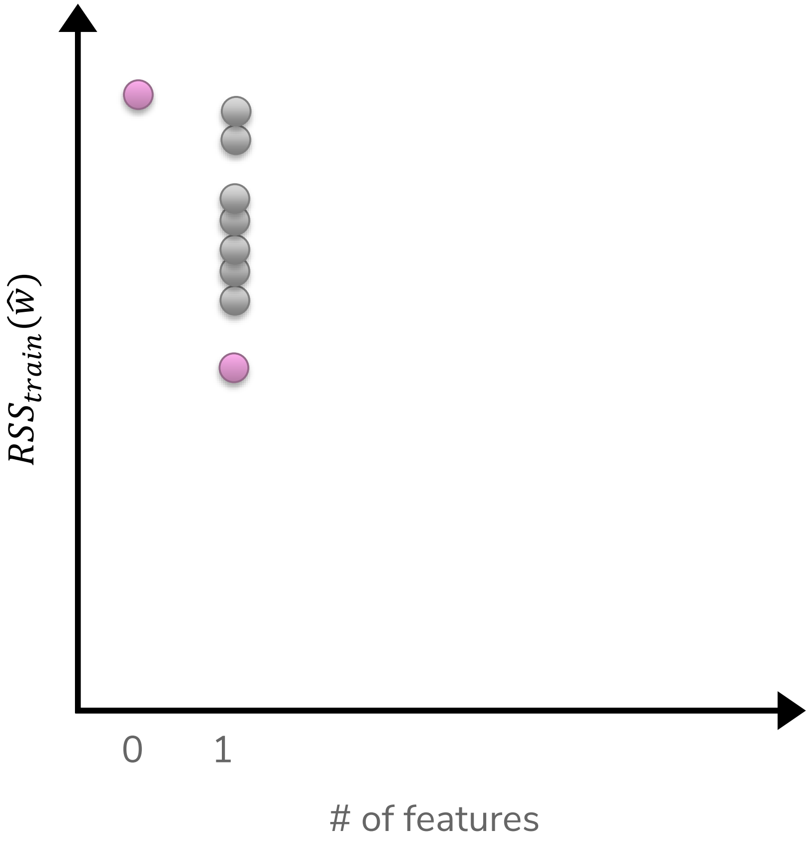
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



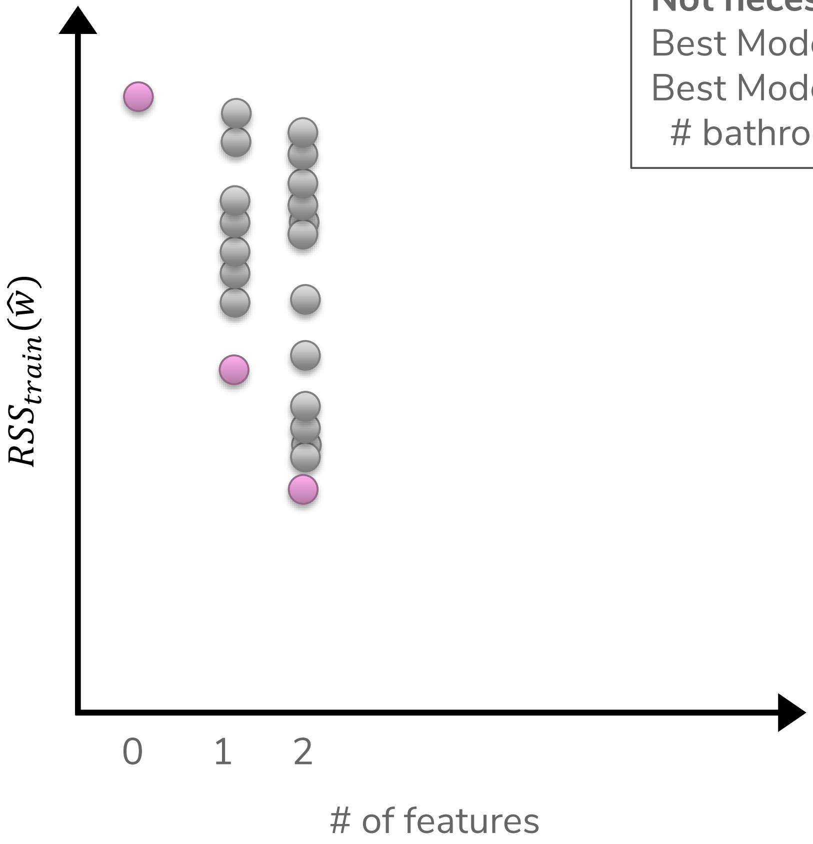
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

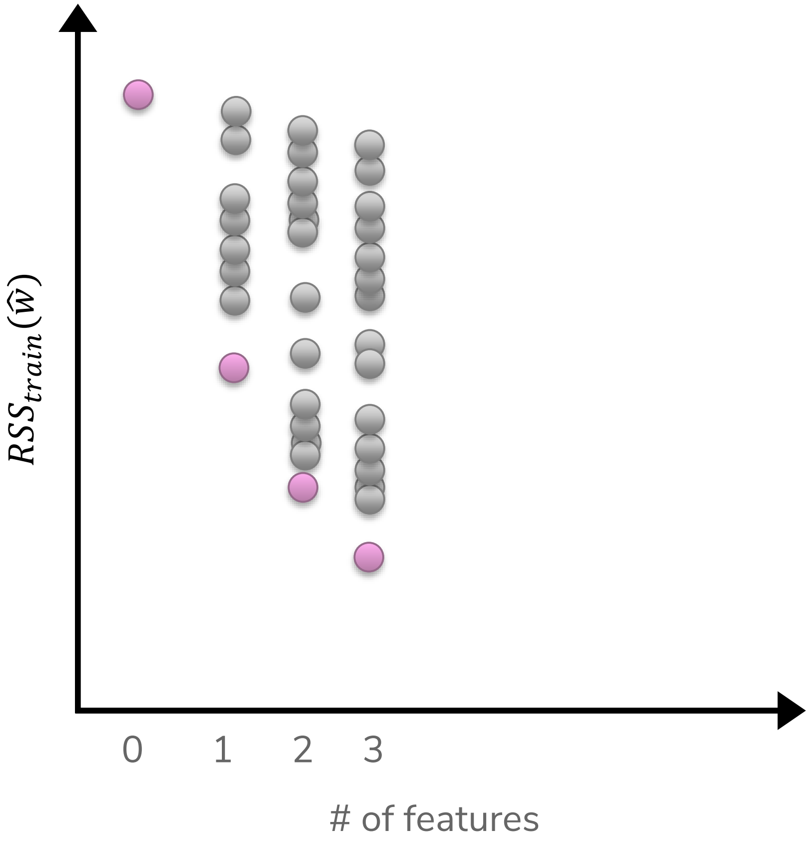
Best Model Size 2



Not necessarily nested!
Best Model – Size 1: sq.ft living
Best Model – Size 2:
bathrooms & # bedrooms

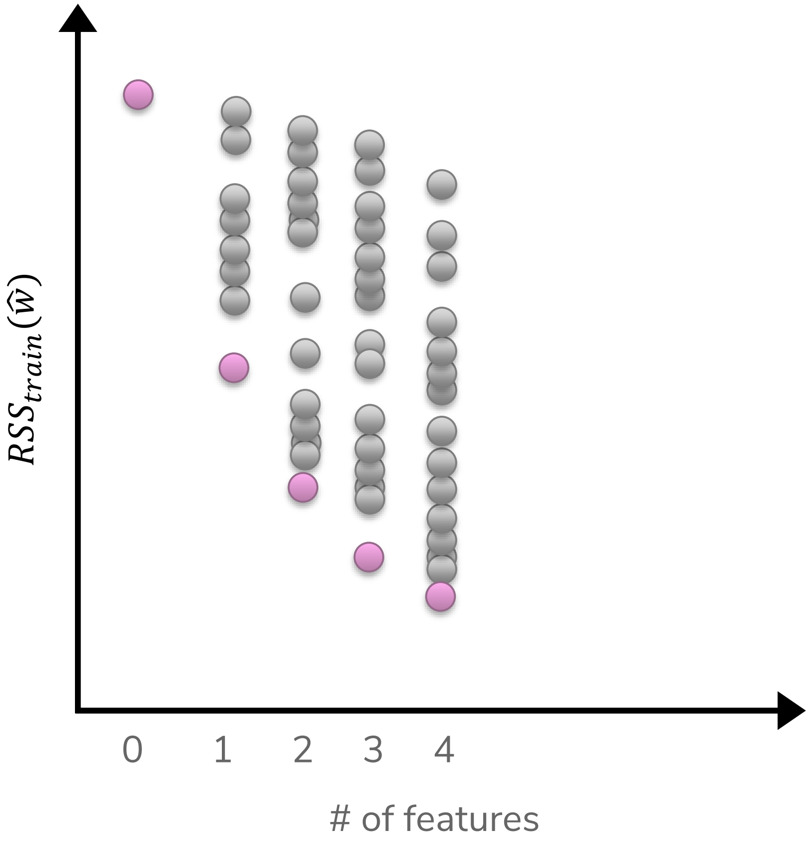
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 3



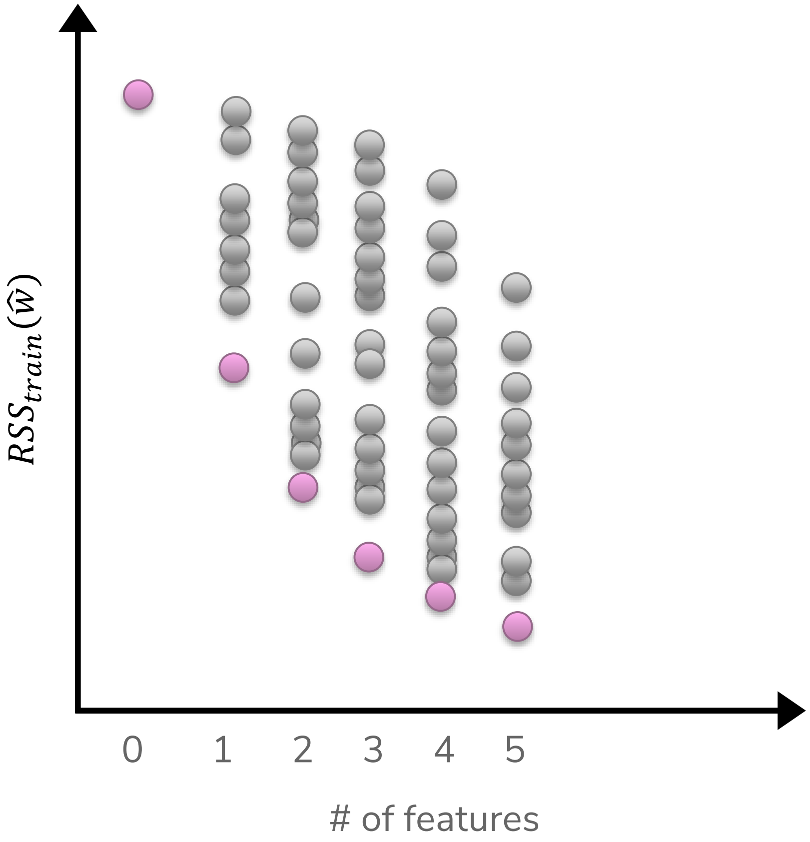
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 4



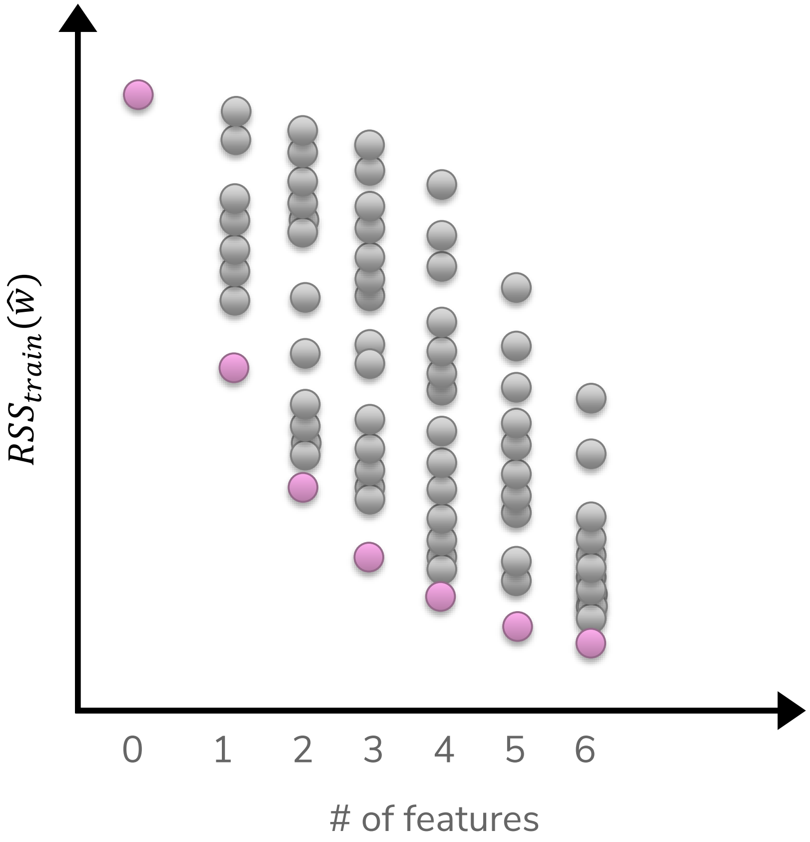
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 5



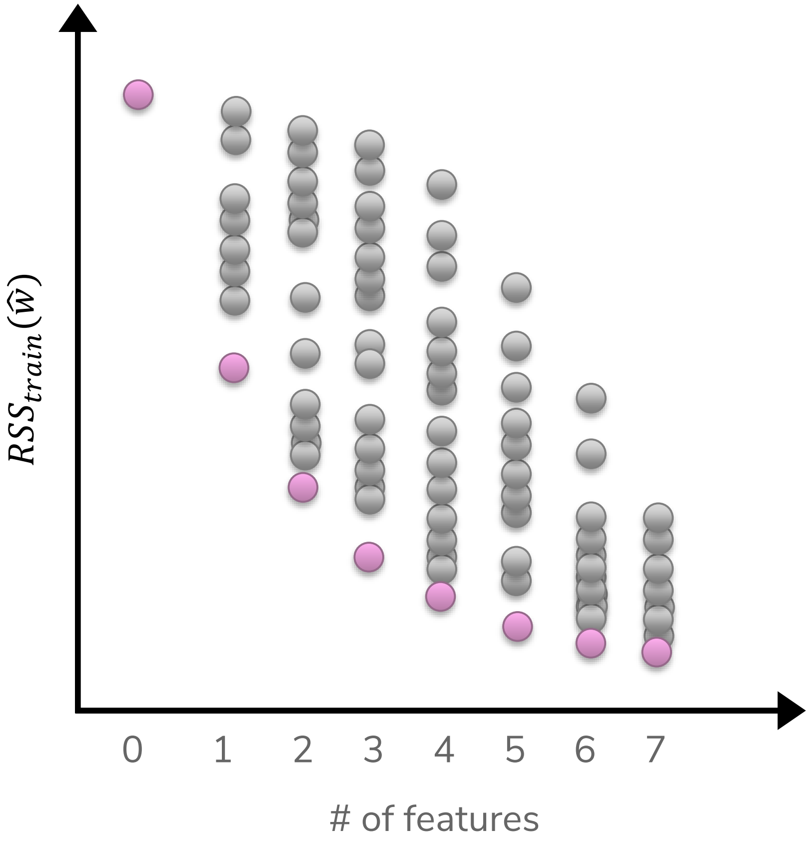
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model
Size 6



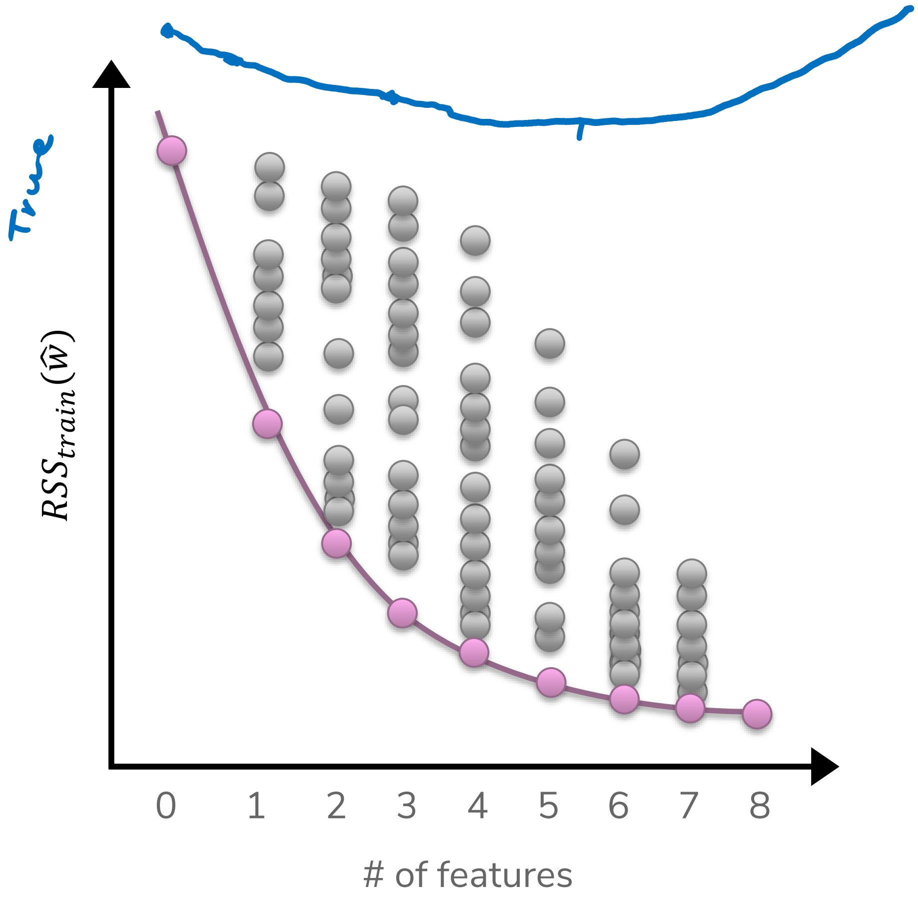
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 7



Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 8



Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Choose Num Features?

Option 1

Assess on a validation set

Option 2

Cross validation

Option 3+

Other metrics for penalizing model complexity like BIC

Class Session

Efficiency of All Subsets

How many models did we evaluate?

$\hat{y}_i = \epsilon_i$

$\hat{y}_i = w_0 h_0(x) + \epsilon_i$

$\hat{y}_i = w_1 h_1(x) + \epsilon_i$

…

$\hat{y}_i = w_0 h_0(x) + w_1 h_1(x) + \epsilon_i$

…

$\hat{y}_i = w_0 h_0(x) + w_1 h_1(x) + \ldots + w_D h_D(x)\ \epsilon_i$

F_0 F_1 F_D

[0 0 0 … 0 0 0]

[1 0 0 … 0 0 0]

[0 1 0 … 0 0 0]

…

[1 1 0 … 0 0 0]

…

$\underbrace{[1\ 1\ 1\ \ldots\ 1\ 1\ 1]}_{D+1}$

2^{D+1}

$2 \cdot 2 \cdot 2 \cdot 2 \cdot \ldots \cdot 2$

If evaluating all subsets of 8 features only took 5 seconds, then

- 16 features would take 21 minutes
- 32 features would take almost 3 years
- 100 features would take almost $7.5 * 10^{20}$ years
  - 50,000,000,000x longer than the age of the universe!

Greedy Algorithms

Knowing it's impossible to find exact solution, approximate it!

Forward stepwise

Start from model with no features, iteratively add features as performance improves.

Backward stepwise

Start with a full model and iteratively remove features that are the least useful.

Combining forward and backwards steps

Do a forward greedy algorithm that eventually prunes features that are no longer as relevant

And many many more!

Example Greedy Algorithm

Start by selecting number of features k

```
S_0 ← {}
for i ← 1..k:
    Find feature f_i not in S_{i-1}, that when combined
    with S_{i-1}, minimizes the loss the most.

    S_i ← S_{i-1} ∪ {f_i}
Return S_k
```

Called greedy because it makes choices that look best at the time.

Option 2
Regularization

Recap: Regularization

Before, we used the quality metric that minimized loss

$$\hat{w} = \min_{w} L(w)$$

Change quality metric to balance loss with measure of overfitting

- $L(w)$ is the measure of fit
- $R(w)$ measures the magnitude of coefficients

$$\hat{w} = \min_{w} L(w) + R(w)$$

How do we actually measure the magnitude of coefficients?

Recap: Magnitude

$w = [w_0, w_1, \dots, w_D]$

$R(w) =$ measure of overfitting

Come up with some number that summarizes the magnitude of the coefficients in w.

Sum?

$R(w) = \sum_{j=0}^{D} w_j$

Doesn't work

$w = [1000000, -1000000]$

$R(w) = 0$

Sum of absolute values?

$R(w) = \sum_{j=0}^{D} |w_j| \triangleq ||w||_1$

L1 norm

(come back Wed!)

Sum of squares?

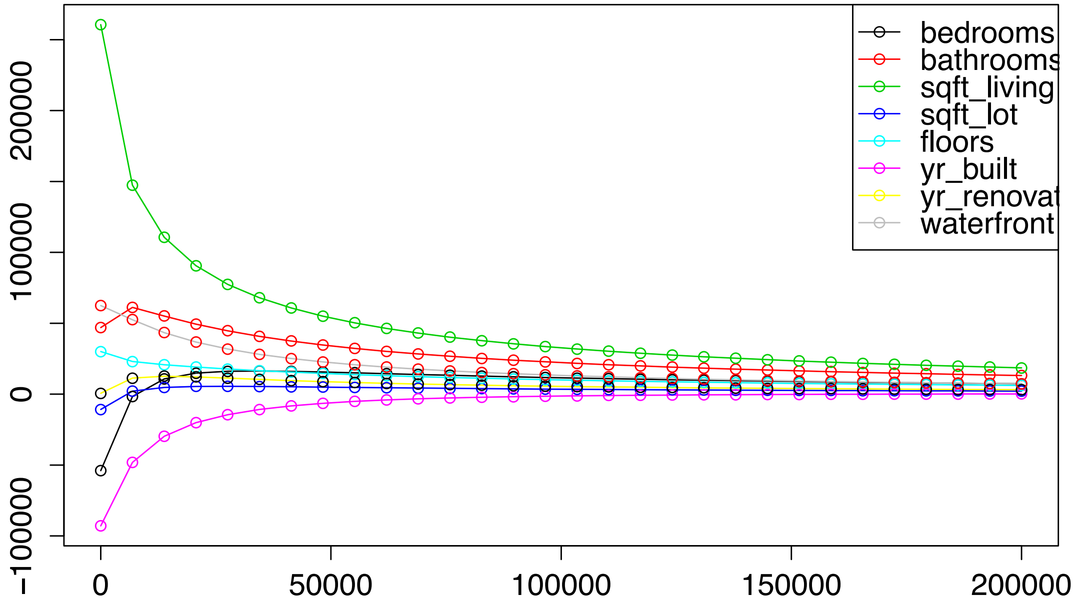
$R(w) = \sum_{j=0}^{D} w_j^2 \triangleq ||w||_2^2$

L2 norm

(today)

Ridge for Feature Selection

We saw that Ridge Regression shrinks coefficients, but they don't become 0. What if we remove weights that are sufficiently small?

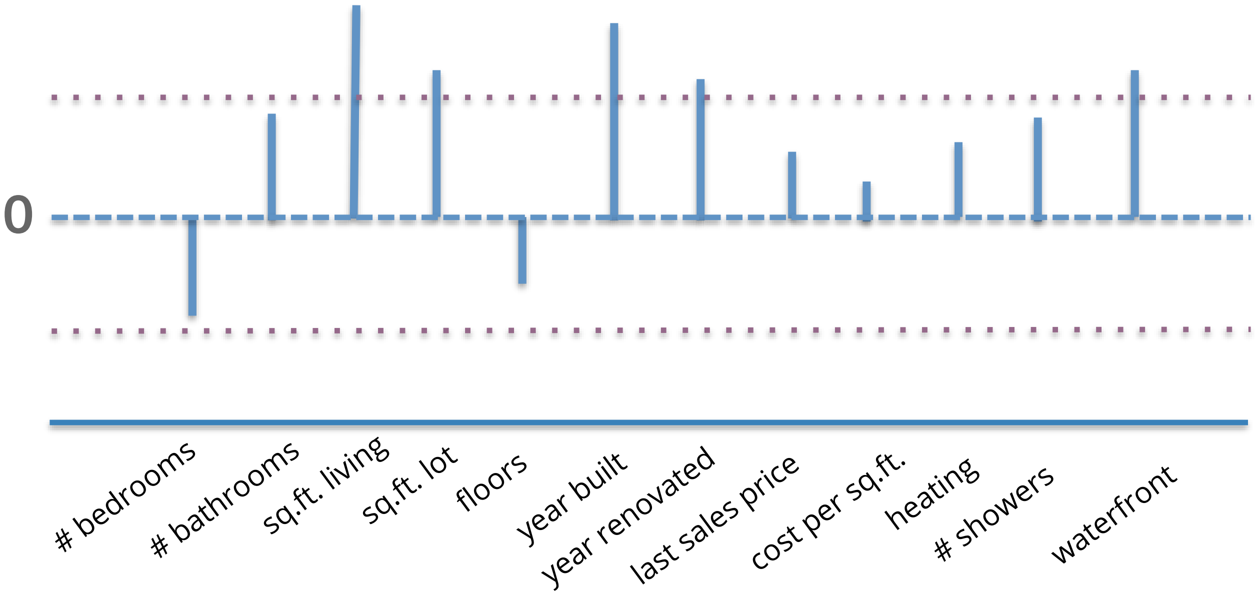


Ridge for Feature Selection

Instead of searching over a **discrete** set of solutions, use regularization to reduce coefficient of unhelpful features.

Start with a full model, and then “shrink” ridge coefficients near 0.

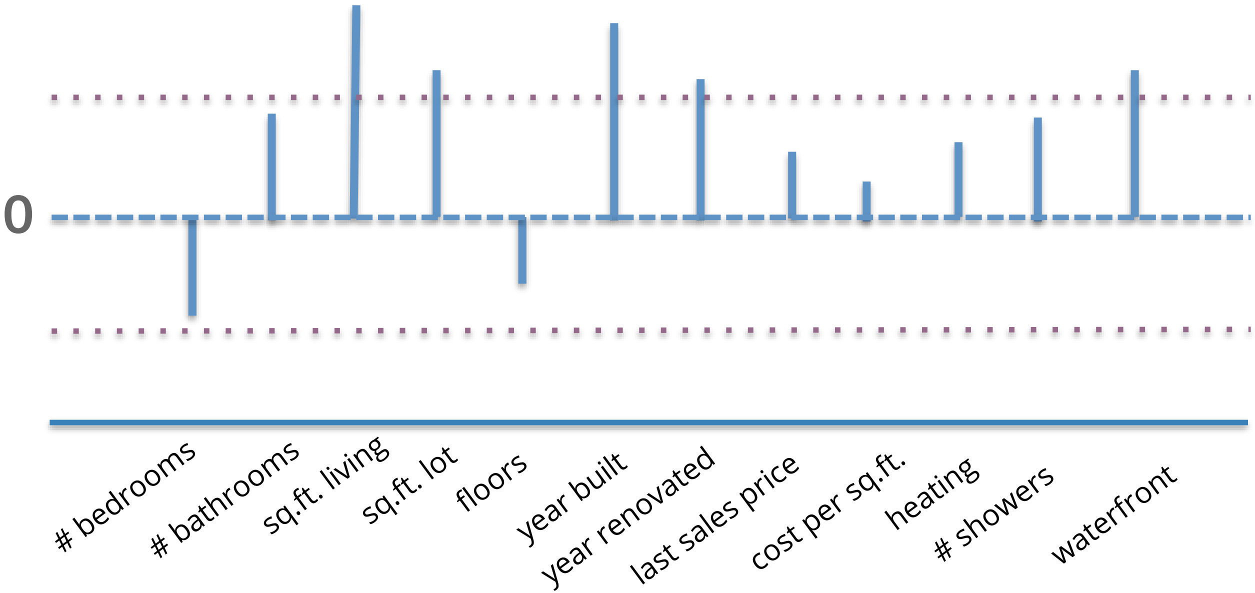
Non-zero coefficients would be considered selected as important.



Ridge for Feature Selection

Look at two related features #bathrooms and # showers.

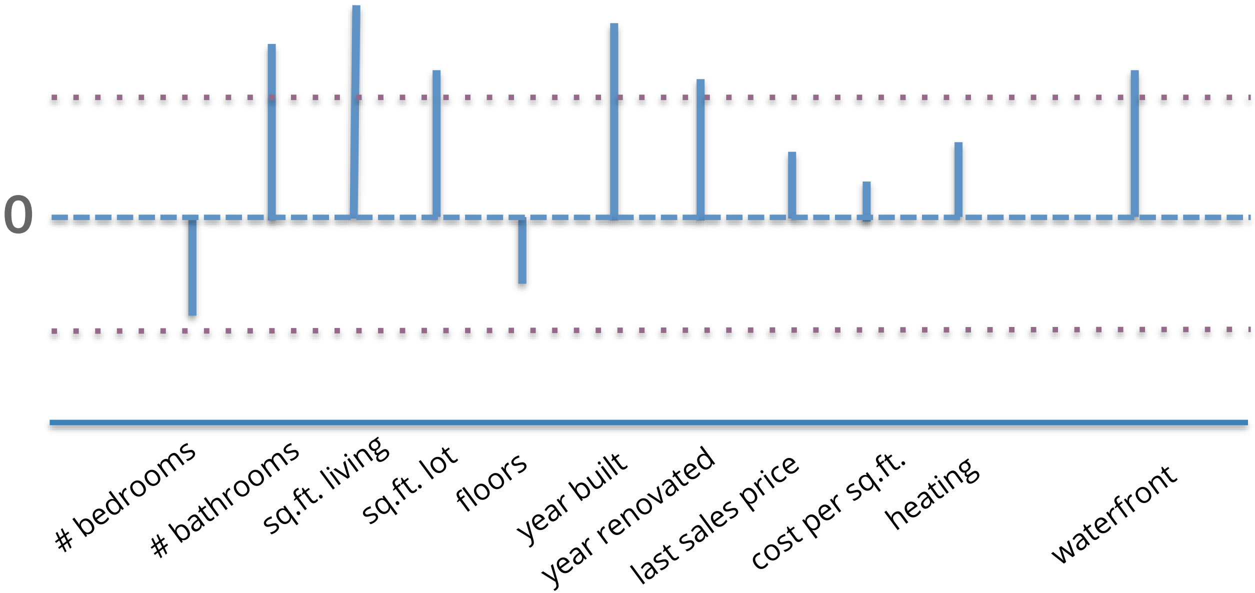
Our model ended up not choosing any features about bathrooms!



Ridge for Feature Selection

What if we had originally removed the # showers feature?

- The coefficient for # bathrooms would be larger since it wasn't "split up" amongst two correlated features
- Instead, it would be nice if there were a regularizer that favors sparse solutions in the first place to account for this…



Brain Break

Come back 3:15

LASSO Regression

L1 norm: $||w||_1 = \sum_{j=0}^{D} |w_j|$

Change quality metric to minimize

$$\hat{w} = \min_w RSS(W) + \lambda \underline{||w||_1}$$

λ is a tuning parameter that changes how much the model cares about the regularization term.

What if $\lambda = 0$?

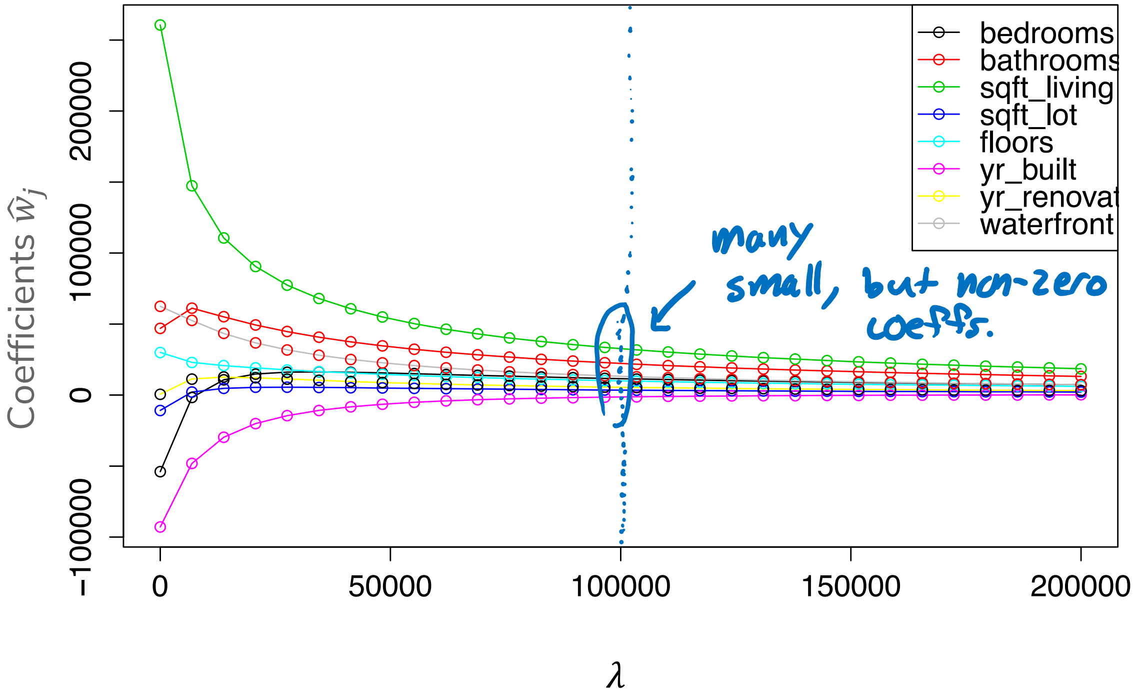
$$\hat{w} = \min_w RSS(w) \Longrightarrow \hat{w}_{LS}$$

What if $\lambda = \infty$?

$$\hat{w} = \vec{0}$$

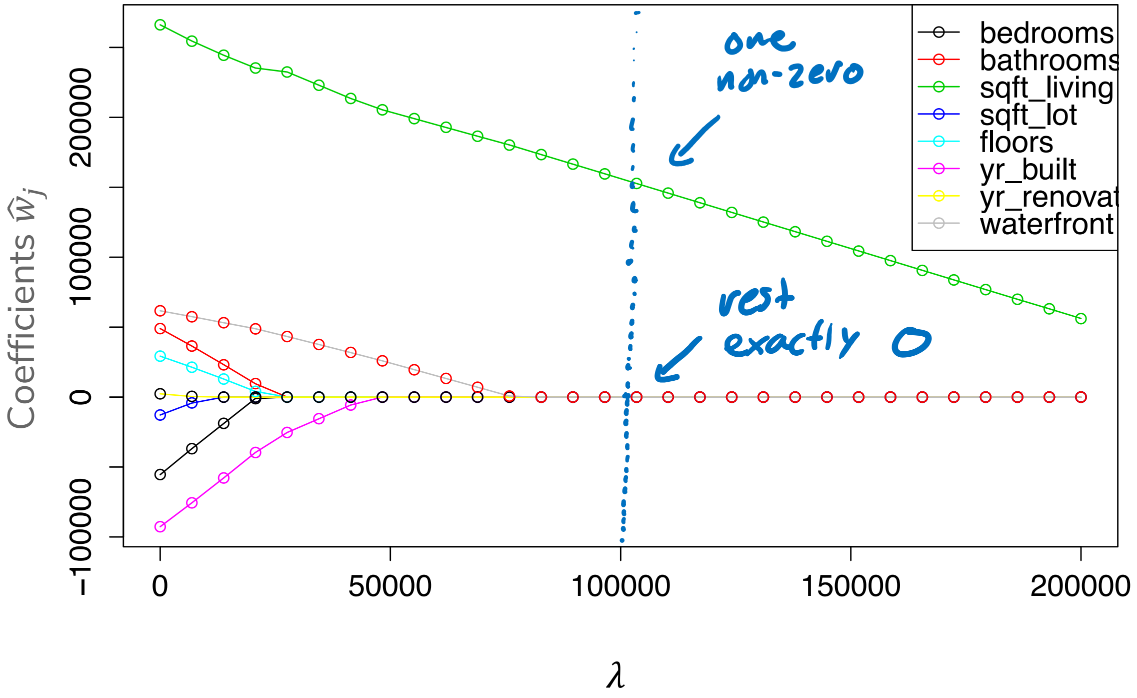
λ in between? $0 \leq ||\hat{w}_{LASSO}||_1 \leq ||\hat{w}_{LS}||_1$

Ridge Coefficient Paths



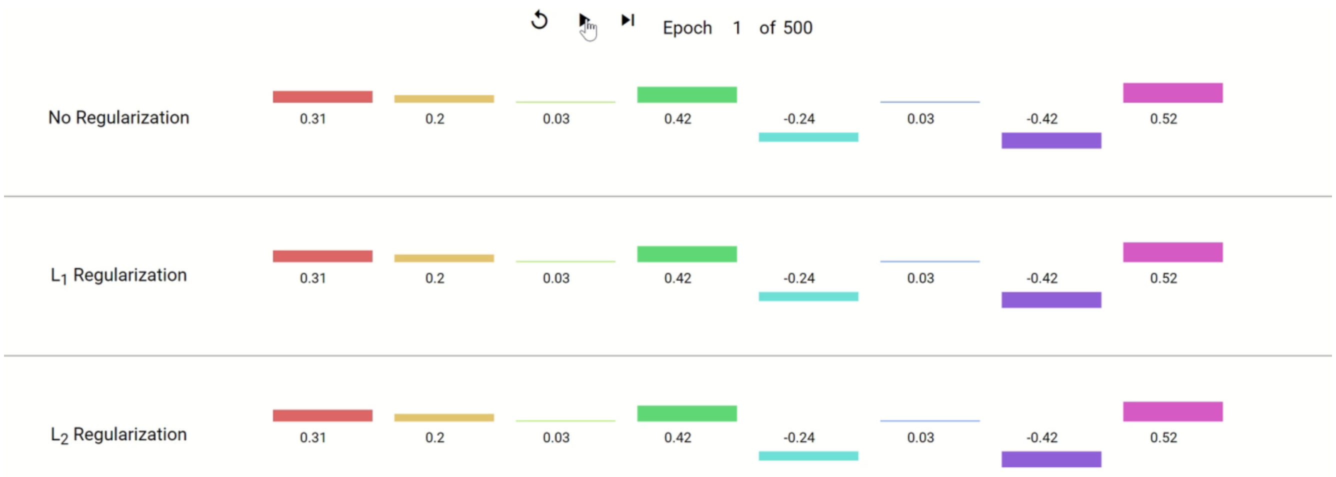
LASSO Coefficient Paths

sparse = many 0 coefs.



Coefficient Paths – Another View

Example from Google's [Machine Learning Crash Course]



Demo

See EdStem

Similar demo to last time's with Ridge but using the LASSO penalty

Poll Everywhere

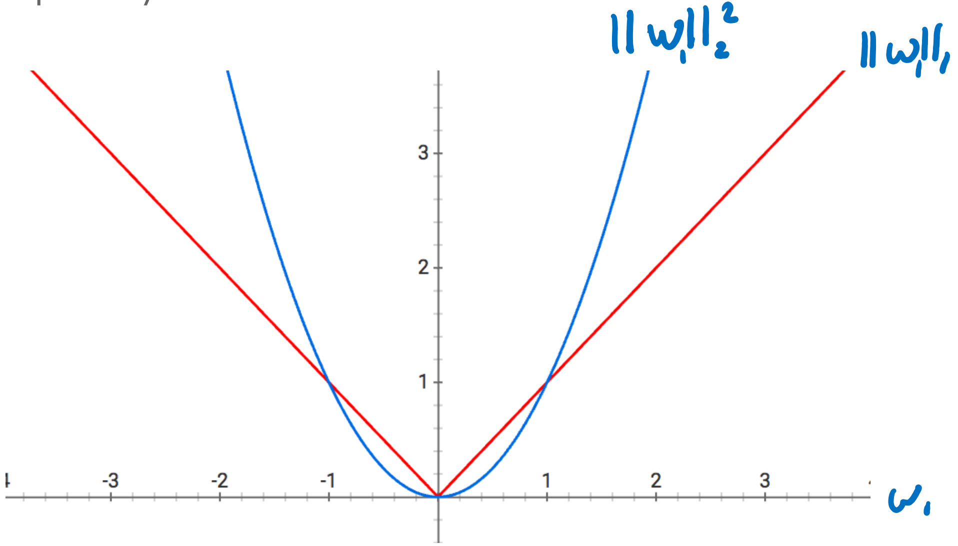
Group

2 minutes

~~pollev.com/cs416~~

There is no poll to answer for this question. This is an open-ended question.

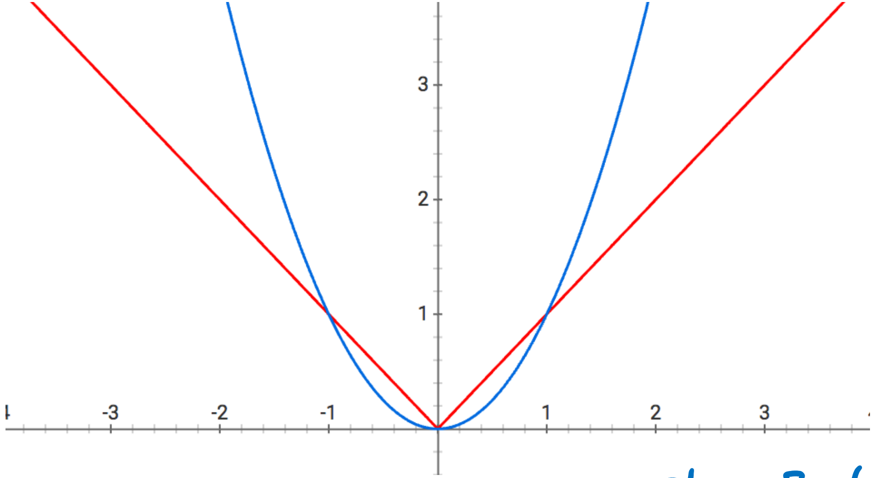
Why might the shape of the L1 penalty cause more sparsity than the L2 penalty?



Sparsity

When using the L1 Norm ($||w||_1$) as a regularizer, it favors solutions that are **sparse**. Sparsity for regression means many of the learned coefficients are 0.

This has to do with the shape of the norm

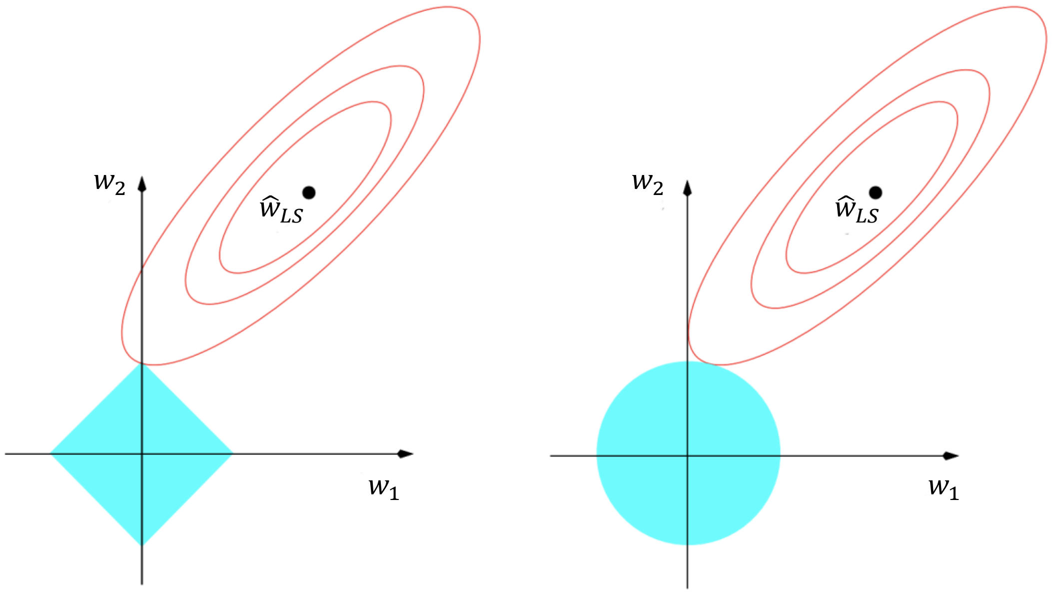


When w_j is small, w_j^2 is VERY small!

or slope for L2 decreases closer to 0!

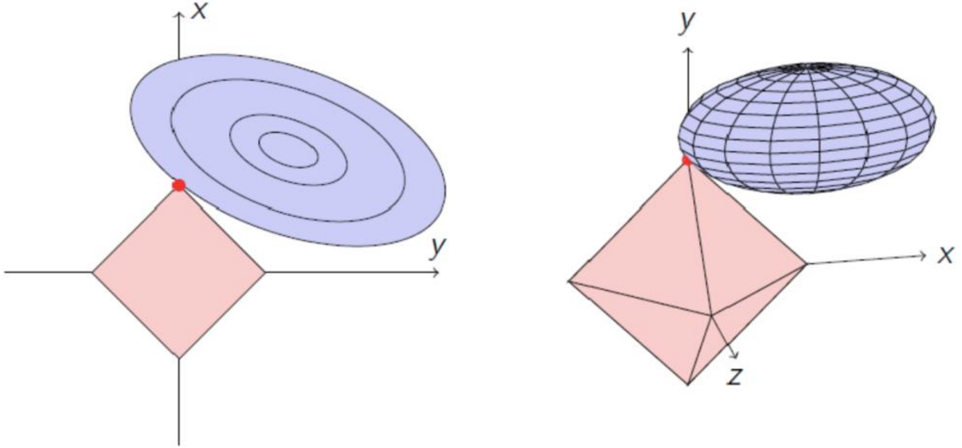
Sparsity Geometry

Another way to visualize why LASSO prefers sparse solutions



The L1 ball has corners (places where some coefficients are 0)

Sparsity Geometry



Brain Break

Poll Everywhere

Think

1 min

pollev.com/cs416

How should we choose the best value of λ for LASSO?

- Pick the λ that has the smallest $RSS(\hat{w})$ on the **validation set**
- Pick the λ that has the smallest $RSS(\hat{w}) + \lambda||\hat{w}||_2^2$ on the **validation set**
- Pick the λ that results in the most zero coefficients
- Pick the λ that results in the fewest zero coefficients
- None of the above

Choosing λ

Exactly the same as Ridge Regression :)

This will be true for almost every **hyper-parameter** we talk about

A **hyper-parameter** is a parameter you specify for the model that influences which parameters (e.g. coefficients) are learned by the ML aglorithm

Hyper parameter tuning:

for each setting of HPs:

train model with current hp setting

validate predictor w/ validation set / cross-val

track hp w/ lowest val error

return best hp and estimate of test error.

LASSO in Practice

A very common usage of LASSO is in feature selection. If you have a model with potentially many features you want to explore, you can use LASSO on a model with all the features and choose the appropriate λ to get the right complexity.

Then once you find the non-zero coefficients, you can identify which features are the most important to the task at hand*

De-biasing LASSO

LASSO adds bias to the Least Squares solution (this was intended to avoid the variance that leads to overfitting)

- Recall Bias-Variance Tradeoff

It's possible to try to remove the bias from the LASSO solution using the following steps

1. Run LASSO to select the which features should be used (those with non-zero coefficients)
2. Run regular Ordinary Least Squares on the dataset with only those features

Coefficients are no longer shrunk from their true values

Issues with LASSO

1. Within a group of highly correlated features (e.g. # bathroom and # showers), LASSO tends to select amongst them arbitrarily.
   - Maybe it would be better to select them all together?
2. Often, empirically Ridge tends to have better predictive performance

Elastic Net aims to address these issues

$$\hat{w}_{ElasticNet} = \min_{w} RSS(w) + \lambda_1 \left|\left|w\right|\right|_1 + \lambda_2 \left|\left|w\right|\right|_2^2$$

Combines both to achieve best of both worlds!

Poll Everywhere

Think

1 min

pollev.com/cs416

Suppose you wanted to try out the following models:

- LASSO with hyperparameter choices $\lambda \in [0.01, 1, 10]$
- Ridge with hyperparameter choices $\lambda \in [0.05, 5, 50]$

Of the 6 models you will try, how do you pick the best predictor learned?

- Pick the predictor that has the smallest $RSS(\hat{w})$ on the **validation set**

  same as hyperparameter tuning!
- Pick the predictor that has the smallest $RSS(\hat{w}) + \lambda||\hat{w}||_2^2$ on the **validation set**
- Pick the predictor that has the smallest $RSS(\hat{w}) + \lambda||\hat{w}||_1$ on the **validation set**
- Pick the λ that results in the most zero coefficients
- Pick the λ that results in the fewest zero coefficients
- None of the above

A Big Grain of Salt

Be careful when interpreting results of feature selection or feature importances in Machine Learning!

- Selection only considers features included
- Sensitive to correlations between features
- Results depend on the algorithm used!

Recap

Theme: Use regularization to do feature selection

Ideas:

- Describe "all subsets" approach to feature selection and why it's impractical to implement.
- Formulate LASSO objective
- Describe how LASSO coefficients change as hyper-parameter λ is varied
- Interpret LASSO coefficient path plot
- Compare and contrast LASSO and ridge