# CSE/STAT 416

## Regularization – LASSO Regression

Pemi Nguyen
University of Washington
April 6, 2022

Slides by Hunter Schafer

# Administrivia

**Homework 1** due this Friday. We have late days but use them carefully.

Office Hours info posted on the course website

Please submit anonymous feedback so that we can know how we can improve the course quality

Sorry for the logistical issues on Monday, hopefully it'll get better ☺. I'll post a re-recording later this weekend.

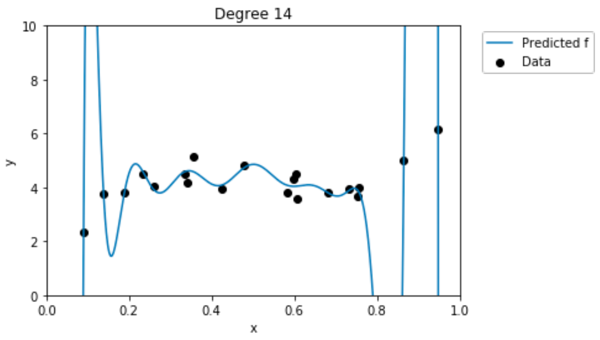
# Pre-Lecture Video 1

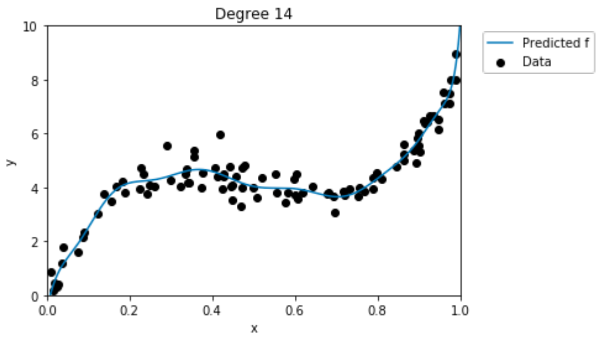
*Recap Ridge*

# 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.



# Recap: Ridge Regression

Change quality metric to minimize

$$\hat{w} = \underset{w}{\operatorname{argmin}}\, MSE(W) + \lambda \|w\|_2^2$$

$\lambda$ is tuning parameter that changes how much the model cares about the regularization term.

**What if $\lambda = 0$?**

**What if $\lambda = \infty$?**

**$\lambda$ in between?**

Poll Everywhere

Think

2 min

pollev.com/cs416

**How should we choose the best value of $\lambda$?**

After we train each model with a certain $\lambda_i$ in order to find

$\hat{w}_i = \text{argmin}_w \, MSE(w) + \lambda_i ||w||_2^2$:

a) Pick the $\lambda_i$ that has the smallest $MSE(\hat{w}_i)$ on the **train set**

b) Pick the $\lambda_i$ that has the smallest $MSE(\hat{w}_i)$ on the **validation set**

c) Pick the $\lambda_i$ that has the smallest $MSE(\hat{w}_i) + \lambda_i ||\hat{w}_i||_2^2$ on the **train set**

d) Pick the $\lambda_i$ that has the smallest $MSE(\hat{w}_i) + \lambda_i ||\hat{w}_i||_2^2$ on the **validation set**

# Choosing $\lambda$

For any particular setting of $\lambda$, use Ridge Regression objective

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

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

How do we choose the right value of $\lambda$? 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 $MSE(w) + \lambda \left|\left|w_{1:D}\right|\right|_2^2$ 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 $\lambda$. They are in different "units" in some sense.

# Choosing $\lambda$

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

for $\lambda$ in $\lambda$s:

  Train a model using using Gradient Descent

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

  Compute validation error

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

  Track $\lambda$ with smallest $validation\_error$

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

Overfitting concern to validation set is less you never directly trained on it!

# 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). $\hat{w}$ 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

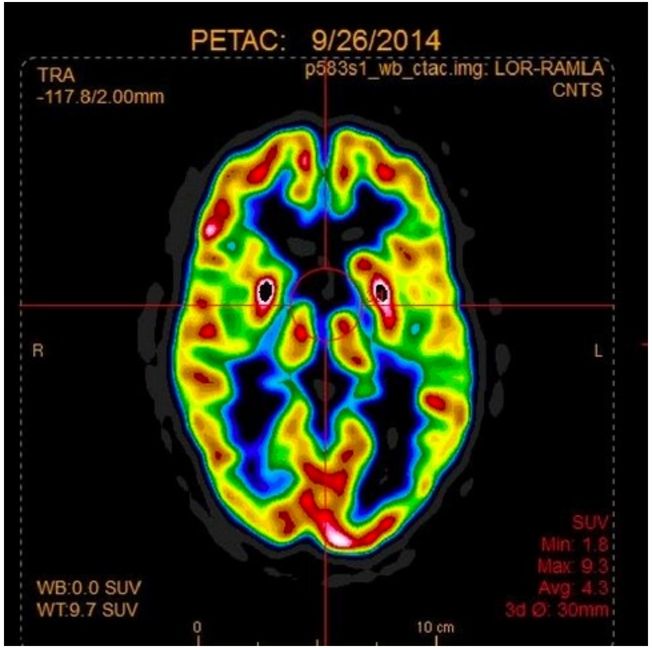
Dishwasher
Garbage disposal
Microwave
Range / Oven
Refrigerator
Washer
Dryer
Laundry location
Heating type
Jetted Tub
Deck
Fenced Yard
Lawn
Garden
Sprinkler System

…

# Sparsity: Reading Minds

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

$x$

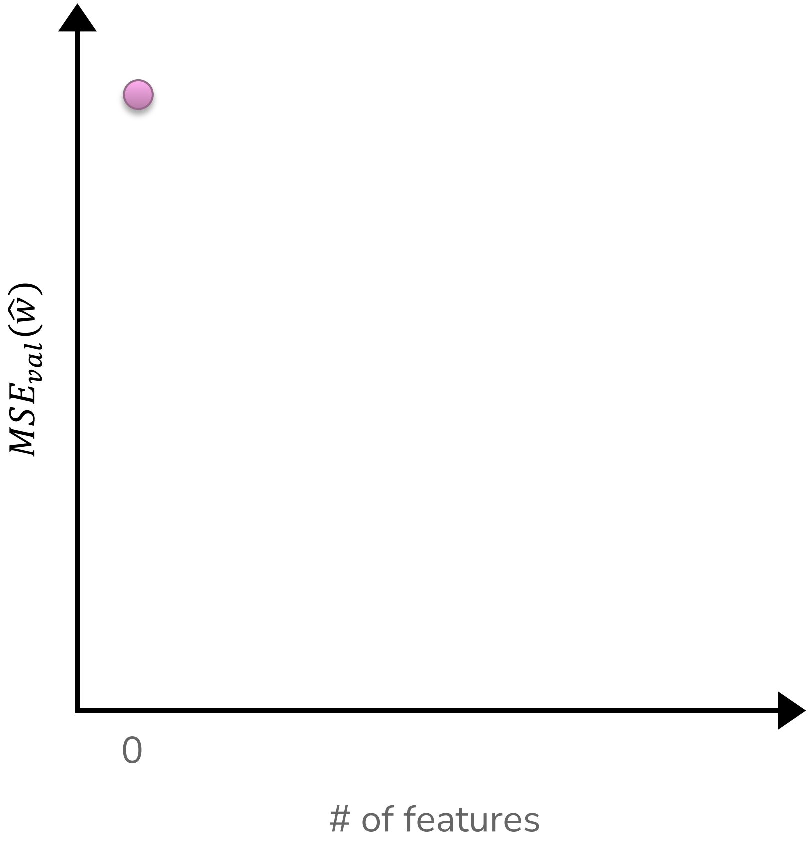



$y$

10 – Happy ☺

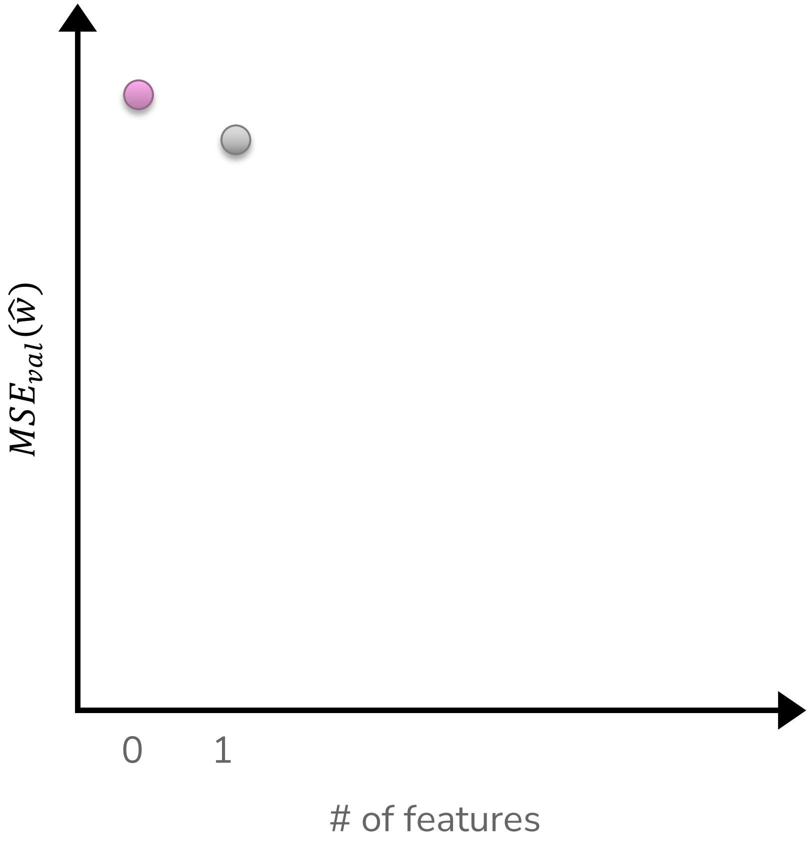
0 – Sad ☹

# Best Model Size 0



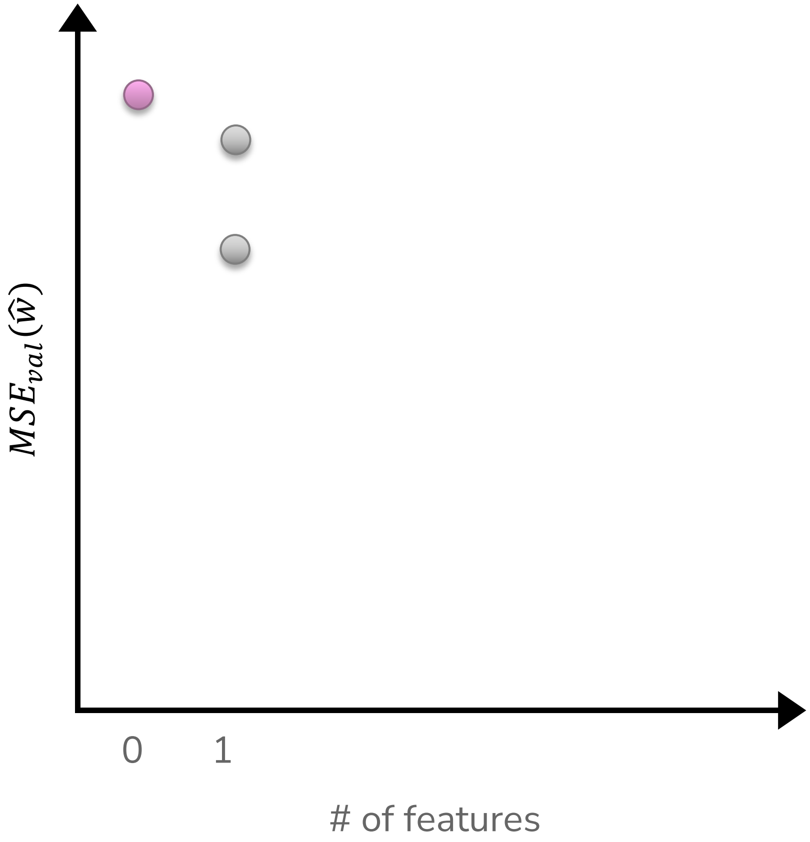

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1



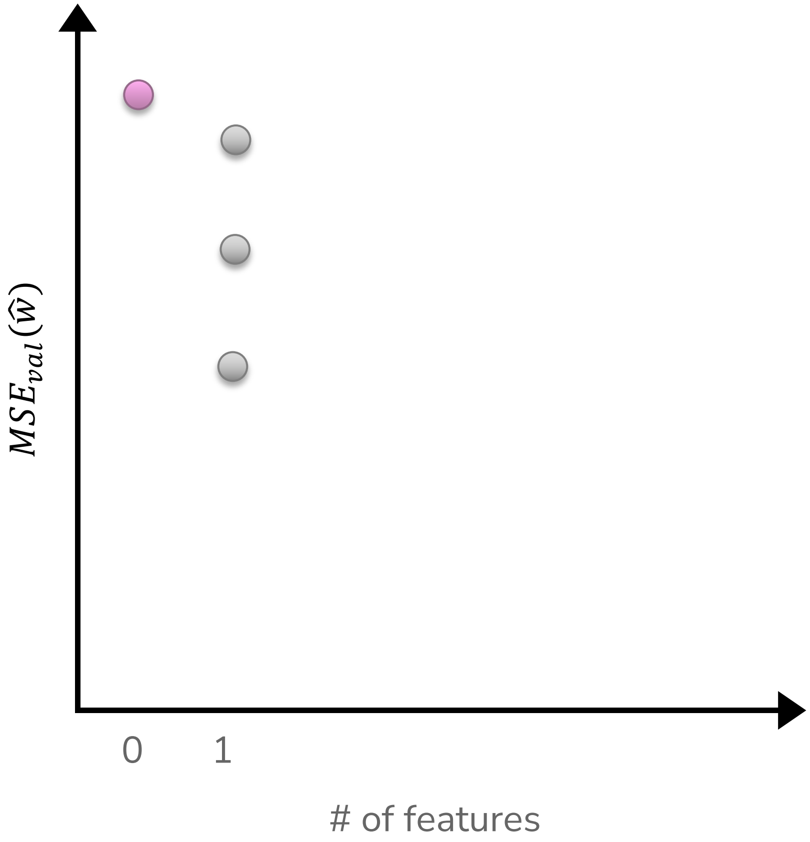

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1



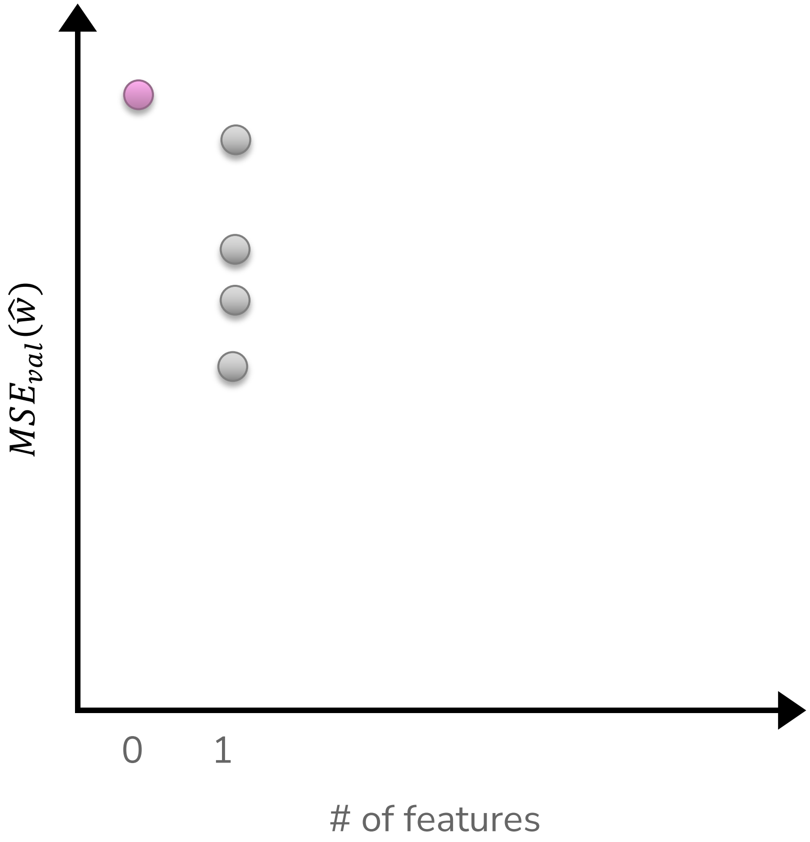

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1



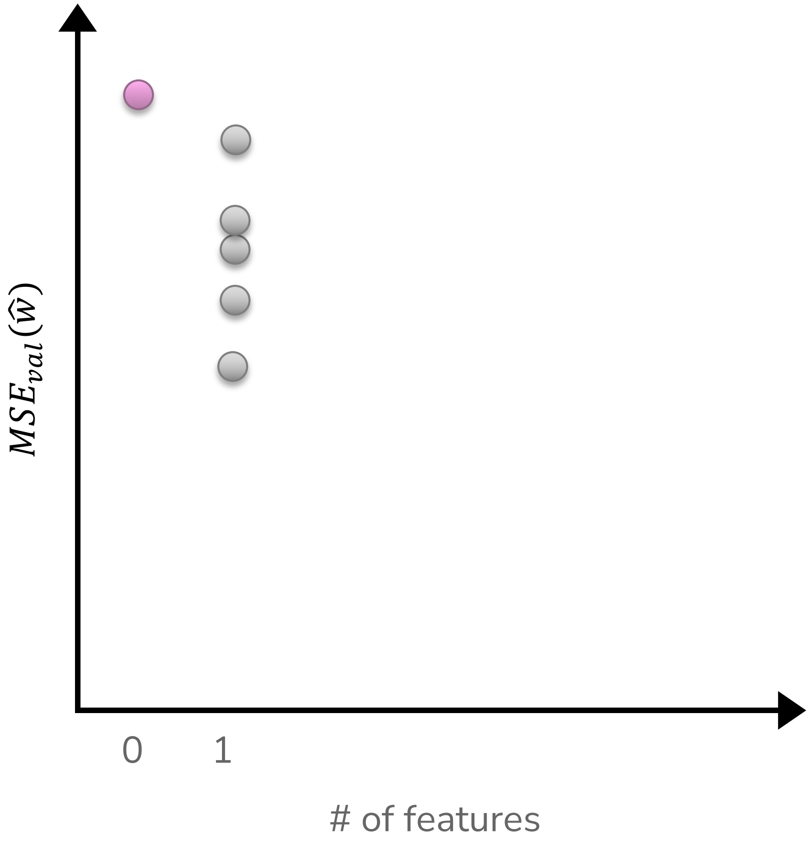

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1



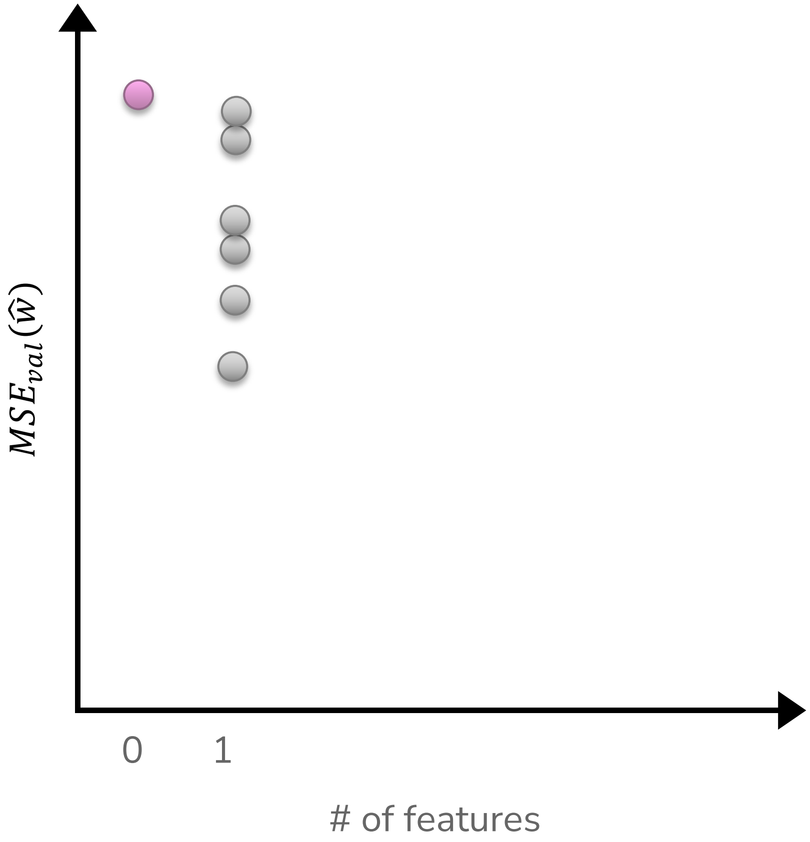

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1



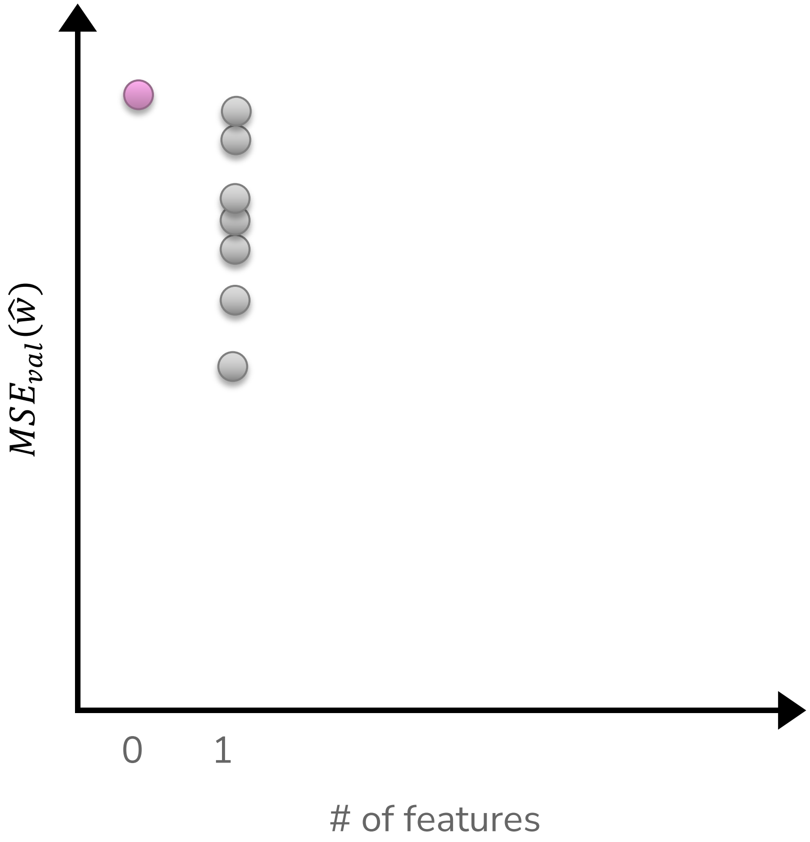

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1



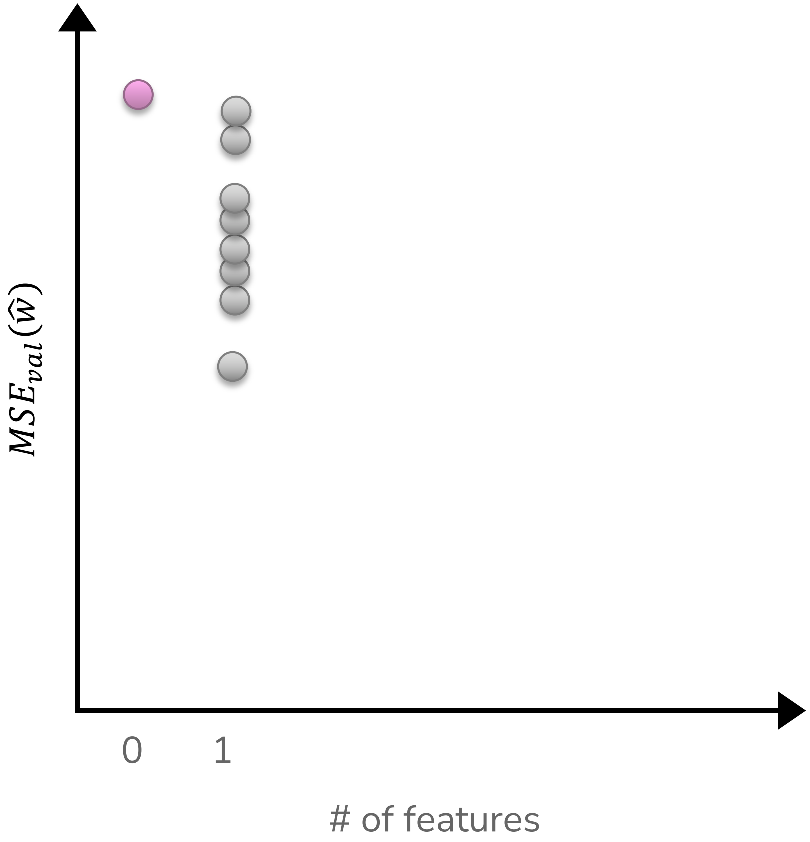

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1



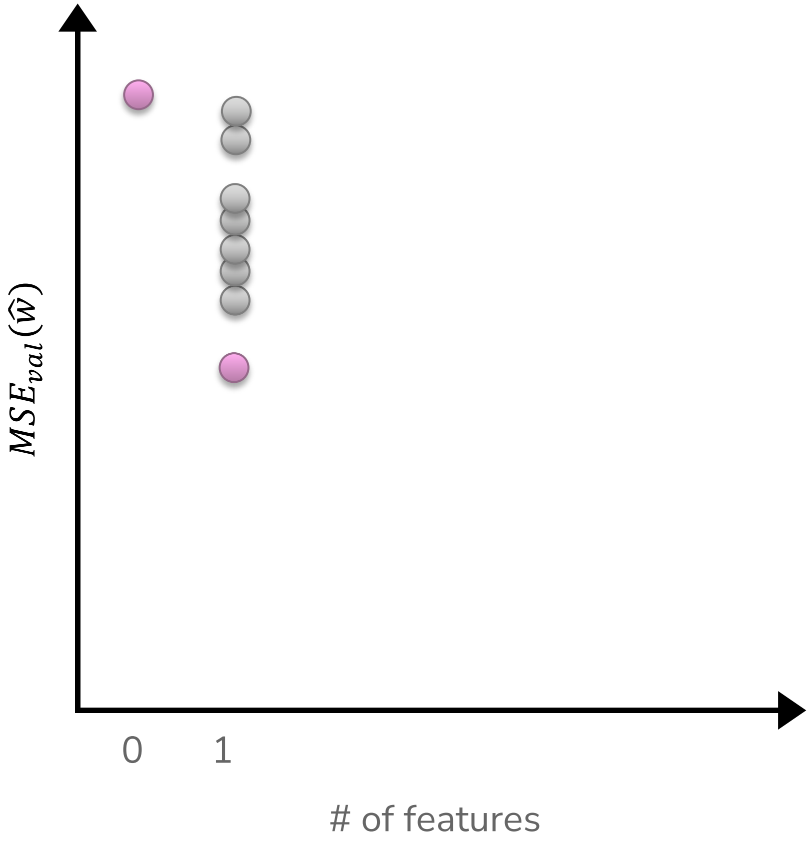

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 1




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

# Best Model Size 1



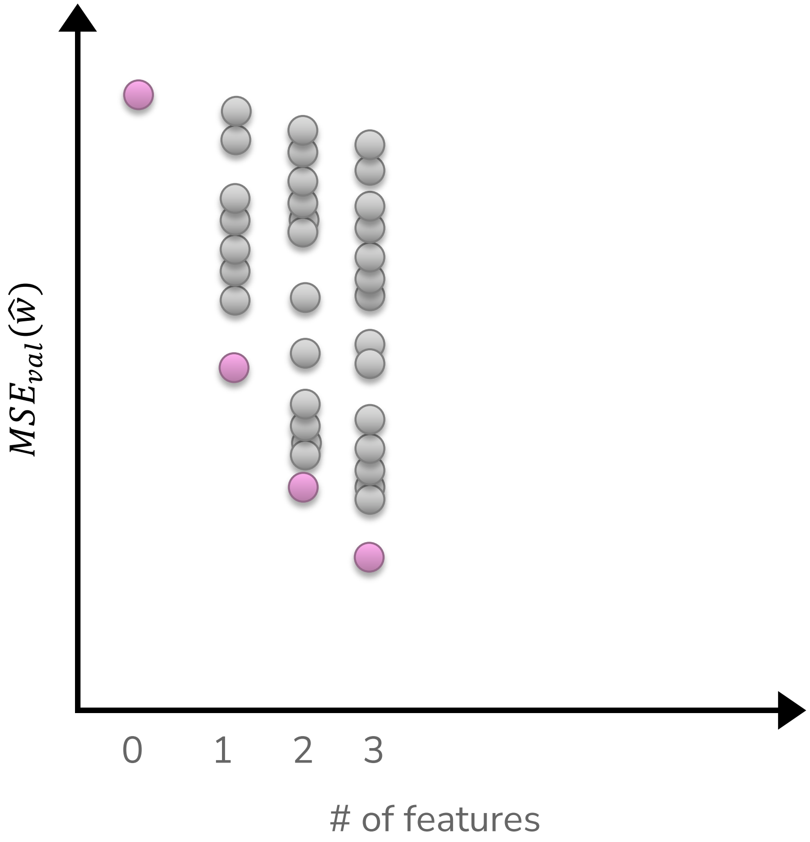

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 2


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

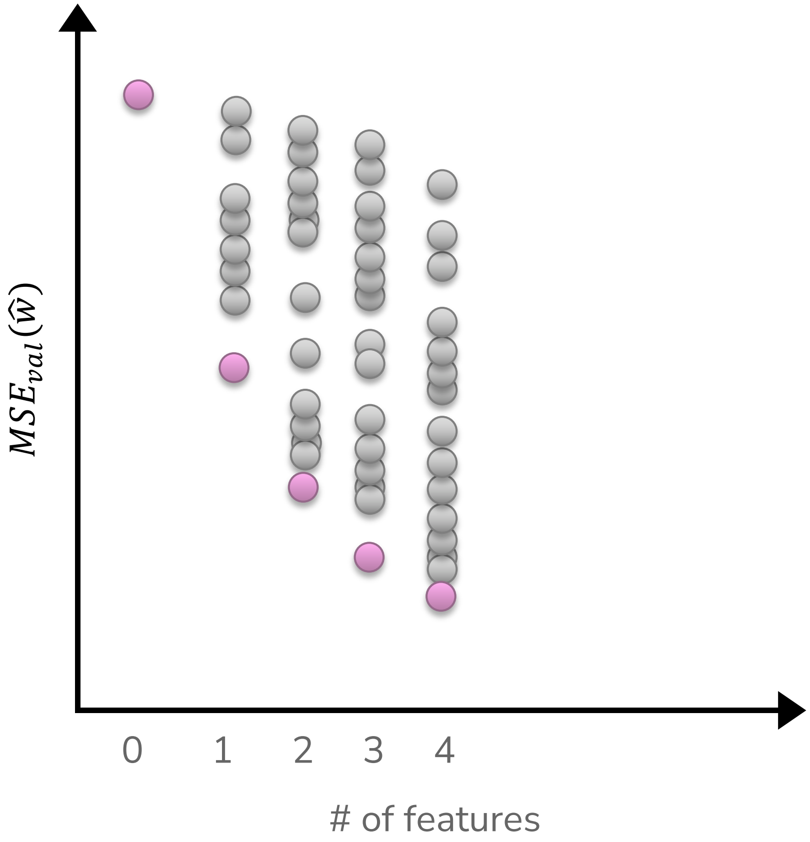
**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 3



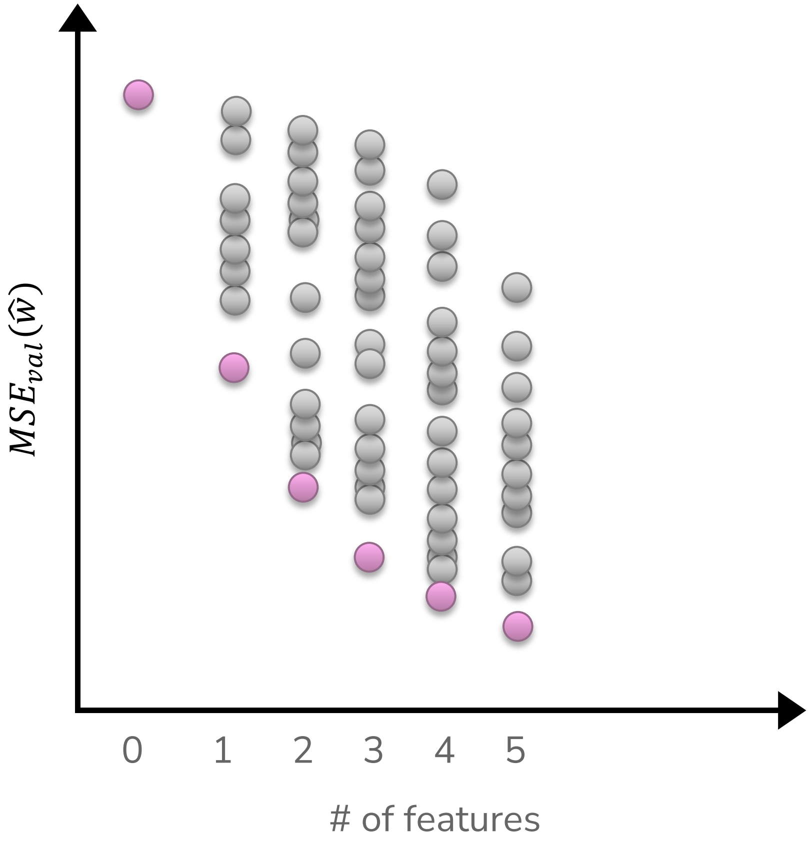

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 4



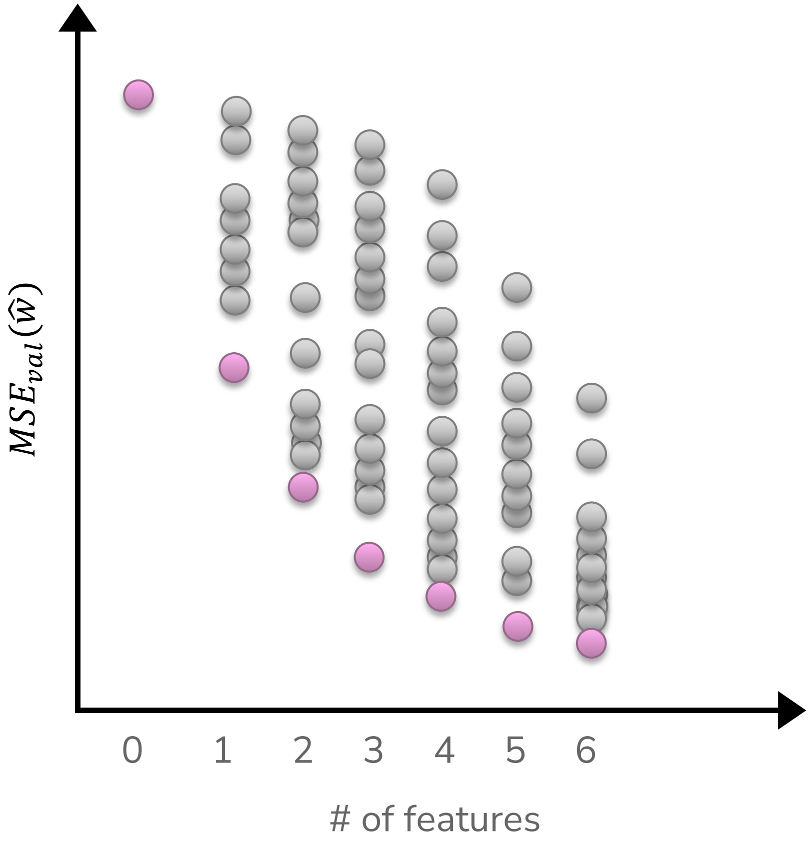

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 5



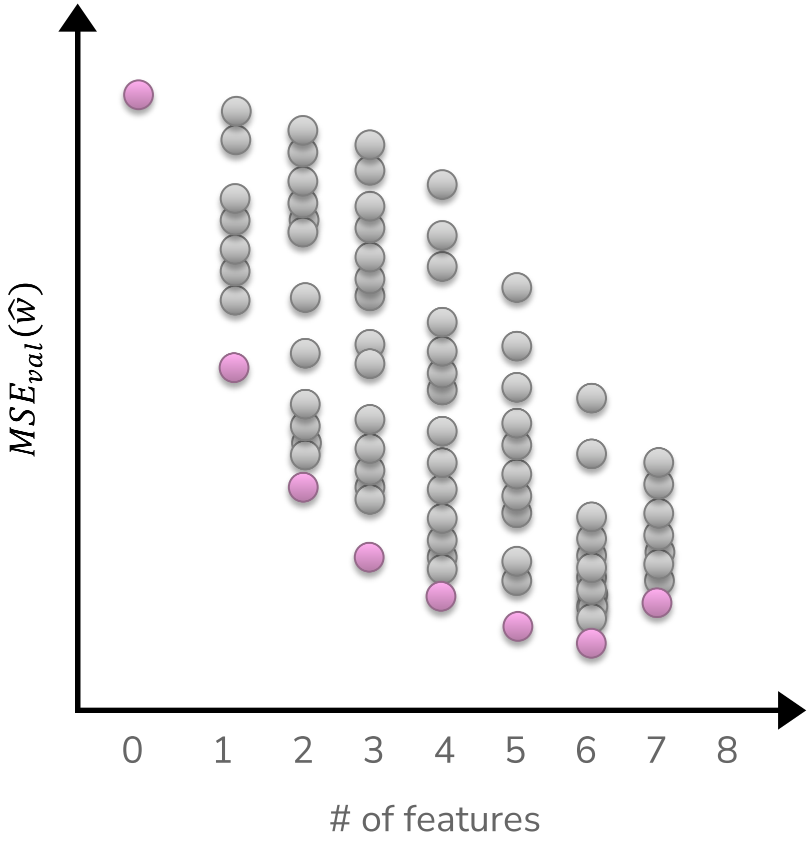

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# Best Model Size 6



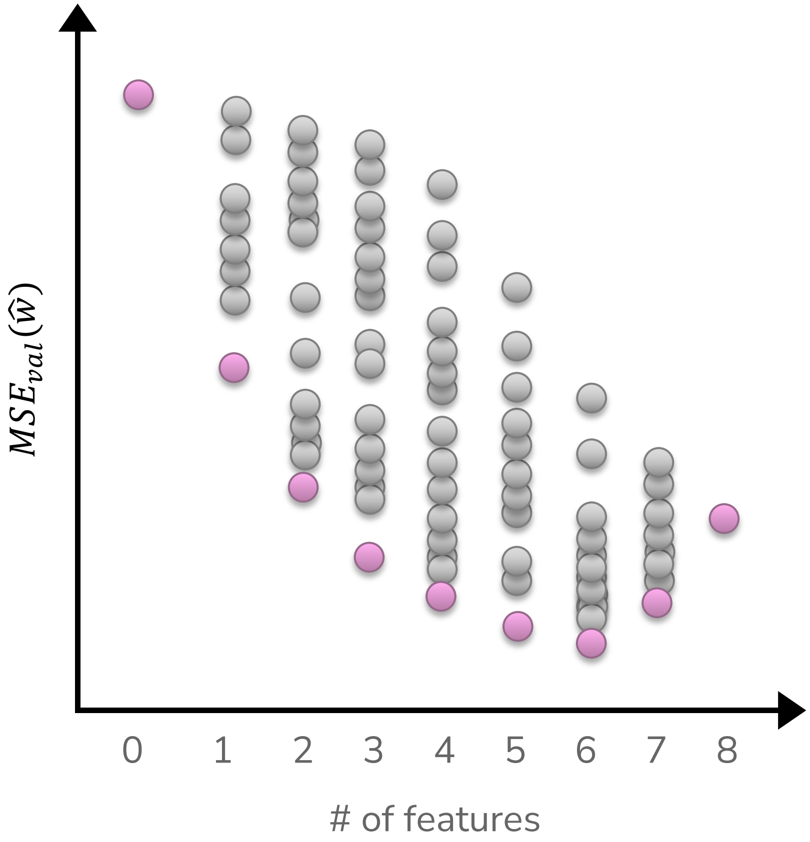

**Features**
# bathrooms
# bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

# 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?**

$\hat{y}_i = w_0$ [0 0 0 … 0 0 0]

$\hat{y}_i = w_0 + w_1 h_1(x)$ [1 0 0 … 0 0 0]

$\hat{y}_i = w_0 + w_2 h_2(x)$ [0 1 0 … 0 0 0]

… …

$\hat{y}_i = w_0 + w_1 h_1(x) + w_2 h_2(x)$ [1 1 0 … 0 0 0]

… …

$\hat{y}_i = w_0 + w_1 h_1(x) + \ldots + w_D h_D(x)$ [1 1 1 … 1 1 1]

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 $7.5 \times 10^{20}$ years

- 50,000,000,000x longer than the age of the universe!

Poll Everywhere

Think

2 min

pollev.com/cs416

How many linear regression models in total do we have to evaluate for a dataset with d features in order to find the global optimum?

a) $d^2$

b) $d$

c) $2^d$

d) $d + 12$

# 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 $k$

```
min_val = ∞
for i ← 1..k:
    Find feature f_i not in S_{i-1}, that when combined
    with S_{i-1}, minimizes the validation loss the most.
    S_i ← S_{i-1} ∪ {f_i}
    if val_loss(S_i) > min_val:
        break
```

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

Poll Everywhere

Think

2 min

pollev.com/cs416

What is the runtime of this greedy algorithm?

a) $O(k)$

b) $O(k^2)$

c) $O(k^3)$

d) $O(1)$

# Option 2
## Regularization

# Recap: Regularization

Before, we used the quality metric that minimize loss

$$\hat{w} = \underset{w}{\operatorname{argmin}}\, L(w)$$

Change quality metric to balance loss with measure of overfitting

$L(w)$ is the measure of fit

$R(w)$ measures the magnitude of coefficients

$$\hat{w} = \underset{w}{\operatorname{argmin}}\, L(w) + R(w)$$

How do we actually measure the magnitude of coefficients?

# Recap: Magnitude

Come up with some number that summarizes the magnitude of the weights $w$.

$$\hat{w} = \underset{w}{\operatorname{argmin}} \, MSE(w) + \lambda R(w)$$

**Sum?**

$$R(w) = w_0 + w_1 + \cdots + w_d$$

Doesn't work because the weights can cancel out (e.g. $w_0 = 1000, w_1 = -1000$) which so $R(w)$ doesn't reflect the magnitudes of the weights

**Sum of absolute values?**

$$R(w) = |w_0| + |w_1| + \cdots + |w_d| = \|w\|_1$$

It works! We're using L1-norm, for L1-regularization (LASSO)

**Sum of squares?**

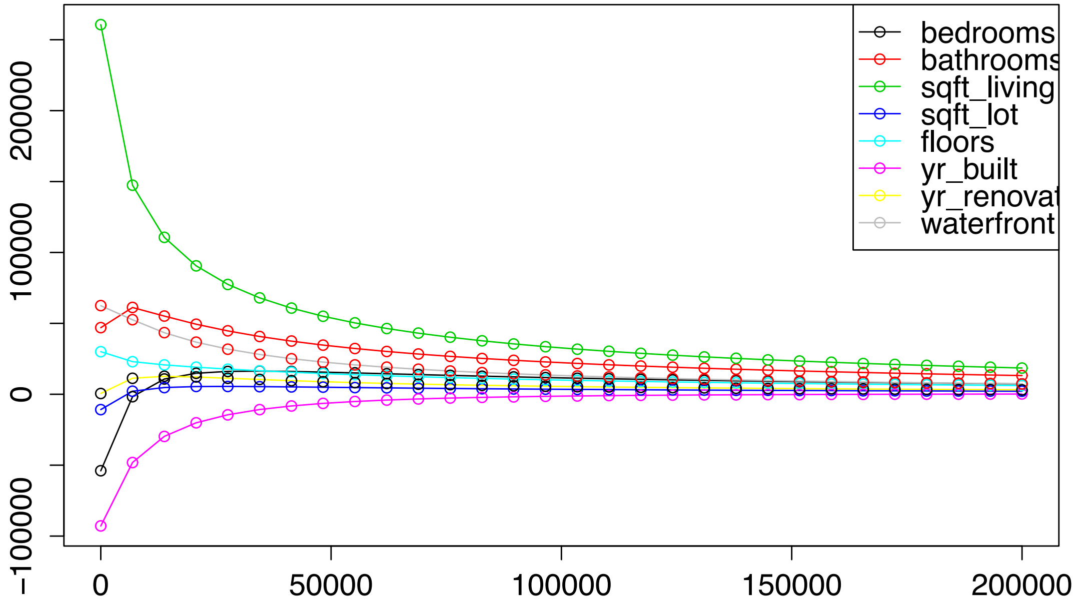
$$R(w) = |w_0|^2 + |w_1|^2 + \ldots + |w_d|^2 = w_0^2 + w_1^2 + \ldots + w_d^2 = \|w\|_2^2$$

It works! We're using L2-norm, for L2-regularization (Ridge Regression)

**Note:** Definition of p-Norm: $\|w\|_p^p = |w_0|^p + |w_1|^p + \ldots + |w_d|^p$

# 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?

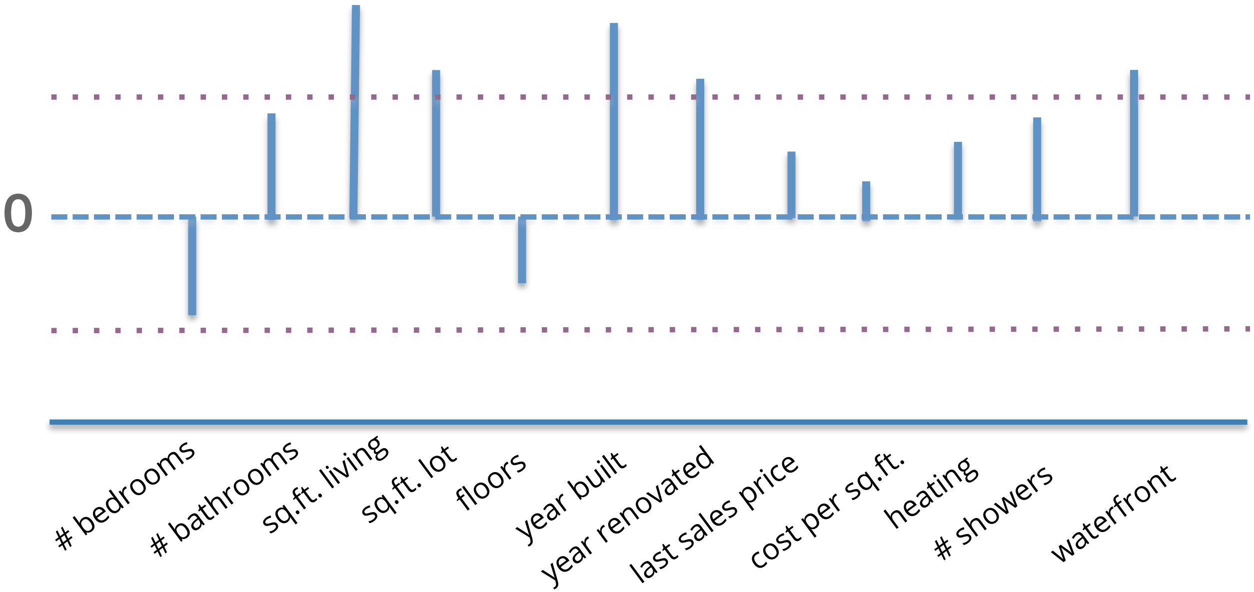


# Ridge for Feature Selection

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.

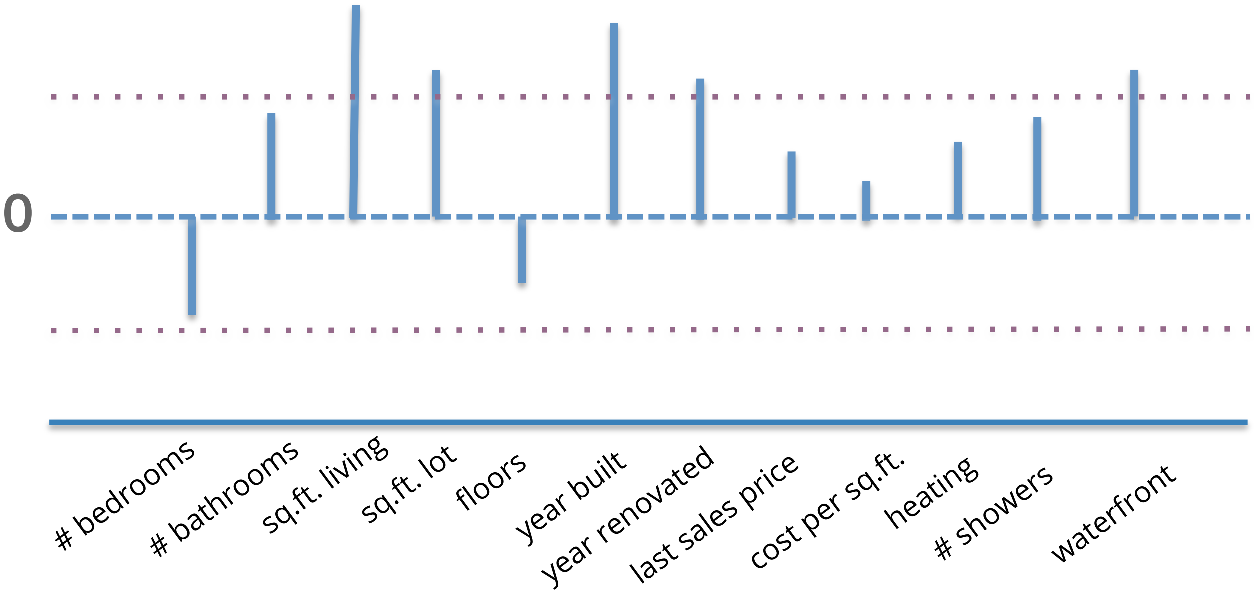
Non-zero coefficients would be considered selected as important.



# Ridge for Feature Selection

Look at two related features #bathrooms and # showers.

Our model ended up not choosing any features about bathrooms!

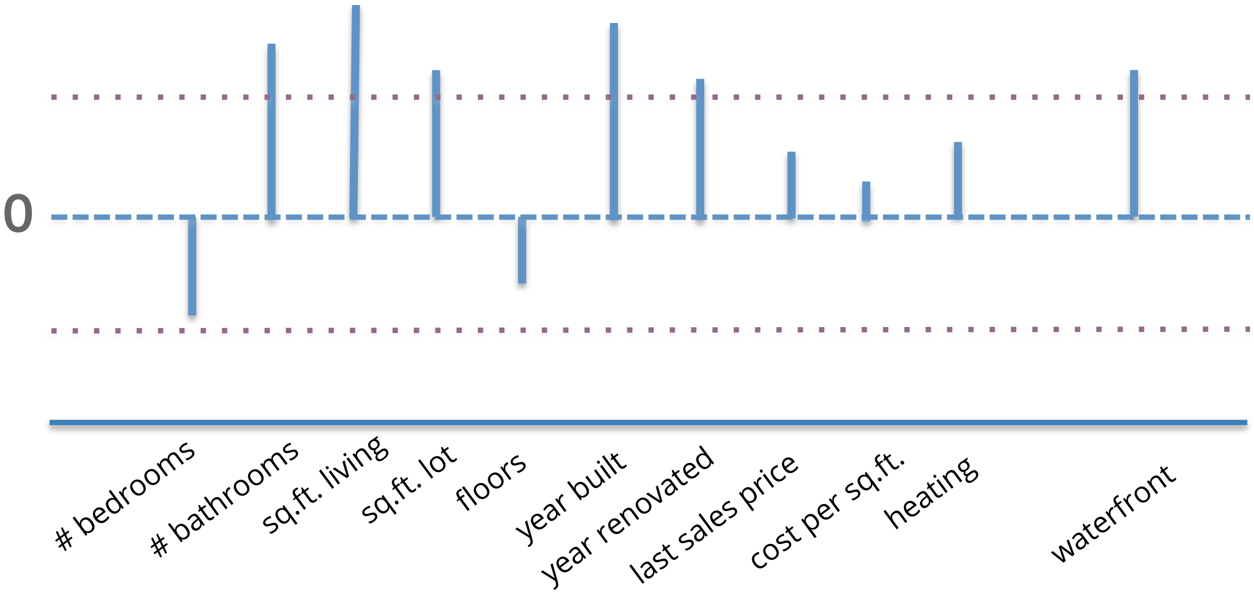


# Ridge for Feature Selection

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

# LASSO Regression

Change quality metric to minimize

$$\hat{w} = \min_{w} MSE(W) + \lambda \left|\left|w\right|\right|_1$$

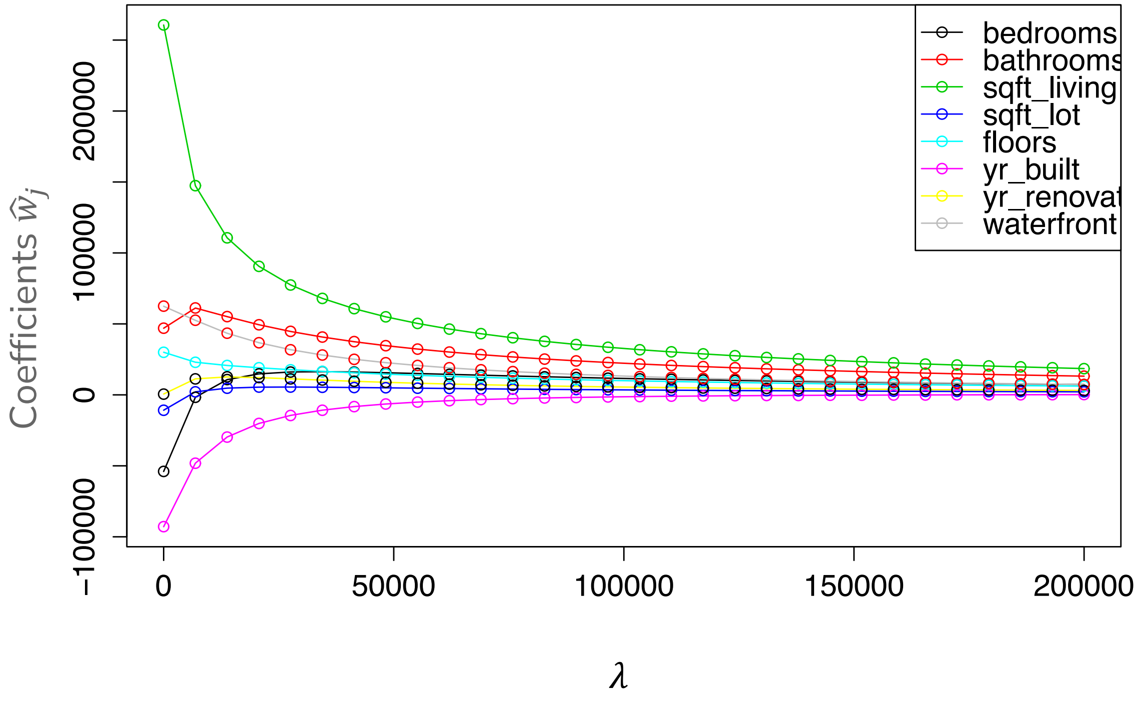
$\lambda$ is a tuning parameter that changes how much the model cares about the regularization term.

**What if $\lambda = 0$?**

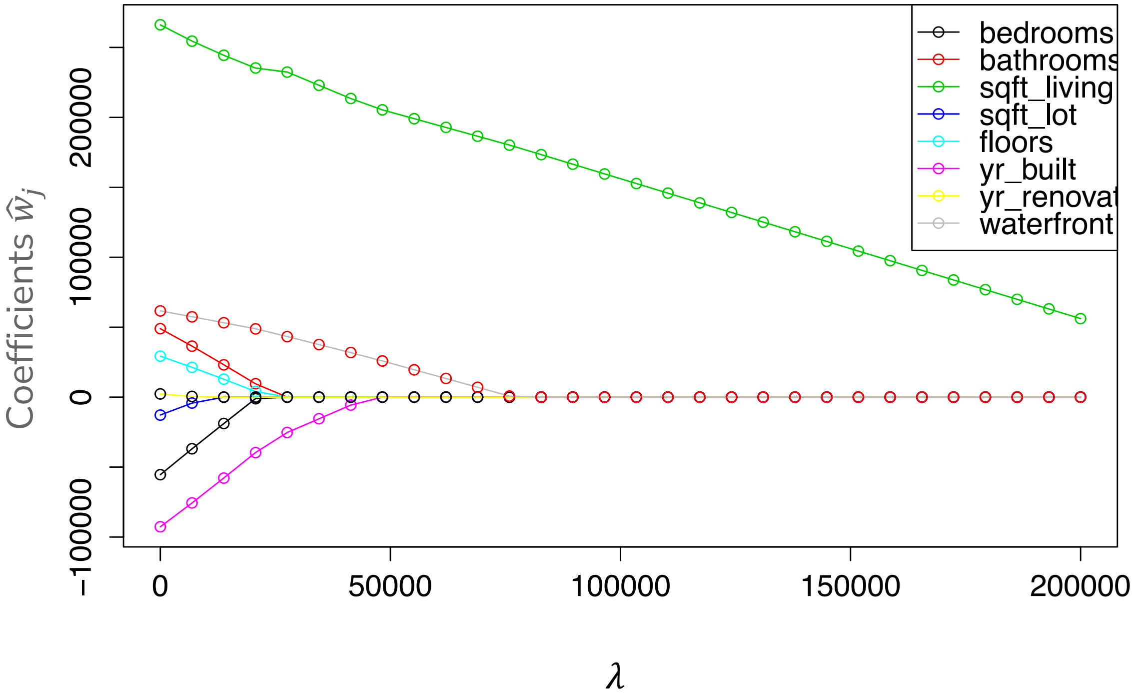
**What if $\lambda = \infty$?**

**$\lambda$ in between?**

# Ridge (L2) Coefficient Paths



# LASSO (L1) Coefficient Paths



Poll Everywhere

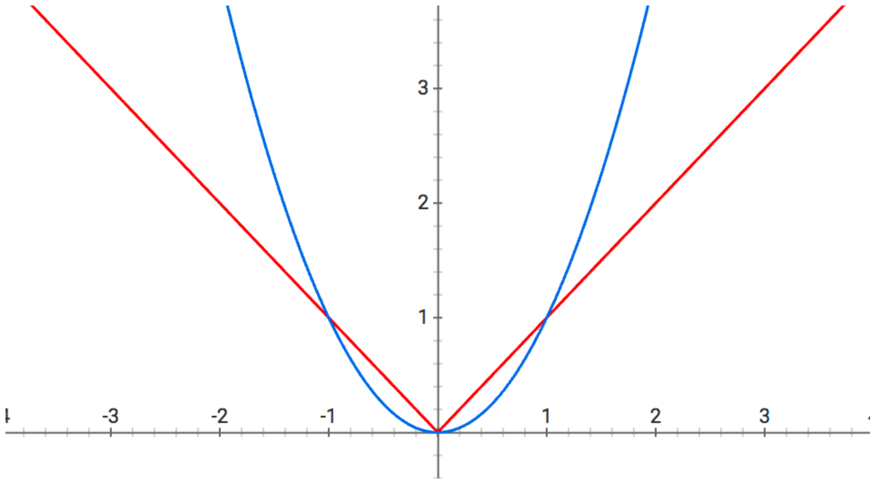
Group

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.

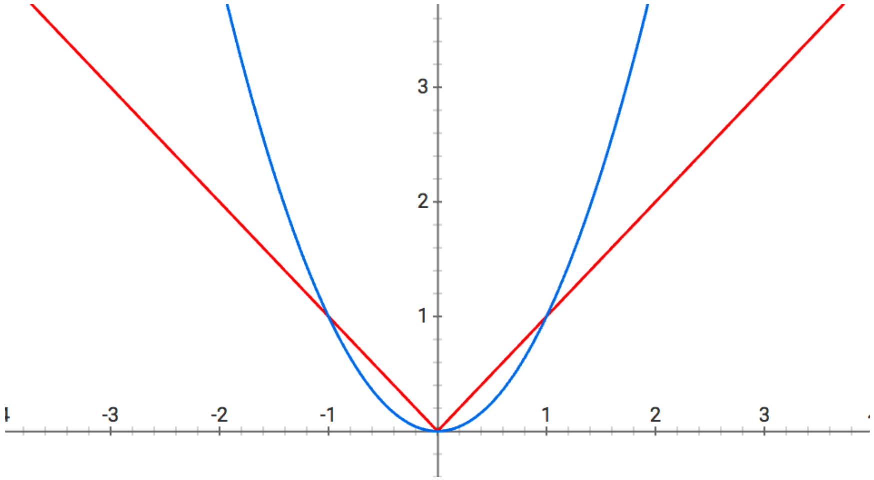
This has to do with the shape of the norm

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

# 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.

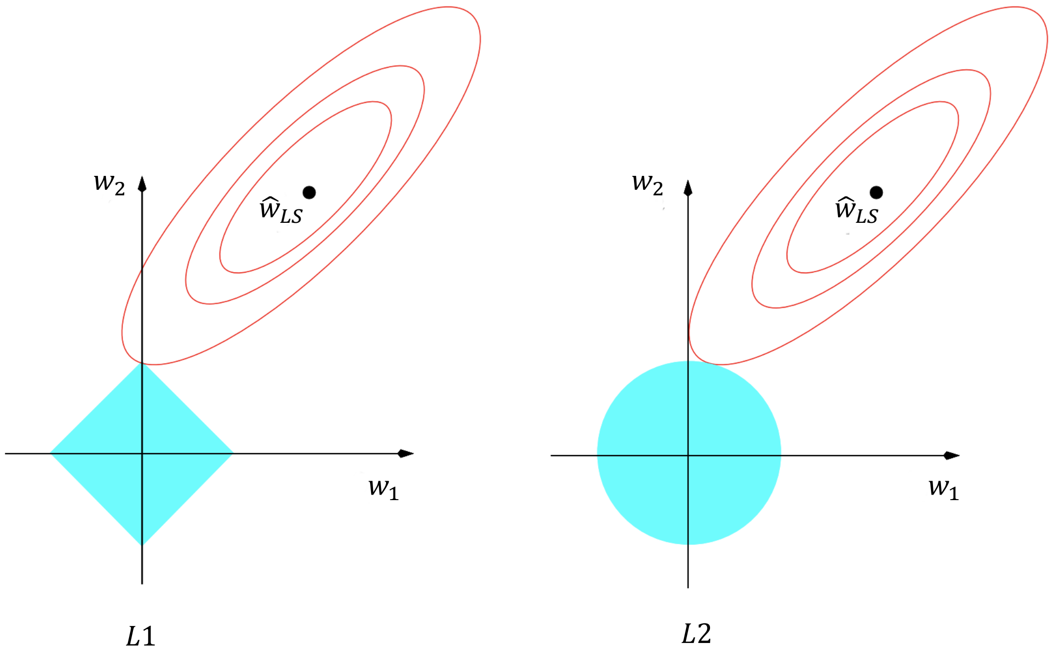
This has to do with the shape of the norm




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

# Sparsity Geometry

Another way to visualize why LASSO prefers sparse solutions




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

# Brain Break

# Choosing $\lambda$

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

# 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 $\lambda$ 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

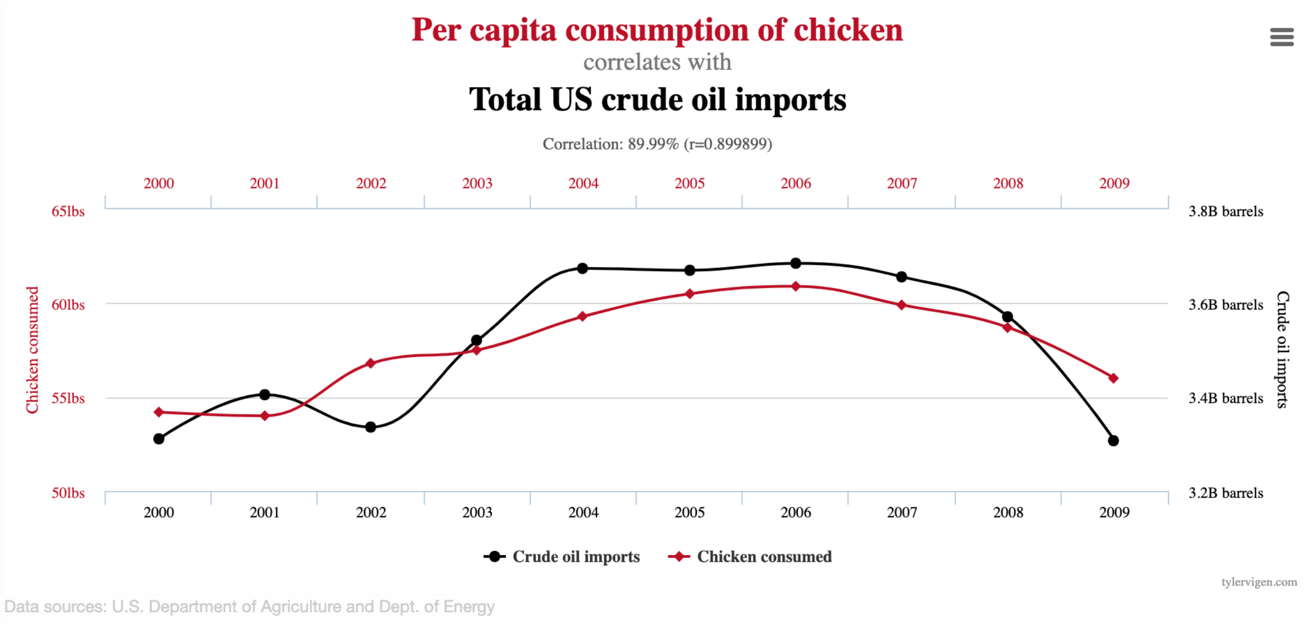
It’s possible to try to remove the bias from the LASSO solution using the following steps

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

# Interesting visualizations of “Correlation doesn’t imply Causation”

https://www.tylervigen.com/spurious-correlations



# 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

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

Combines both to achieve best of both worlds!

Poll Everywhere

Think

2 minutes

pollev.com/cs416

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

What are some differences between L1 and L2 regularizations? Their implications?

# Differences between L1 and L2 regularizations

L1 (LASSO):

- Introduces more sparsity to the model
- Less sensitive to outliers
- Helpful for feature selection, making the model more interpretable
- More computational efficient as a model (due to the sparse solutions, so you have to compute less dot products)

L2 (Ridge):

- Makes the weights small (but not 0)
- More sensitive to outliers (due to the squared terms)
- Usually works better in practice

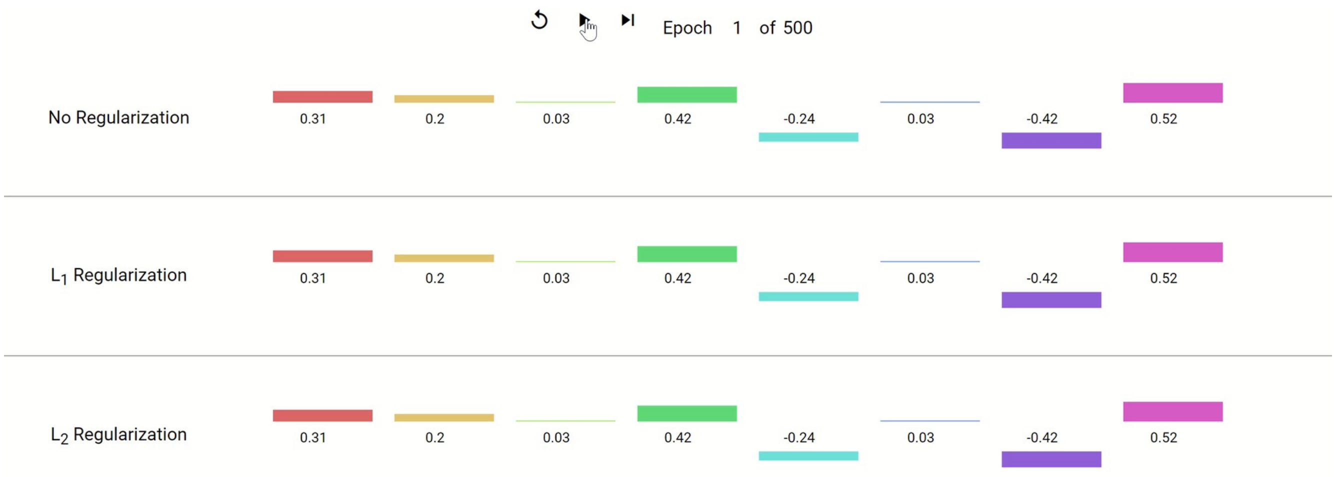
# A Big Grain of Salt

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!

# Coefficient Paths – Another View

Example from Google's [Machine Learning Crash Course]



# 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 $\lambda$ is varied

Interpret LASSO coefficient path plot

Compare and contrast LASSO (L1) and Ridge (L2)