CSE/STAT 416

Regularization – LASSO Regression

Pre-Class Videos

Tanmay Shah
University of Washington
July 1, 2024

Pre-Class Video 1

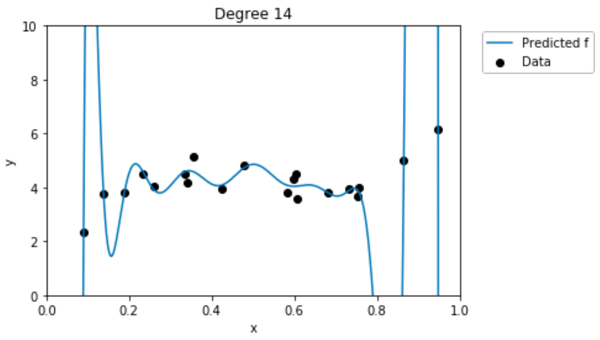
Ridge Regression

Recap: Number of Features

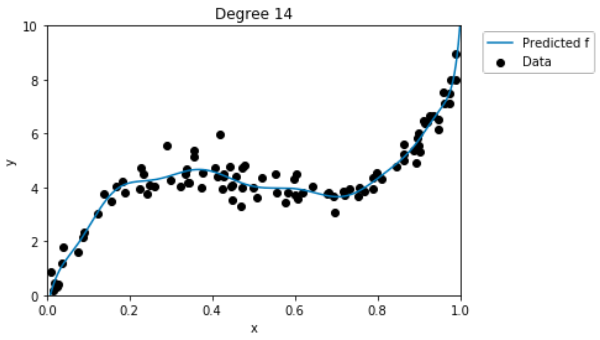
Overfitting is not limited to polynomial regression of large degree. It can also happen if you use a large number of features!

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



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

Recap: Ridge Regression

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.

Hyperparameter tuning

Choosing λ

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
```

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

```
  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-Class Video 2

Feature Selection & 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

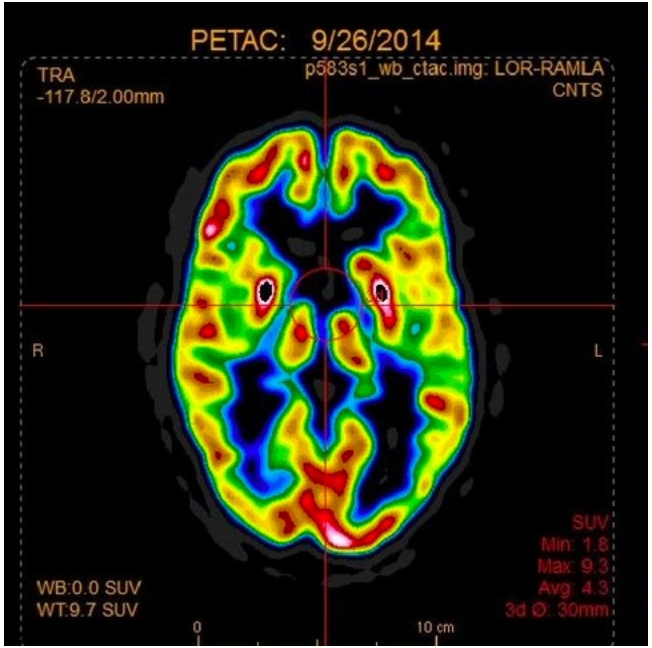
Dishwasher
Garbage disposal
Microwave
Range / Oven
Refrigerator
Washer
Dryer
Laundry location
Heating type
Jetted Tub
Deck
Fenced Yard
Lawn
Garden
Sprinkler System

…

Sparsity: Reading Minds

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

x

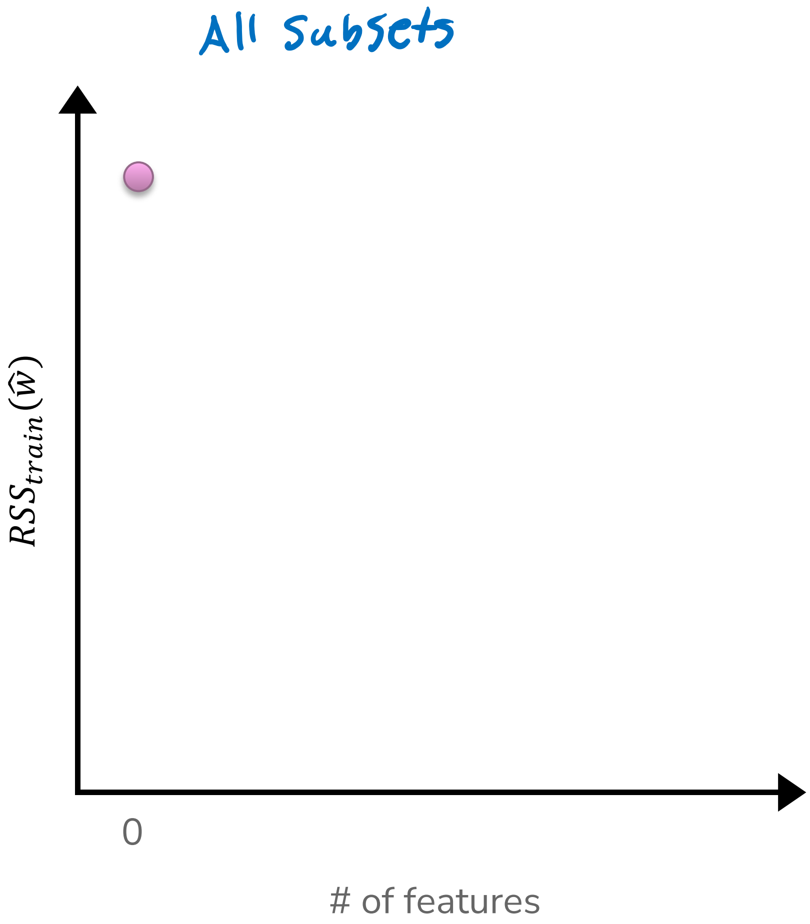


y

10 – Happy ☺

0 – Sad ☹

Best Model Size 0



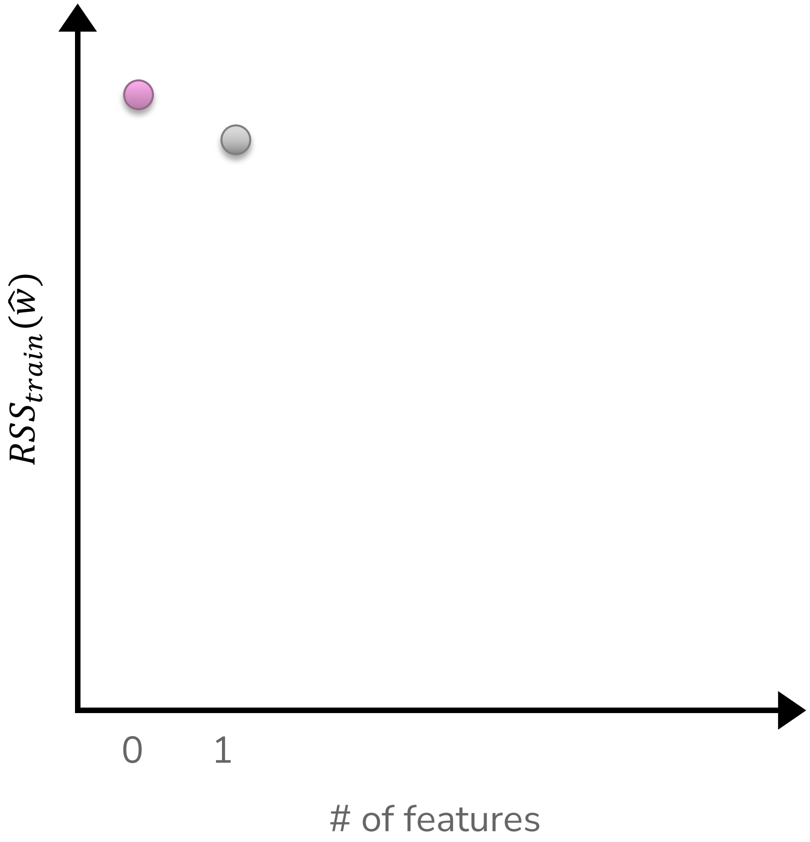
Noise only:

$y_i = \varepsilon_i$

Features

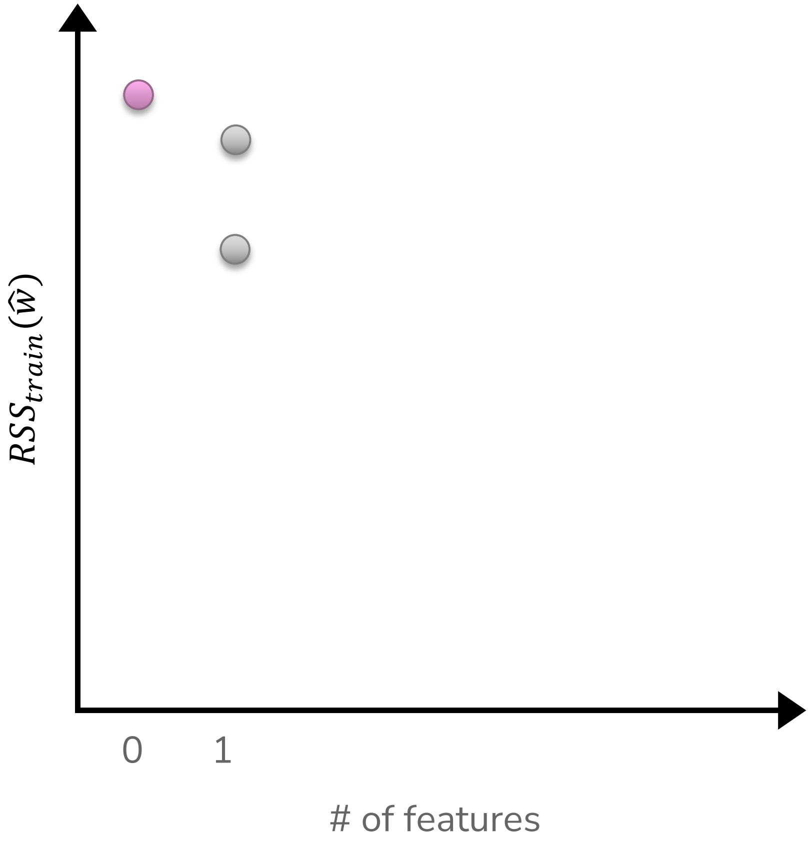
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



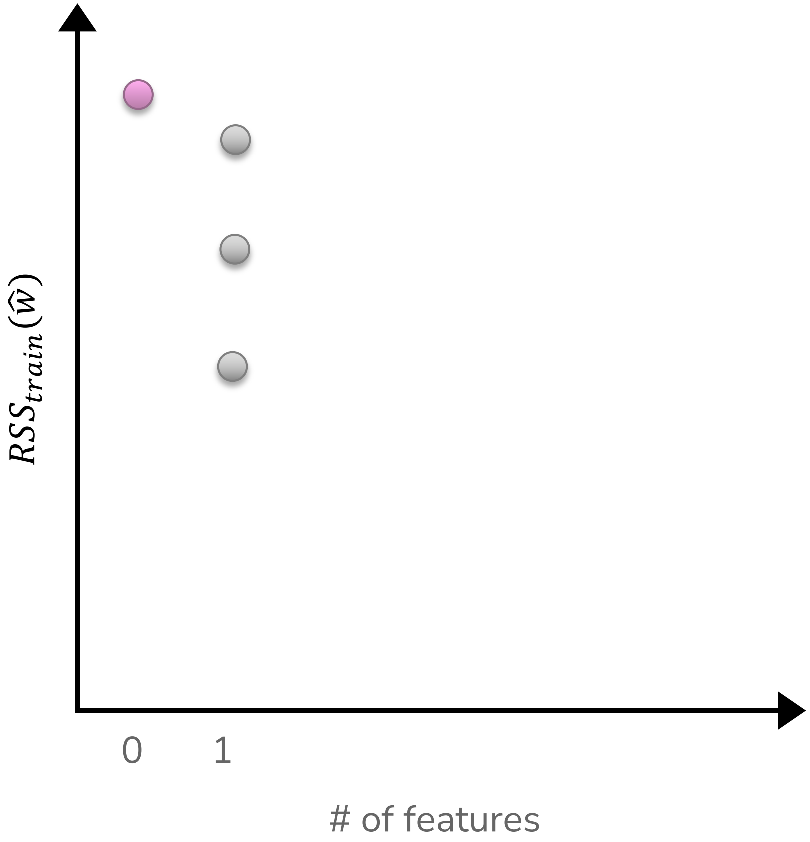
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



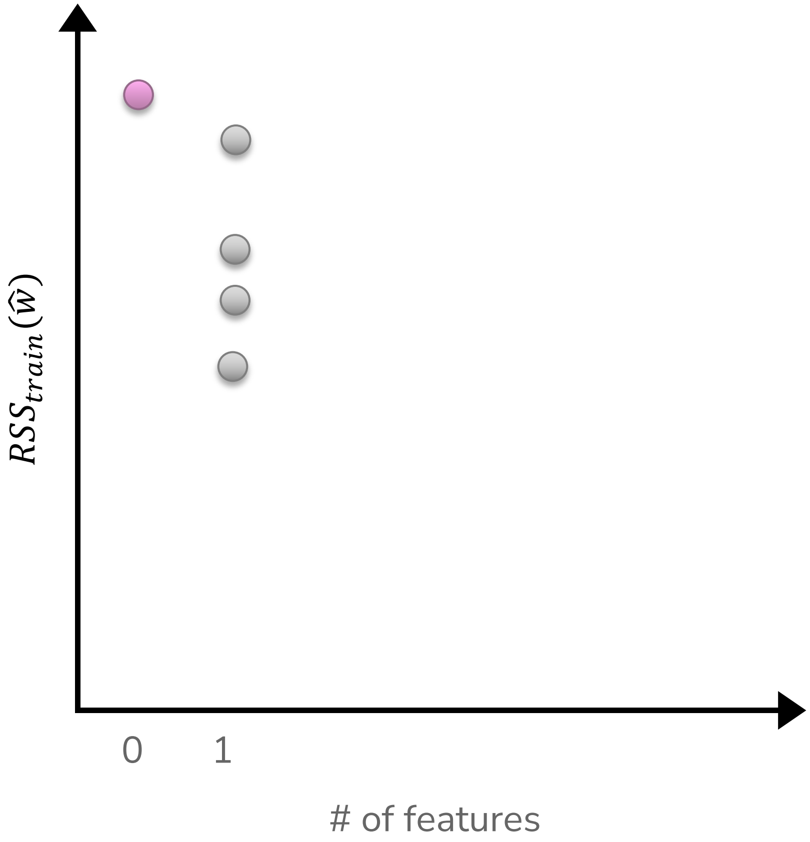
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



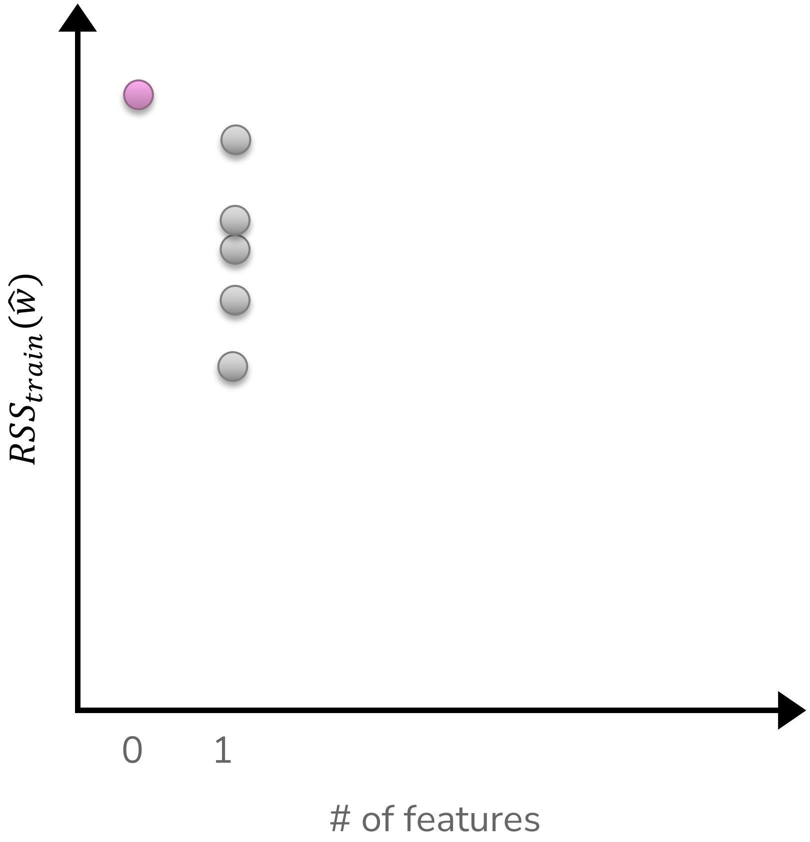
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



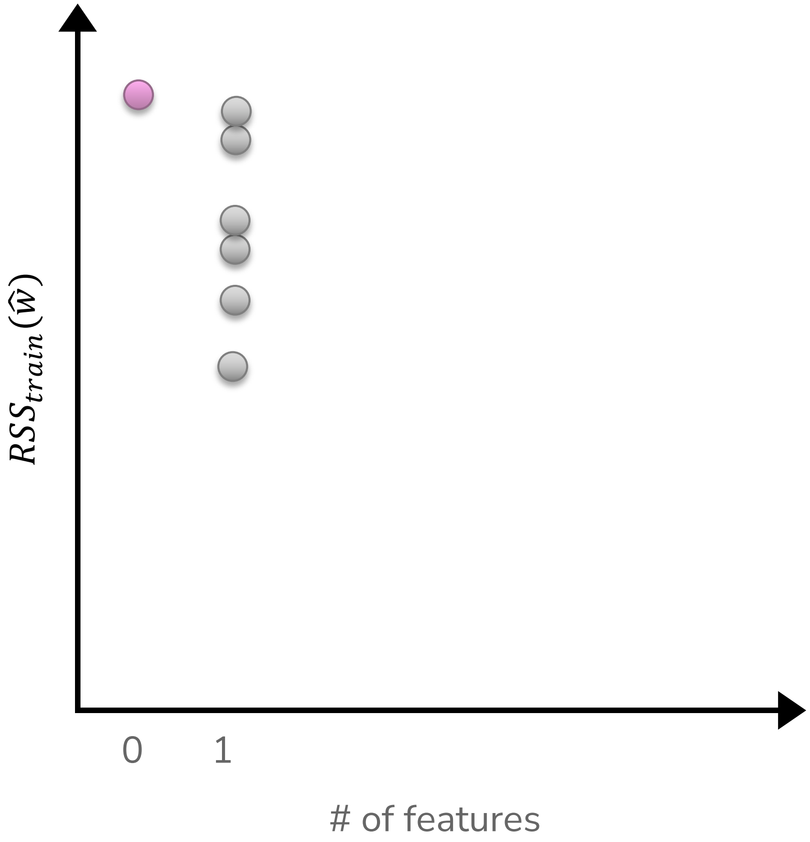
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



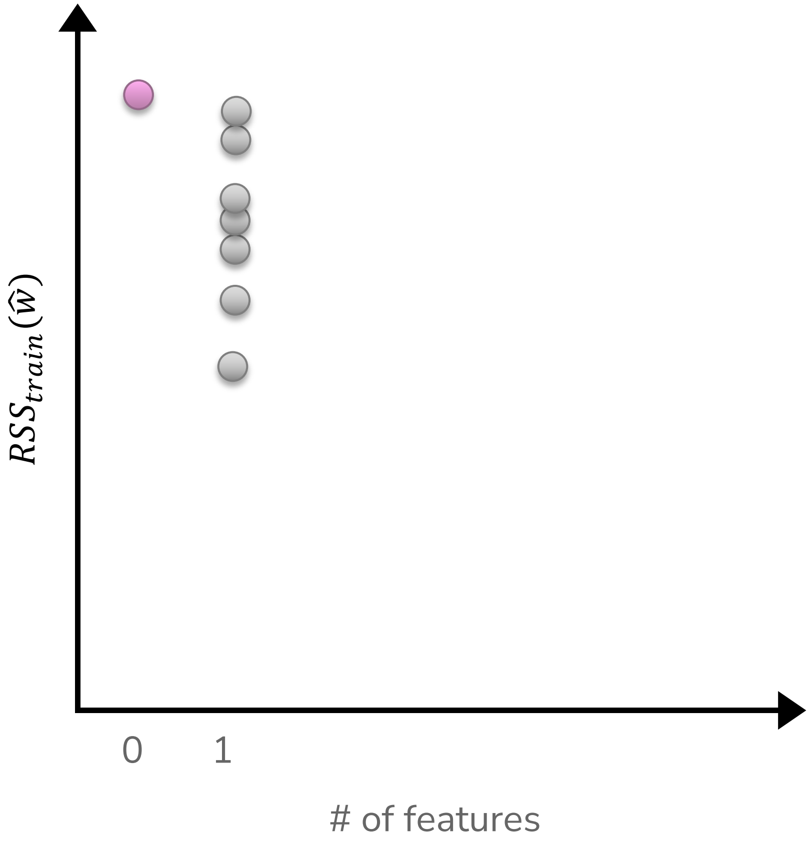
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



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

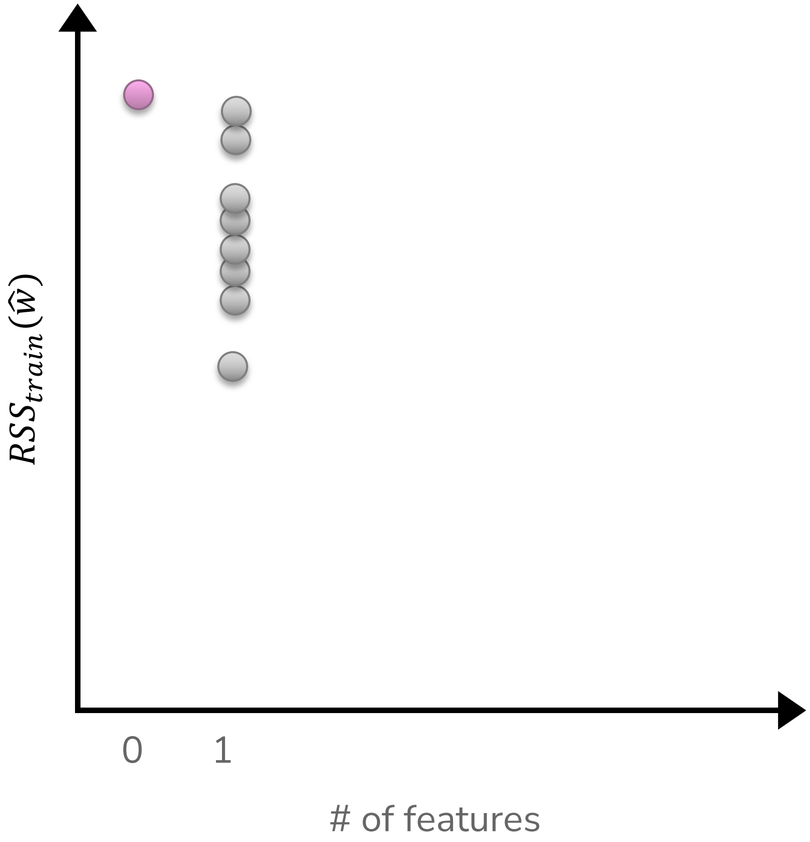
Best Model Size 1



Features

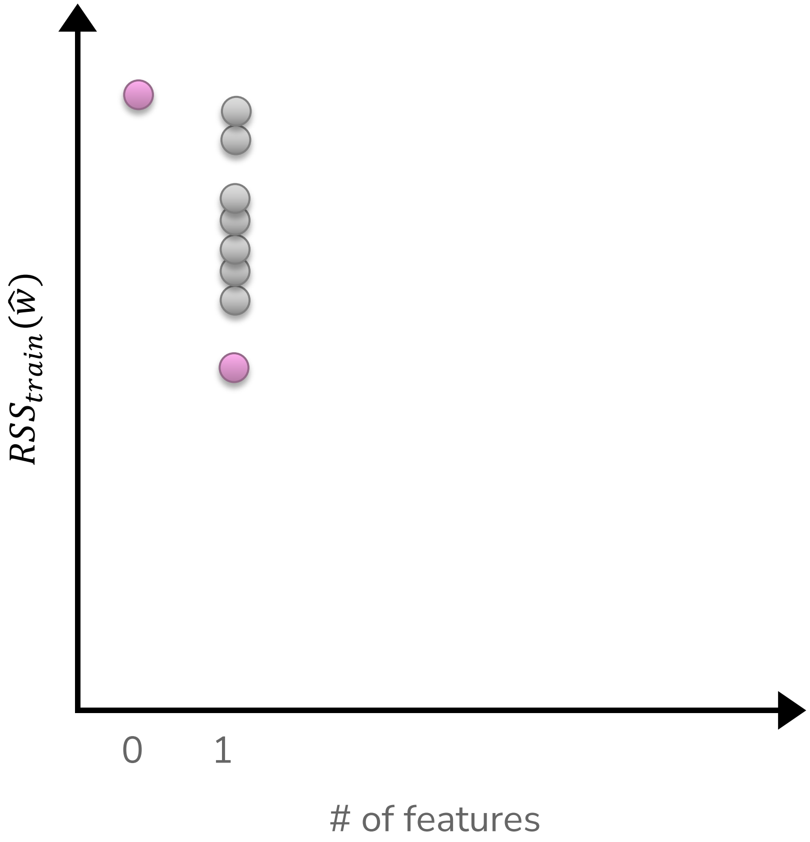
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 1



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

Best Model Size 1



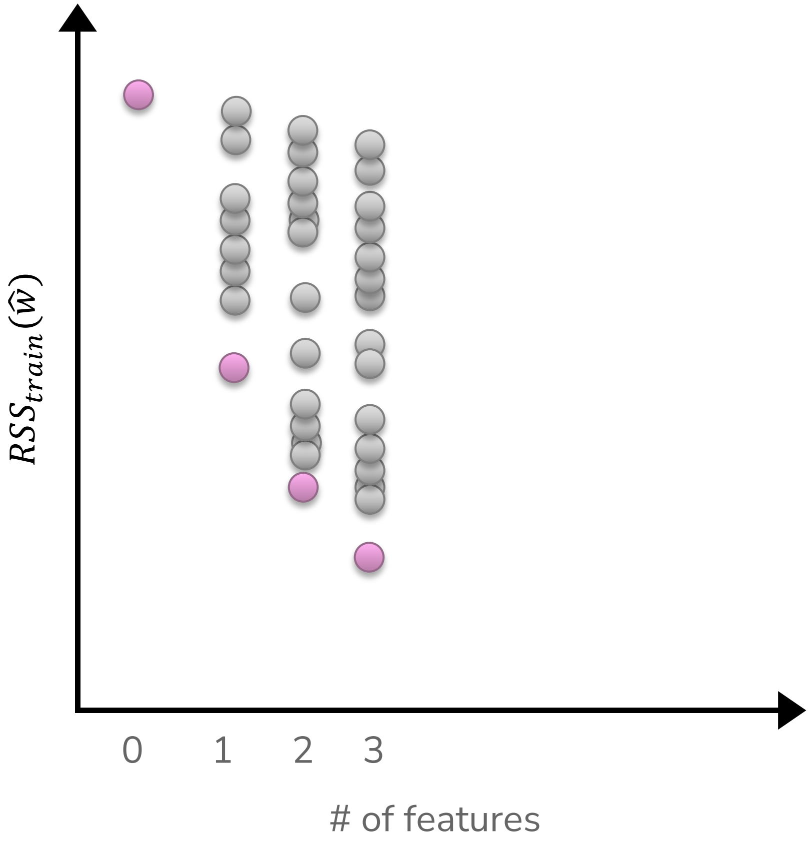
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 2

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

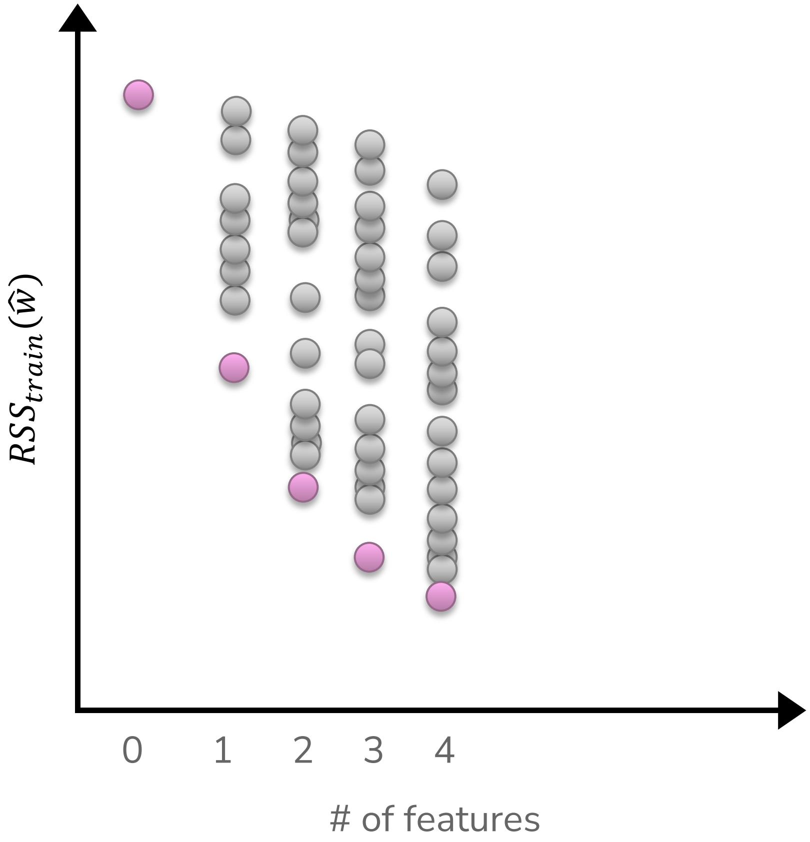
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 3



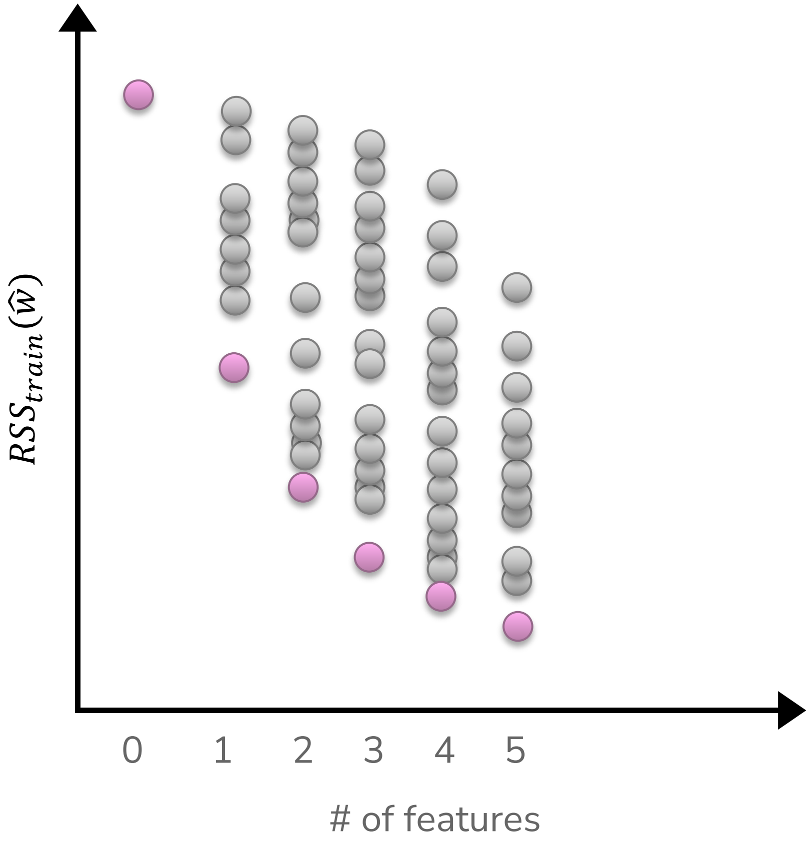
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 4



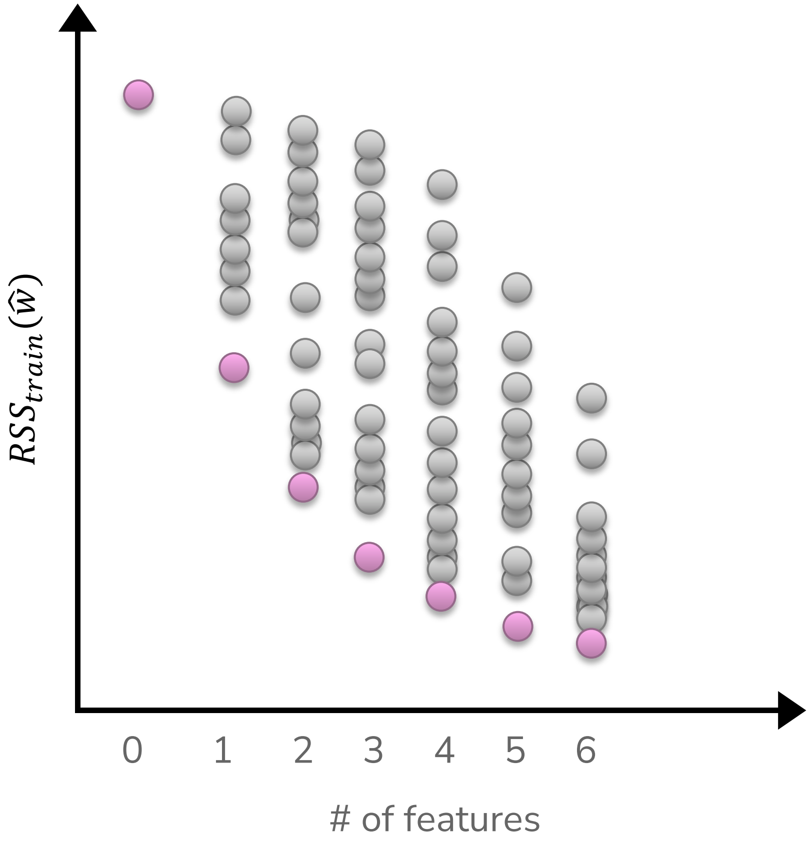
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 5



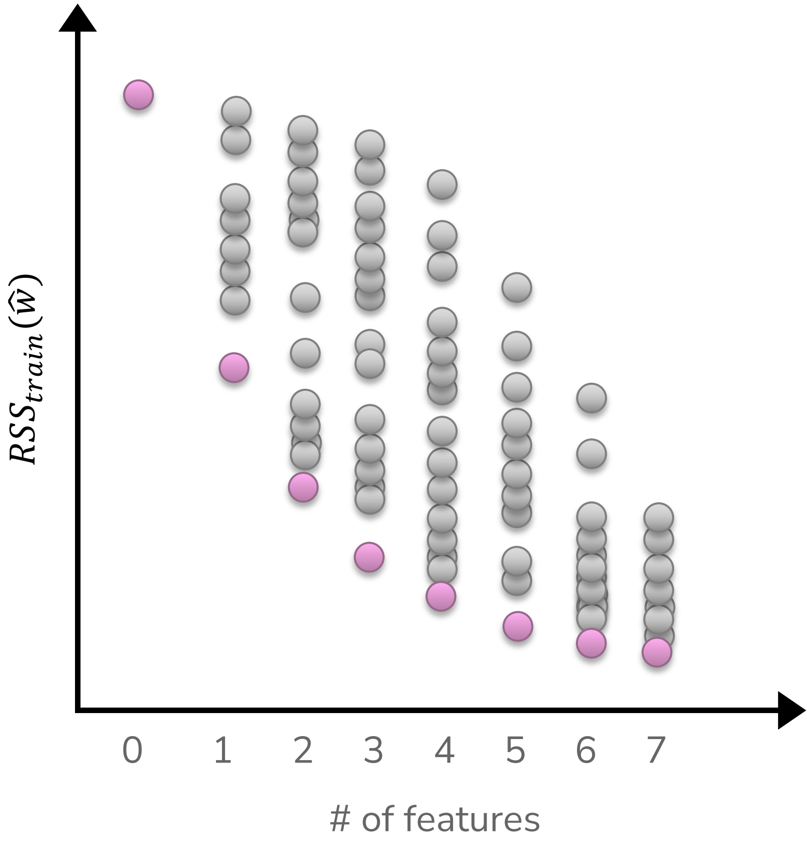
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

Best Model Size 6



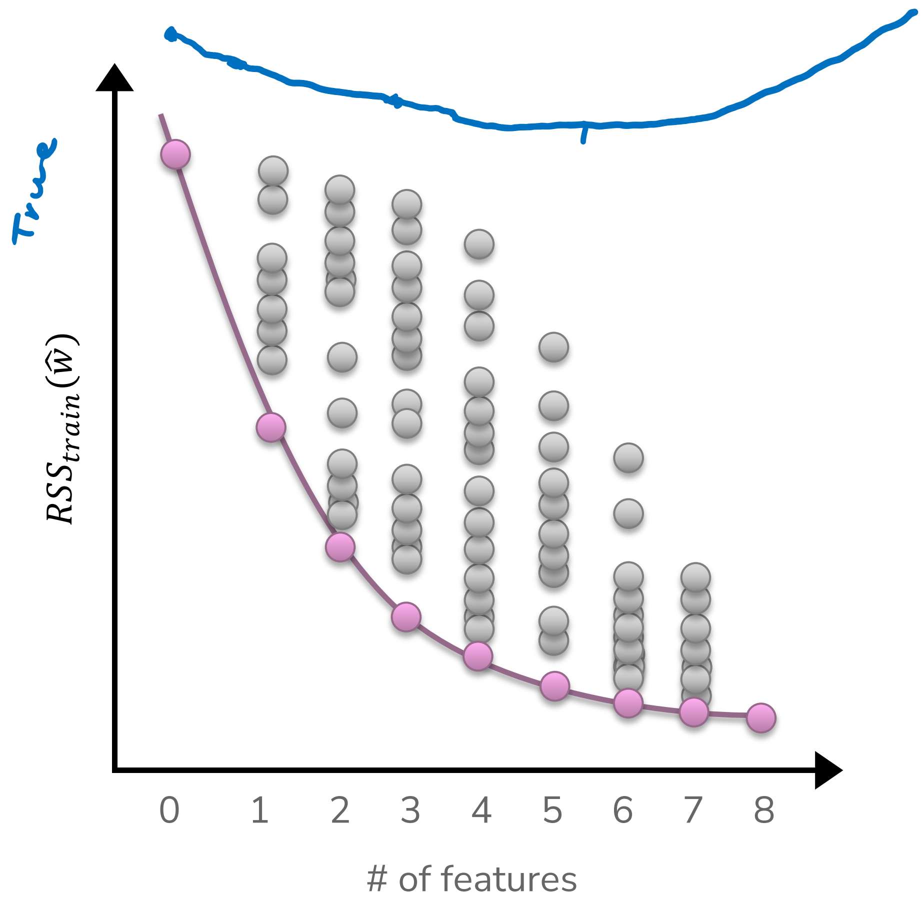
Features
bathrooms
bedrooms
sq.ft. living
sq.ft lot
floors
year built
year renovated
waterfront

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 Bayesian Information Criterion (BIC)

CSE/STAT 416

Regularization – LASSO Regression

Tanmay Shah
University of Washington
July 1, 2024

❓ **Questions?** Raise hand or **sli.do #cs416**
💬 **Before Class:** Favorite food near campus?
🎵 **Listening to:** [Sammy Rae & The Friends]

Administrivia

Last lecture in the “Regression” case study!

- Next 2 weeks: Classification
- Following 1 week: Deep Learning

Upcoming Due Dates:

- HW1 due tomorrow
- Learning Reflection 2 due Fri 11:59PM

OH is a great place to ask your learning reflection questions!

Reminder of resources

Regularization recap

Recap: Ridge Regression

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

Change quality metric to minimize

$$\hat{w} = \min_{w} \overset{MSE}{\cancel{RSS}}(W) + \underline{\lambda \|w\|_2^2}$$

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

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

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

LASSO Regression

L1 norm: $||w||_1 = \sum_{j=1}^{D} |w_j|$

Change quality metric to minimize

$$\hat{w} = \underset{w}{\operatorname{argmin}}\, MSE(w) + \lambda||w||_1$$

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

What if $\lambda = 0$?

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

$= \hat{w}_{OLS}$

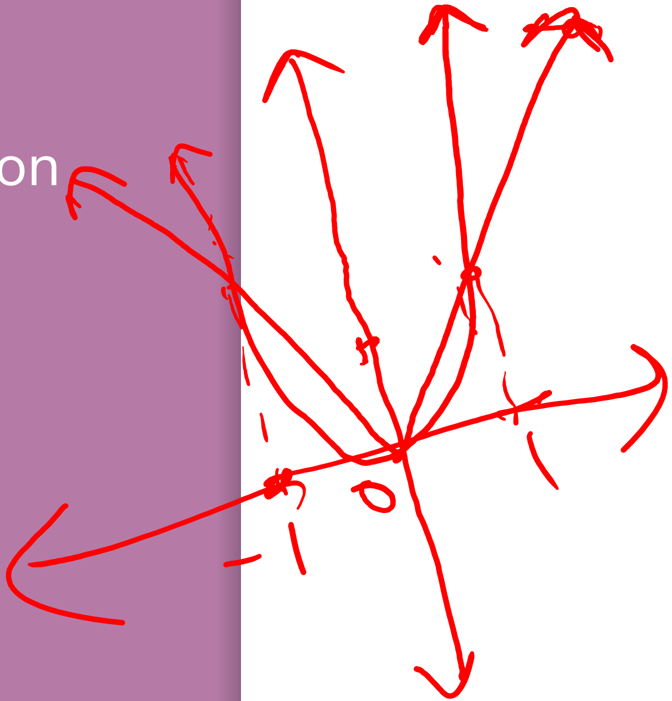
What if $\lambda = \infty$?

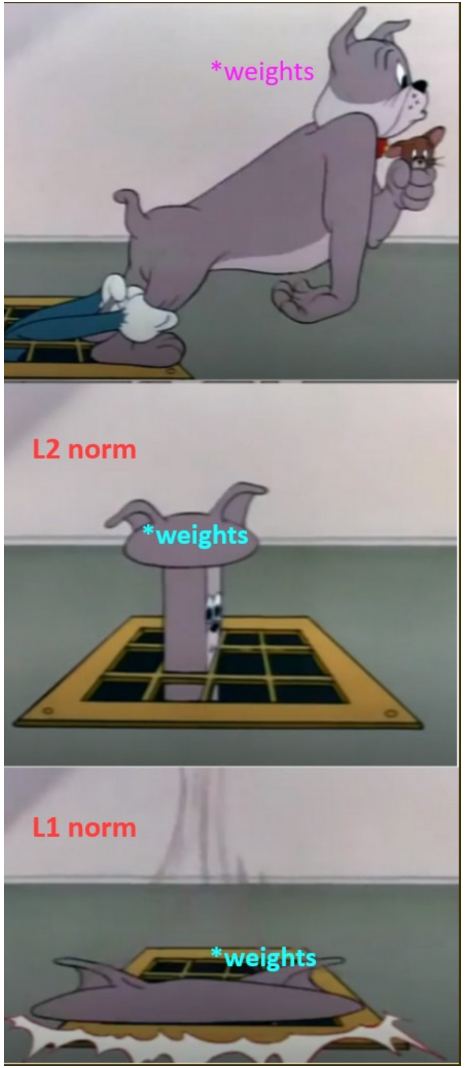
$\hat{w} = \underset{w}{\operatorname{argmin}}\, \lambda||w||_1 = \vec{0} = [0, 0, \ldots, 0]$

λ in between?

$\boxed{0 \leq ||\hat{w}_{LASSO}||_1 \leq ||\hat{w}_{OLS}||_1}$

Ridge vs Lasso Regression





Poll Everywhere

Think

1 min

pollev.com/cs416

How should we choose the best value of λ?

After we train each model with a certain λ_i and find

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

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

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

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

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

e) None of the above

Poll Everywhere

Think

2 min

pollev.com/cs416

How should we choose the best value of λ?

After we train each model with a certain λ_i and find

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

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

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

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

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

e) None of the above

Used only during Training process

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

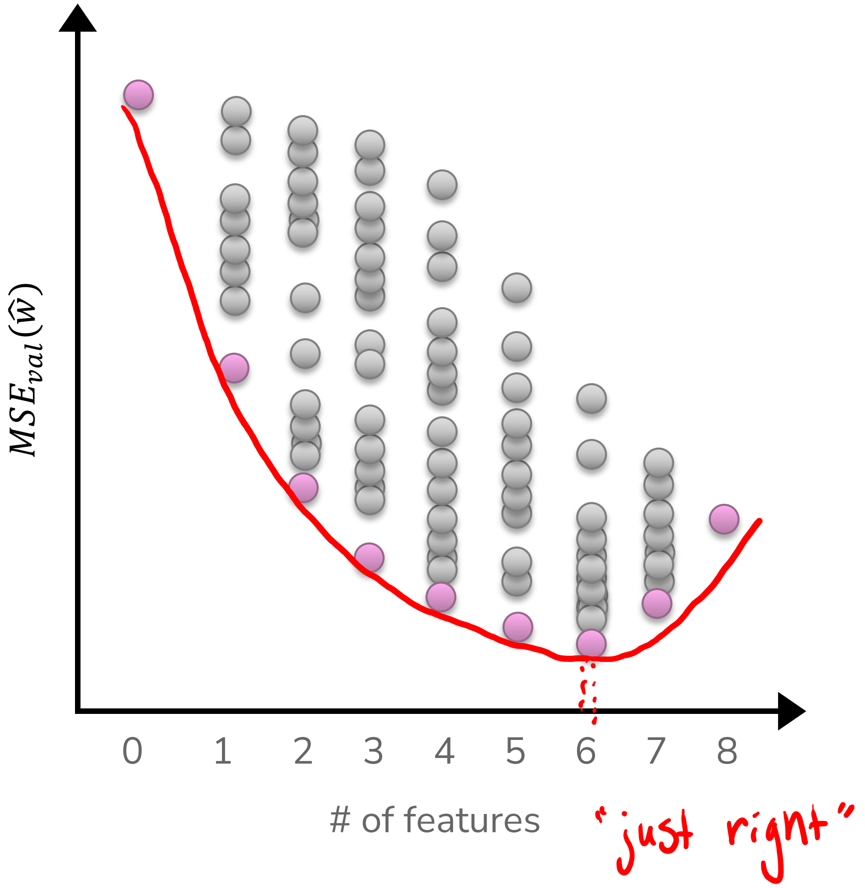
Imagine if we had MANY features (e.g. DNA). $\hat{w}$ could have 10^{11} coefficients. Evaluating $\hat{y} = \hat{w}^T h(x)$ would be very slow!

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

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

Best Model Size 8

Note: Video showed $MSE_{train}(\hat{w})$



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

Efficiency of All Subsets

whether or not $h_1(x)$ is in model

... $h_D(x)$

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]

2^D possible combos

$2 \cdot 2 \cdot 2 \cdot \ldots \cdot 2 \cdot 2 \cdot 2 = 2^D$

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!

Choose Num Features?

Clearly all subsets is unreasonable. How can we choose how many and which features to include?

Option 1

Greedy Algorithm

Option 2

LASSO Regression (L1 Regularization)

L2 => Ridge

L1 => LASSO

Greedy Algorithms

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: Forward Stepwise

Start by selecting number of desired features k

```
min_val = ∞
S₀ ← Ø
for i ← 1..k:
    Find feature fᵢ not in Sᵢ₋₁, that when combined
    with Sᵢ₋₁, minimizes the validation loss the most.
    Sᵢ ← Sᵢ₋₁ ∪ {fᵢ}
    if val_loss(Sᵢ) > min_val:
        break  # No need to look at more features
```

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

- Greedily optimal != Global optimum

slido

Think

1 min

sli.do #cs416

Say you want to find the optimal two-feature model, using the forward stepwise algorithm. What model would the forward stepwise algorithm choose?

Subsets of Size 1

Features	Val Loss
# bath	201
# bed	300
sq ft	157
year built	224

$S_1 = \{\text{sq ft}\}$

Subsets of Size 2

Features *(unordered)*	Val Loss
(# bath, # bed)	120
(# bath, sq ft)	131
(# bath, year built)	190
(# bed, sq ft)	137
(# bed, year built)	209
(sq ft, year built)	145

slido

Group

1 min

sli.do #cs416

Say you want to find the optimal two-feature model, using the forward stepwise algorithm. What model would the forward stepwise algorithm choose?

Subsets of Size 1

Features	**Val Loss**
# bath	201
# bed	300
sq ft	157
year built	224

Subsets of Size 2

Features *(unordered)*	**Val Loss**
(# bath, # bed)	120
(# bath, sq ft)	131
(# bath, year built)	190
(# bed, sq ft)	137
(# bed, year built)	209
(sq ft, year built)	145

Brain Break

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

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

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?

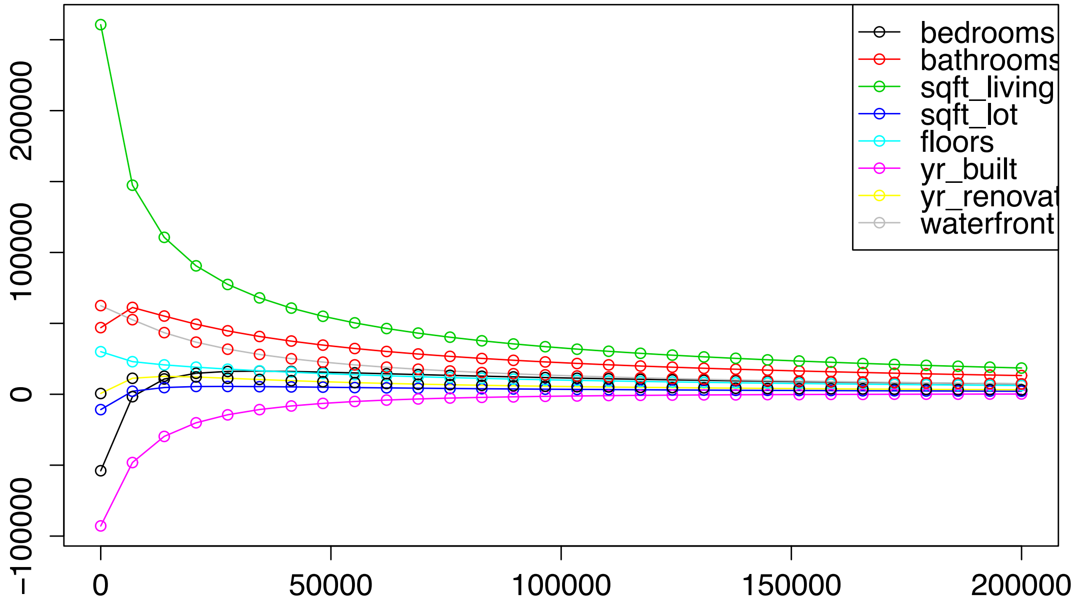
$$R(w) = |w_0|^2 + |w_1|^2 + \ldots + |w_d|^2 = w_0^2 + w_1^2 + \ldots + w_d^2 = \|w\|_2^2$$

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

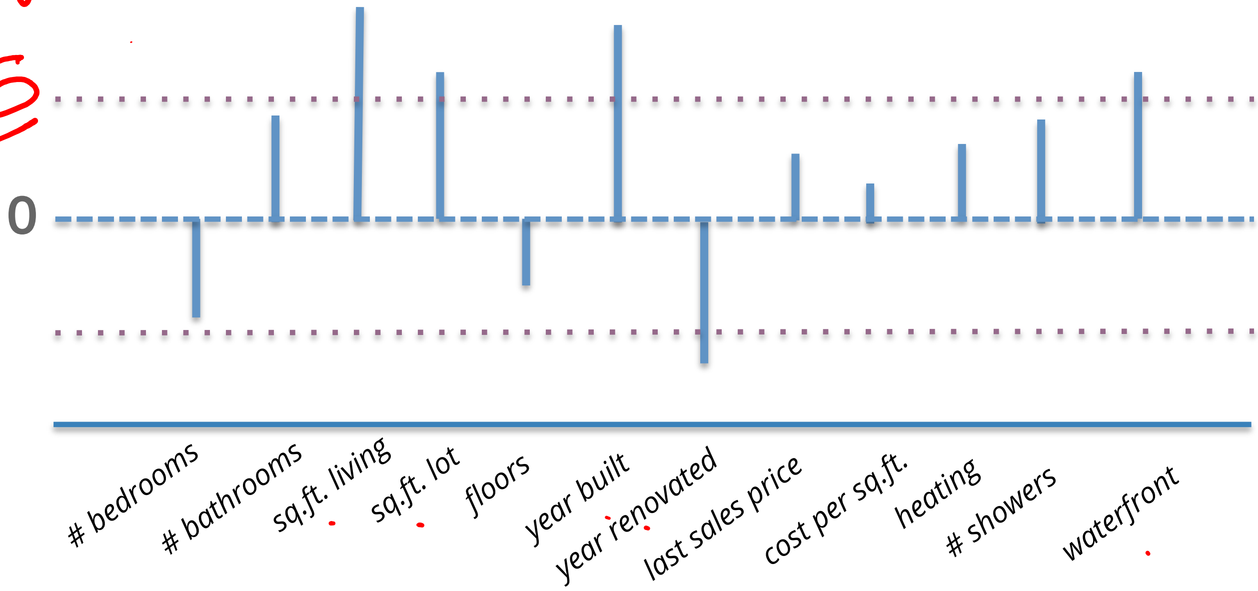
Instead of searching over a **discrete** set of solutions, use regularization to reduce coefficient of unhelpful features.

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

Non-zero coefficients would be considered selected as important.

threshold

0.5



Group

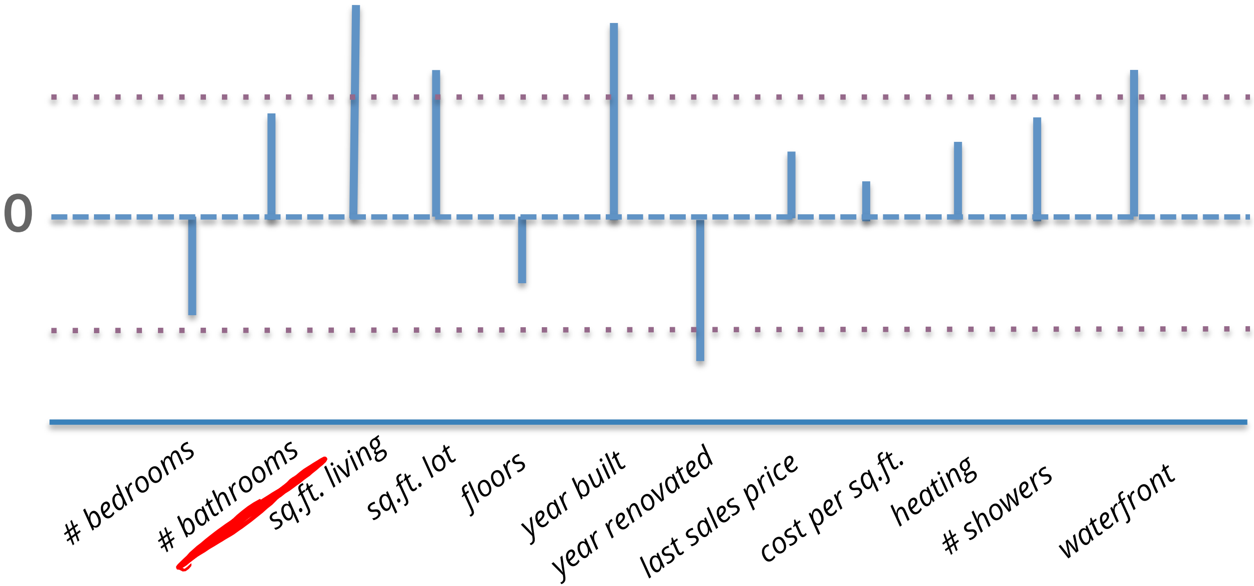
1 min

What do you think about this approach to feature selection? Will it work? Why or why not. Use your logic!

Ridge for Feature Selection

Look at two related features #bathrooms and # showers.

Our model ended up not choosing any features about bathrooms!

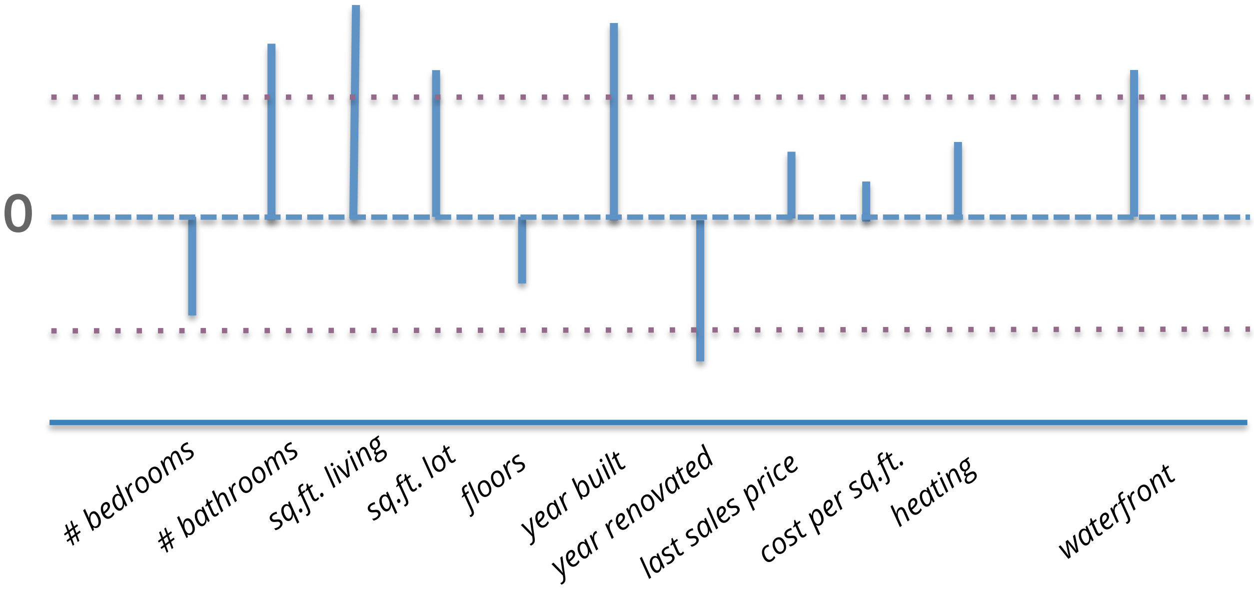


Ridge for Feature Selection

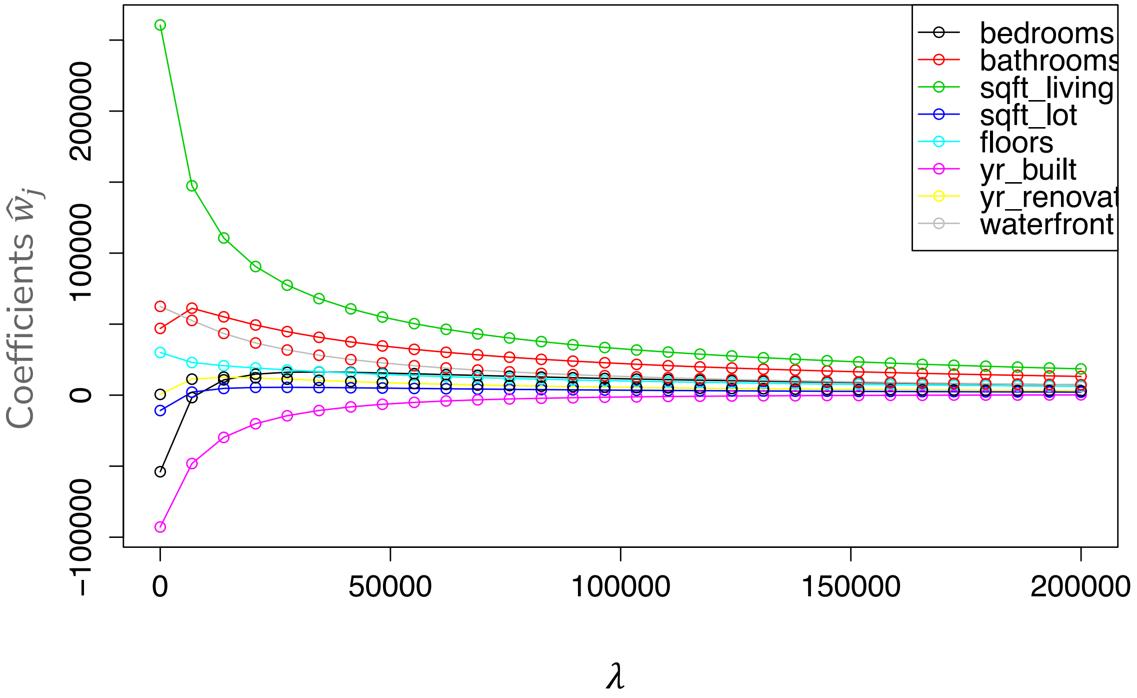
What if we had originally removed the # showers feature?

The coefficient for # bathrooms would be larger since it wasn't "split up" amongst two correlated features

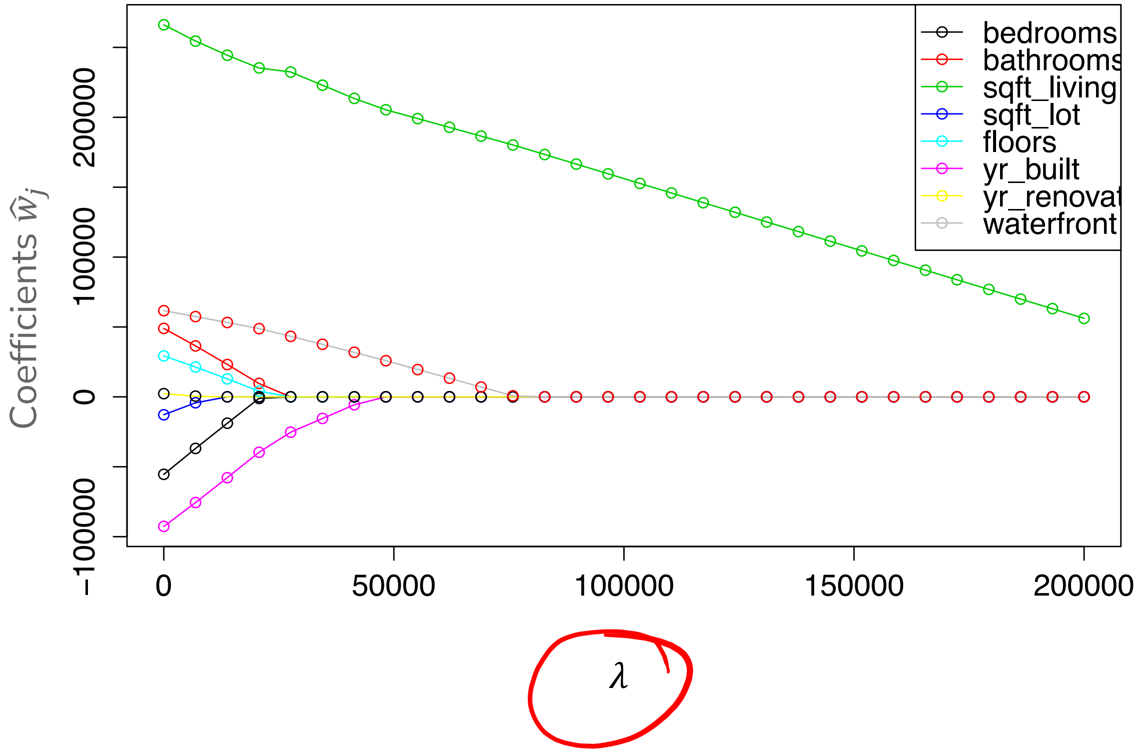
Instead, it would be nice if there were a regularizer that favors sparse solutions in the first place to account for this...



Ridge (L2) Coefficient Paths

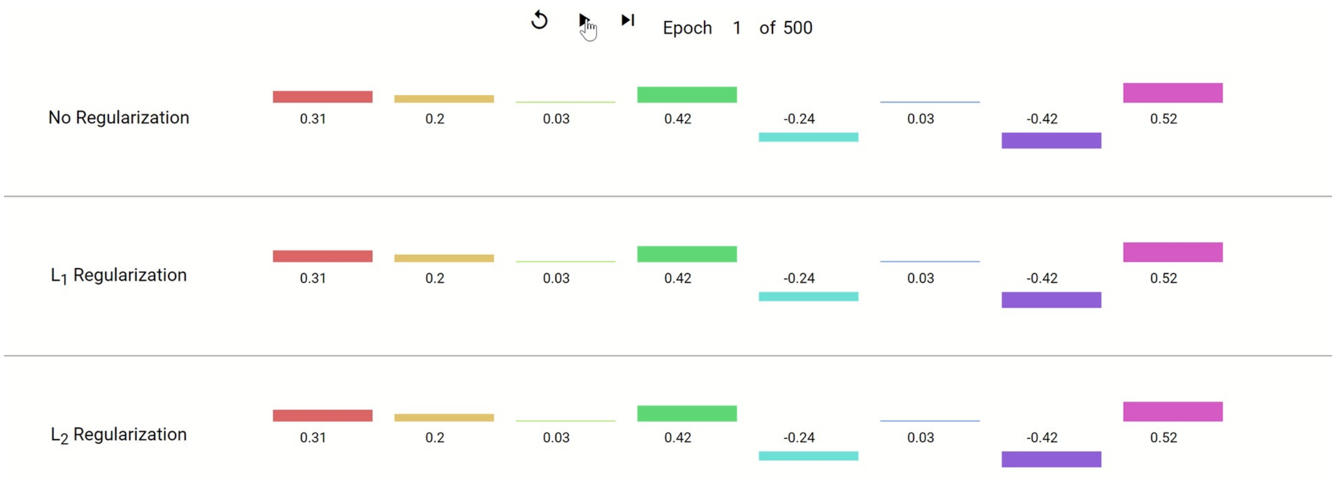


LASSO (L1) Coefficient Paths



Coefficient Paths – Another View

Example from Google's Machine Learning Crash Course



Demo

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

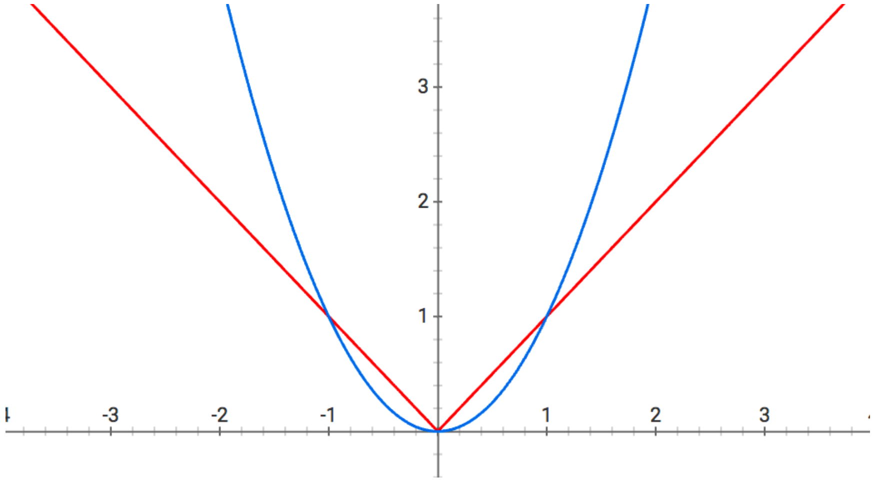
slido

Group

2 minutes

No poll

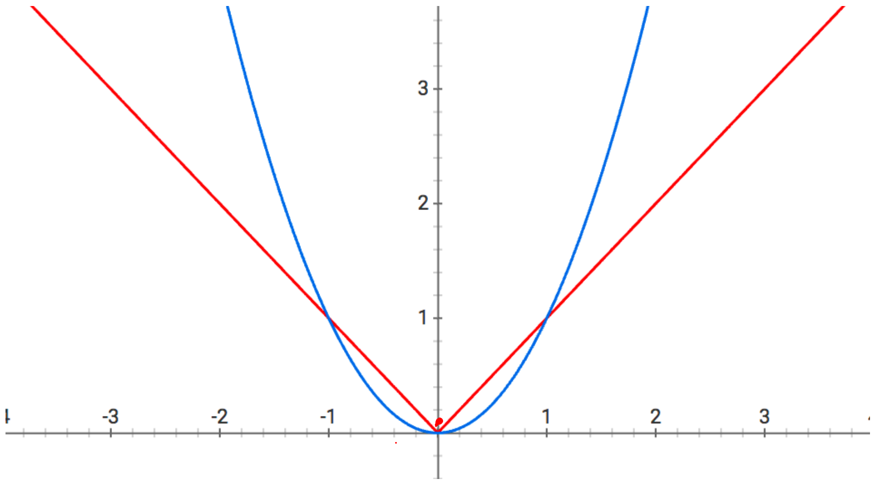
Why might the shape of the L1 penalty cause more sparsity than the L2 penalty?



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.

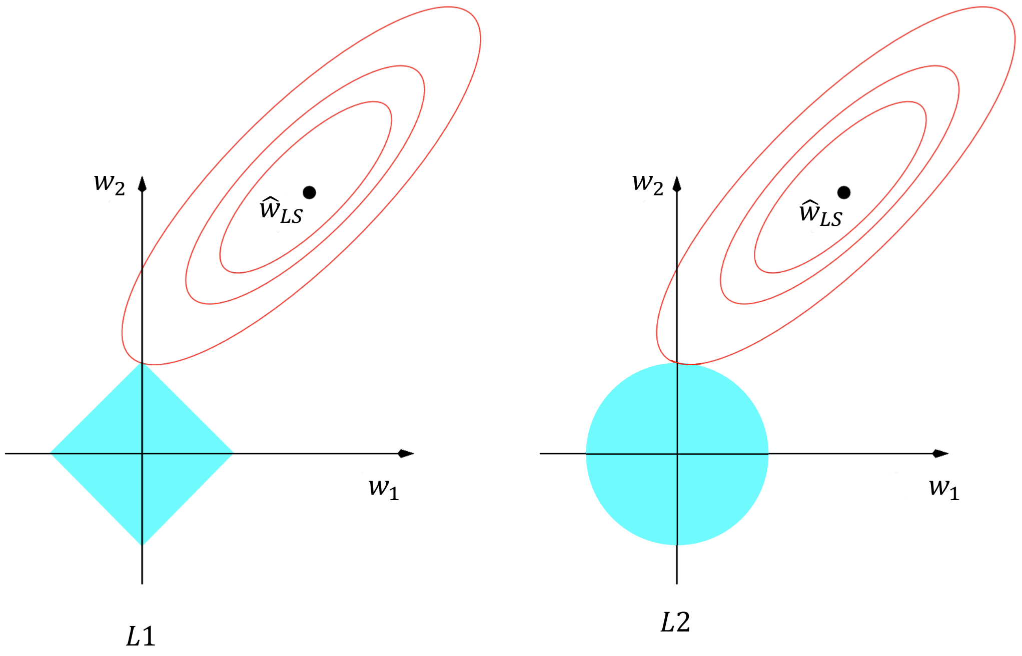
This has to do with the shape of the norm



When w_j is small, w_j^2 is VERY small! Diminishing returns on decreasing w_j with Ridge penalty

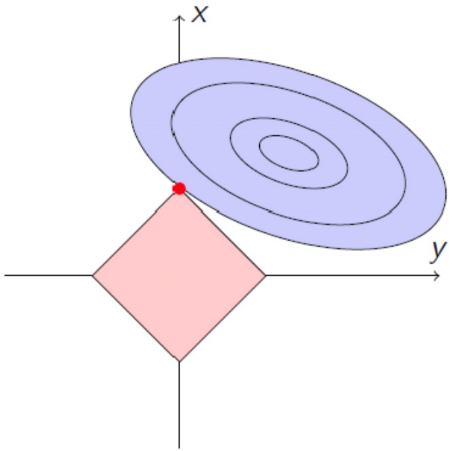
Sparsity Geometry

Another way to visualize why LASSO prefers sparse solutions

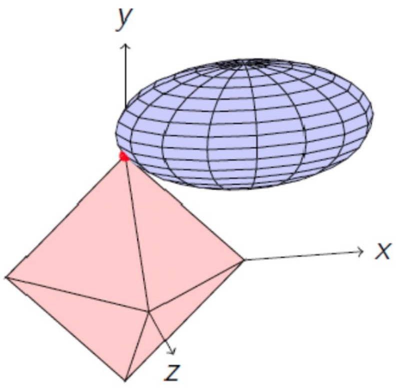


The L1 ball has spikes (places where some coefficients are 0)

Sparsity Geometry



$L1$ (2 features)



$L1$ (3 features)

Brain Break

slido

Think

1 min

sli.do #cs416

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

a) Pick the λ that has the smallest $MSE(\hat{w})$ on the **validation set**

b) Pick the λ that has the smallest $MSE(\hat{w}) + \lambda||\hat{w}||_2^2$ on the **validation set**

c) Pick the λ that results in the most zero coefficients

d) Pick the λ that results in the fewest zero coefficients

e) 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

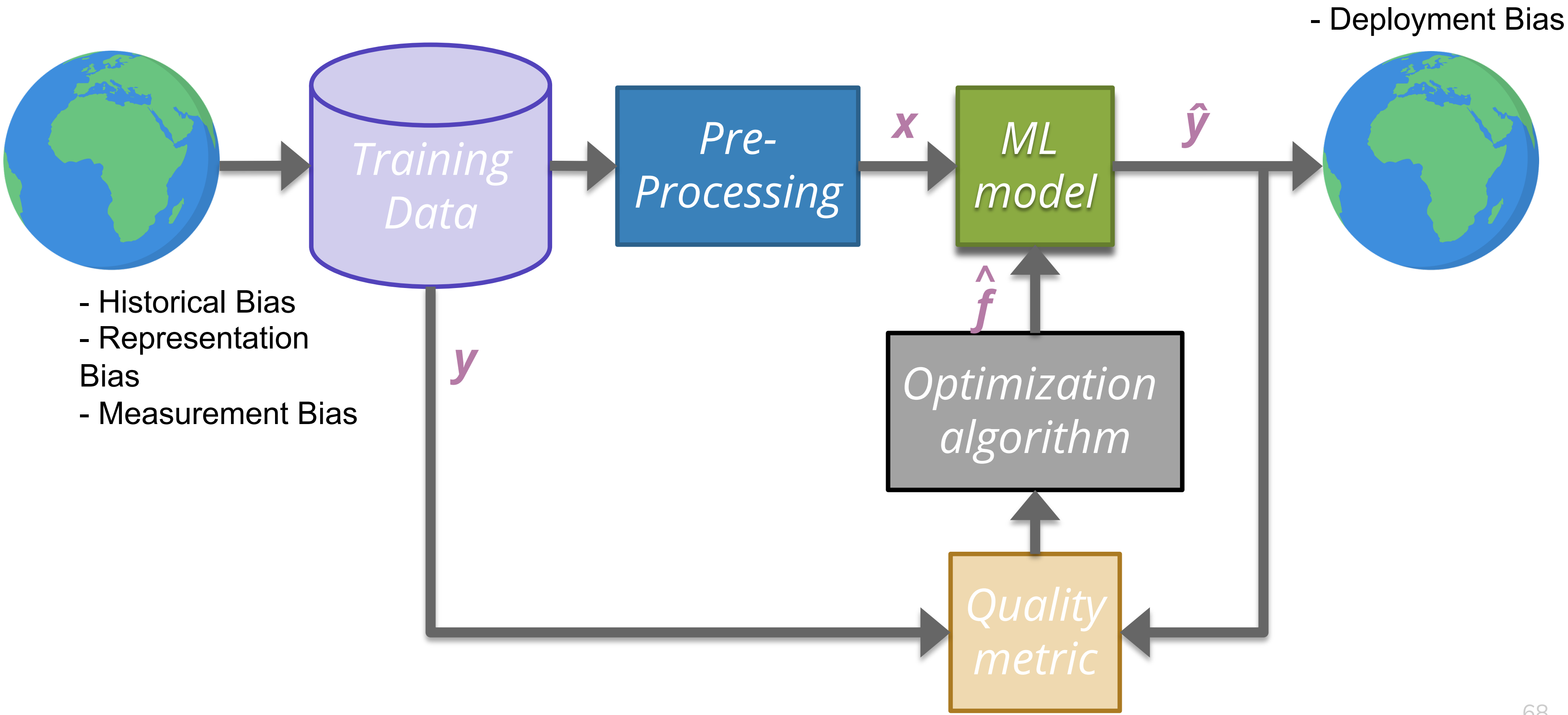
LASSO in Practice

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*

* e.g., using domain-specific expertise

ML Pipeline



- Deployment Bias
Training Data
Pre-Processing
x
ML model
ŷ
f̂
y
Optimization algorithm
Quality metric
- Historical Bias
- Representation Bias
- Measurement Bias

De-biasing LASSO

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

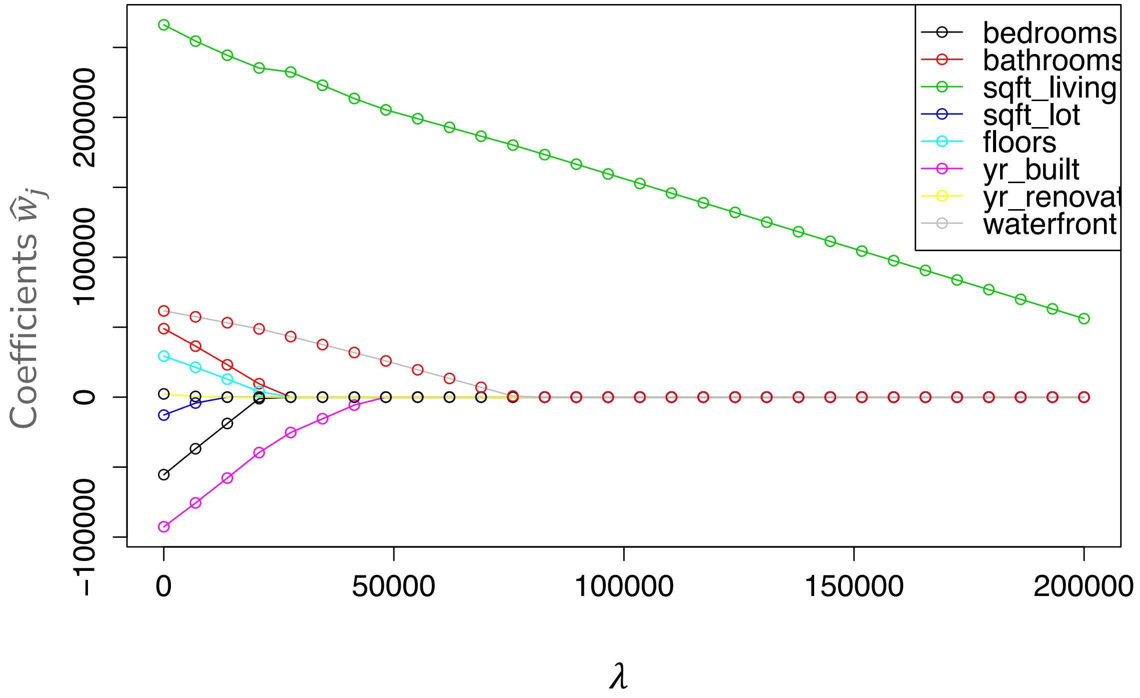
Recall Bias-Variance Tradeoff

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

1. Run LASSO to select 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

LASSO (L1) Coefficient Paths



(De-biased) LASSO In Practice

1. Split the dataset into train, val, and test sets
2. Normalize features. Fit the normalization on the train set, apply that normalization on the train, val, and test sets.
3. Use validation or cross-validation to find the value of λ that that results in a LASSO model with the lowest validation error.
4. Select the features of that model that have non-zero weights.
5. Train a Linear Regression model with only those features.
6. Evaluate on the test set.

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

$$\hat{w}_{ElasticNet} = \underset{w}{\operatorname{argmin}}\, MSE(w) + \lambda_1 \left|\left|w\right|\right|_1 + \lambda_2 \left|\left|w\right|\right|_2^2$$

Combines both to achieve best of both worlds!

A Big Grain of Salt

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

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

At the end of the day, the best models combine statistical insights with domain-specific expertise!

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

Recap

Theme: Using regularization to do feature selection

Ideas:

Describe “all subsets” approach to feature selection and why it’s impractical to implement.

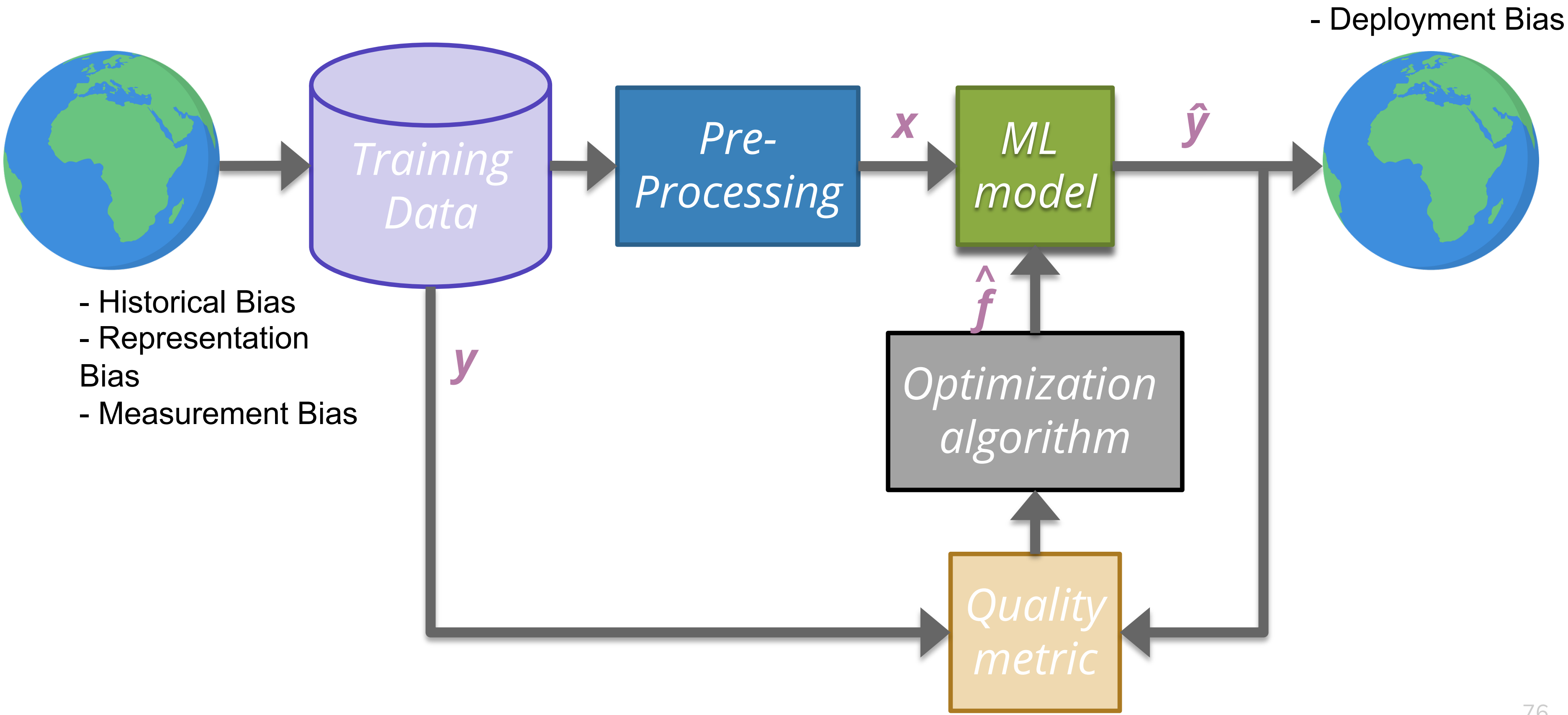
Formulate LASSO objective

Describe how LASSO coefficients change as hyper-parameter λ is varied

Interpret LASSO coefficient path plot

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

ML Pipeline


- Deployment Bias
Training Data
Pre-Processing
x
ML model
ŷ
- Historical Bias
- Representation Bias
- Measurement Bias
y
f̂
Optimization algorithm
Quality metric

CSE/STAT 416

Classification

Tanmay Shah
University of Washington
June 1, 2024

❓ **Questions?** Raise hand or **sli.do #cs416**
💬 **Before Class:** Does a straw have two holes or one?
🎵 **Listening to:** nothing, enjoy the calm

Roadmap So Far

1. Housing Prices - Regression
   - Regression Model
   - Assessing Performance
   - Ridge Regression
   - LASSO
2. Sentiment Analysis – Classification
   - Classification Overview
   - Logistic Regression

Regression vs. Classification

Regression problems involve predicting **continuous values**.

- E.g., house price, student grade, population growth, etc.

Classification problems involve predicting **discrete labels**

- e.g., spam detection, object detection, loan approval, etc.

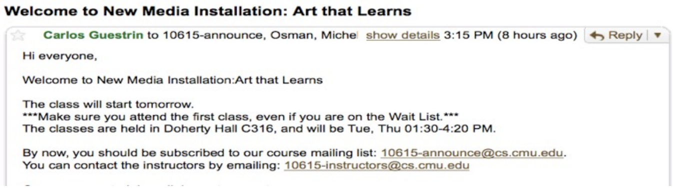
Spam Filtering


Osman Khan to Carlos show details Jan 7 (6 days ago) Reply

sounds good
+ok

Carlos Guestrin wrote:
Let's try to chat on Friday a little to coordinate and more on Sunday in person?

Carlos


Welcome to New Media Installation: Art that Learns

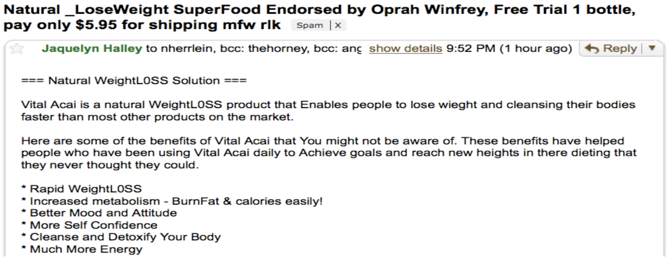
Carlos Guestrin to 10615-announce, Osman, Miche show details 3:15 PM (8 hours ago) Reply

Hi everyone,

Welcome to New Media Installation:Art that Learns

The class will start tomorrow.
Make sure you attend the first class, even if you are on the Wait List.
The classes are held in Doherty Hall C316, and will be Tue, Thu 01:30-4:20 PM.

By now, you should be subscribed to our course mailing list: 10615-announce@cs.cmu.edu.
You can contact the instructors by emailing: 10615-instructors@cs.cmu.edu


Natural _LoseWeight SuperFood Endorsed by Oprah Winfrey, Free Trial 1 bottle, pay only $5.95 for shipping mfw rlk Spam | x

Jaquelyn Halley to nherrlein, bcc: thehorney, bcc: ang show details 9:52 PM (1 hour ago) Reply

=== Natural WeightL0SS Solution ===

Vital Acai is a natural WeightL0SS product that Enables people to lose wieght and cleansing their bodies faster than most other products on the market.

Here are some of the benefits of Vital Acai that You might not be aware of. These benefits have helped people who have been using Vital Acai daily to Achieve goals and reach new heights in there dieting that they never thought they could.

* Rapid WeightL0SS
* Increased metabolism - BurnFat & calories easily!
* Better Mood and Attitude
* More Self Confidence
* Cleanse and Detoxify Your Body
* Much More Energy

Output: y

Spam

Not Spam (ham)

Input: x

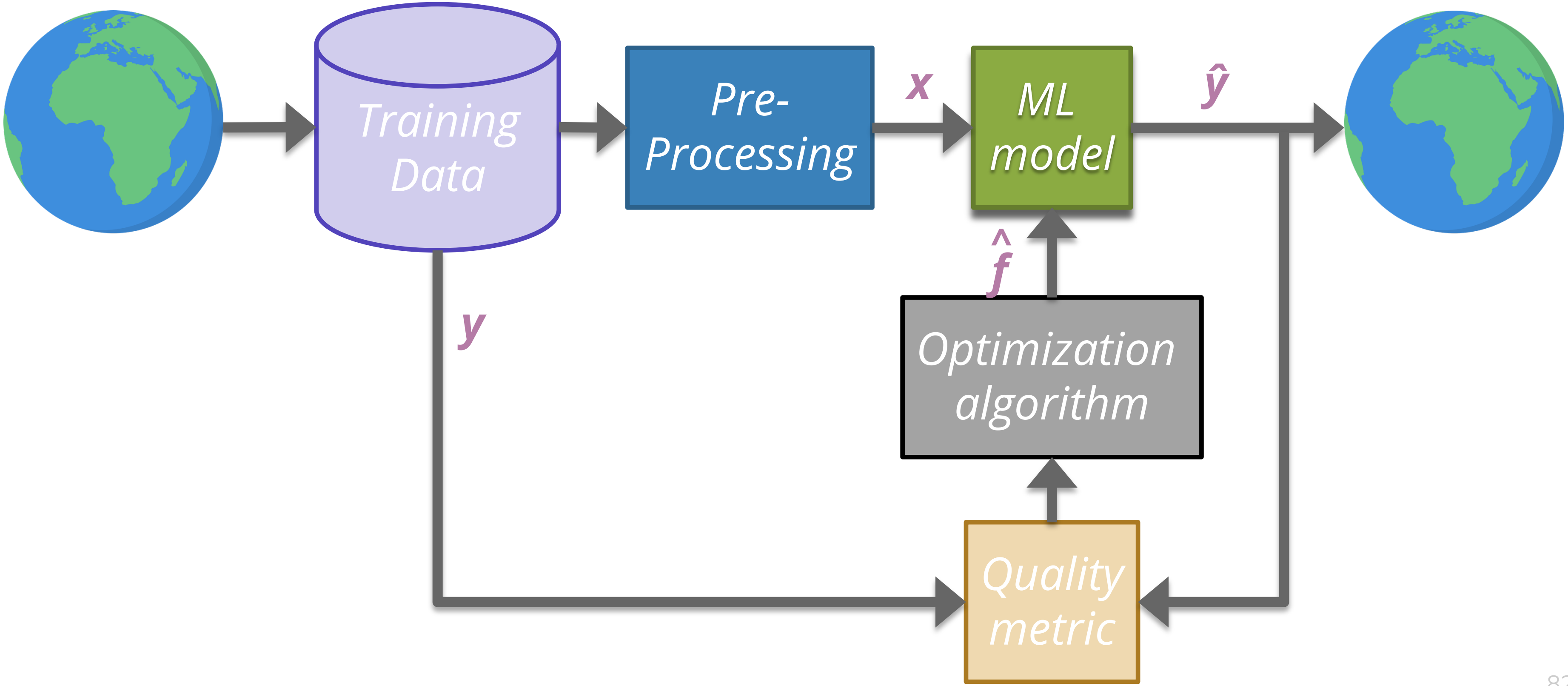
Text of email
Sender
Subject
...

Object Detection

Input: x
Pixels

Output: y
Class
(+ Probability)

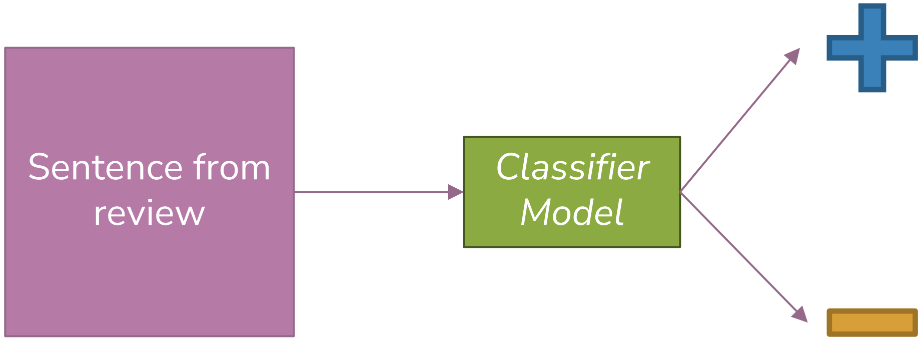
ML Pipeline



Training Data
Pre-Processing
x
ML model
ŷ
f̂
y
Optimization algorithm
Quality metric

Sentiment Classifier

In our example, we want to classify a restaurant review as positive or negative.



Input: x

Output: y
Predicted class

Converting Text to Numbers (Vectorizing):

Bag of Words

Idea: One feature per word!

Example: ”Sushi was great, the food was awesome, but the service was terrible”

sushi	was	great	the	food	awesome	but	service	terrible

This **has** to be too simple, right?

Stay tuned (today and Wed) for issues that arise and how to address them ☺

Pre-Processing: Sample Dataset

Review	Sentiment
"Sushi was great, the food was awesome, but the service was terrible"	+1
…	…
"Terrible food; the sushi was rancid."	-1

Vectorizer

Sushi	was	great	the	food	awesome	but	service	terrible	rancid	Sentiment
1	3	1	2	1	1	1	1	1	0	+1
…	…	…	…	…	…	…	…	…	…	…
1	1	0	1	1	0	0	0	1	1	-1

How to Implement Sentiment Analysis?

Attempt 1: Simple Threshold Analysis

Attempt 2: Linear Classifier

Attempt 3 (Wed): Logistic Regression

Attempt 1: Simple Threshold Classifier

Idea: Use a list of good words and bad words, classify review by the most frequent type of word

Word	Good?
sushi	None
was	None
great	Good
the	None
food	None
but	None
awesome	Good
service	None
terrible	Bad
rancid	Bad

Simple Threshold Classifier

Input x: Sentence from review

Count the number of positive and negative words, in x

If $\text{num\_positive} > \text{num\_negative}$:

- $\hat{y} = +1$

Else:

- $\hat{y} = -1$

Example: "Sushi was great, the food was awesome, but the service was terrible"

Limitations of Attempt 1 (Simple Threshold Classifier)

Words have different degrees of sentiment.

- Awesome > Great
- How can we weigh them differently?

Single words are not enough sometimes…

- “Good” → Positive
- “Not Good” → Negative

How do we get list of positive/negative words?

Words Have Different Degrees of Sentiments

What if we generalize good/bad to a numeric weighting per word?

Word	Good?
sushi	None
was	None
great	Good
the	None
food	None
but	None
awesome	Good
service	None
terrible	Bad
rancid	Bad

Word	Weight
sushi	0
was	0
great	1
the	0
food	0
but	0
awesome	2
service	0
terrible	-1
rancid	-2

How do we get the word weights?

What if we learn them from the data?

$h_1(x)$	$h_2(x)$	$h_3(x)$	$h_4(x)$	$h_5(x)$	$h_6(x)$	$h_7(x)$	$h_8(x)$	$h_9(x)$
sushi	**was**	**great**	**the**	**food**	**awesome**	**but**	**service**	**terrible**
1	3	1	2	1	1	1	1	1

In linear regression we learnt the weights for each feature. Can we do something similar here?

Word	**Weight**
sushi	w_1
was	w_2
great	w_3
the	w_4
food	w_5
awesome	w_6
but	w_7
service	w_8
terrible	w_9

Attempt 2: Linear Classifier

Idea: Use labelled training data to learn a weight for each word. Use weights to score a sentence.

Model:

$$\hat{y}_i = sign\big(Score(x_i)\big) = sign(s_i)$$

$$= sign\left(\sum_{j=0}^{D} w_j h_j(x_i)\right) = sign(w^T h(x_i))$$

Word	Weight
sushi	0
was	0
great	1
the	0
food	0
awesome	2
but	0
service	0
terrible	-1

$h_1(x)$	$h_2(x)$	$h_3(x)$	$h_4(x)$	$h_5(x)$	$h_6(x)$	$h_7(x)$	$h_8(x)$	$h_9(x)$
sushi	**was**	**great**	**the**	**food**	**awesome**	**but**	**service**	**terrible**
1	3	1	2	1	1	1	1	1

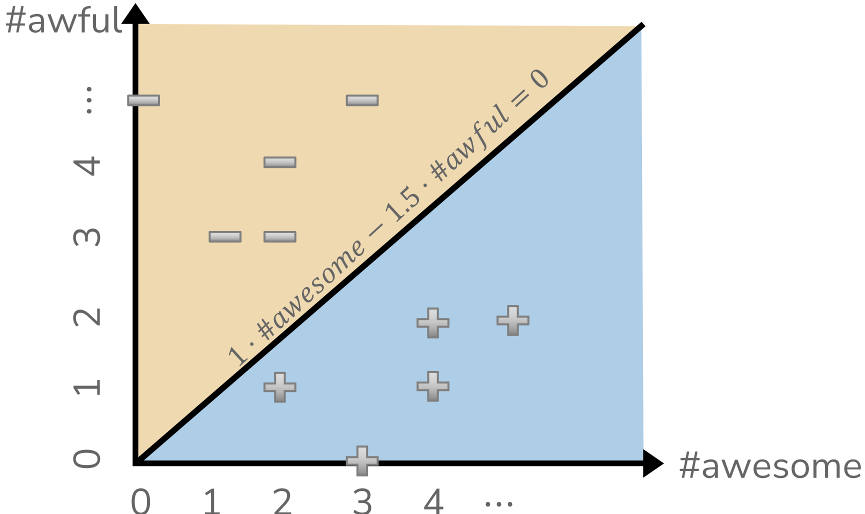
"Sushi was great, the food was awesome, but the service was terrible"

Decision Boundary

Consider if only two words had non-zero coefficients

Word	Coefficient	Weight
	w_0	0.0
awesome	w_1	1.0
awful	w_2	-1.5

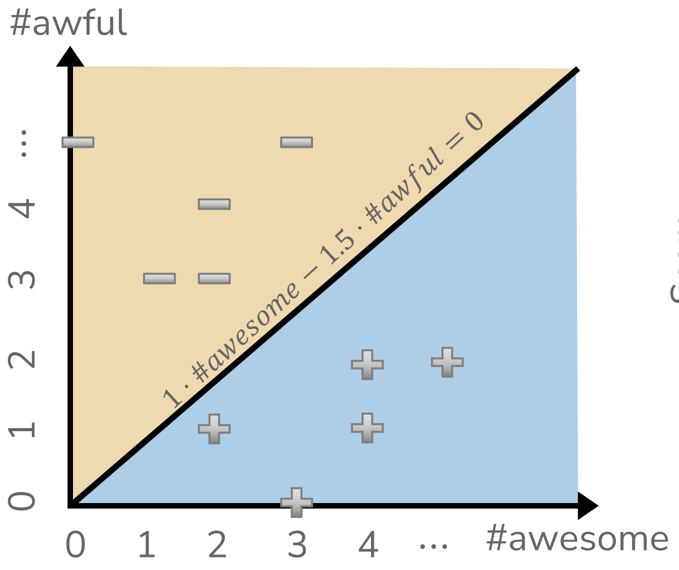
$$\hat{s} = 1 \cdot \#awesome - 1.5 \cdot \#awful$$



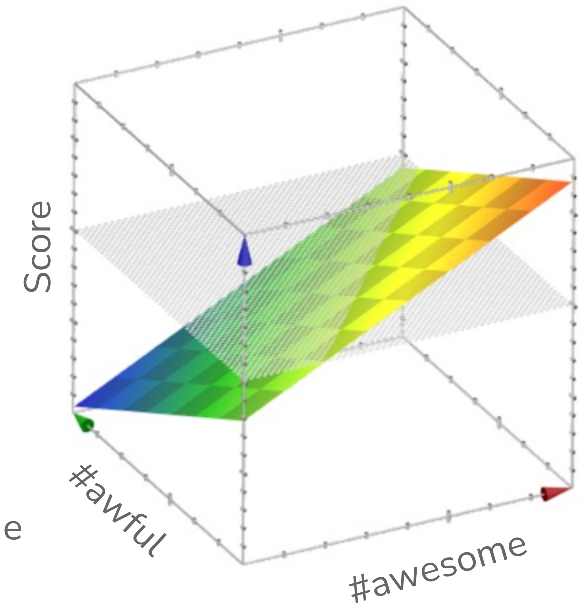
Decision Boundary

$$Score(x) = 1 \cdot \#awesome - 1.5 \cdot \#awful$$

2-dimensional view



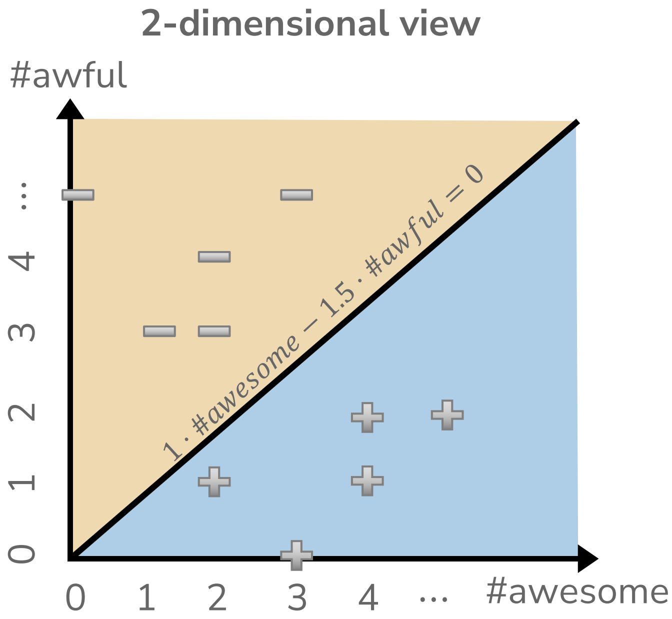
3-dimensional view

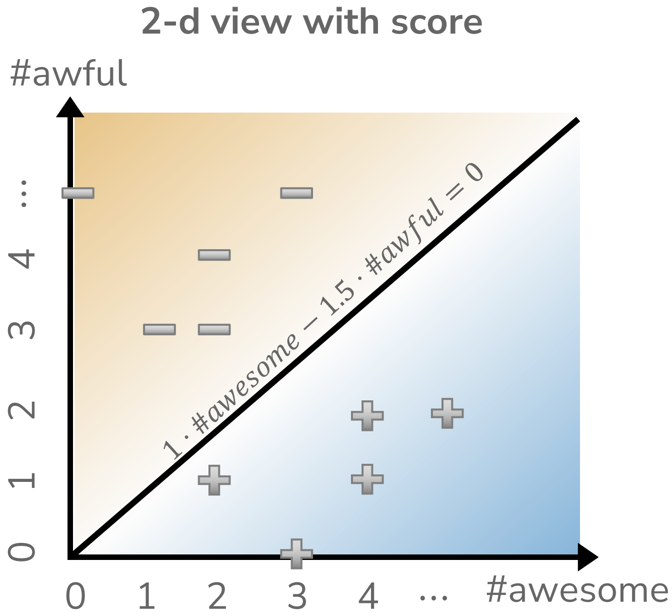


Generally, with classification we don't us a plot like the 3d view since it's hard to visualize, instead use 2d plot with decision boundary

Decision Boundary with Score

$$Score(x) = 1 \cdot \#awesome - 1.5 \cdot \#awful$$



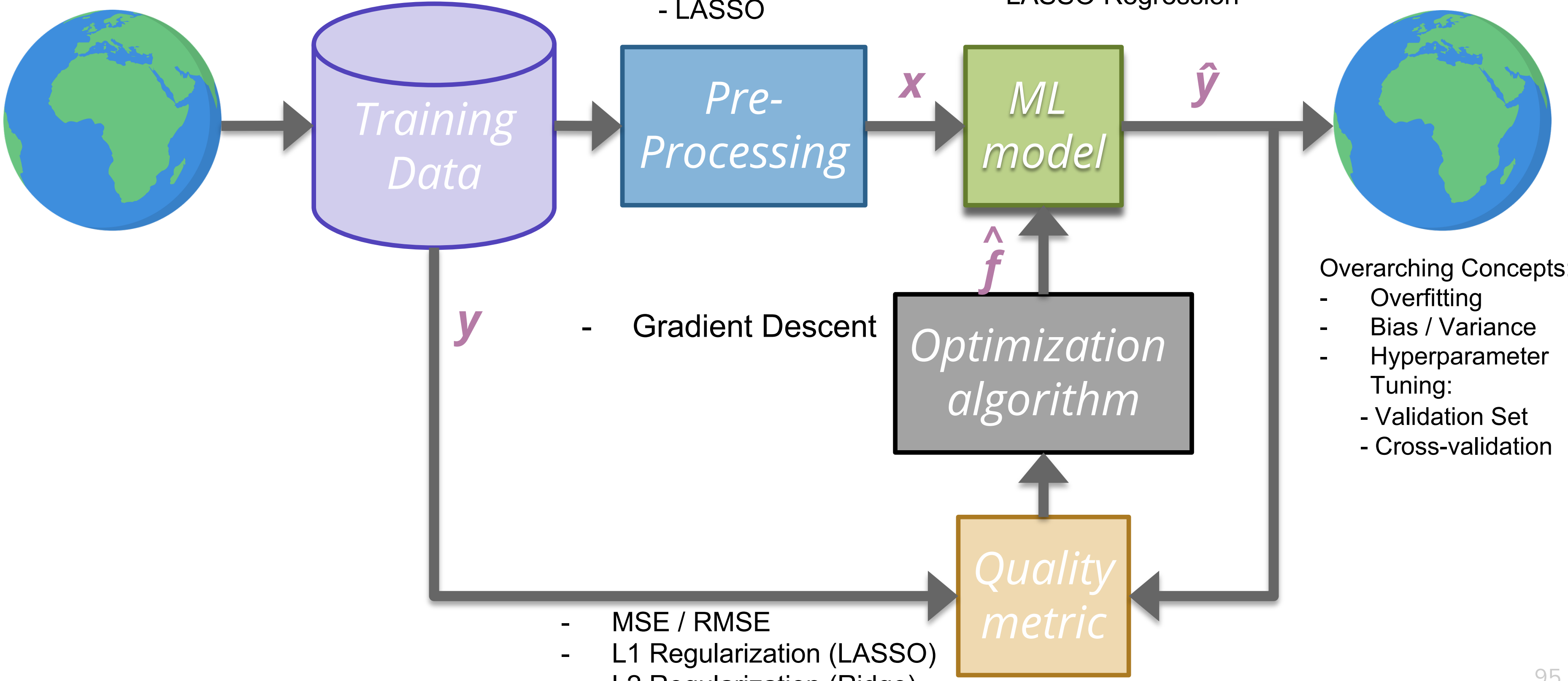


ML Pipeline

- Train/val/test -split

- Scaling / Normalization
- Feature Selection
  - All Subsets
  - Greedy
  - LASSO

- Linear Regression
- Polynomial Regression
- Ridge Regression
- LASSO Regression



- Gradient Descent

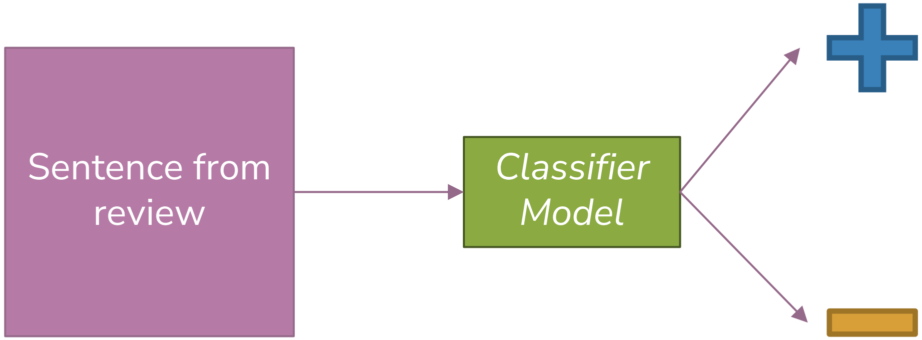
- MSE / RMSE
- L1 Regularization (LASSO)
- L2 Regularization (Ridge)

Overarching Concepts:
- Overfitting
- Bias / Variance
- Hyperparameter Tuning:
  - Validation Set
  - Cross-validation

Classification

Sentiment Classifier

In our example, we want to classify a restaurant review as positive or negative.



Input: x

Output: y
Predicted class

Attempt 1: Simple Threshold Classifier

Idea: Use a list of good words and bad words, classify review by the most frequent type of word

Word	**Good?**
sushi	None
was	None
great	Good
the	None
food	None
but	None
awesome	Good
service	None
terrible	Bad
rancid	Bad

Simple Threshold Classifier

Input x: Sentence from review

Count the number of positive and negative words, in x

If $\text{num\_positive} > \text{num\_negative}$:

- $\hat{y} = +1$

Else:

- $\hat{y} = -1$

+

Example: "Sushi was great, the food was awesome, but the service was terrible"

+ -

2+ 1- → +ve

Attempt 2: Linear Classifier

Idea: Use labelled training data to learn a weight for each word. Use weights to score a sentence.

$\mathrm{sign}(2) = +ve$

Model:

$$\hat{y}_i = sign\big(Score(x_i)\big) = sign(s_i)$$

$$= sign\left(\sum_{j=0}^{D} w_j h_j(x_i)\right) = sign(w^T h(x_i))$$

$h_1(x)$	$h_2(x)$	$h_3(x)$	$h_4(x)$	$h_5(x)$	$h_6(x)$	$h_7(x)$	$h_8(x)$	$h_9(x)$
sushi	**was**	**great**	**the**	**food**	**awesome**	**but**	**service**	**terrible**
1	3	1	2	1	1	1	1	1

Word	**Weight**
sushi	0
was	0
great	1
the	0
food	0
awesome	2
but	0
service	0
terrible	-1

"Sushi was great, the food was awesome, but the service was terrible"

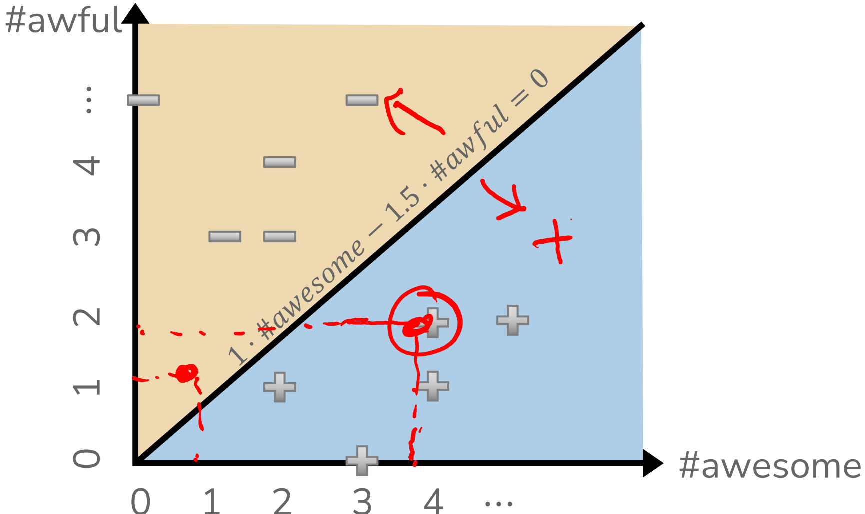
$1 \cdot 1 + 1 \cdot 2 + 1 \cdot (-1) = 2 \rightarrow +ve$

Decision Boundary

Consider if only two words had non-zero coefficients

Word	*Coefficient*	*Weight*
	w_0	*0.0*
awesome	w_1	*1.0*
awful	w_2	*-1.5*

$$\hat{s} = 1 \cdot \#awesome - 1.5 \cdot \#awful$$



Poll Everywhere

Think

1 min

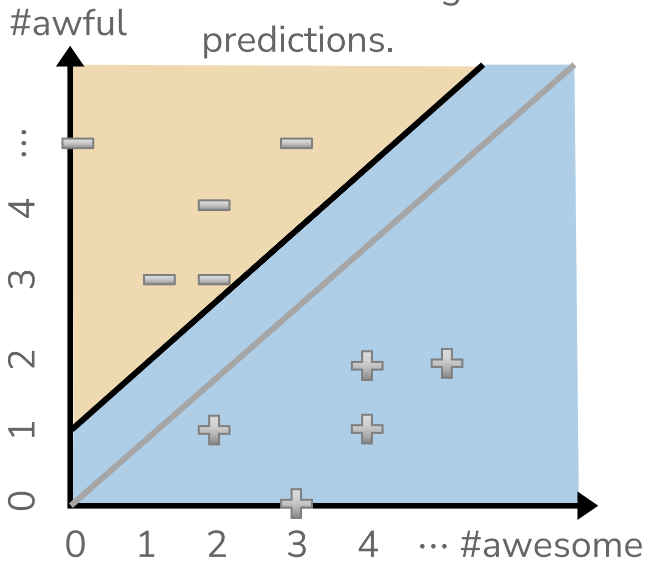
pollev.com/cs416

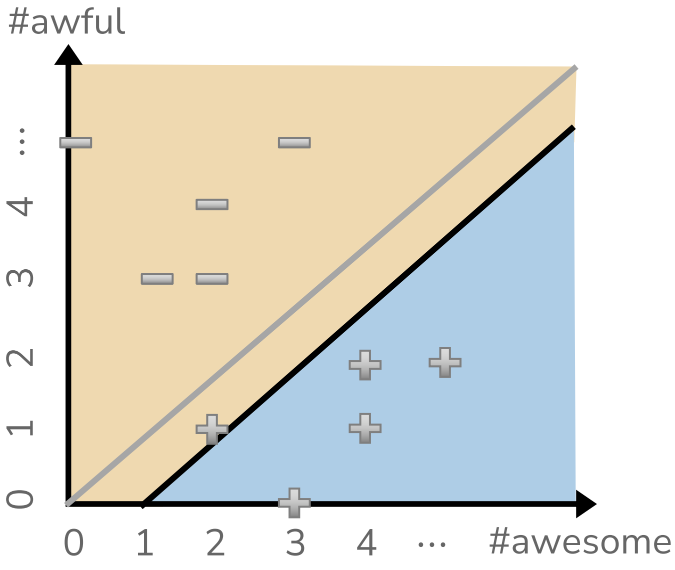
What happens to the decision boundary if we add an intercept?

$$Score(x) = 1.0 + 1 \cdot \#awesome - 1.5 \cdot \#awful$$

Which graph shows the new decision boundary (black)?

Describe in English what we are doing in terms of the model's predictions.





Poll Everywhere

Think

2 min

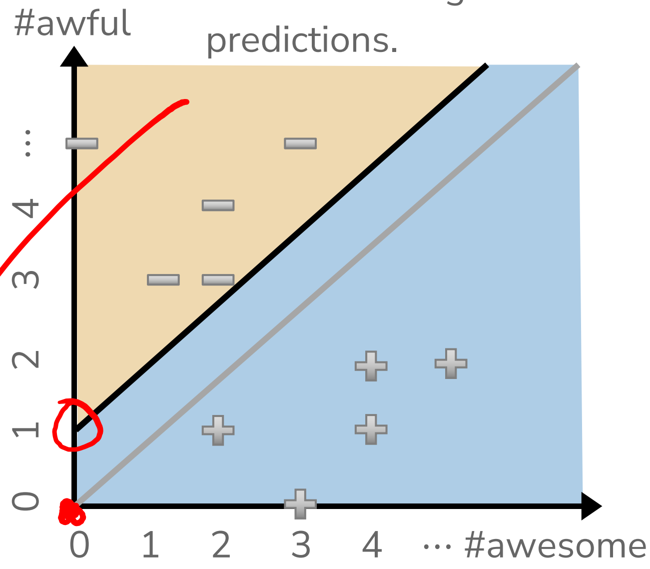
pollev.com/cs416

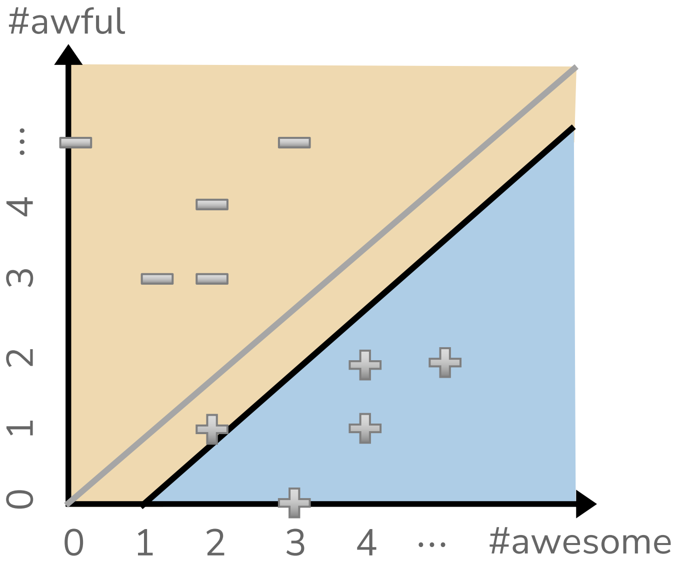
What happens to the decision boundary if we add an intercept?

$$Score(x) = 1.0 + 1 \cdot \#awesome - 1.5 \cdot \#awful$$

Which graph shows the new decision boundary (black)?

Describe in English what we are doing in terms of the model's predictions.

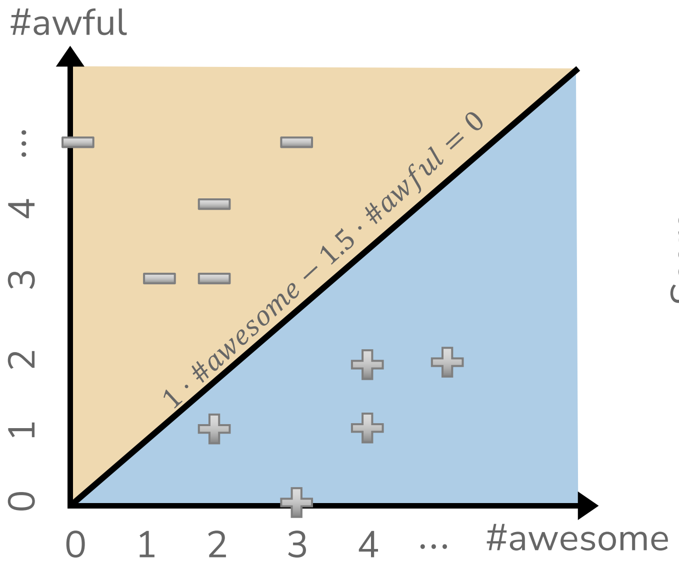




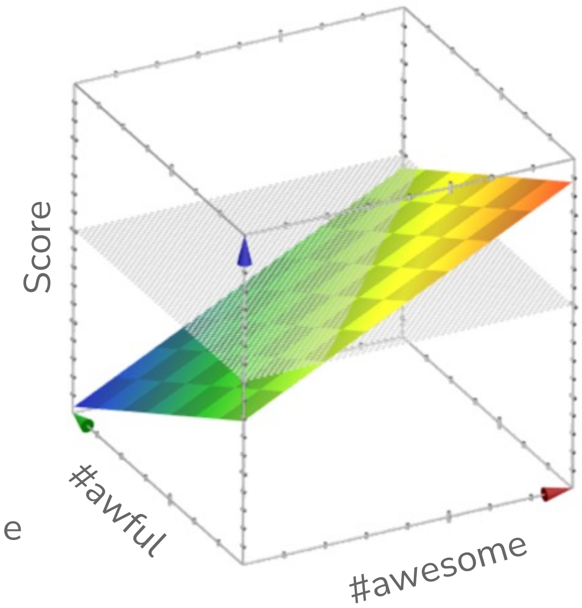
Decision Boundary

$$Score(x) = 1 \cdot \#awesome - 1.5 \cdot \#awful$$

2-dimensional view



3-dimensional view

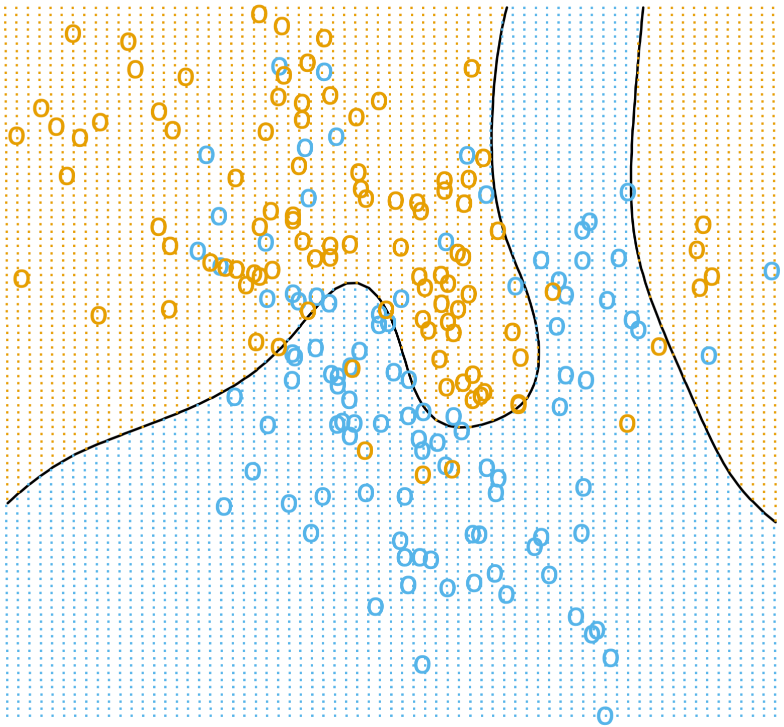


Generally, with classification we don't us a plot like the 3d view since it's hard to visualize, instead use 2d plot with decision boundary

Complex Decision Boundaries?

What if we want to use a more complex decision boundary?

- Need more complex model/features! (Come back Wed)



Single Words Are Sometimes Not Enough!

What if instead of making each feature one word, we made it two?

- **Unigram**: a sequence of one word
- **Bigram**: a sequence of two words
- **N-gram**: a sequence of n-words

"Sushi was good, the food was good, the service was not good"

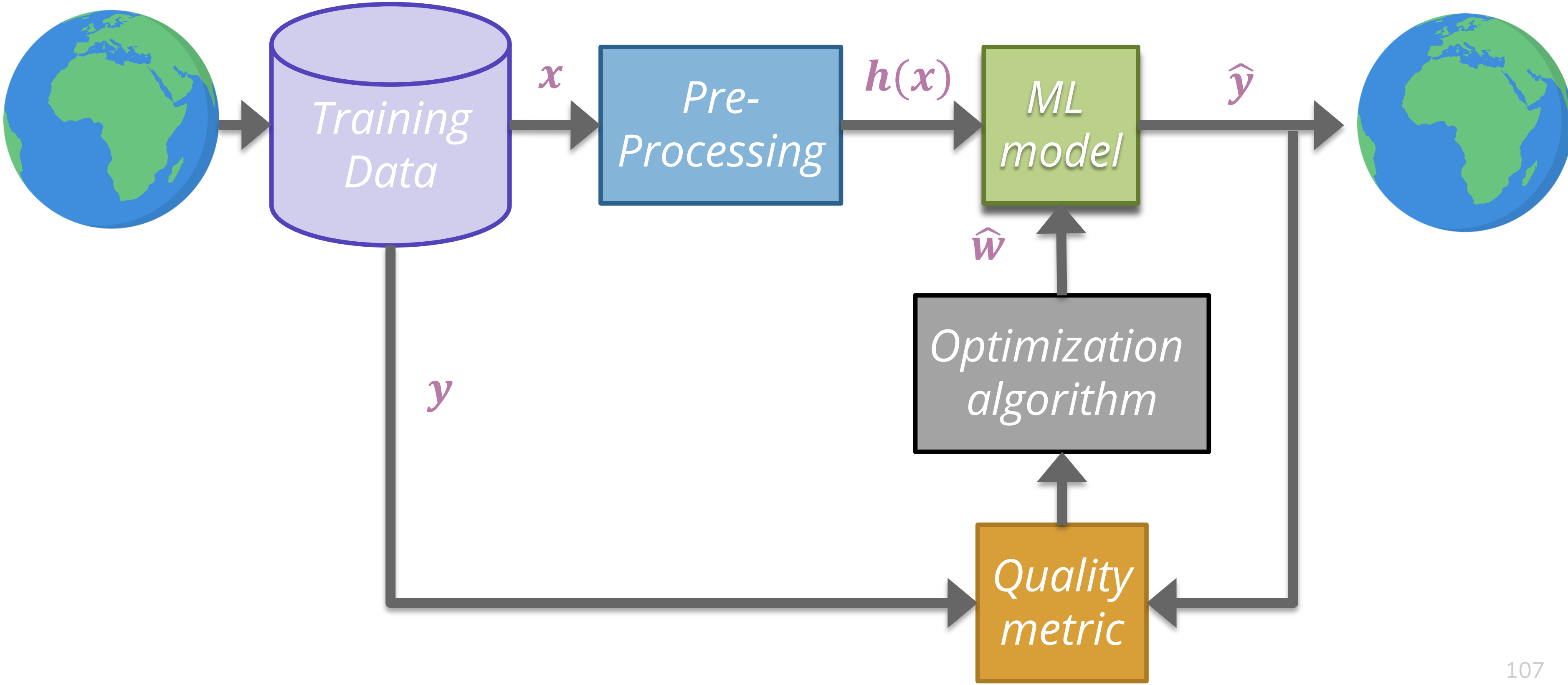
sushi	was	good	the	food	service	not
1	3	3	2	1	1	1

sushi was	was good	good the	the food	food was	the service	service was	was not	not good
1	2	2	1	1	1	1	1	1

Longer sequences of words results in more context, more features, and a greater chance of overfitting.

Evaluating Classifiers

ML Pipeline



Training Data
x
Pre-Processing
h(x)
ML model
ŷ
ŵ
Optimization algorithm
y
Quality metric

Classification Error

Ratio of examples where there was a mistaken prediction

What's a mistake?

If the true label was positive ($y = +1$), but we predicted negative ($\hat{y} = -1$)

If the true label was negative ($y = -1$), but we predicted positive ($\hat{y} = +1$)

Classification Error

$$\text{error} = \frac{1}{n} \sum_{i=1}^{n} \left[\mathbb{1}\{ y_i \neq \hat{y}_i \} \right]$$

Classification Accuracy

$$\text{accuracy} = 1 - \text{error}$$

What's a good accuracy?

For binary classification: 2 $\rightarrow \frac{1}{2}$

Should at least beat random guessing...

Accuracy should be at least 0.5

For multi-class classification (k classes):

Should still beat random guessing

Accuracy should be at least: $\boxed{1 / k}$

- 3-class: 0.33
- 4-class: 0.25
- ...

Besides that, higher accuracy means better, right?

Detecting Spam

Imagine I made a "Dummy Classifier" for detecting spam

The classifier ignores the input, and always predicts spam.

This actually results in 90% accuracy! Why?

- Most emails are spam…

This is called the **majority class classifier.**

A classifier as simple as the majority class classifier can have a high accuracy if there is a **class imbalance**.

A class imbalance is when one class appears much more frequently than another in the dataset

This might suggest that accuracy isn't enough to tell us if a model is a good model.

Assessing Accuracy

Always digging in and ask critical questions of your accuracy.

Is there a **class imbalance**?

How does it compare to a baseline approach?

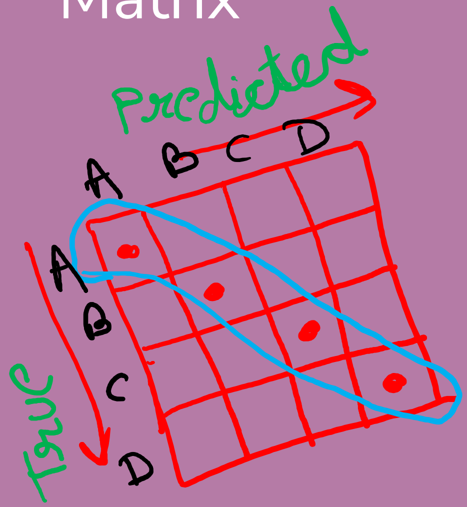
- Random guessing
- Majority class
- ...

Most important: **What does my application need?**

- What's good enough for user experience?
- What is the impact of a mistake we make?

Brain Break

Confusion Matrix



For binary classification, there are only two types of mistakes

$$\hat{y} = +1, \quad y = -1$$

$$\hat{y} = -1, \quad y = +1$$

Generally we make a **confusion matrix** to understand mistakes.

		Predicted Label	
		+	−
True Label	+	*True Positive (TP)*	*False Negative (FN)*
	−	*False Positive (FP)*	*True Negative (TN)*

Tip on remembering: complete the sentence “My prediction was a …”

Confusion Matrix Example

		Predicted Label	
		+	−
True Label	+	*True Positive (TP)*	*False Negative (FN)*
	−	*False Positive (FP)*	*True Negative (TN)*

Which is Worse?

What's worse, a false negative or a false positive?

It entirely depends on your application!

Detecting Spam

False Negative: Annoying

False Positive: Email lost

Medical Diagnosis

False Negative: Disease not treated

False Positive: Wasteful treatment

In almost every case, how treat errors depends on your context.

Errors and Fairness

We mentioned on the first day how ML is being used in many contexts that impact crucial aspects of our lives.

Models making errors is a given, what we do about that is a choice:

- Are the errors consequential enough that we shouldn't use a model in the first place?
- Do different demographic groups experience errors at different rates?
  - If so, we would hopefully want to avoid that model!

Will talk more about how to define whether or a not a model is fair / discriminatory next week. Will use these notions of error as a starting point!

Binary Classification Measures

Notation

$$C_{TP} = \#\text{TP},\quad \mathrm{C_{FP}} = \#\text{FP},\quad \mathrm{C_{TN}} = \#\text{TN},\quad \mathrm{C_{FN}} = \#\text{FN}$$

$$N = C_{TP} + C_{FP} + C_{TN} + C_{FN}$$

$$N_P = C_{TP} + C_{FN},\quad N_N = C_{FP} + C_{TN}$$

Error Rate

$$\frac{C_{FP} + C_{FN}}{N}$$

Accuracy Rate

$$\frac{C_{TP} + C_{TN}}{N}$$

False Positive rate (FPR)

$$\frac{C_{FP}}{N_N}$$

False Negative Rate (FNR)

$$\frac{C_{FN}}{N_P}$$

True Positive Rate or Recall

$$\frac{C_{TP}}{N_P}$$

Precision

$$\frac{C_{TP}}{C_{TP} + C_{FP}}$$

F1-Score

$$2\,\frac{Precision \cdot Recall}{Precison + Recall}$$

See more!

Multiclass Confusion Matrix

Consider predicting (*Healthy, Cold, Flu*)

Predicted Label

True Label	*Healthy*	*Cold*	*Flu*
Healthy	*60*	*8*	*2*
Cold	*4*	*12*	*4*
Flu	*0*	*2*	*8*

Poll Everywhere

Think

1 min

pollev.com/cs416

Suppose we trained a classifier and computed its confusion matrix on the training dataset. **Is there a class imbalance in the dataset and if so, which class has the highest representation?**

Predicted Label (columns) / **True Label** (rows)

	Pupper	*Doggo*	*Woofer*
Pupper	2	27	4
Doggo	4	25	4
Woofer	1	30	2

Poll Everywhere

Think

2 min

pollev.com/cs416

Suppose we trained a classifier and computed its confusion matrix on the training dataset. **Is there a class imbalance in the dataset and if so, which class has the highest representation?**

True Label \ Predicted Label	*Pupper*	*Doggo*	*Woofer*
Pupper	2	27	4
Doggo	4	25	4
Woofer	1	30	2

Learning Theory

How much data?

The more the merrier

- But data quality is also an extremely important factor

Theoretical techniques can bound how much data is needed

- Typically too loose for practical applications
- But does provide some theoretical guarantee

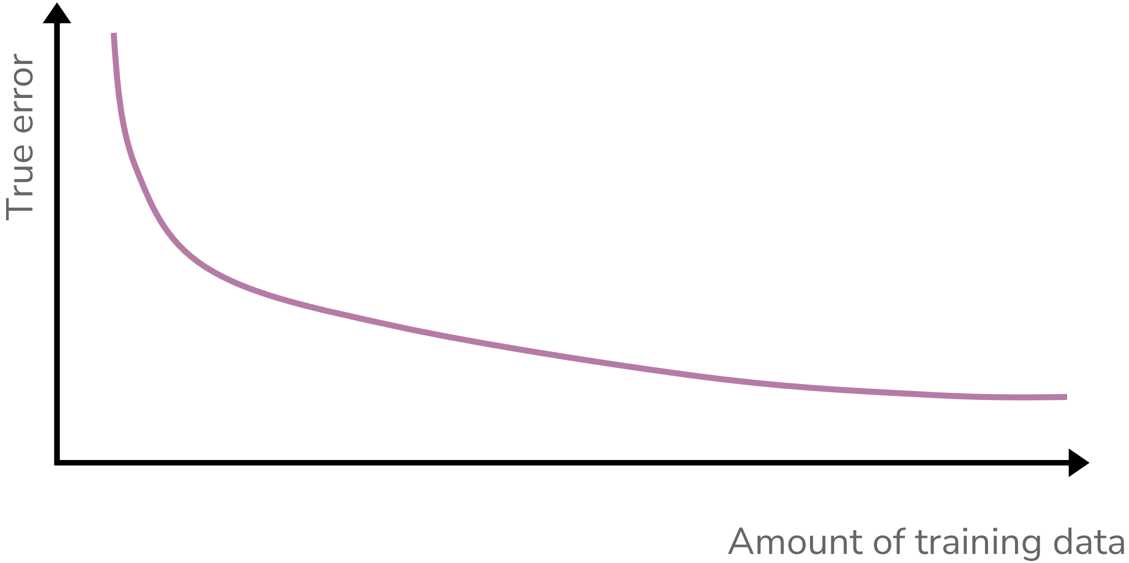
In practice

- More complex models need more data

Learning Curve

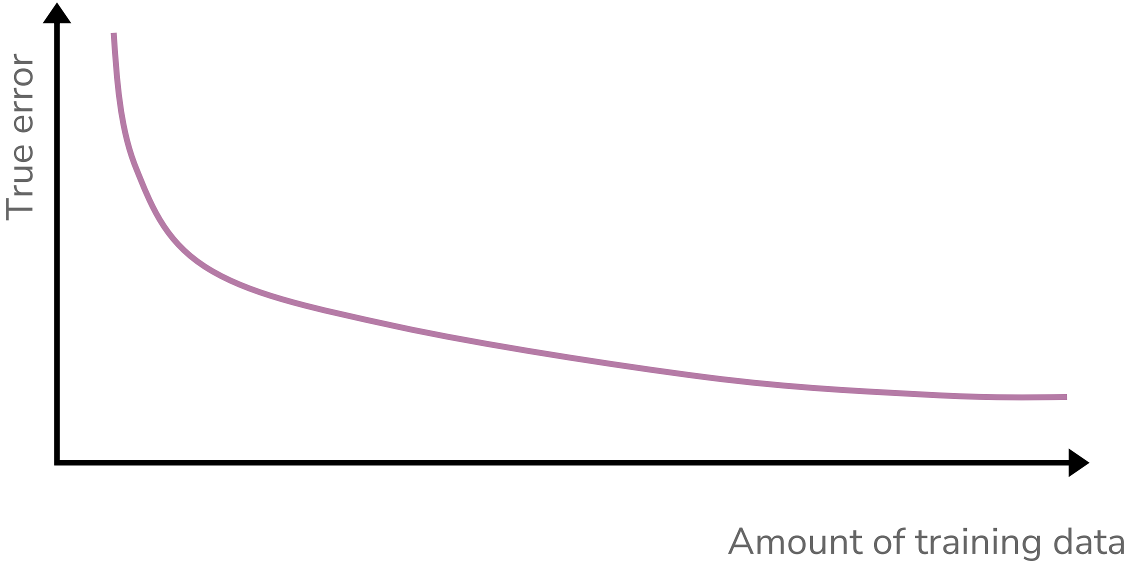
How does the true error of a model relate to the amount of training data we give it?

Hint: We've seen this picture before



Learning Curve

What if we use a more complex model?



Next Time

We will address the issues highlighted with the Linear Classifier approach from today by predicting the probability of a sentiment, rather than the sentiment itself.

$$P(y|x)$$

Normally assume some structure on the probability (e.g., linear)

$$P(y|x,w) \approx w^T x$$

Use machine learning algorithm to learn approximate $\hat{w}$ such that $\hat{P}(y|x)$ is close to $P(y|x)$, where:

$$\hat{P}(y|x) = P(y|x,\hat{w})$$

Recap

Theme: Describe high level idea and metrics for classification

Ideas:

- Applications of classification
- Linear classifier
- Decision boundaries
- Classification error / Classification accuracy
- Class imbalance
- Confusion matrix
- Learning theory