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.
Recap:
Ridge Regression

Change quality metric to minimize

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

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

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

$$\hat{w} = \min_w RSS(w)$$

This is called the least squares solution.

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

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

If $w = \hat{0}$ (all $w_j = 0$), then $RSS(w) + \lambda ||w||^2_2 = RSS(w)_{\text{least}}$

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

**$\lambda$ in between?**

$$0 \leq ||\hat{w}||^2_2 \leq ||\hat{w}_{\text{LS}}||^2_2$$
How should we choose the best value of $\lambda$?

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

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

$$\hat{\mathbf{w}}_{\text{ridge}} = \min_w \text{RSS}(\mathbf{w}) + \lambda \| \mathbf{w}_{1:D} \|^2_2$$

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

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 $\text{RSS}(\mathbf{w}) + \lambda \| \mathbf{w}_{1:D} \|^2_2$ 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

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

Compute validation error

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

Track $\lambda$ with smallest $\text{validation\_error}$

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

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
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

\[
\hat{y} = \sum_{\hat{w}_j \neq 0} \hat{w}_j h_j(x)
\]
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
...
How happy are you? What part of the brain controls happiness?
Best Model
Size 0

All Subsets

Noise only: $y_i = e_i$

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

RSS_{train} (\hat{w})

0

# of features
Best Model
Size 1

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

$\text{RSS}_{\text{train}}(\hat{w})$

# of features

0 1
Best Model
Size 1

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

\text{RSS}_{\text{train}}(\hat{w})

# of features

0 1
Best Model

Size 1

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

RSS_{\text{train}}(\hat{\theta}) vs. # of features
Best Model
Size 1

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

RSS_{train}(\hat{w})

# of features
Best Model
Size 1

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

RSS_{train}(\hat{w})

# of features

0 1
Best Model
Size 1

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

RSSt{\text{train}}(\vec{w})

# of features

0 1
Best Model
Size 1

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

### # of features

<table>
<thead>
<tr>
<th># of features</th>
<th>RSS_{\text{train}}(\hat{\omega})</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Features</th>
<th># of features</th>
</tr>
</thead>
<tbody>
<tr>
<td># bathrooms</td>
<td>0</td>
</tr>
<tr>
<td># bedrooms</td>
<td>1</td>
</tr>
<tr>
<td>sq.ft. living</td>
<td>0</td>
</tr>
<tr>
<td>sq.ft lot</td>
<td>1</td>
</tr>
<tr>
<td>floors</td>
<td></td>
</tr>
<tr>
<td>year built</td>
<td></td>
</tr>
<tr>
<td>year renovated</td>
<td></td>
</tr>
<tr>
<td>waterfront</td>
<td></td>
</tr>
</tbody>
</table>

RSS_{\text{train}}(\hat{\theta})

# of features

0 1
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

RSS_{\text{train}}(\tilde{w})

# of features

0 1 2 3 4 5
Best Model
Size 6

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

\[ \text{RSS}_{\text{train}}(\hat{w}) \]

# of features
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

RSS_{train}(\hat{\theta})

# of features

0 1 2 3 4 5 6 7 8
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)
Last lecture in the “Regression” case study!
- Next 2 weeks: Classification
- Following 1 week: Deep Learning

Section Tomorrow:
- Coding up RIDGE and Lasso (helpful for HW1!)

Upcoming Due Dates:
- HW0 Late due date Thurs 4/6 11:59PM (if using 2 late days)
- HW1 out right after class, due Tues 4/11 11:59PM
- Learning Reflection 1 due Fri 11:59PM

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

Reminder of resources
Recap:
Ridge Regression

Change quality metric to minimize:

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

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

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

$$\hat{w} = \min_w RSS(w)$$

Exactly the old problem! This is called the least squares solution.

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

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

If $w = \hat{0}$ (all $w_j = 0$), then $RSS(w) + \lambda \|w\|^2_2 = RSS(w)_{\text{min}}$

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

**$\lambda$ in between?**

$$0 \leq \|\hat{w}\|^2_2 \leq \|\hat{w}_{\text{LS}}\|^2_2$$
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 \( \hat{y} = \widehat{w}^T h(x) \) would be very slow!

If \( \widehat{w} \) is **sparse**, only need to look at the non-zero coefficients

\[
\hat{y} = \sum_{\widehat{w}_j \neq 0} \widehat{w}_j h_j(x)
\]
Best Model
Size 8

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

Note: Video showed $\text{MSE}_{\text{train}}(W)$

```
<table>
<thead>
<tr>
<th># bathrooms</th>
<th># bedrooms</th>
<th>sq.ft living</th>
<th>sq.ft lot</th>
<th>floors</th>
<th>year built</th>
<th>year renovated</th>
<th>waterfront</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
```
Efficiency of All Subsets

How many models did we evaluate?

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

\[ \ldots \]
\[ \hat{y}_i = w_0 + w_1 h_1(x) + w_2 h_2(x) \]

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

There are \(2^D\) possible combinations.

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!
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)

\[ L_2 \Rightarrow \text{Ridge} \]
\[ L_1 \Rightarrow \text{LASSO} \]
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!
Start by selecting number of desired features $k$

$$\text{min_val} = \infty$$
$$S_0 \leftarrow \emptyset$$

for $i \leftarrow 1..k$:

- Find feature $f_i$ not in $S_{i-1}$, that when combined with $S_{i-1}$, minimizes the validation loss the most.
- $S_i \leftarrow S_{i-1} \cup \{f_i\}$
- if $\text{val_loss}(S_i) > \text{min_val}$:
  - break  # No need to look at more features

Called greedy because it makes choices that look best at the time.
- Greedily optimal !=

Example:
Forward Stepwise
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

<table>
<thead>
<tr>
<th>Features</th>
<th>Val Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td># bath</td>
<td>201</td>
</tr>
<tr>
<td># bed</td>
<td>300</td>
</tr>
<tr>
<td>sq ft</td>
<td>157</td>
</tr>
<tr>
<td>year built</td>
<td>224</td>
</tr>
</tbody>
</table>

### Subsets of Size 2

<table>
<thead>
<tr>
<th>Features (unordered)</th>
<th>Val Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>(# bath, # bed)</td>
<td>120</td>
</tr>
<tr>
<td>(# bath, sq ft)</td>
<td>131</td>
</tr>
<tr>
<td>(# bath, year built)</td>
<td>190</td>
</tr>
<tr>
<td>(# bed, sq ft)</td>
<td>137</td>
</tr>
<tr>
<td>(# bed, year built)</td>
<td>209</td>
</tr>
<tr>
<td>(sq ft, year built)</td>
<td>145</td>
</tr>
</tbody>
</table>
Say you want to find the optimal two-feature model, using the forward stepwise algorithm. What model would the forward stepwise algorithm choose?

<table>
<thead>
<tr>
<th>Subsets of Size 1</th>
<th>Features</th>
<th>Val Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td># bath</td>
<td>201</td>
<td></td>
</tr>
<tr>
<td># bed</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>sq ft</td>
<td>157</td>
<td></td>
</tr>
<tr>
<td>year built</td>
<td>224</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subsets of Size 2 (unordered)</th>
<th>Val Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>(# bath, # bed)</td>
<td>120</td>
</tr>
<tr>
<td>(# bath, sq ft)</td>
<td>131</td>
</tr>
<tr>
<td>(# bath, year built)</td>
<td>190</td>
</tr>
<tr>
<td>(# bed, sq ft)</td>
<td>137</td>
</tr>
<tr>
<td>(# bed, year built)</td>
<td>209</td>
</tr>
<tr>
<td>(sq ft, year built)</td>
<td>145</td>
</tr>
</tbody>
</table>
Brain Break
Option 2
Regularization
Recap: Regularization

Before, we used the quality metric that minimize loss
\[
\hat{w} = \arg\min_w L(w)
\]

Change quality metric to balance loss with measure of overfitting
- \( L(w) \) is the measure of fit
- \( R(w) \) measures the magnitude of coefficients

\[
\hat{w} = \arg\min_w L(w) + \lambda R(w)
\]

How do we actually measure the magnitude of coefficients?
Recap: Magnitude

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

$$\hat{w} = \arg\min_w MSE(w) + \lambda R(w)$$

**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?**

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

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

**Note:** Definition of $p$-Norm: $\|w\|_p^p = |w_0|^p + |w_1|^p + \cdots + |w_d|^p$
We saw that Ridge Regression shrinks coefficients, but they don’t become 0. What if we remove weights that are sufficiently small?

MUST NORMALIZE

Coefficient/weight

\(\lambda\)
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.
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...
LASSO Regression

\[ \hat{\omega} = \arg\min_w \text{MSE}(w) + \lambda ||w||_1 \]

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

**What if \( \lambda = 0 \)?**

\[ \hat{\omega} = \arg\min_w \text{MSE}(w) = \hat{\omega}_{OLS} \]

**What if \( \lambda = \infty \)?**

\[ \hat{\omega} = \arg\min_w \lambda ||w||_1 = \hat{\omega} = [0, 0, \ldots, 0] \]

\( \lambda \) in between?

\[ 0 \leq ||\hat{\omega}_{LASSO}||_1 \leq ||\hat{\omega}_{OLS}||_1 \]
Ridge (L2) Coefficient Paths

$R(\omega) = \|\omega\|_2^2$

Coefficients near zero, but not equal to 0.
LASSO (L1) Coefficient Paths

\[ R(\omega) = ||w||_1, \]

- Sparse \(\rightarrow\) more zero coefficients
- \(\rightarrow\) Fewer important features

\[ \text{Coefficients } \hat{w}_j \]

\[ \text{Coefficients vs. } \lambda \]

- \text{bedrooms}
- \text{bathrooms}
- \text{sqft\_living}
- \text{sqft\_lot}
- \text{floors}
- \text{yr\_built}
- \text{yr\_renovat}
- \text{waterfront}

^ goes to 0
Coefﬁcient Paths – Another View

Example from Google’s Machine Learning Crash Course

<table>
<thead>
<tr>
<th>No Regularization</th>
<th>L₁ Regularization</th>
<th>L₂ Regularization</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.31</td>
<td>0.31</td>
<td>0.31</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>0.03</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>0.42</td>
<td>0.42</td>
<td>0.42</td>
</tr>
<tr>
<td>-0.24</td>
<td>-0.24</td>
<td>-0.24</td>
</tr>
<tr>
<td>0.03</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>-0.42</td>
<td>-0.42</td>
<td>-0.42</td>
</tr>
<tr>
<td>0.52</td>
<td>0.52</td>
<td>0.52</td>
</tr>
</tbody>
</table>
Demo

Similar demo to last time’s with Ridge but using the LASSO penalty
$w = [w_0, w_1]$

Why might the shape of the L1 penalty cause more sparsity than the L2 penalty?
When using the L1 Norm ($\|w\|_1$) as a regularizer, it favors solutions that are \textbf{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 \textbf{VERY} small! Diminishing returns on decreasing $w_j$ with Ridge penalty.
Can also learn regularization

\[ \hat{w} = \arg\min_w \text{MSE}(w) \]
\[ \text{such that } R(w) \leq B \]

Another way to visualize why LASSO prefers sparse solutions

Optimal weights for OLS

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

More likely to hit a min at a spike

Sparsity

Geometry
$L_1$ (2 features)  

$L_1$ (3 features)
Brain Break
How should we choose the best value of $\lambda$ for LASSO?

- **a)** Pick the $\lambda$ that has the smallest $MSE(\hat{\omega})$ on the **validation set**
- **b)** Pick the $\lambda$ that has the smallest $MSE(\hat{\omega}) + \lambda||\hat{\omega}||^2_2$ on the **validation set**
- **c)** Pick the $\lambda$ that results in the most zero coefficients
- **d)** Pick the $\lambda$ that results in the fewest zero coefficients
- **e)** None of the above

Same process as Ridge
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 algorithm

For almost every hyperparameter:

Pick hyperparameter that has the lowest $\text{err}_{\text{val}} = \text{MSE}_{\text{val}}(\omega)$
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*

* e.g., using domain-specific expertise
ML Pipeline

Training Data → Pre-Processing → ML model → Output

- Historical Bias
- Representation Bias
- Measurement Bias

Lasso for feature selection

Optimization algorithm

Quality metric

Deployment Bias

- Feature

\[ y \] - \[ \hat{y} \] - \[ \hat{f} \]
De-biasing LASSO

As \( \lambda \) increases, the resulting model have higher bias and less variance.

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

Suppose $\lambda_3 \geq \lambda$ has lowest error.

OLS

sp. ft. coeff shrunk from OLS so!
(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 $\lambda$ 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.
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

\[
\hat{\mathbf{w}}_{\text{ElasticNet}} = \underset{\mathbf{w}}{\operatorname{argmin}} \ MSE(\mathbf{w}) + \lambda_1 ||\mathbf{w}||_1 + \lambda_2 ||\mathbf{w}||_2^2
\]

Combines both to achieve best of both worlds!
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!

Results always in context of your experimental setup.

i.e., change setup, possibly find completely different results for "most important"
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 in terms of accuracy

*not always! “No free lunch”*
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 $\lambda$ is varied
- Interpret LASSO coefficient path plot
- Compare and contrast LASSO (L1) and Ridge (L2)
ML Pipeline

- Training Data
- Pre-Processing
- ML Model
- Optimization algorithm
- Quality metric

- Historical Bias
- Representation Bias
- Measurement Bias
- Deployment Bias