PSY9511: Seminar 3

Regularization and variable selection

Esten H. Leonardsen

07.09.23

Outline

1. Assignment 1
2. Assignment 2
3. Regularization
    - Variable selection
    - Shrinkage (+ live coding 🥳)
    - Dimensionality reduction

Assignment 1

UNIVERSITETET
I OSLO

Assignment 1: Coding

- Create a vector of 100 standard normally distributed numbers and visualize them with a histogram.
- Show rows 5, 8, 9, and 10 of the Auto dataset.
- Show the last three columns of the Auto dataset.
- Show all cars with five cylinders in the Auto dataset.

Assignment 1: Coding

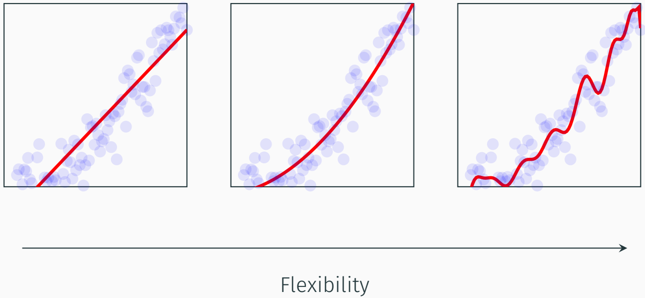
- Create a vector of 100 standard normally distributed numbers and visualize them with a histogram.
- Show rows 5, 8, 9, and 10 of the Auto dataset.
- Show the last three columns of the Auto dataset.
- Show all cars with five cylinders in the Auto dataset.

http://localhost:8889/notebooks/notebooks%2FAssignment%201.ipynb

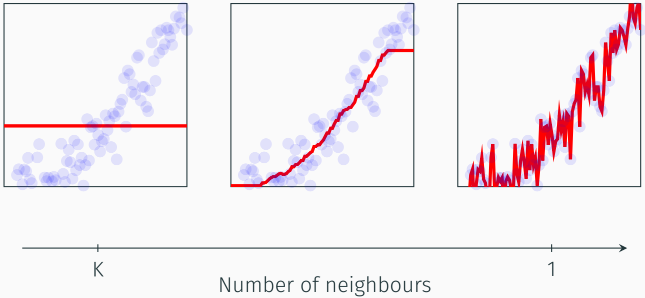
Assignment 2: Bias-variance trade-off

Flexibility

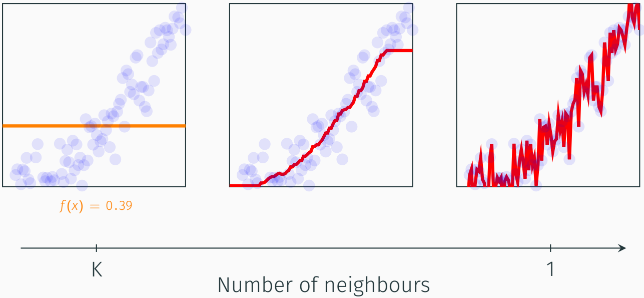
Assignment 2: Bias-variance trade-off

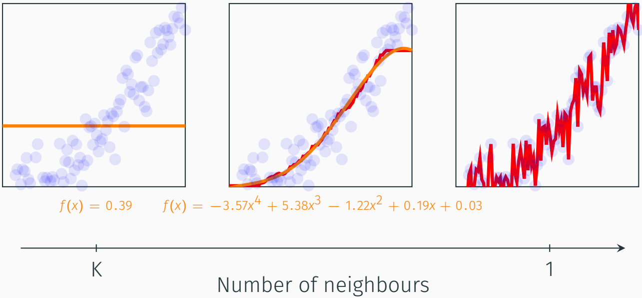


Assignment 2: Bias-variance trade-off



Assignment 2: Bias-variance trade-off




$f(x) = 0.39$
$f(x) = -3.57x^4 + 5.38x^3 - 1.22x^2 + 0.19x + 0.03$
K
1
Number of neighbours

Assignment 2: Bias-variance trade-off

$f(x) = 0.39$ $\quad$ $f(x) = -3.57x^4 + 5.38x^3 - 1.22x^2 + 0.19x + 0.03$

Assignment 2: Bias-variance trade-off

$f(x) = 0.39$ $\quad$ $f(x) = -3.57x^4 + 5.38x^3 - 1.22x^2 + 0.19x + 0.03$

1

Assignment 2: Bias-variance trade-off

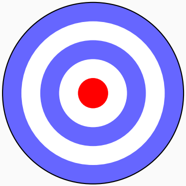
$$\underset{\substack{\uparrow\\1}}{f(x) = 0.39} \qquad f(x) = \underset{\substack{\uparrow\\1}}{-3.57}x^4 + \underset{\substack{\uparrow\\2}}{5.38}x^3 - \underset{\substack{\uparrow\\3}}{1.22}x^2 + \underset{\substack{\uparrow\\4}}{0.19}x + \underset{\substack{\uparrow\\5}}{0.03}$$

Assignment 2: Bias-variance trade-off

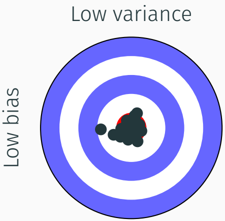
Model flexibility: Denotes the complexity of the approximated function $\hat{y} = \hat{f}(x)$.

- Informally: Wigglyness of the line
- Formally: Number of parameters in the function (degrees of freedom)

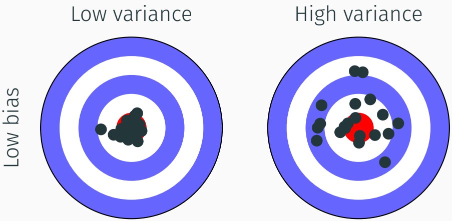
Assignment 2: Bias-variance trade-off



Assignment 2: Bias-variance trade-off

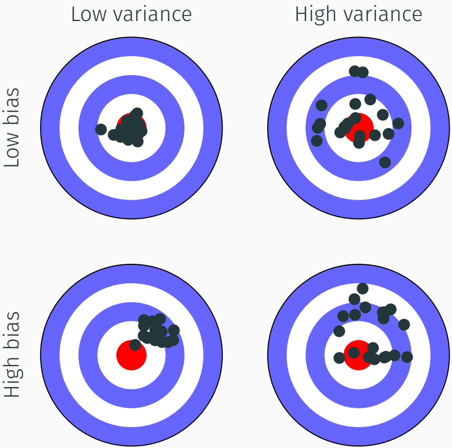


Assignment 2: Bias-variance trade-off



Assignment 2: Bias-variance trade-off

Assignment 2: Bias-variance trade-off



Assignment 2

Assignment 2: Data splitting

Assignment 2: Random seeds

Assignment 2: Log-odds vs probability vs class

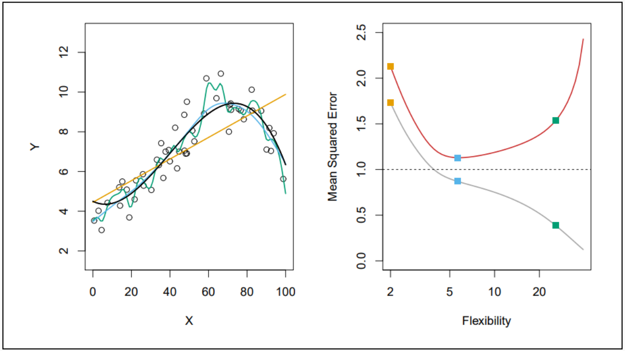
Assignment 2: Eye test

Regularization

Regularization: Motivation

$$y \sim \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \beta_3 * x_3$$

Regularization: Motivation



Regularization: Motivation

$$y \sim \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \beta_3 * x_3$$

Regularization: Out-of-sample testing

In[1]:
```
import pandas as pd


df = pd.read_csv('/Users/esten/Downloads/Auto.csv')
train = df.iloc[:int(len(df) * 0.8)]
validation = df.iloc[int(len(df) * 0.8):]

print(f'Using {len(train)} samples for training')
print(f'Using {len(validation)} samples for validation')
```

Out[1]:
```
Using 317 samples for training
Using 80 samples for validation
```

Regularization: Methods

1. Variable selection
    a. Best subset selection
    b. Forward stepwise selection
    c. Backward stepwise selection
2. Shrinkage
    a. LASSO
    b. Ridge Regression
3. Dimensionality reduction
    a. Principal Component Regression
    b. Partial Least Squares

Variable selection

Variable selection: Motivation

The number of predictors we are using in our model directly impacts model complexity.

Variable selection: Outline

Problem

We have a set of predictors $P = \{x_0, x_1, \ldots\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Variable selection: Outline

Problem

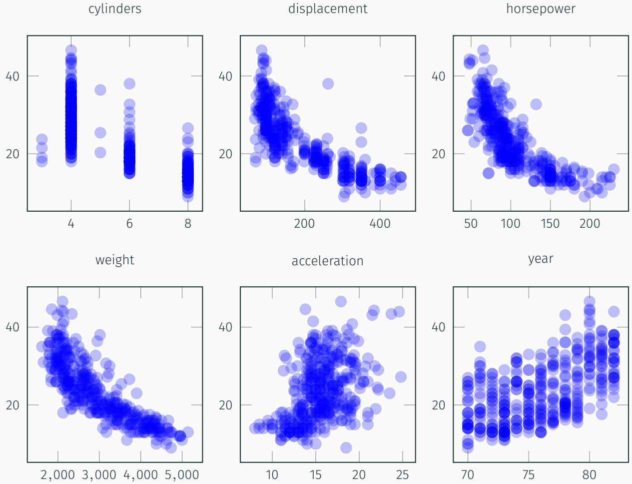
We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Motivation

1. Reduce model complexity (overfitting)
2. Simplify interpretation

Problem

We have a set of predictors $P = \{x_0, x_1, \ldots\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.



Variable selection: Best subset selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Train models on all subsets p and select the best one.

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

In[1]:

```python
import numpy as np

from itertools import chain, combinations
from sklearn.linear_model import LinearRegression

subsets = list(chain.from_iterable(combinations(predictors, r) \
                                   for r in range(len(predictors)+1)))

best = {'mse': float('inf'), 'subset': None}

for subset in subsets:
    if len(subset) == 0:
        continue

    model = LinearRegression()
    model.fit(train[list(subset)], train[target])
    predictions = model.predict(validation[list(subset)])
    mse = np.mean((predictions - validation[target]) ** 2)

    if mse < best['mse']:
        best = {'mse': mse, 'subset': subset}

print(f'MSE: {best["mse"]:.2f}, predictors: {best["subset"]}')
```

Out[1]:

```
MSE: 29.68, predictors: ('cylinders', 'displacement', 'horsepower', 'weight', 'year')
```

Variable selection: Best subset selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Train models on all subsets p and select the best one.

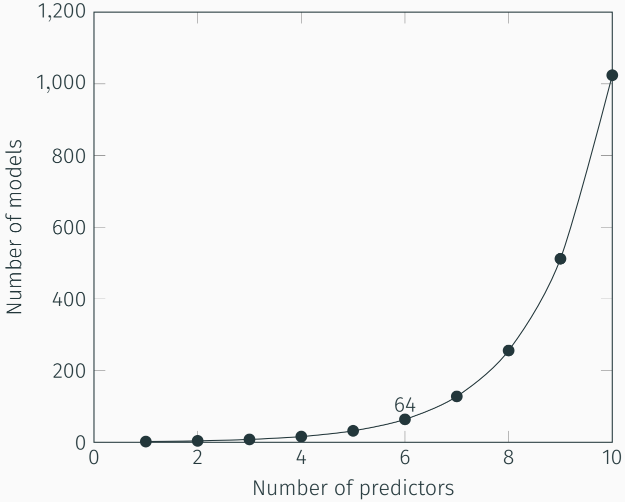
\+ Positives

Guaranteed to find the optimal solution.
Simple implementation

\- Drawbacks

Need to train many ($2^{|P|}$) models.

Variable selection: Best subset selection



Variable selection: Forward stepwise selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Start with no predictors. Iteratively add the predictor that yields the best model until all are included.

Variable selection: Forward stepwise selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Start with no predictors. Iteratively add the predictor that yields the best model until all are included.

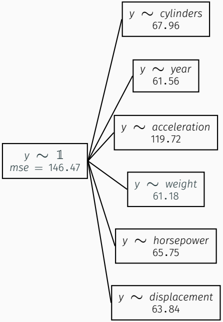
$$\boxed{\begin{array}{c} y \sim \mathbb{1} \\ mse = 146.47 \end{array}}$$

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Start with no predictors. Iteratively add the predictor that yields the best model until all are included.



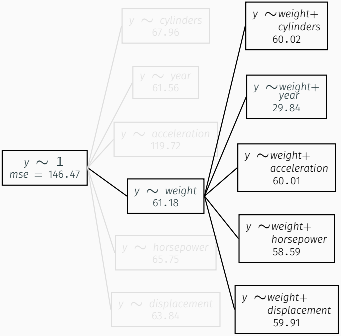
Variable selection: Forward stepwise selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Start with no predictors. Iteratively add the predictor that yields the best model until all are included.



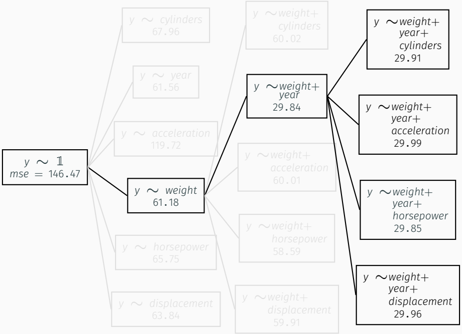
Variable selection: Forward stepwise selection

<u>Problem</u>

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

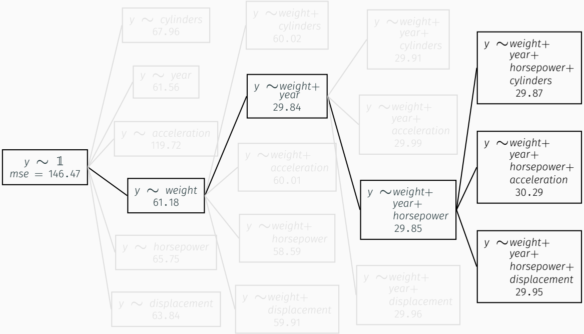
<u>Solution</u>

Start with no predictors. Iteratively add the predictor that yields the best model until all are included.



Problem
We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution
Start with no predictors. Iteratively add the predictor that yields the best model until all are included.



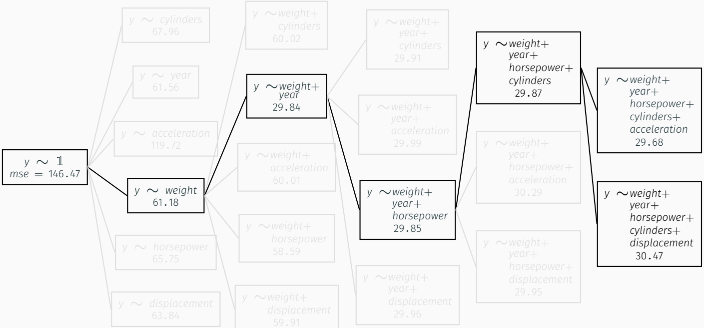
Variable selection: Forward stepwise selection

Problem

We have a set of predictors $P = \{x_0, x_1, \ldots\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

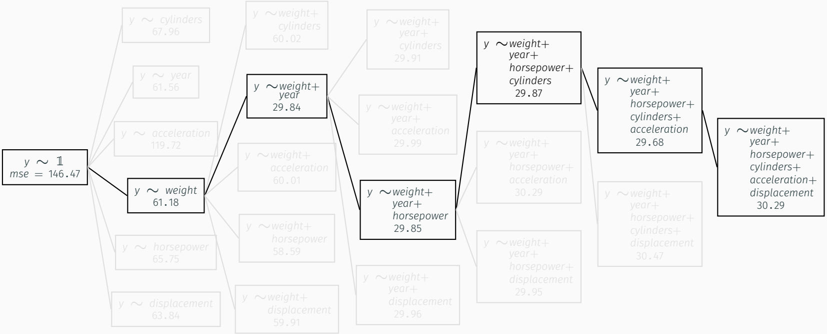
Start with no predictors. Iteratively add the predictor that yields the best model until all are included.



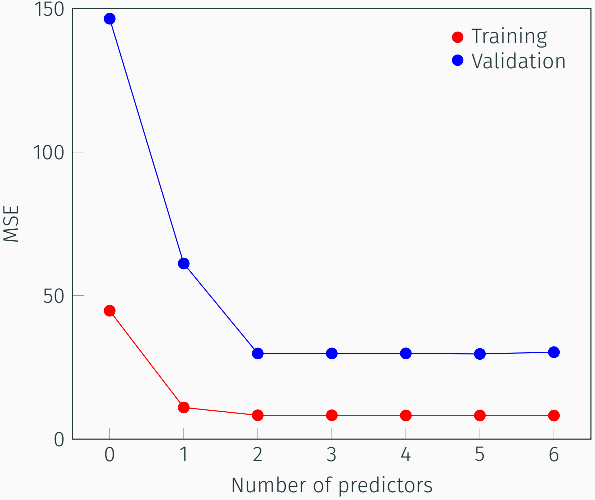
Variable selection: Forward stepwise selection

Problem
We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

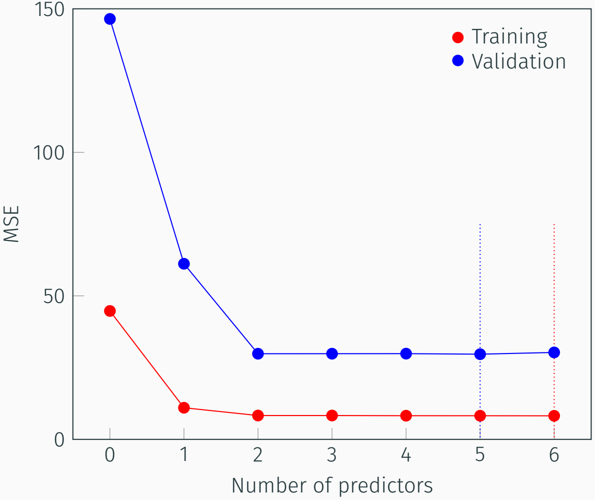
Solution
Start with no predictors. Iteratively add the predictor that yields the best model until all are included.



Variable selection: Forward stepwise selection



Variable selection: Forward stepwise selection



Problem

We have a set of predictors $P = \{x_0, x_1, \ldots\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

In[1]:

```
def fit_and_evaluate(train: pd.DataFrame, validation: pd.DataFrame,
                     predictors: List[str], target: str):
   model = LinearRegression()
   model.fit(train[predictors], train[target])

   train_predictions = model.predict(train[predictors])
   validation_predictions = model.predict(validation[predictors])

   return np.mean((train_predictions - train[target]) ** 2), \
          np.mean((validation_predictions - validation[target]) ** 2)

predictors = ['cylinders', 'displacement', 'horsepower', 'weight', 'acceleration', 'year']
target = 'mpg'

train['intercept'] = 1
validation['intercept'] = 1
train_mse, validation_mse = fit_and_evaluate(train, validation,
                                             predictors=['intercept'],
                                             target=target)
print(f'[]: {validation_mse:.2f} ({train_mse:.2f})')

chosen_predictors = []

while len(chosen_predictors) < len(predictors):
   best_predictor = {'train_mse': None, 'validation_mse': float('inf'),
                     'predictor': None}

   for predictor in set(predictors) - set(chosen_predictors):
       train_mse, validation_mse = fit_and_evaluate(train, validation,
                                   predictors=chosen_predictors + [predictor],
                                   target=target)

       if validation_mse < best_predictor['validation_mse']:
           best_predictor = {'train_mse': train_mse, 'validation_mse': validation_mse, 'predictor': predictor}

   chosen_predictors.append(best_predictor['predictor'])
```

Variable selection: Forward stepwise selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Start with no predictors. Iteratively add the predictor that yields the best model until all are included.

\+ Positives

Need to train fewer models.

\- Drawbacks

Not guaranteed to find the optimal solution.

Variable selection: Backward stepwise selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Start with all predictors. Iteratively remove the predictor that yields the best model until all you have none left.

Variable selection: Backward stepwise selection

Problem

We have a set of predictors $P = \{x_0, x_1, ...\}$ and a target variable y, and we want to find the subset $p \subseteq P$ that yields the best (linear) model for predicting y.

Solution

Start with all predictors. Iteratively remove the predictor that yields the best model until all you have none left.

\+ Positives

Need to train fewer models.

\- Drawbacks

Not guaranteed to find the optimal solution.

Shrinkage

Shrinkage: Outline

$$y \sim \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 + \beta_6 x_6$$

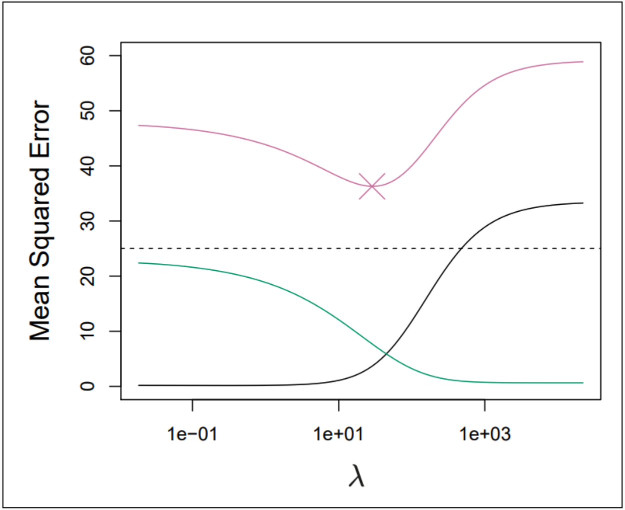
Out[1]:

	coef	std err	P>|t|	[0.025	0.975]
Intercept	-14.5353	4.764	0.002	-23.90	-5.16
cylinders	-0.3299	0.332	0.321	-0.98	0.32
displacement	0.0077	0.007	0.297	-0.00	0.02
horsepower	-0.0004	0.014	0.977	-0.02	0.02
weight	-0.0068	0.001	0.000	-0.00	-0.00
acceleration	0.0853	0.102	0.404	-0.11	0.28
year	0.7534	0.053	0.000	0.65	0.85

$$y \sim \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 + \beta_6 x_6$$

$$mse = bias^2 + variance + irreducible\ error$$

Shrinkage: Outline



Shrinkage: Outline

$$salary \sim \beta_0 + \beta_1 * age$$

Shrinkage: Outline

$$salary \sim \beta_0 + \beta_1 * age$$

$$salary \sim 300000 + 10000 * age$$

$$salary \sim 600000 + 0 * age$$

Shrinkage: Ridge regression

$$loss_{mse} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2$$

Shrinkage: Ridge regression

$$loss_{ridge} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} \beta_j^2$$

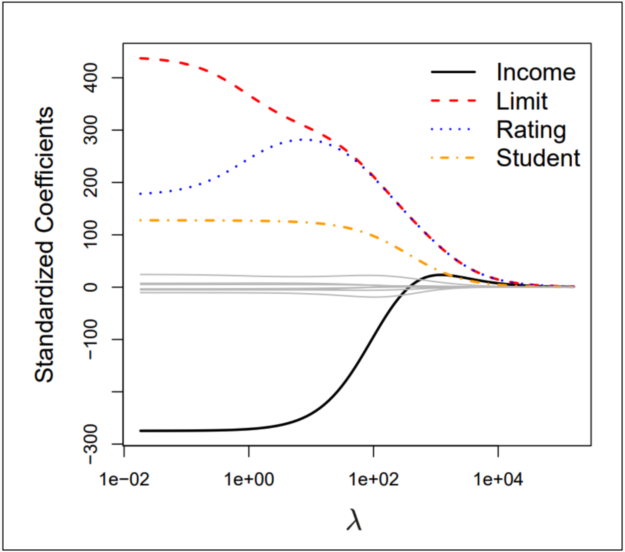
Shrinkage: Ridge regression

$$loss_{ridge} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 \boxed{+\lambda \sum_{j=0}^{p} \beta_j^2}$$

$$\Downarrow$$

$$\lambda \to \infty \Rightarrow \beta \to 0$$

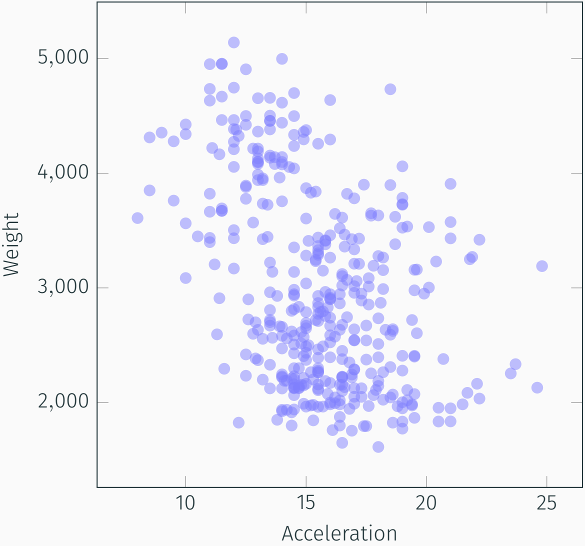
Shrinkage


Income
Limit
Rating
Student
Standardized Coefficients
400
300
200
100
0
−100
−300
1e−02
1e+00
1e+02
1e+04
λ

Shrinkage: Ridge regression

$$loss_{ridge} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} \beta_j^2$$

Shrinkage: Feature standardization



Shrinkage: Feature standardization

z-score standardization

Shrinkage: Feature standardization

z-score standardization

$$x = \frac{x - \mu_X}{\sigma_X^2}$$

Shrinkage: Feature standardization

z-score standardization

$$x = \frac{x - \mu_X}{\sigma_X^2}$$

In[1]:
```
for col in predictors:
    print(f'{col}: {np.mean(df[col]):.2f} ({np.std(df[col]):.2f})')

# z-score standardization
for col in predictors:
    df[col] = (df[col] - np.mean(df[col])) / np.std(df[col])

for col in predictors:
    print(f'{col} after: {np.mean(df[col]):.2f} ({np.std(df[col]):.2f})')
```

z-score standardization

$$x = \frac{x - \mu_X}{\sigma_X^2}$$

In[1]:
```
for col in predictors:
    print(f'{col}: {np.mean(df[col]):.2f} ({np.std(df[col]):.2f})')

# z-score standardization
for col in predictors:
    df[col] = (df[col] - np.mean(df[col])) / np.std(df[col])

for col in predictors:
    print(f'{col} after: {np.mean(df[col]):.2f} ({np.std(df[col]):.2f})')
```

Out[1]:
```
cylinders: 5.47 (1.70)
displacement: 194.41 (104.51)
horsepower: 104.47 (38.44)
weight: 2977.58 (848.32)
acceleration: 15.54 (2.76)
year: 75.98 (3.68)
cylinders after: -0.00 (1.00)
displacement after: -0.00 (1.00)
horsepower after: -0.00 (1.00)
weight after: -0.00 (1.00)
acceleration after: 0.00 (1.00)
year after: -0.00 (1.00)
```

Shrinkage: Ridge regression

http://localhost:8888/notebooks/notebooks/Live%20coding.ipynb

Shrinkage: Ridge regression

$$loss_{ridge} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} \beta_j^2$$

Regularization through shrinking the model covariates towards zero.

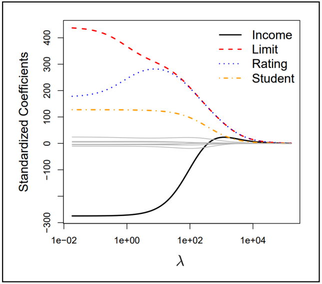
Shrinkage: LASSO

$$loss_{ridge} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} \beta_j^2$$

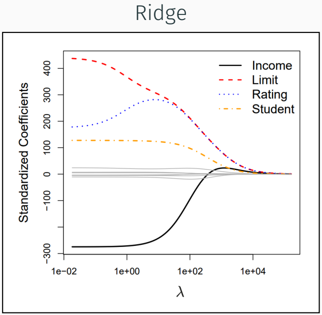
$$loss_{lasso} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} |\beta_j|$$

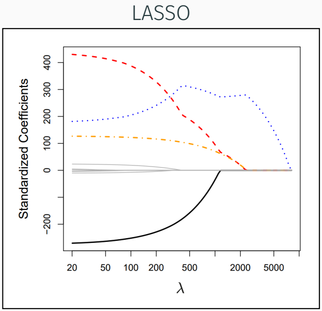
Shrinkage: LASSO

Ridge



Shrinkage: LASSO





Shrinkage: LASSO

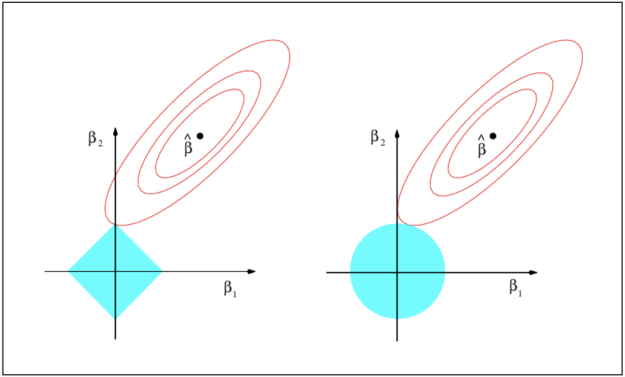
Predictor	Ridge	LASSO
Intercept	23.44	23.44
Weight	-5.59	-4.78
Year	2.75	2.00
Horsepower	-0.07	-0.09
Cylinders	-0.54	0
Acceleration	0.19	0
Displacement	0.66	0

Shrinkage: LASSO

Predictor	Ridge	LASSO
Intercept	23.44	23.44
Weight	-5.59	-4.78
Year	2.75	2.00
Horsepower	-0.07	-0.09
Cylinders	-0.54	0
Acceleration	0.19	0
Displacement	0.66	0

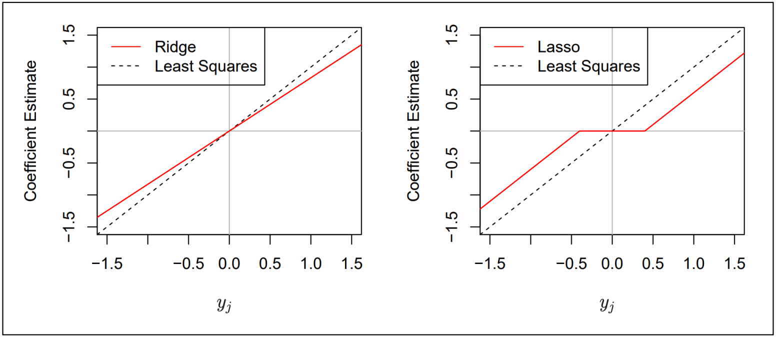
A coefficient of 0 does not mean the predictor has no predictive value for the outcome!

Shrinkage: LASSO

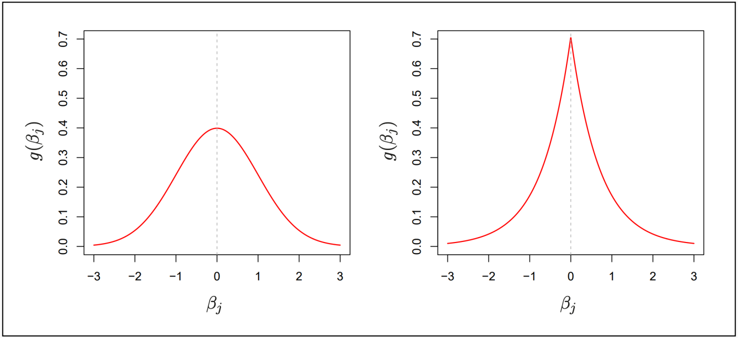


Whiteboard! 🥳

Shrinkage: LASSO



Shrinkage: LASSO



Shrinkage: Summary

$$loss_{mse} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2$$

Fits the **best** model to the data.

Shrinkage: Summary

$$loss_{mse} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2$$

Fits the **best** model to the data.

$$loss_{ridge} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} \beta_j^2$$

Fits the **best** model to the data while **shrinking** coefficients towards zero.

Shrinkage: Summary

$$loss_{mse} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2$$

Fits the **best** model to the data.

$$loss_{ridge} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} \beta_j^2$$

Fits the **best** model to the data while **shrinking** coefficients towards zero.

$$loss_{lasso} = \sum_{i=0}^{n} \left(y_i - \sum_{j=0}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=0}^{p} |\beta_j|$$

Fits the **best** model to the data while **shrinking** coefficients towards zero such that some variables are effectively **removed**.