

Lecture 16:

Regularization/Shrinkage Methods

Big Data and Machine Learning for Applied Economics

Econ 4676

Ignacio Sarmiento-Barbieri

Universidad de los Andes

October 1, 2020

Agenda

- 1 Recap & Implementation
 - K-Fold Cross Validation
- 2 Regularization
 - Lasso
 - Ridge
 - Extension
 - Demo Regularization
- 3 Further Readings

Overfit and out of Sample Prediction

- ▶ ML we care about prediction out of sample
- ▶ Overfit: complex models predict very well inside a sample but "bad" outside
- ▶ Choose the right complexity level
- ▶ Trade off Bias/Variance
- ▶ Challenge: accept some bias to decrease variance

Motivation

► Estimating test error: two approaches

- 1 We can directly estimate the test error, using either a validation set approach or a cross-validation approach
- 2 We can indirectly estimate test error by making an adjustment to the training error to account for overfitting.
 - AIC, BIC, C_p and Adjusted R^2
 - These techniques adjust the training error for the model size, and can be used to select among a set of models with different numbers of variables.
 - AIC and BIC are very closely related to classical notions of hypothesis testing.

Demo: Prelogomenon

```
#Load the required packages
library("McSpatial") #loads the package
library("dplyr") #for data wrangling
library("caret") #ML
data(matchdata) #loads the data set
matchdata <- matchdata %>% mutate(price=exp(lnprice)) %>% select(-lnprice)
#transforms log prices to standard prices
set.seed(123) #set the seed for replication purposes
str(matchdata) #compact display
```

```
## 'data.frame':   3204 obs. of  18 variables:
## $ year      : num  1995 1995 1995 2005 1995 ...
## $ lnland    : num  8.23 8.63 8.7 8.63 8.63 ...
## $ lnldg     : num  6.98 7.02 7.22 6.87 7.2 ...
## $ rooms     : int   5 5 5 4 6 7 7 6 4 6 ...
## $ bedrooms  : int   3 3 3 2 3 3 4 3 2 3 ...
## $ bathrooms : num   1 1.5 1.5 1 1 2 1 2 1.5 1.5 ...
## $ centair   : int   0 1 1 0 1 0 1 1 1 1 ...
## $ fireplace: int   0 0 0 0 0 1 0 0 0 0 ...
## $ brick     : num   1 1 1 1 1 1 1 1 1 1 ...
## $ garage1   : num   1 0 0 1 0 1 0 0 1 0 ...
## $ garage2   : num   0 1 1 0 1 0 1 0 0 0 ...
## $ dcdbd     : num  13.6 13.5 13.6 13.5 13.6 ...
## $ rr        : num   0 0 0 0 0 0 0 0 0 0 ...
## $ yrbuilt   : num  1953 1952 1952 1949 1953 ...
## $ carea     : Factor w/ 11 levels "Albany Park",...: 3 3 3 3 3 3 3 3 3 3 ...
## $ latitude  : num  42 42 42 42 42 ...
## $ longitude : num -87.8 -87.8 -87.8 -87.8 -87.8 ...
## $ price     : num 170750 168000 192500 398000 215000 ...
```

Demo: K-fold CV

```
model2 <- train(price ~ bedrooms,   # model to fit
                data = matchdata,
                trControl = trainControl(method = "cv", number = 10),
                # Method: crossvalidation, 10 folds
                method = "lm")# specifying regression model
```

model2

```
## Linear Regression
##
## 3204 samples
##    1 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2883, 2884, 2884, 2883, 2884, 2885, ...
## Resampling results:
##
##    RMSE      Rsquared    MAE
## 147308.8  0.01385314 122117.8
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Demo: K-fold CV

```
model2 <- train(price ~ bedrooms+bathrooms+ rooms+centair+fireplace+brick+
  lnland+lnbldg+garage1+garage2+
  dcdbd+ rr +
  yrbuilt+ factor(year) +
  carea+ latitude+longitude,
  data = matchdata,
  trControl = trainControl(method = "cv", number = 10),
  method = "lm")
```

model2

```
## Linear Regression
##
## 3204 samples
## 17 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2883, 2884, 2884, 2883, 2883, 2884, ...
## Resampling results:
##
## RMSE      Rsquared  MAE
## 74297.54  0.7490674  48170.37
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Model Subset Selection

- ▶ We have M_k models
- ▶ We want to find the model that best predicts out of sample
- ▶ We have a number of ways to go about it
 - ▶ Best Subset Selection
 - ▶ Stepwise Selection
 - ▶ Forward selection
 - ▶ Backward selection

Best Subset Selection

$$y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + u \quad (1)$$

- 1 Estimate **all** possible models with $k = 0, 1, \dots, p$ predictors.
- 2 Compute the prediction error using cross validation
- 3 Pick the one with the smallest prediction error

Demo: Best Subset Selection

- Caret is usually the solution but it doesn't have everything =(

| | | | | |
|---|--------------|------------|-------|-------|
| Linear Regression with Backwards Selection | leapBackward | Regression | leaps | nvmax |
| Linear Regression with Forward Selection | leapForward | Regression | leaps | nvmax |
| Linear Regression with Stepwise Selection | leapSeq | Regression | leaps | nvmax |

Note: <https://topepo.github.io/caret/available-models.html>

Demo: Best Subset Selection

```
require("leaps")
```

```
class(matchdata$careas)
```

```
## [1] "factor"
```

```
class(matchdata$year)
```

```
## [1] "numeric"
```

```
matchdata$year<-factor(matchdata$year)
```

```
best<-regsubsets(price ~ ., method="exhaustive",data = matchdata)  
summary(best)
```

Demo: Best Subset Selection

```
## Subset selection object
## Call: regsubsets.formula(price ~ ., method = "exhaustive", data = matchdata)
## 26 Variables (and intercept)
## ...
## Selection Algorithm: exhaustive
##      year2005 lnland lnblgd rooms bedrooms bathrooms centair fireplace
## 1  ( 1 ) "*"      " "      " "      " "      " "      " "      " "
## 2  ( 1 ) "*"      " "      "*"      " "      " "      " "      " "
## 3  ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
## 4  ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
## 5  ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
## 6  ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
## 7  ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
## 8  ( 1 ) "*"      "*"      "*"      " "      " "      " "      "*"
##      brick garage1 garage2 dcdbd rr  yrbuilt careaEdgewater careaEdison Park
## 1  ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 2  ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 3  ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 4  ( 1 ) " "      " "      " "      " "      " "      "*"      " "
## 5  ( 1 ) " "      " "      " "      " "      " "      "*"      " "
## 6  ( 1 ) " "      " "      " "      " "      " "      "*"      " "
## 7  ( 1 ) " "      " "      " "      " "      " "      "*"      " "
## 8  ( 1 ) " "      " "      " "      " "      " "      "*"      " "
```

Demo: Best Subset Selection

```
##          careaForest Glen careaJefferson Park careaLincoln Square
## 1 ( 1 ) " " " " " "
## 2 ( 1 ) " " " " " "
## 3 ( 1 ) " " " " " "
## 4 ( 1 ) " " " " " "
## 5 ( 1 ) "*" " " " "
## 6 ( 1 ) "*" " " " "
## 7 ( 1 ) "*" " " "*"
## 8 ( 1 ) "*" " " "*"
##          careaNorth Park careaNorwood Park careaRogers Park careaUptown
## 1 ( 1 ) " " " " " "
## 2 ( 1 ) " " " " " "
## 3 ( 1 ) " " " " " "
## 4 ( 1 ) " " " " " "
## 5 ( 1 ) " " " " " "
## 6 ( 1 ) " " " " "*"
## 7 ( 1 ) " " " " "*"
## 8 ( 1 ) " " " " "*"
##          careaWest Ridge latitude longitude
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " "
## 4 ( 1 ) " " " "
## 5 ( 1 ) " " " "
## 6 ( 1 ) " " " "
## 7 ( 1 ) " " " "
## 8 ( 1 ) " " " "
```

Stepwise Selection

1 Forward Stepwise Selection

- ▶ Start with no predictors
- ▶ Test all models with 1 predictor. Choose the one with smallest prediction error using cross validation
- ▶ Add 1 predictor at a time, without taking away.
- ▶ Of the $p+1$ models, choose the one with smallest prediction error using cross validation

2 Backward Stepwise Selection

- ▶ Same idea but start with a complete model and go backwards, taking one at a time.

Demo Stepwise Selection

```
forward <- train(price ~ ., data = matchdata,  
  method = "leapForward",  
  trControl = trainControl(method = "cv", number = 10))  
forward
```

```
## Linear Regression with Forward Selection  
##  
## 3204 samples  
## 17 predictor  
##  
## No pre-processing  
## Resampling: Cross-Validated (10 fold)  
## Summary of sample sizes: 2884, 2884, 2884, 2883, 2883, 2883, ...  
## Resampling results across tuning parameters:  
##  
##   nvmax  RMSE      Rsquared  MAE  
##   2      84219.28  0.6768962  55184.58  
##   3      79212.49  0.7147519  51009.82  
##   4      78595.00  0.7193113  50631.17  
##  
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was nvmax = 4.
```

Demo Stepwise Selection

```
summary(forward$finalModel)
```

```
## Subset selection object
## 26 Variables (and intercept)
##           Forced in Forced out
## ...

## 1 subsets of each size up to 4
## Selection Algorithm: forward
##           year2005 lnland lnldg rooms bedrooms bathrooms centair fireplace
## 1 ( 1 ) "*"      " "      " "      " "      " "      " "      " "
## 2 ( 1 ) "*"      " "      "*"      " "      " "      " "      " "
## 3 ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
## 4 ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
##           brick garage1 garage2 dcdbd rr yrbuilt careaEdgewater careaEdison Park
## 1 ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 2 ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 3 ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 4 ( 1 ) " "      " "      " "      " "      " "      "*"      " "
##           careaForest Glen careaJefferson Park careaLincoln Square
## 1 ( 1 ) " "      " "      " "      " "
## 2 ( 1 ) " "      " "      " "      " "
## 3 ( 1 ) " "      " "      " "      " "
## 4 ( 1 ) " "      " "      " "      " "
##           careaNorth Park careaNorwood Park careaRogers Park careaUptown
## 1 ( 1 ) " "      " "      " "      " "
## 2 ( 1 ) " "      " "      " "      " "
## 3 ( 1 ) " "      " "      " "      " "
## 4 ( 1 ) " "      " "      " "      " "
##           careaWest Ridge latitude longitude
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      " "
```


Demo Stepwise Selection

```
backwards <- train(price ~ ., data = matchdata,  
  method = "leapBackward",  
  trControl = trainControl(method = "cv", number = 10))  
  
backwards
```

```
## Linear Regression with Backwards Selection  
##  
## 3204 samples  
## 17 predictor  
##  
## No pre-processing  
## Resampling: Cross-Validated (10 fold)  
## Summary of sample sizes: 2884, 2882, 2884, 2885, 2883, 2884, ...  
## Resampling results across tuning parameters:  
##  
##   nvmax  RMSE      Rsquared  MAE  
##   2      84353.39  0.6769755  55217.19  
##   3      79280.53  0.7153318  51000.22  
##   4      78137.63  0.7235894  49820.72  
##  
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was nvmax = 4.
```

Demo Stepwise Selection

```
summary(backwards$finalModel)
```

```
## Subset selection object
## 26 Variables (and intercept)
## ...
## Selection Algorithm: backward
##      year2005 lnland lnldg rooms bedrooms bathrooms centair fireplace
## 1 ( 1 ) "*"      " "      " "      " "      " "      " "      " "
## 2 ( 1 ) "*"      " "      "*"      " "      " "      " "      " "
## 3 ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
## 4 ( 1 ) "*"      "*"      "*"      " "      " "      " "      " "
##      brick garage1 garage2 dcdb rr yrbuilt careaEdgewater careaEdison Park
## 1 ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 2 ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 3 ( 1 ) " "      " "      " "      " "      " "      " "      " "
## 4 ( 1 ) " "      " "      " "      " "      " "      " "      " "
##      careaForest Glen careaJefferson Park careaLincoln Square
## 1 ( 1 ) " "      " "      " "      " "
## 2 ( 1 ) " "      " "      " "      " "
## 3 ( 1 ) " "      " "      " "      " "
## 4 ( 1 ) "*"      " "      " "      " "
##      careaNorth Park careaNorwood Park careaRogers Park careaUptown
## 1 ( 1 ) " "      " "      " "      " "
## 2 ( 1 ) " "      " "      " "      " "
## 3 ( 1 ) " "      " "      " "      " "
## 4 ( 1 ) " "      " "      " "      " "
##      careaWest Ridge latitude longitude
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      " "
## 4 ( 1 ) " "      " "      " "
```

Regularization

- ▶ Isn't it worth starting with the general model (p-variables) and crossing out the non-significant coefficients?
- ▶ Or we could go the following path:
 - ▶ Run model, get coefficients and p-values
 - ▶ Take out those with p-values below a certain α (we can adjust for FDR)
 - ▶ Why is this a bad idea?
- ▶ Backward selection approximates that idea (not exactly) and does a better job than the previous point

Regularization

- ▶ The key of modern statistics is regularization
- ▶ The strategy involves penalizing complexity so as to depart from optimality and stabilize the system
- ▶ 'Crossing out' variables / coefficients is an extreme way to 'shrink' them.
- ▶ Lasso: a formal and algorithmic way of accomplishing this task.

Lasso

- ▶ For $\lambda \geq 0$ given, consider minimizing the following objective function

$$\min_{\beta} L(\beta) = \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda \sum_{s=1}^p |\beta_s| \quad (2)$$

- ▶ Note:
 - ▶ First coef. constant
 - ▶ $\lambda = 0$?
 - ▶ $\lambda = \infty$?

Lasso

$$\min_{\beta} L(\beta) = \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda \sum_{s=1}^p |\beta_s| \quad (3)$$

- ▶ LASSO magic: it automatically chooses which variables go in ($\beta_s \neq 0$) and which are out ($\beta_s = 0$)
- ▶ Why? Coefficients that don't go in are corner solutions
- ▶ $L(\beta)$ is non differentiable

Lasso Intuition

► Lasso Intuition

$$\min_{\beta} L(\beta) = \sum_{i=1}^n (y_i - x_i \beta)^2 + \lambda |\beta| \quad (4)$$

► Only one predictor, i.e., one coefficient.

► Standardize predictor $\sum x_i^2 = 1$ so

$$\hat{\beta}_{OLS} = \frac{\sum_i x_i y_i}{\sum_i x_i^2} \quad (5)$$

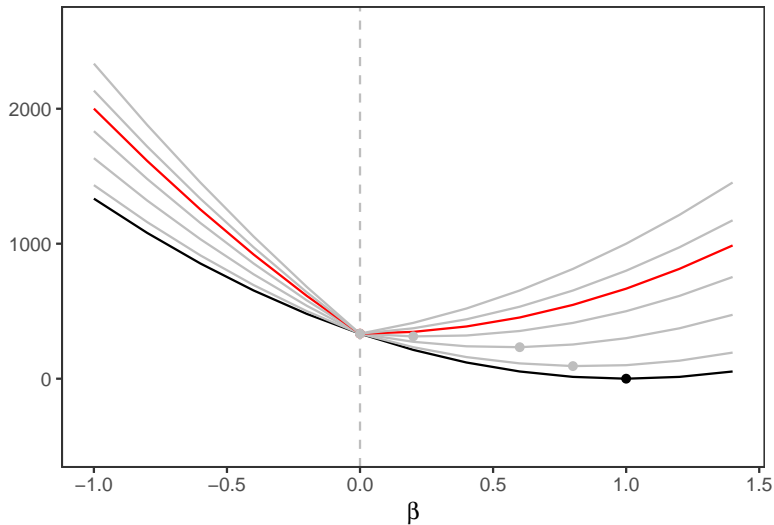
Lasso Intuition

$$\min_{\beta} L(\beta) = \sum_{i=1}^n (y_i - x_i \beta)^2 + \lambda |\beta| \quad (6)$$

$$L(\beta) = \sum_{i=1}^n (y_i - x_i \beta)^2 + \lambda |\beta| = \begin{cases} \sum_{i=1}^n (y_i - x_i \beta)^2 + \lambda \beta & \text{if } \beta \geq 0 \\ \sum_{i=1}^n (y_i - x_i \beta)^2 - \lambda \beta & \text{if } \beta \leq 0 \end{cases} \quad (7)$$

- ▶ Non differentiable at $\beta = 0$
- ▶ Differentiable otherwise $\beta \neq 0$

Lasso Intuition



Lasso Intuition

$$\begin{aligned}\frac{dL(\beta)^+}{d\beta} &= -2 \sum y_i x_i + 2\beta \sum x_i^2 + \lambda \\ &= -2 \sum y_i x_i + \beta + \lambda\end{aligned}\tag{8}$$

$$\frac{dL(0)^+}{d\beta} = -2 \sum y_i x_i + \lambda$$

then, if

$$\lambda \geq 2 \sum y_i x_i\tag{9}$$

we have $\hat{\beta}_{lasso} = 0$

Lasso Intuition

if $\lambda < 2\sum y_i x_i$ we have an interior solution

$$-2\sum y_i x_i + \hat{\beta}_{lasso} + \lambda = 0 \quad (10)$$

$$\hat{\beta}_{lasso} = \sum y_i x_i - \frac{\lambda}{2} \quad (11)$$

$$\hat{\beta}_{lasso} = \hat{\beta}_{OLS} - \frac{\lambda}{2} \quad (12)$$

Ridge

- ▶ For $\lambda \geq 0$ given, consider minimizing the following objective function

$$\min_{\beta} R(\beta) = \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda \sum_{s=2}^p (\beta_s)^2 \quad (13)$$

- ▶ Note:
 - ▶ Intuition is similar to Lasso, however the problem is completely different

Ridge

$$\min_{\beta} R(\beta) = \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda (\beta_s)^2 \quad (14)$$

FOC

$$-2 \sum_{i=1}^n y_i x_i + 2\beta + 2\lambda \beta_s = 0 \quad (15)$$

$$\begin{aligned} \hat{\beta}_{ridge} &= \frac{\sum_{i=1}^n y_i x_i}{(1 + \lambda)} \\ &= \frac{\beta_{OLS}}{(1 + \lambda)} \end{aligned} \quad (16)$$

- ▶ Solution is *always* interior (unlike Lasso)
- ▶ Solutions is "shrunk"

Lasso and Ridge Intuition

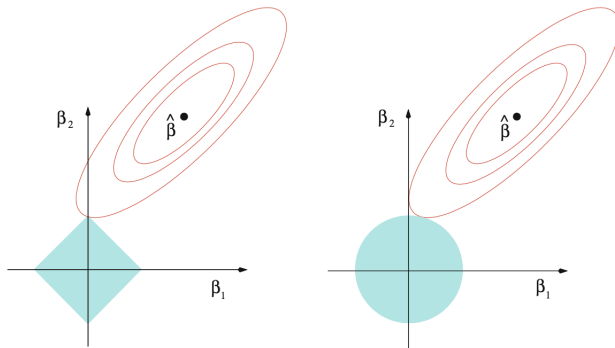


FIGURE 3.11. Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the least squares error function.

Technical comments

- ▶ We showed that Ridge is biased, but for certain λ
 $MSE(\beta_{ridge}) < MSE(\beta_{OLS})$
- ▶ Not possible to derive an exact result for Lasso, but ridge approximates to lasso
- ▶ Lasso shrinks everything towards zero, Ridge not quite so
- ▶ Application wise:
 - ▶ Standardize the data
 - ▶ Selection of λ ?

Technical comments: λ Selection

- ▶ Selection of λ ?
- ▶ Use CV
- ▶ Choose a grid of λ values, and compute the CV error for each value
- ▶ Select the λ^* that minimizes the prediction error
- ▶ Estimate using all the observations and the selected λ^*

Extension

► Family of penalized regressions

$$\min_{\beta} R(\beta) = \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda \sum_{s=2}^p |\beta_s|^p \quad (17)$$

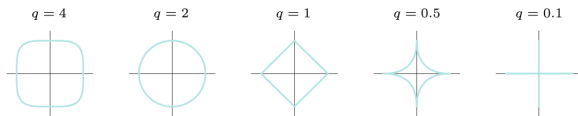


FIGURE 3.12. Contours of constant value of $\sum_j |\beta_j|^q$ for given values of q .

► Combination? Elastic Net

$$\begin{aligned} \min_{\beta} EL(\beta) &= \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda_1 \sum_{s=2}^p |\beta_s| + \lambda_2 \sum_{s=2}^p \beta_s^2 \\ &= \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda \sum_{s=2}^p ((1 - \alpha) \beta_s^2 + \alpha |\beta_s|) \end{aligned} \quad (18)$$

Demo Regularization

```
lambda <- 10^seq(-2, 3, length = 10)
lasso <- train(
  price ~., data = matchdata, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneGrid = expand.grid(alpha = 1,
    lambda=lambda), preProcess = c("center", "scale")
)
lasso
```

```
## 3204 samples
## 17 predictor
##
## Pre-processing: centered (26), scaled (26)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2884, 2885, 2882, 2883, 2885, 2883, ...
## Resampling results across tuning parameters:
##
##   lambda      RMSE      Rsquared    MAE
##   1.000000e-02  74410.32  0.7487304  48120.44
##   3.593814e-02  74410.32  0.7487304  48120.44
##   1.291550e-01  74410.32  0.7487304  48120.44
##   4.641589e-01  74410.32  0.7487304  48120.44
##   1.668101e+00  74410.32  0.7487304  48120.44
##   5.994843e+00  74410.32  0.7487304  48120.44
##   2.154435e+01  74410.32  0.7487304  48120.44
##   7.742637e+01  74411.91  0.7487131  48102.41
##   2.782559e+02  74485.72  0.7481557  48011.74
##   1.000000e+03  74702.02  0.7467291  47865.45
##
## Tuning parameter 'alpha' was held constant at a value of 1
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 1 and lambda = 21.54435.
```

Demo Regularization

```
ridge <- train(
  price ~., data = matchdata, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneGrid = expand.grid(alpha = 0, lambda=lambda), preProcess = c("center", "scale")
)
ridge
```

```
## 3204 samples
## 17 predictor
##
## Pre-processing: centered (26), scaled (26)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2884, 2883, 2883, 2884, 2883, ...
## Resampling results across tuning parameters:
##
##  lambda      RMSE      Rsquared    MAE
##  1.000000e-02  75133.92  0.7479252  47361.73
##  3.593814e-02  75133.92  0.7479252  47361.73
##  1.291550e-01  75133.92  0.7479252  47361.73
##  4.641589e-01  75133.92  0.7479252  47361.73
##  1.668101e+00  75133.92  0.7479252  47361.73
##  5.994843e+00  75133.92  0.7479252  47361.73
##  2.154435e+01  75133.92  0.7479252  47361.73
##  7.742637e+01  75133.92  0.7479252  47361.73
##  2.782559e+02  75133.92  0.7479252  47361.73
##  1.000000e+03  75133.92  0.7479252  47361.73
##
## Tuning parameter 'alpha' was held constant at a value of 0
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0 and lambda = 1000.
```

Demo Regularization

```
el <- train(
  price ~., data = matchdata, method = "glmnet",
  trControl = trainControl("cv", number = 10), preProcess = c("center", "scale")
)
el
```

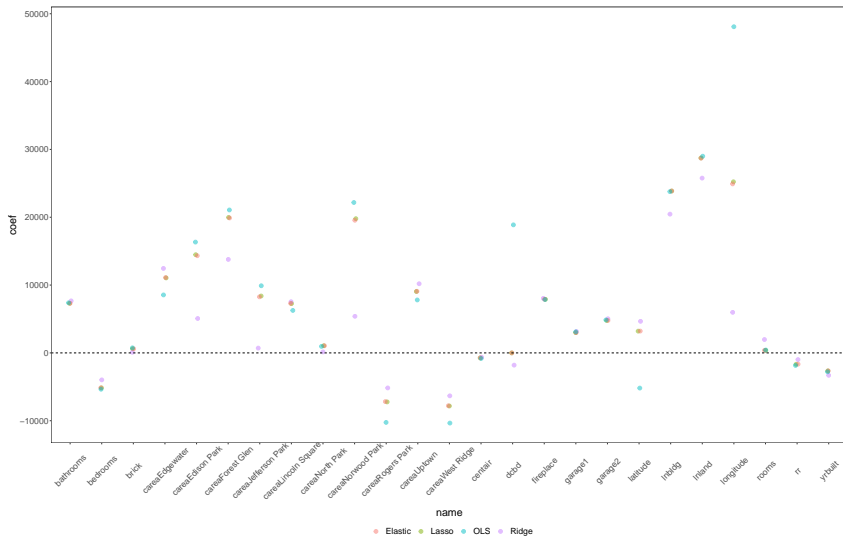
```
## glmnet
##
## 3204 samples
## 17 predictor
##
## Pre-processing: centered (26), scaled (26)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2885, 2885, 2883, 2883, 2884, ...
## Resampling results across tuning parameters:
##
##  alpha  lambda      RMSE      Rsquared    MAE
##  0.10    230.7569  74163.82  0.7525328  48112.63
##  0.10    2307.5689  74241.34  0.7518849  47772.50
##  0.10    23075.6893  76941.03  0.7489116  48065.77
##  0.55    230.7569  74161.27  0.7524710  48056.67
##  0.55    2307.5689  74420.59  0.7505525  47656.06
##  0.55    23075.6893  82931.37  0.7161268  52413.46
##  1.00    230.7569  74189.84  0.7522040  48029.77
##  1.00    2307.5689  74622.97  0.7493269  47527.50
##  1.00    23075.6893  88016.04  0.6846917  57561.95
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0.55 and lambda = 230.7569.
```

Demo Regularization

```
models <- list(ridge = ridge, lasso = lasso, elastic = el)
resamples(models) %>% summary( metric = "RMSE")
```

```
##
## Call:
## summary.resamples(object = ., metric = "RMSE")
##
## Models: ridge, lasso, elastic
## Number of resamples: 10
##
## RMSE
##           Min.   1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## ridge  64485.44 70582.29 74874.23 75133.92 77717.74 89880.65    0
## lasso  66245.34 71294.79 74792.57 74410.32 78080.16 80946.67    0
## elastic 59004.20 69395.53 71960.20 74161.27 78485.49 88578.06    0
```

Demo Regularization



Further Readings

- ▶ Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.
- ▶ James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.
- ▶ Kuhn, M. (2012). The caret package. R Foundation for Statistical Computing, Vienna, Austria.
<https://topepo.github.io/caret/index.html>
- ▶ Taddy, M. (2019). Business data science: Combining machine learning and economics to optimize, automate, and accelerate business decisions. McGraw Hill Professional.