STA 235H - Model Selection II: Stepwise and Shrinkage

Fall 2022

McCombs School of Business, UT Austin

Last class

- Started with our prediction chapter
  - Bias vs. Variance
  - Validation set approach and Cross-validation
  - How to choose a model for a continuous outcome (RMSE)

Today: Continuing our journey

- Some ways to **select models**:
  - Stepwise selection
  - Shrinkage methods: Ridge and Lasso.

One step at a time

Stepwise selection

- We have seen how to choose between some given models. **But what if we want to test all possible models?**
- **Stepwise selection**: Computationally-efficient algorithm to select a model based on the data we have (subset selection).

Algorithm for forward stepwise selection:

1. Start with the *null model*, M_0 (no predictors)
2. For $k = 0, \ldots, p-1$:

   a) Consider all $p-k$ models that augment M_k with one additional predictor.
   b) Choose the *best* among these $p-k$ models and call it M_{k+1}.

3. Select the single best model from $M_0, \ldots, M_p$ using CV.

Backwards stepwise follows the same procedure, but starts with the full model.

Q1: Will forward stepwise subsetting yield the same results as backwards stepwise selection?

a) Yes

b) No

How do we do stepwise selection in R?

```r
library(caret)

set.seed(100)

train.control <- trainControl(method = "cv", number = 10) #set up a 10-fold cv

lm.fwd <- train(logins ~ . - unsubscribe, data = train.data, method = "leapForward",
                tuneGrid = data.frame(nvmax = 1:5), trControl = train.control)
lm.fwd$results
```

```
##   nvmax     RMSE  Rsquared      MAE     RMSESD RsquaredSD      MAESD
## 1     1 2.269469 0.6101788 1.850376 0.04630907 0.01985045 0.04266950
## 2     2 2.087184 0.6702660 1.639885 0.04260047 0.01784601 0.04623508
## 3     3 2.087347 0.6702094 1.640405 0.04258030 0.01804773 0.04605074
## 4     4 2.088230 0.6699245 1.641402 0.04270561 0.01808685 0.04620206
## 5     5 2.088426 0.6698623 1.641528 0.04276883 0.01810569 0.04624618
```

- Which one would you choose out of the 5 models? Why?

How do we do stepwise selection in R?

```
# We can see the number of covariates that is optimal to choose:
lm.fwd$bestTune
```

```
##   nvmax
## 2     2
```

```
# And how does that model looks like:
summary(lm.fwd$finalModel)
```

```
## Subset selection object
## 5 Variables  (and intercept)
##        Forced in Forced out
## id         FALSE      FALSE
## female     FALSE      FALSE
## city       FALSE      FALSE
## age        FALSE      FALSE
## got        FALSE      FALSE
## 1 subsets of each size up to 2
## Selection Algorithm: forward
##          id  female city age got
## 1  ( 1 ) " " " "    " "  " " "*"
## 2  ( 1 ) " " " "    "*"  " " "*"
```

```
# If we want to recover the coefficient names, we can use the coef() function:
coef(lm.fwd$finalModel, lm.fwd$bestTune$nvmax)
```

```
## (Intercept)        city         got
##    7.035888    2.570454   -6.306371
```

Honey, I shrunk the coefficients!

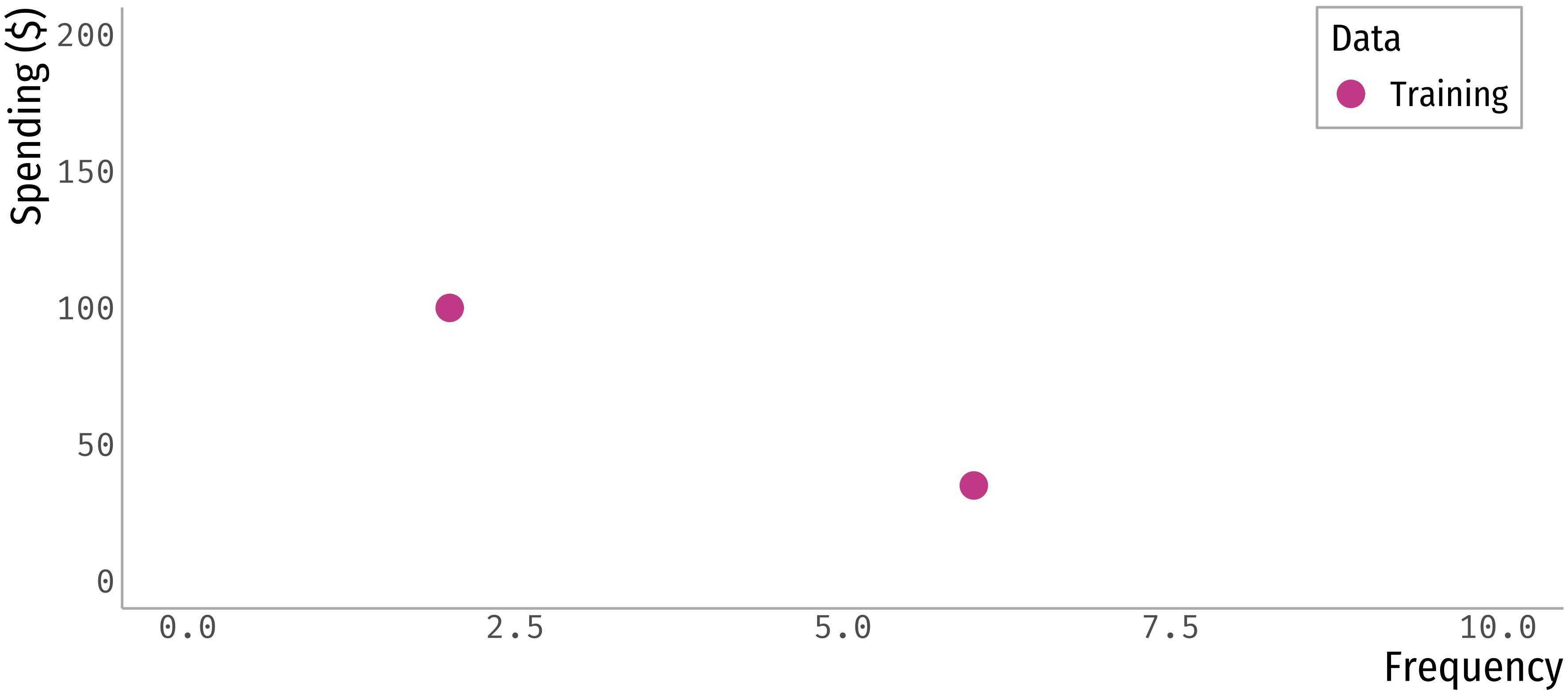
What is shrinkage?

- We just reviewed the **stepwise procedure**: Subsetting model selection approach.
  - Select k out of p total predictors
- **Shrinkage** ***(a.k.a Regularization)***: Fitting a model with all p predictors, but introducing bias (i.e. shrinking coefficients towards 0) for improvement in variance.
  - Ridge regression
  - Lasso regression

Let's build a ridge.

Ridge Regression: An example

- Predict spending based on frequency of visits to a website

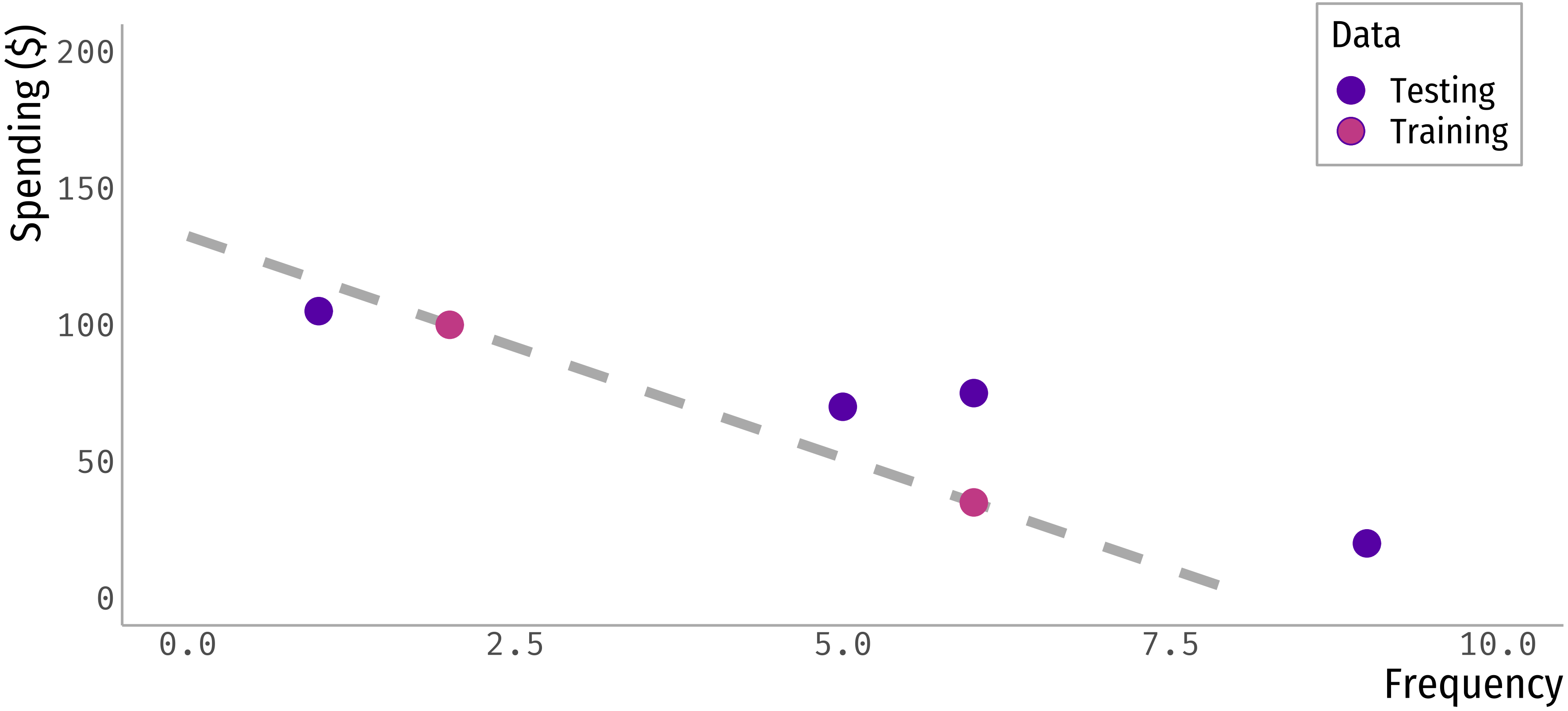


Ordinary Least Squares

- In an **OLS**: Minimize sum of squared-errors, i.e. $\min_{\beta} \sum_{i=1}^{n} (\text{spend}_i - (\beta_0 + \beta_1 \text{freq}_i))^2$

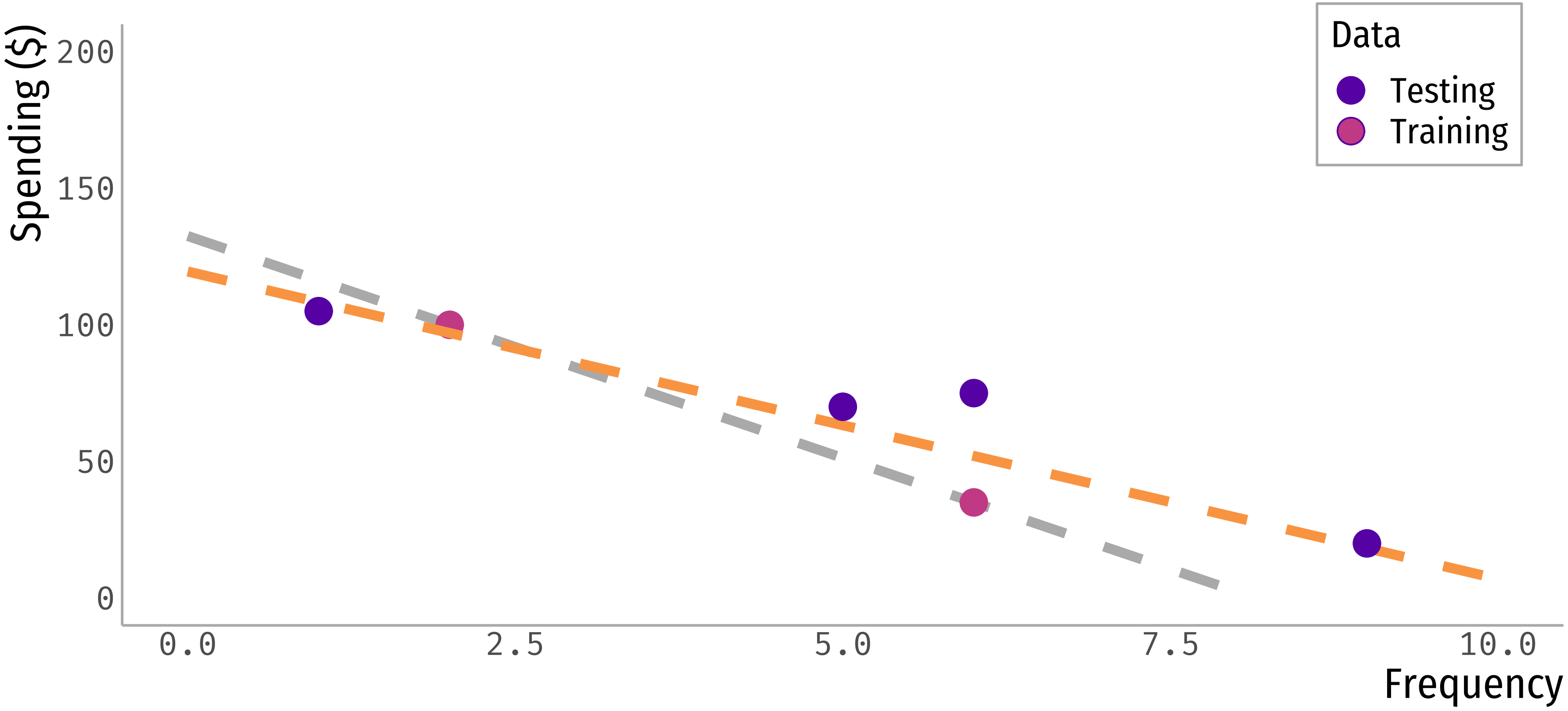
What about fit?

- Does the OLS fit the testing data well?



Ridge Regression

- Let's shrink the coefficients!: Ridge Regression



Ridge Regression: What does it do?

- Ridge regression **introduces bias to reduce variance** in the testing data set.

- In a simple regression (i.e. one regressor/covariate):

$$\min_{\beta} \sum_{i=1}^{n} \underbrace{(y_i - \beta_0 - x_i\beta_1)^2}_{OLS}$$

Ridge Regression: What does it do?

- Ridge regression **introduces bias to reduce variance** in the testing data set.
- In a simple regression (i.e. one regressor/covariate):

$$\min_{\beta} \sum_{i=1}^{n} \underbrace{(y_i - \beta_0 - x_i\beta_1)^2}_{OLS} + \underbrace{\lambda \cdot \beta_1^2}_{RidgePenalty}$$

- λ is the **penalty factor** $\rightarrow$ indicates how much we want to shrink the coefficients.

Q2: In general, which model will have smaller β coefficients?

a) A model with a larger λ

b) A model with a smaller λ

Remember... we care about accuracy in the testing dataset!

RMSE on the testing dataset: OLS

$$RMSE = \sqrt{\frac{1}{4}\sum_{i=1}^{4}(\text{spend}_i - (132.5 - 16.25 \cdot \text{freq}_i))^2} = 28.36$$

RMSE on the testing dataset: Ridge Regression

$$RMSE = \sqrt{\frac{1}{4}\sum_{i=1}^{4}(\text{spend}_i - (119.5 - 11.25 \cdot \text{freq}_i))^2} = 12.13$$

Ridge Regression in general

- For regressions that include **more than one regressor**:

$$\min_{\beta} \sum_{i=1}^{n} \underbrace{(y_i - \sum_{k=0}^{p} x_i \beta_k)^2}_{OLS} + \underbrace{\lambda \cdot \sum_{k=1}^{p} \beta_k^2}_{Ridge Penalty}$$

- In our previous example, if we had two regressors, $female$ and $freq$:

$$\min_{\beta} \sum_{i=1}^{n} (\text{spend}_i - \beta_0 - \beta_1 \text{female}_i - \beta_2 \text{freq}_i)^2 + \lambda \cdot (\beta_1^2 + \beta_2^2)$$

- Because the ridge penalty includes the β's coefficients, **scale matters**:
  - Standardize variables (*you will do that as an option in your code*)

How do we choose λ?

Cross-validation!

1) Choose a grid of λ values

- The grid you choose will be context dependent (play around with it!)

2) Compute cross-validation error (e.g. RMSE) for each

3) Choose the smallest one.

λ vs RMSE?

λ vs RMSE? A zoom

How do we do this in R?

```r
library(caret)

set.seed(100)

hbo <- read.csv("https://raw.githubusercontent

lambda_seq <- c(0,10^seq(-3, 1, length = 100))

ridge <- train(logins ~ . - unsubscribe - id,
            data = train.data,
            method = "glmnet",
            preProcess = "scale",
            trControl = trainControl("cv", numl
            tuneGrid = expand.grid(alpha = 0,
                          lambda = lambda_seq)
  )

cv_lambda <- data.frame(lambda = ridge$results
                        rmse = ridge$results$RI
```

- We will be using the `caret` package

How do we do this in R?

```
library(caret)

set.seed(100)

hbo <- read.csv("https://raw.githubusercontent

lambda_seq <- c(0,10^seq(-3, 1, length = 100))

ridge <- train(logins ~ . - unsubscribe - id,
            data = train.data,
            method = "glmnet",
            preProcess = "scale",
            trControl = trainControl("cv", num
            tuneGrid = expand.grid(alpha = 0,
                          lambda = lambda_seq)
  )

cv_lambda <- data.frame(lambda = ridge$results
                        rmse = ridge$results$R
```

- We will be using the `caret` package
- We are doing cross-validation, so remember to set a seed!

How do we do this in R?

```
library(caret)

set.seed(100)

hbo <- read.csv("https://raw.githubusercontent

lambda_seq <- c(0,10^seq(-3, 1, length = 100))

ridge <- train(logins ~ . - unsubscribe - id,
            data = train.data,
            method = "glmnet",
            preProcess = "scale",
            trControl = trainControl("cv", num
            tuneGrid = expand.grid(alpha = 0,
                         lambda = lambda_seq)
  )

cv_lambda <- data.frame(lambda = ridge$results
                        rmse = ridge$results$R
```

- We will be using the `caret` package
- We are doing **cross-validation**, so remember to set a seed!
- You need to create a grid for the λ's **that will be tested**

How do we do this in R?

```
library(caret)

set.seed(100)

hbo <- read.csv("https://raw.githubusercontent

lambda_seq <- c(0,10^seq(-3, 1, length = 100))

ridge <- train(logins ~ . - unsubscribe - id,
               data = train.data,
               method = "glmnet",
               preProcess = "scale",
               trControl = trainControl("cv", numl
               tuneGrid = expand.grid(alpha = 0,
                                      lambda = lambda_seq)
   )

cv_lambda <- data.frame(lambda = ridge$results
                        rmse = ridge$results$R
```

- We will be using the `caret` package
- We are doing **cross-validation**, so remember to set a seed!
- You need to create a grid for the λ's **that will be tested**
- The function we will use is `train`: Same as before
    - `method="glmnet"` means that it will run an elastic net.
    - `alpha=0` means is a **ridge regression**
    - `lambda = lambda_seq` is not necessary (you can provide your own grid)

How do we do this in R?

```
library(caret)

set.seed(100)

hbo <- read.csv("https://raw.githubusercontent

lambda_seq <- c(0,10^seq(-3, 1, length = 100))

ridge <- train(logins ~ . - unsubscribe - id,
            data = train.data,
            method = "glmnet",
            preProcess = "scale",
            trControl = trainControl("cv", numl
            tuneGrid = expand.grid(alpha = 0,
                         lambda = lambda_seq)
   )

cv_lambda <- data.frame(lambda = ridge$results
                        rmse = ridge$results$R
```

- We will be using the `caret` package
- We are doing **cross-validation**, so remember to set a seed!
- You need to create a grid for the λ's **that will be tested**
- The function we will use is `train`: Same as before
- Important objects in `cv`:
  - `results$lambda`: Vector of λ that was tested
  - `results$RMSE`: RMSE for each λ
  - `bestTune$lambda`: λ that minimizes the error term.

How do we do this in R?

OLS regression:

```
lm1 <- lm(logins ~ got + city,
          data = train.data)

coef(lm1)
```

```
## (Intercept)         got        city
##    7.035888   -6.306371    2.570454
```

```
rmse(lm1, test.data)
```

```
## [1] 2.089868
```

Ridge regression:

```
coef(ridge$finalModel, ridge$bestTune$lambda)
```

```
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  6.564243424
## female       0.002726465
## city         0.824387472
## age          0.046468790
## got         -2.639308962
```

```
rmse(ridge, test.data)
```

```
## [1] 2.097452
```

Throwing a lasso

Lasso regression

- Very similar to ridge regression, except it **changes the penalty term**:

$$\min_{\beta} \sum_{i=1}^{n} \underbrace{(y_i - \sum_{k=0}^{p} x_i \beta_k)^2}_{OLS} + \underbrace{\lambda \cdot \sum_{k=1}^{p} |\beta_k|}_{LassoPenalty}$$

- In our previous example:

$$\min_{\beta} \sum_{i=1}^{n} (\text{spend}_i - \beta_0 - \beta_1 \text{female}_i - \beta_2 \text{freq}_i)^2 + \lambda \cdot (|\beta_1| + |\beta_2|)$$

- Lasso regression is also called l_1 regularization:

$$||\beta||_1 = \sum_{k=1}^{p} |\beta|$$

Ridge vs Lasso

Ridge	Lasso
Final model will have p coefficients	Can set coefficients = 0
Usually better with multicollinearity	Improves interpretability of model
	Can be used for model selection

And how do we do Lasso in R?

```r
library(caret)

set.seed(100)

hbo <- read.csv("https://raw.githubusercontent

lambda_seq <- c(0,10^seq(-3, 1, length = 100))

lasso <- train(logins ~ . - unsubscribe - id, 
            method = "glmnet",
            preProcess = "scale",
            trControl = trainControl("cv", num
            tuneGrid = expand.grid(alpha = 1,
                                   lambda = lambda_seq)
  )

cv_lambda <- data.frame(lambda = lasso$results
                        rmse = lasso$results$R
```

Exactly the same!

- … But change `alpha=1`!!

And how do we do Lasso in R?

Ridge regression:

```
coef(ridge$finalModel, ridge$bestTune$lambda)
```

```
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  6.564243424
## female       0.002726465
## city         0.824387472
## age          0.046468790
## got         -2.639308962
```

```
rmse(ridge, test.data)
```

```
## [1] 2.097452
```

Lasso regression:

```
coef(lasso$finalModel, lasso$bestTune$lambda)
```

```
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (Intercept)  6.85434931
## female       .
## city         0.87800876
## age          0.02926514
## got         -2.83314945
```

```
rmse(lasso, test.data)
```

```
## [1] 2.09153
```

Your Turn

In-class Activity

- Go to https://sta235h.rocks/Week11 and read the exercise.
- Answer the questions in the activity and write them in your sheet.
- Submit your sheet at the end of the class.
- Important note: We will be doing a classification task (binary outcome). In this case, we do not use RMSE as a measure to compare outcomes, but accuracy.
    - Accuracy: How many observations (%) can I classify correctly?

Main takeway points

- You can **shrink coefficients** to introduce bias and decrease variance.
- Ridge and Lasso regression are **similar**:
  - Lasso can be used for model selection.
- Importance of understanding **how to estimate the penalty coefficient**.

References

- James, G. et al. (2021). "Introduction to Statistical Learning with Applications in R". *Springer. Chapter 6.*
- STDHA. (2018). "Penalized Regression Essentials: Ridge, Lasso & Elastic Net"