Class 15: ElasticNet

MGSC 310

Prof. Jonathan Hersh

Class 15: Announcements

1. Problem Set 4 posted, due Monday, Oct 26
2. Midterm exam next week (Oct 27 – 29)

Midterm Exam details

1. Exam: Posted 12:30 on Tuesday, due 5pm Thursday
2. Structured much like the problem sets
    - Mix of conceptual and coding questions
3. Open note, open internet, BUT DO NOT COPY CODE FROM ANYWHERE FOUND ONLINE OR FROM YOUR PEERS
4. How to study?
    - Read slides and textbook and ensure you know all core concepts
    - Practice running and interpreting models, producing output
    - Prepare code snippets to do common tasks

- Extra Instructor Office Hours Friday 11 – 12:30

Class 15: Outline

1. Comparing Ridge vs Lasso
2. ElasticNet Algorithm
3. ElasticNet in R
4. ElasticNet Lab

Another way to write Lasso

Lasso

$$\min_{\beta} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

Lasso with two variables

$$\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_{i1} - \beta_1 x_{i2})^2 \quad \text{subject to} \quad |\beta_1| + |\beta_2| \leq s$$

In other words: I give you s as a budget (like setting some lambda)

You can increase your coefficients but the sum of the absolute value of them must be less than s

Another way to write Ridge

Ridge

$$\min_{\beta} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} (\beta_j)^2 \leq s$$

Ridge with two variables

$$\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_{i1} - \beta_1 x_{i2})^2 \quad \text{subject to} \quad (\beta_1)^2 + (\beta_2)^2 \leq s$$

In other words: I give you s as a budget (like setting some lambda)

You can increase your coefficients but the sum of the absolute value of them must be less than s

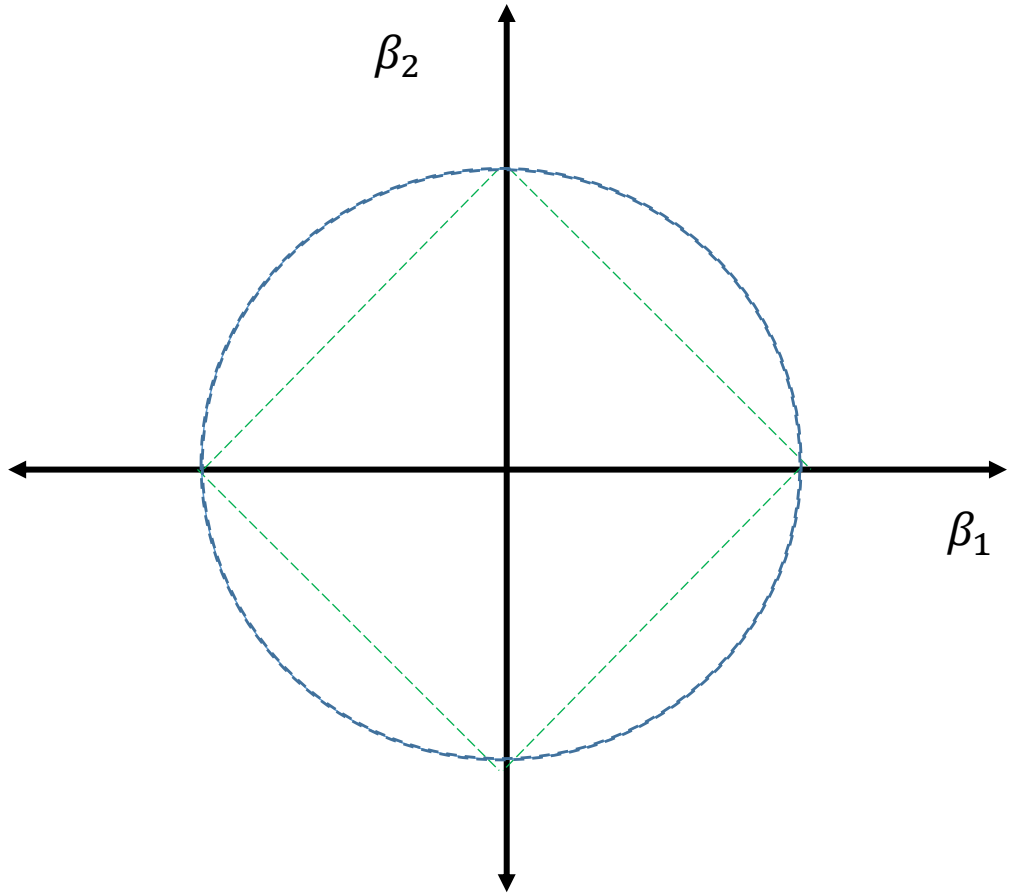
Ridge Versus Lasso Penalty

Ridge penalty $(\beta_1)^2 + (\beta_2)^2 \leq 1$

Lasso penalty $|\beta_1| + |\beta_2| \leq 1$

Let's pick an arbitrary value of s = 1

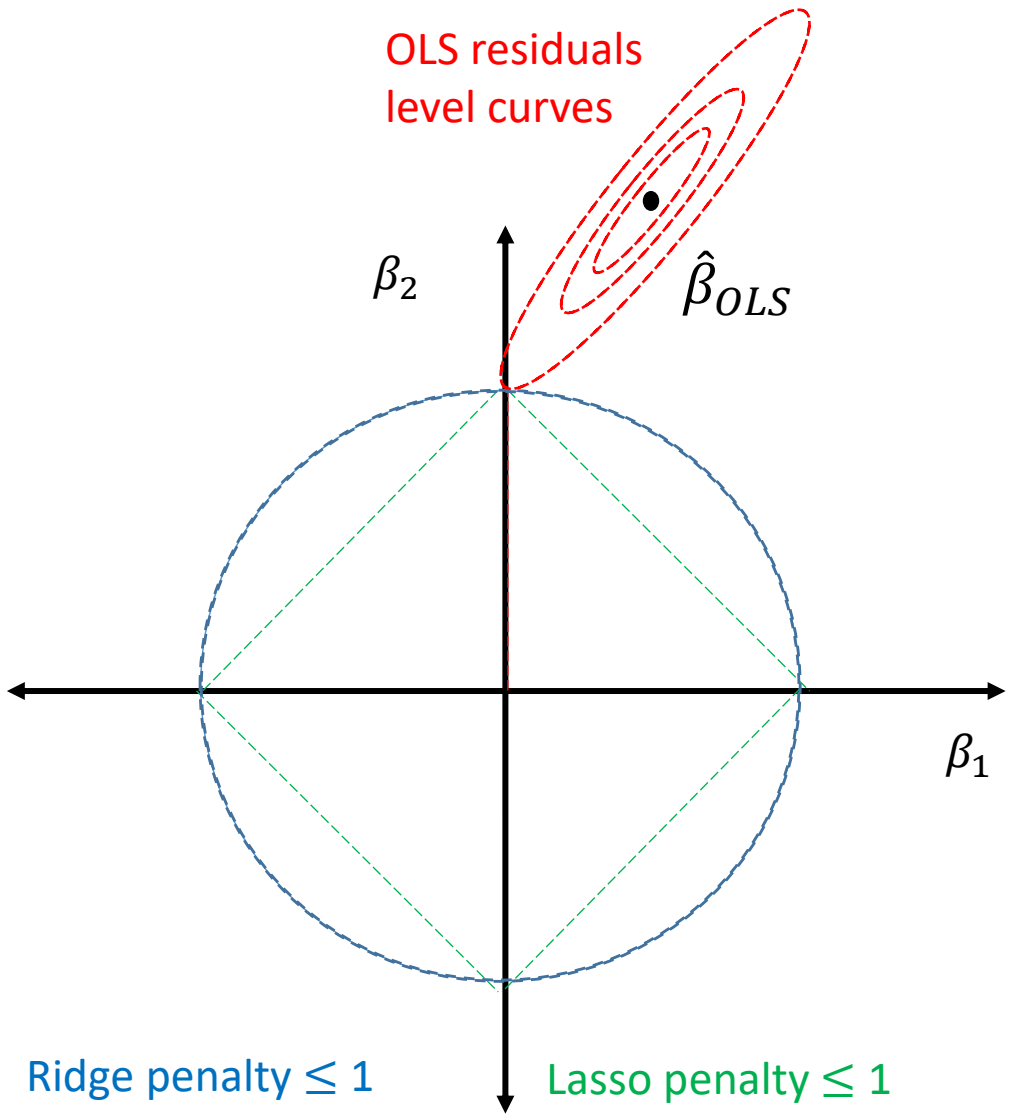
What do these look like graphically?



Ridge penalty ≤ 1

Lasso penalty ≤ 1

Ridge and Lasso Equations Redux



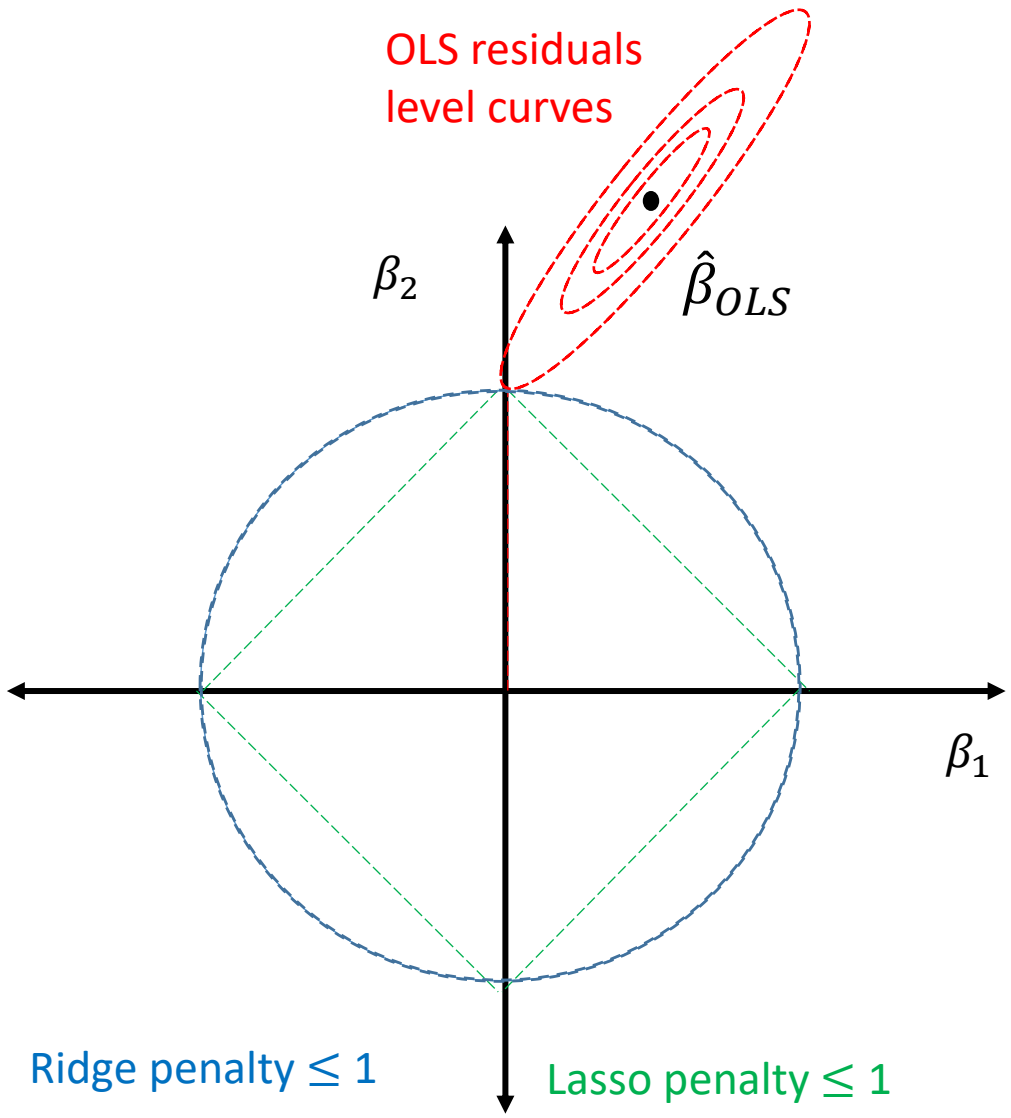
Suppose the optimal OLS beta is this point in black

Meaning, without constraints this point achieves a minimum of the residuals

We can represent that graphically as a series of contour lines where the black dot (OLS beta) is the minimum

Level curves farther from the OLS point are higher residuals

Ridge and Lasso Equations Redux



Graphically what the ridge equation is asking is: "find the lowest residual level curve while staying within the blue circle"

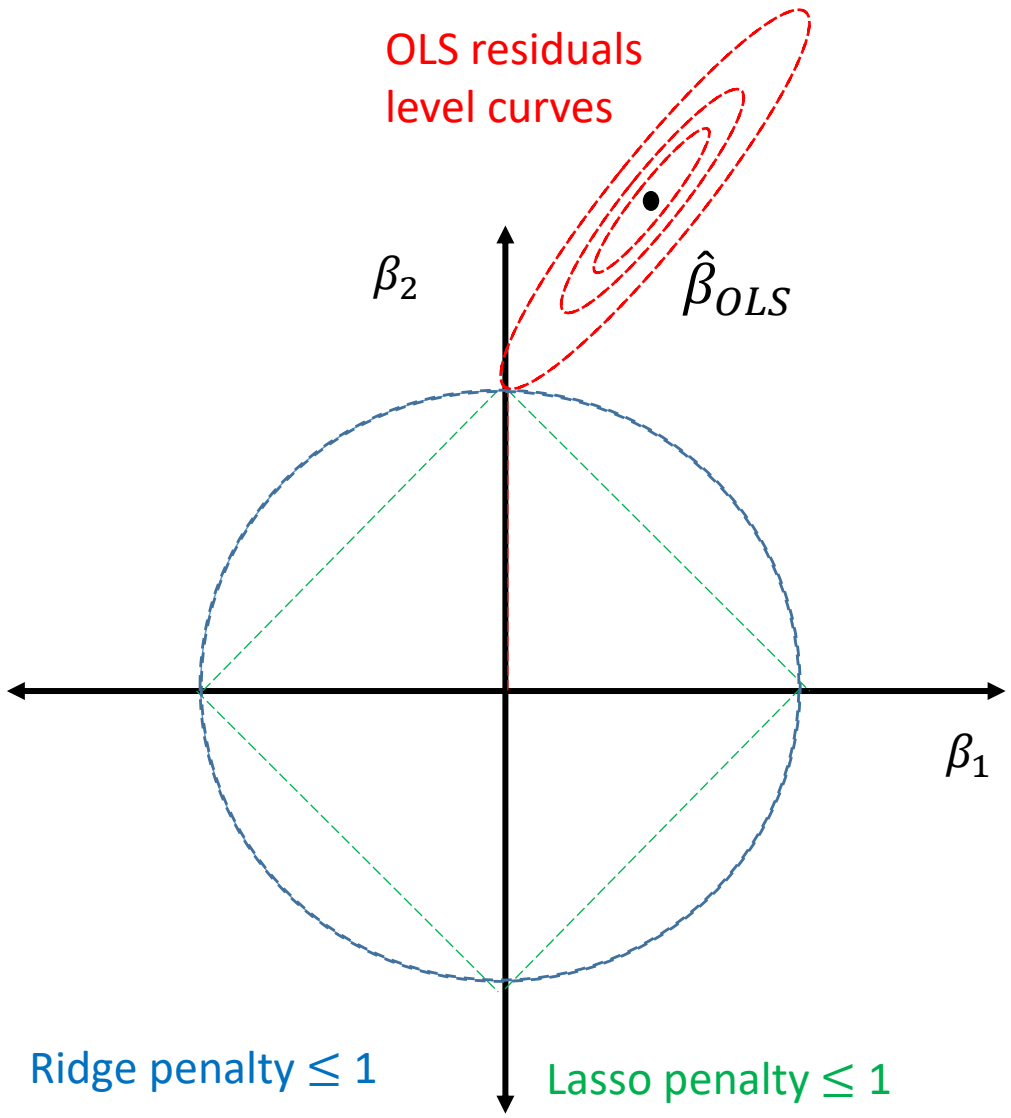
That is the level curve tangent to the blue line

Ridge

$$\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_{i1} - \beta_1 x_{i2})^2$$

subject to $(\beta_1)^2 + (\beta_2)^2 \leq s$

Ridge and Lasso Equations Redux



Graphically what the Lasso equation is asking is: "find the lowest residual level curve while staying within the green diamond"

That is the level curve tangent to the green line

Lasso

$$\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_{i1} - \beta_1 x_{i2})^2$$

subject to $|\beta_1| + |\beta_2| \leq s$

Ridge and Lasso Equations Redux

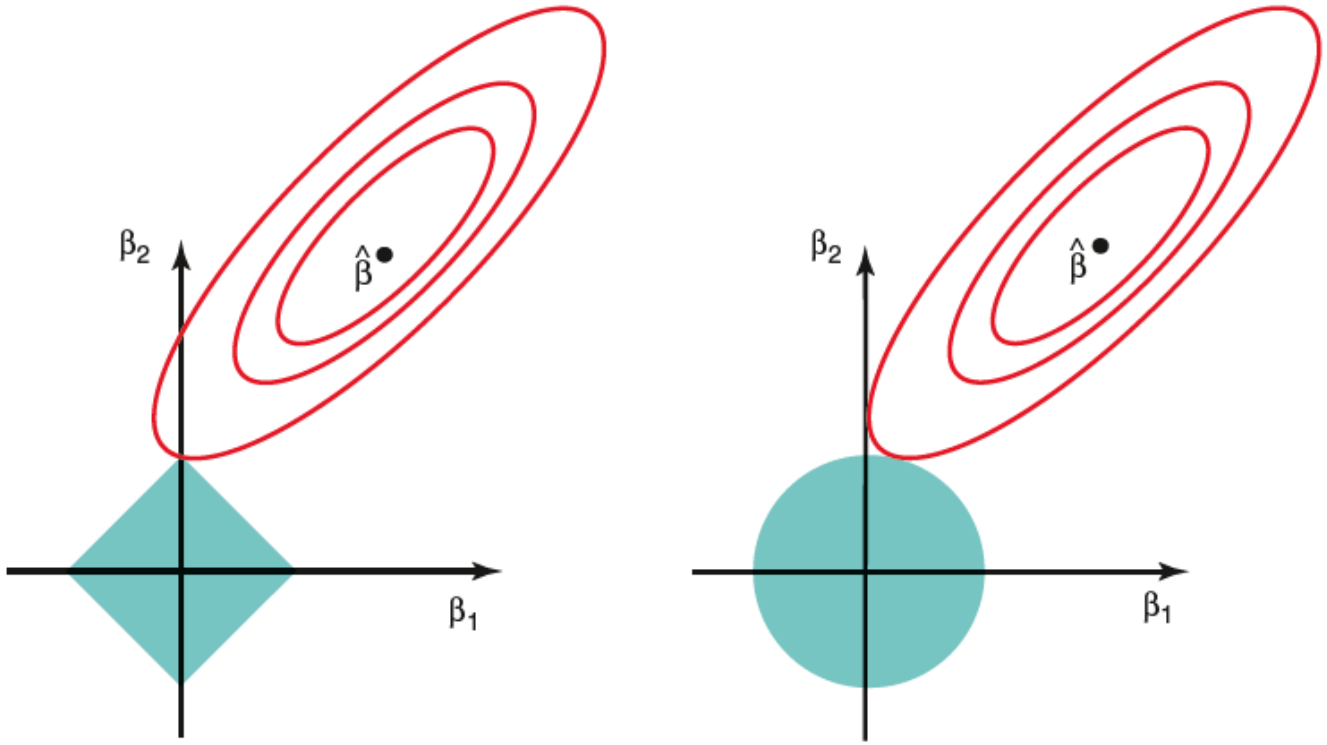


FIGURE 6.7. *Contours of the error and constraint functions for the lasso* (left) *and ridge regression* (right). *The solid blue areas are the constraint regions,* $|\beta_1| + |\beta_2| \leq s$ *and* $\beta_1^2 + \beta_2^2 \leq s$, *while the red ellipses are the contours of the RSS.*

Lasso acts as a variable selector because the point of tangency for Lasso is often such that one of the variables (here beta_1) is zero

Ridge does not have this property, and we see there's still some small value for beta_1 in the right plot

Ridge versus Lasso

- Use Lasso when the “data generating process” (DGP, how the data is really formed) is **sparse**
- What is a sparse DGP?
  - Only a few variables really matter!
- Ridge should be used when many variables matter a little

Why Choose? ElasticNet Uses Both Ridge and Lasso Penalty

$$\beta_{ENet} = \min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_{i1} - \beta_1 x_{i2})^2$$

$$+ \lambda[\underbrace{\alpha(|\beta_1| + |\beta_2|)}_{\text{Lasso penalty}} + \underbrace{(1-\alpha)(\beta_1^2 + \beta_2^2)}_{\text{Ridge penalty}}]$$

- $\alpha \in [0,1]$ controls the amount of ridge versus lasso penalty
- λ functions as before -> controlling total amount of shrinkage penalty

How to choose λ and α? Grid Search

$$\beta_{ENet} = \min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_{i1} - \beta_1 x_{i2})^2$$

$$+ \lambda[\alpha(|\beta_1| + |\beta_2|) + (1-\alpha)(\beta_1^2 + \beta_2^2)]$$

λ	$\alpha = 0$	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 1$
0.5	0.3	0.6	1.3	1.7	4.0
1.0	2.6	3.1	2.1	3.2	4.3
1.5	3.1	3.9	3.2	4.3	5.3
2	3.8	4.6	5.4	6.0	7.4

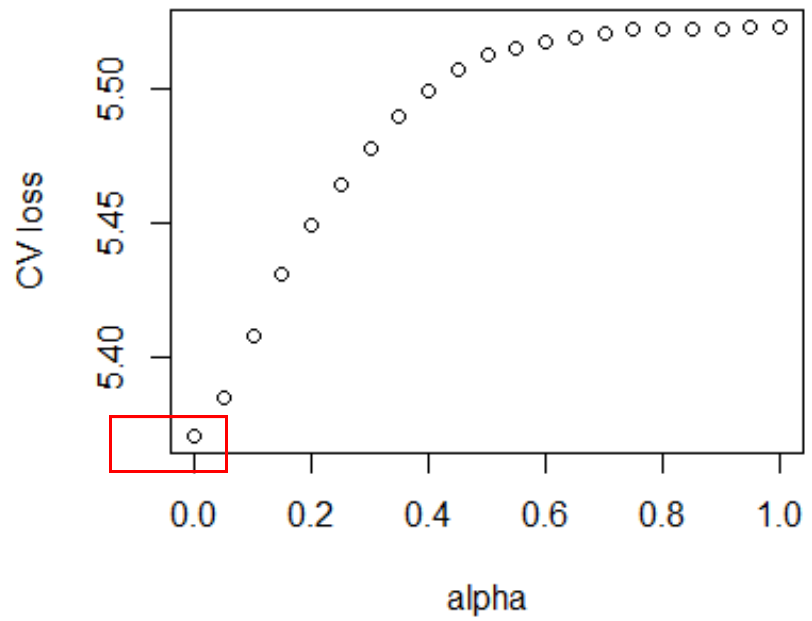
- We try out a number of different combinations of hyper-parameters
- For each hyper-parameter combination we calculate cross-validate MSE
- Optimal combination has lowest cross-validated MSE

Estimating ElasticNet with cva.glmnet

```
# ------------------------------------------------------
# ElasticNet Model
# ------------------------------------------------------
enet_mod <- cva.glmnet(hwy ~ .,
                       data = mpg_clean,
                       alpha = seq(0,1, by = 0.05))

plot(enet_mod)


minlossplot(enet_mod,
            cv.type = "min")
```



- cva.glmnet will estimate a variety of elasticNet models varying alpha from 0 (all ridge) to 1 (all lasso)
- We must specify a sequence of alphas (between zero and 1) to estimate
- The function minlossplot() shows us how cross-validated MSE varies as we change alpha
- This plot reveals the minimum alpha value is at alpha = 0 or full lasso

Helper Functions for ElasticNet

```
# Use this function to find the best alpha
get_alpha <- function(fit) {
  alpha <- fit$alpha
  error <- sapply(fit$modlist,
                  function(mod) {min(mod$cvm)})
  alpha[which.min(error)]
}

# Get all parameters.
get_model_params <- function(fit) {
  alpha <- fit$alpha
  lambdaMin <- sapply(fit$modlist, `[[`, "lambda.min")
  lambdaSE <- sapply(fit$modlist, `[[`, "lambda.1se")
  error <- sapply(fit$modlist, function(mod) {min(mod$cvm)})
  best <- which.min(error)
  data.frame(alpha = alpha[best], lambdaMin = lambdaMin[best],
             lambdaSE = lambdaSE[best], eror = error[best])
}
```

```
> best_alpha <- get_alpha(enet_mod)
> print(best_alpha)
[1] 0
> get_model_params(enet_mod)
  alpha lambdaMin lambdaSE     eror
1     0 0.4551619 3.867754 5.370625
>
```

```
# extract the best model object
best_mod <- enet_mod$modlist[[which(enet_mod$alpha == best_alpha)]]
```

- cva.glmnet returns a list of models, one for every value of alpha
- These functions will extract the optimal alpha and model parameters
- This code is a little gnarly but will extract the best elasticNet model at the optimal value of lambda
- We can treat best_mod like any cv.glmnet object

Lab Time!

```
# ------------------------------------------------------------
# Exercises
# ------------------------------------------------------------

# 1. Load the semiconductor dataset and split into testing and
#    training sets
semi <- read_csv('https://raw.githubusercontent.com/TaddyLab/MBAcourse/master/example
semi_split <- initial_split(semi, 0.9)
semi_train <- training(semi_split)
semi_test <- testing(semi_split)

# 2. Execute the code below to generate a sequence of alphas

alpha_list <- seq(0,1, by = 0.1)

# 3. Estimate an elasticnet model using the semi_train dataset predicting
#    FAIL as a function of all other variables in the dataset.
#    Be sure to use the cva.glmnet function and pass the alpha_list.
#    Store this model as enet_mod2

# 4. In your own words, describe how the vector alpha_list impacts the
#    elasticnet model

# 5. Call the plot function against the enet_mod2 and describe the results

# 6. Call the function minlossplot with the option 'cv.type = "min"'
#    and describe the results.

# 7. Does the above plot suggest you should use a ridge or lasso model?

# 8. Call the functions 'get_model_params()' and 'get_alpha()' on the elasticnet
#    model and describe the results.

# 9. Use the code to extract the best model from the enet_mod2 list and store
#    this as best_mod2.
```

Class 15 Summary

- Lasso and Ridge both penalize magnitude of coefficients, resulting in a less overfit model
- The key parameter is lambda, the amount of shrinkage applied
- We cross-validate to select the optimal lambda
- Lasso penalizes coefficient exactly to zero and acts a variable selector.
- When to use Ridge vs Lasso? Use Lasso if you want fewer variables, desire “sparsity” or optimal model interpretation
- ElasticNet combines Ridge and Lasso penalty and often performs the best.