Class 14: Lasso

MGSC 310

Prof. Jonathan Hersh

Class 14: 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

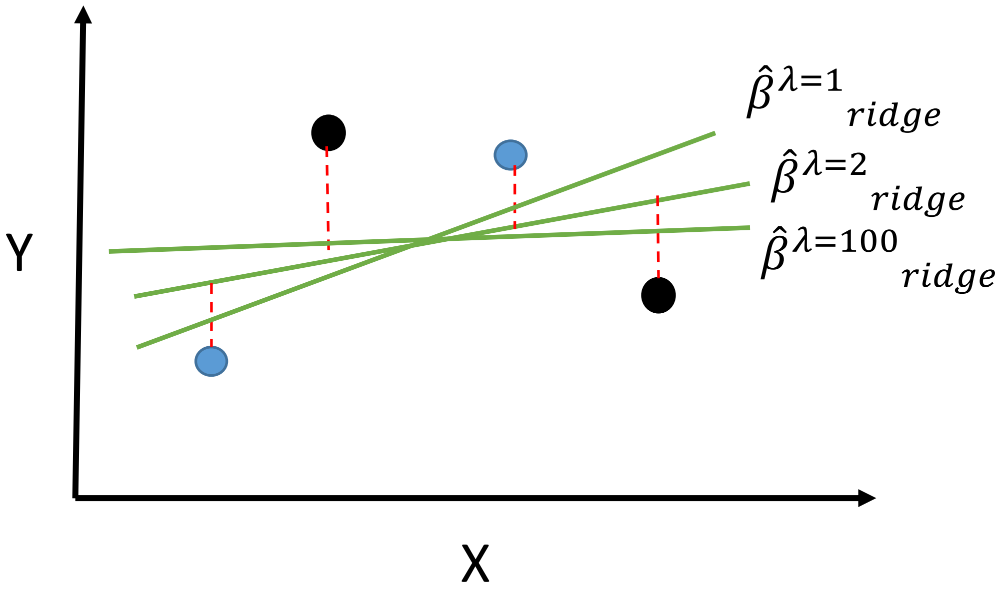
- Extra Instructor Office Hours Friday 11 – 12:30

Class 14: Outline

1. Ridge Regression Review
2. Lasso Regression Algorithm
3. Lasso Regression in R
4. Ridge Regression Lab
5. Comparing Ridge vs Lasso

Recall: Ridge Regression

$\hat{\beta}_{ridge}$ $minimizes: residuals + \lambda \cdot (slope)^2$



So how do we choose λ?

In practice we estimate a many models with many different values of λ

We pick a min and max lambda (say 0 and 100), then choose some points in-between

Optimal λ^* minimizes cross-validated error

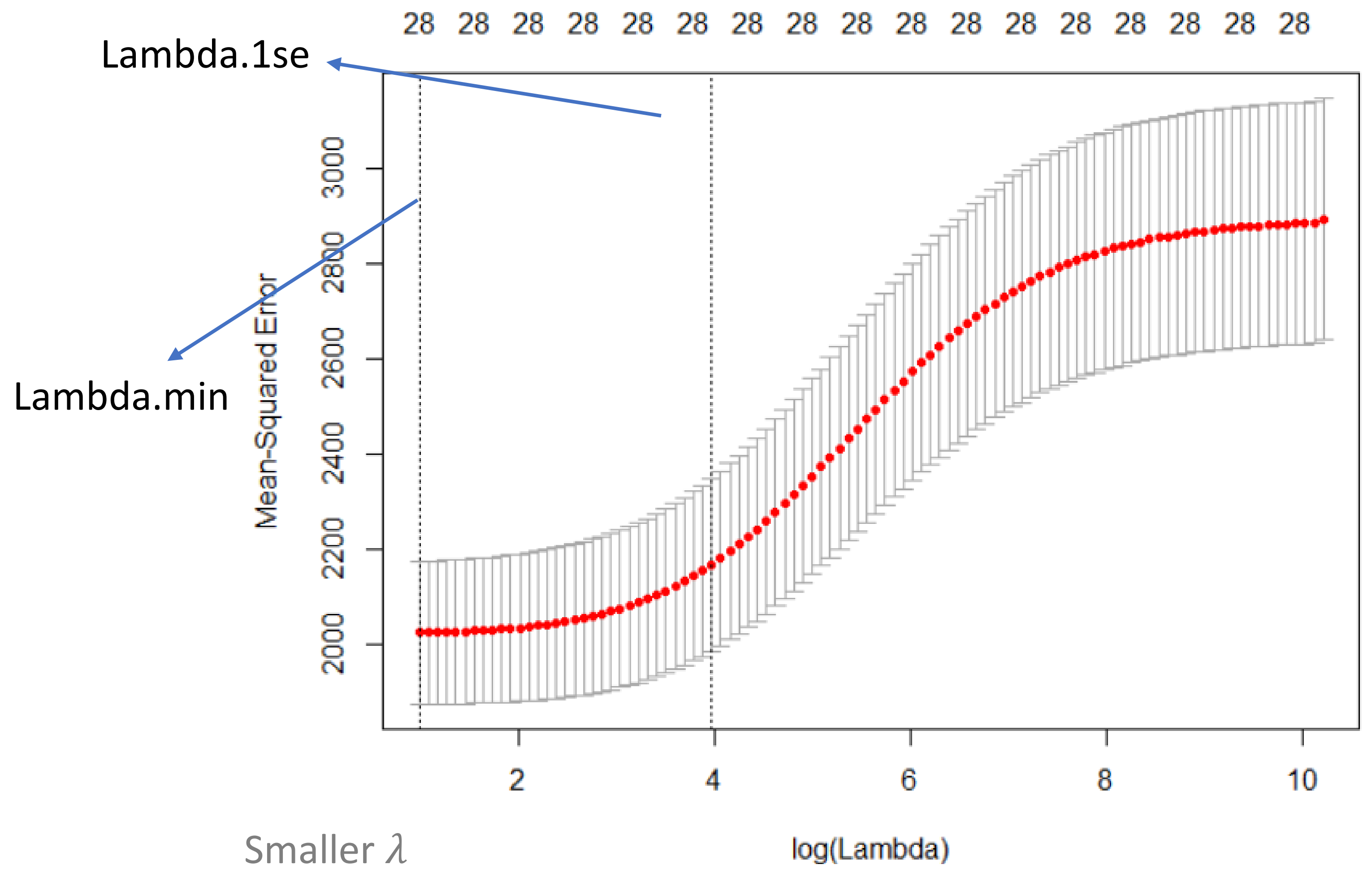
Ridge Model with glmnetUtils

```r
# estimate a Ridge model using glmnet
# note if you get an error make sure you
# have loaded glmnetUtils
ridge_mod <- cv.glmnet(hwy ~ .,
                       data = mpg_clean,
                       # note alpha = 0 sets ridge!
                       alpha = 0)
```

```
> print(ridge_mod$lambda.min)
[1] 0.7247465
> #
> print(ridge_mod$lambda.1se)
[1] 2.665893
```

- cv.glmnet estimates a lasso or ridge model. Automatically performs cross-validation to select optimal lambda!
- We must set alpha = 0 to signify ridge model
- lambda.min stores the value of lambda that minimizes cross-validated error
- lambda.1se stores the value of lambda that minimizes cross-validated error plus one estimated standard error
- Why the difference? Lambda.min gives the best performing value, lambda.1se add extra penalization for more parsimony

Cross-Validated MSE Plot As A Function of Lambda



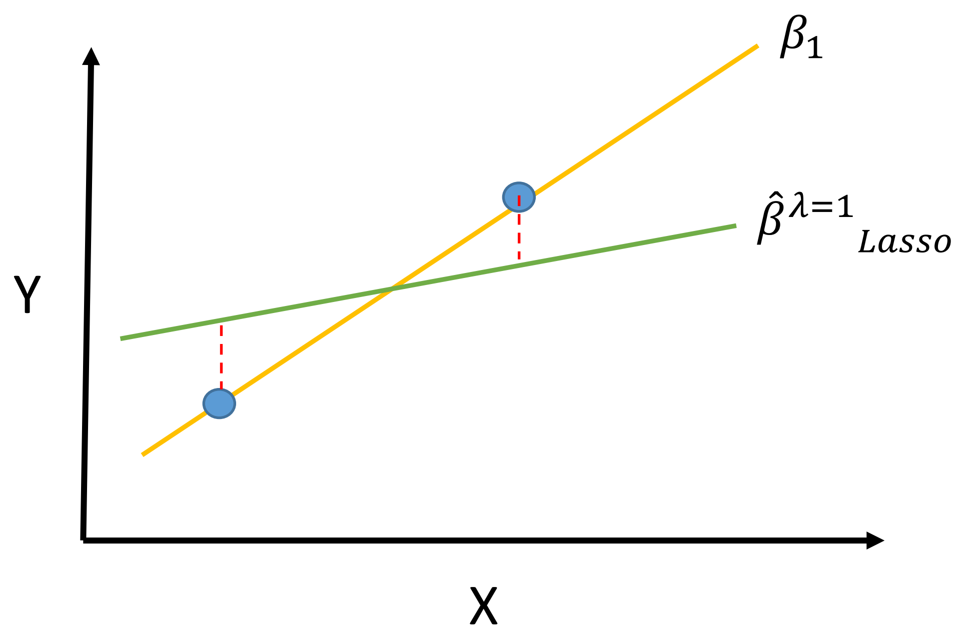
- plot(*model_object*) calls the MSE plot
- This shows how the cross-validated MSE (y-axis) varies as we increase lambda (penalization)
- Model defaults to lambda.1se but either can be appropriate

Smaller λ
Less penalization
Coefficients more
Similar to OLS

Larger λ
More penalization
Smaller coefficients

Lasso Regression Idea

$$\hat{\beta}_{Lasso}\ minimizes:\ residuals + \lambda \cdot (\,|\beta_1| + |\beta_2| + \cdots + |\beta_k|\,)$$



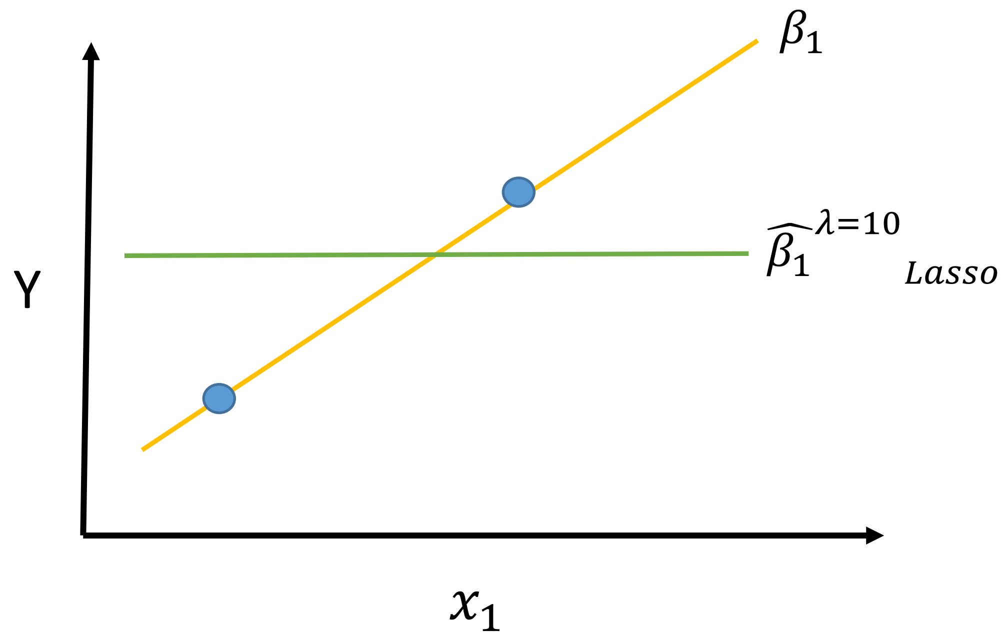
Lasso minimizes the residuals plus lambda times the absolute value of the slope coefficients

Lasso coefficients are still smaller than OLS coefficients

Lasso still accepts a little bias for (hopefully) less variance

Key Lasso Property: Variable Selection

$$\hat{\beta}_{Lasso}\ minimizes:\ residuals + \lambda \cdot (\ |\cancel{\beta_1}| + |\beta_2|)$$



For large values of λ, some slope coefficients will be chosen to be exactly zero

E.g. if we set $\lambda = 10$, maybe $\beta_1^{lasso} = 0$ but $\beta_1^{lasso} != 0$

If that happens we effectively remove β_1 from the equation, and we have a variable selection algorithm

Lasso Model with glmnetUtils

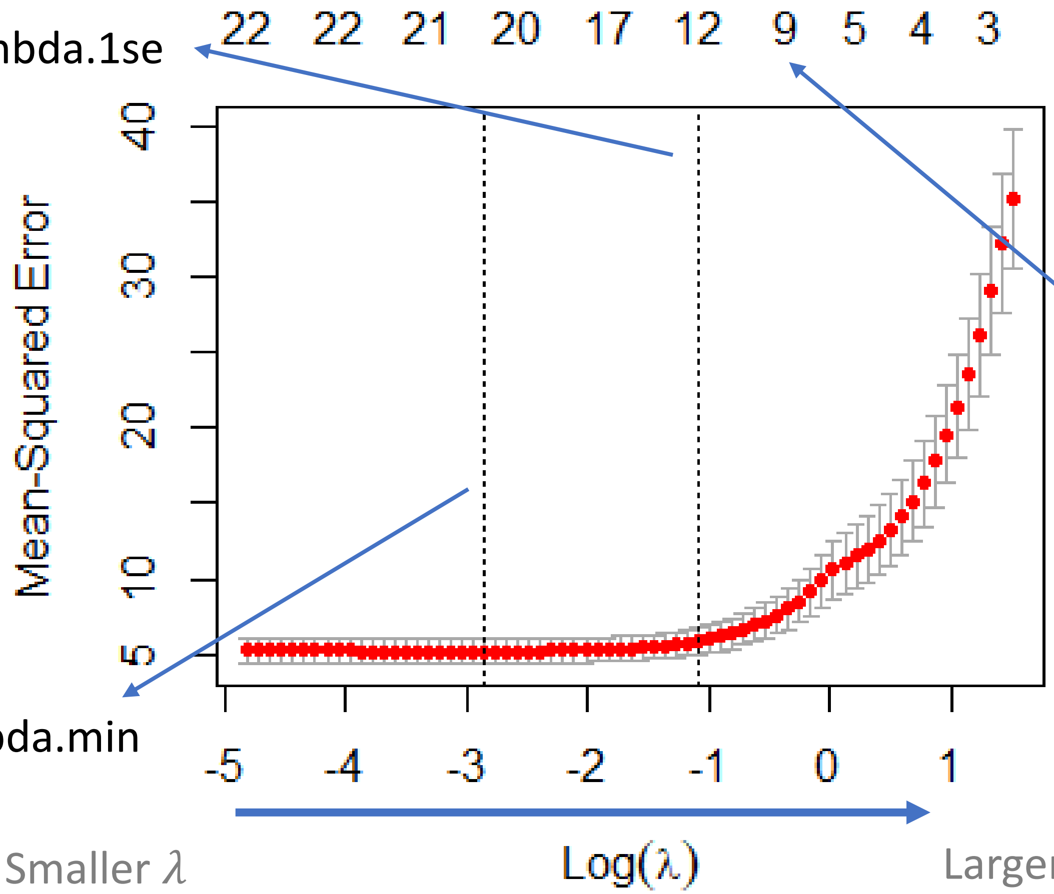
```r
# note cv.glmnet automatically performs
# k-fold cross-validation
lasso_mod <- cv.glmnet(hwy ~ .,
                       data = mpg_clean,
                       # note alpha = 1 sets lasso!
                       alpha = 1)
```

```
> print(lasso_mod$lambda.min)
[1] 0.05743492
> #
> print(lasso_mod$lambda.1se)
[1] 0.3363975
```

- We estimate lasso using cv.glmnet
- Here we must set alpha = 1 to estimate Lasso
- Again we get values of lambda.min (minimizes cross-validated MSE) and lambda.1se (minimum plus 1 SE)

Lasso Cross-Validated MSE Plot

Lambda.1se



Lambda.min

Smaller λ
Less penalization
Coefficients more
Similar to OLS

Larger λ
More penalization
Smaller coefficients

- Lasso MSE plot is very similar
- Top number indicates number of non-zero coefficients for each value of lambda!
- E.g. at this value of lambda, 9 variables are non-zero
- We still have lambda.1se and lambda.min vertical dashed lines but lambda.1se generally shrinks more variables to exactly zero!

Lasso Coefficient Vector

```
> lasso_coefs <- data.frame(
+   `lasso_min` = coef(lasso_mod, s = lasso_mod$lambda.min) %>%
+     round(3) %>% as.matrix() %>% as.data.frame(),
+   `lasso_1se` = coef(lasso_mod, s = lasso_mod$lambda.1se) %>%
+     round(3) %>% as.matrix() %>% as.data.frame()
+ ) %>% rename(`lasso_min` = 1, `lasso_1se` = 2)
> print(lasso_coefs)
                      lasso_min lasso_1se
(Intercept)            -261.494   -99.422
manufacturerchevrolet     0.722     0.000
manufacturerdodge        -0.971    -1.026
manufacturerford         -0.682     0.000
manufacturertoyota        0.172     0.000
manufacturervolkswagen   -0.162     0.000
manufacturerOther         0.000     0.000
displ                    -0.645    -0.646
year                      0.148     0.067
cyl                      -1.046    -1.124
transauto(l4)            -0.327    -0.329
transauto(l5)             0.000     0.000
transmanual(m5)           0.462     0.000
transOther                0.000     0.000
drv4                     -2.319    -2.379
drvf                      0.190     0.485
drvr                      0.000     0.000
flc                       4.977     1.216
fld                       8.752     6.756
fle                      -4.641    -3.088
flp                       0.000     0.000
flr                       0.000     0.000
classcompact              0.139     0.000
classmidsize              0.000     0.000
classpickup              -3.922    -2.696
classsubcompact           0.095     0.000
classsuv                 -4.089    -2.933
classOther               -0.874     0.000
```

- We can build the lasso coefficient vector as we did for Ridge
- Note the higher the lambda (lambda.1se > lambda.min) the more variables that are “shrunk” to zero
- Lasso sets coefficients = 0 if they do not improve the cross-validated MSE
- Ridge will just shrink these coefficients towards zero but will never set them exactly = 0

Lasso Lab

```
# ------------------------------------------------------------
#  Lab Exercises
# ------------------------------------------------------------
# 1. Load the semiconductor dataset and split into testing and
#    training sets
semi <- read_csv('https://raw.githubusercontent.com/TaddyLab/MBAcourse/master/examples/se
semi_split <- initial_split(semi, 0.6)
semi_train <- training(semi_split)
semi_test <- testing(semi_split)

# 2. Estimate a lasso model using the training data, with FAIL as the
#    outcome variable, and every other variable in the data frame as the predictors.
#    Store this model as lasso_mod2

# 3. What does the option "alpha = 1" in cv.glmnet mean?

# 4. What does the option "family = "binomial"" mean?

# 5. How is cv.glmnet different from the function glmnet()?
```

Lasso Lab Continued

```
# 6. Call the plot function against lasso_mod2.
#    Describe the plot as well as the two vertical dashed lines

# 7. Store the lambda.1se Lasso coefficients into a data frame
#    called coef_lasso_1se and print the coefficients

# 8. Store the lambda.min Lasso coefficients into a data frame
#    called coef_lasso_min and print the coefficients

# 9. How many varaibles are non-zero using lambda.min and lambda.1se?
#     Why are they different? When would you use one versus another?

# 10. If you have time, use the coefpath against the lasso mod to see which
#     variables are shrunk first to zero as we increase lambda.
```

Another way to write Lasso

Lasso

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

Lasso with two variables

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

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

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

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

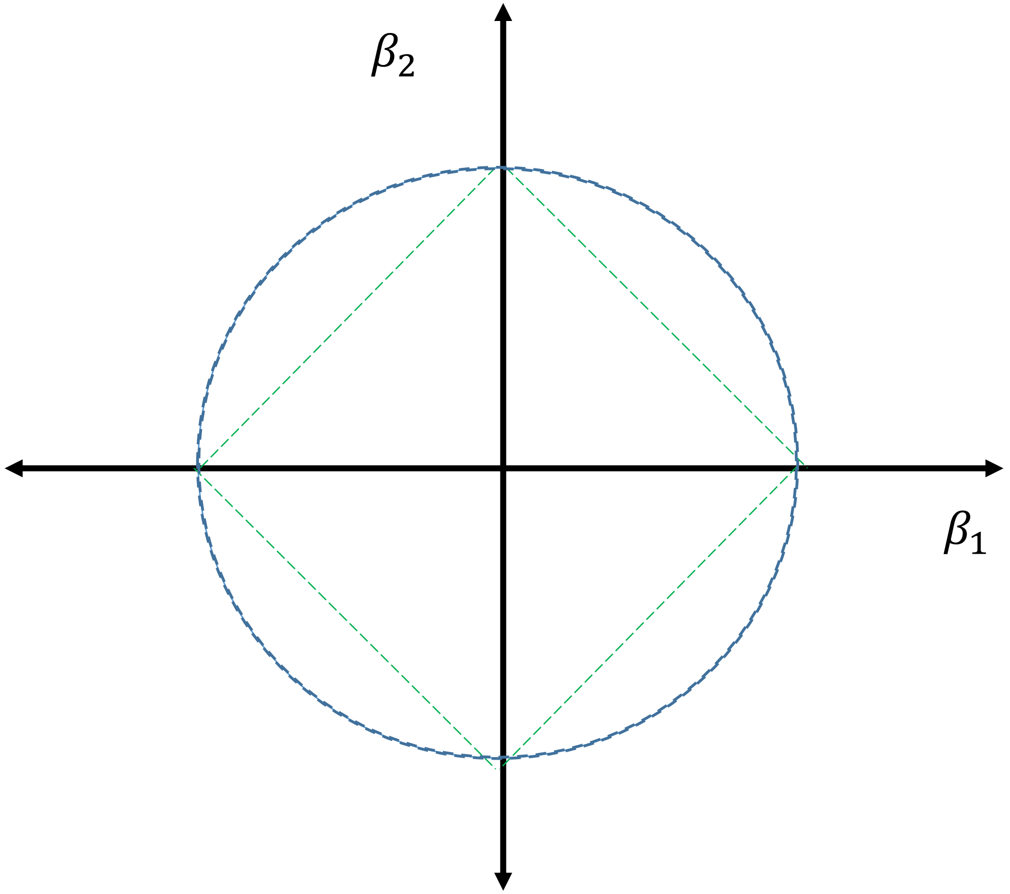
Ridge Versus Lasso Penalty

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

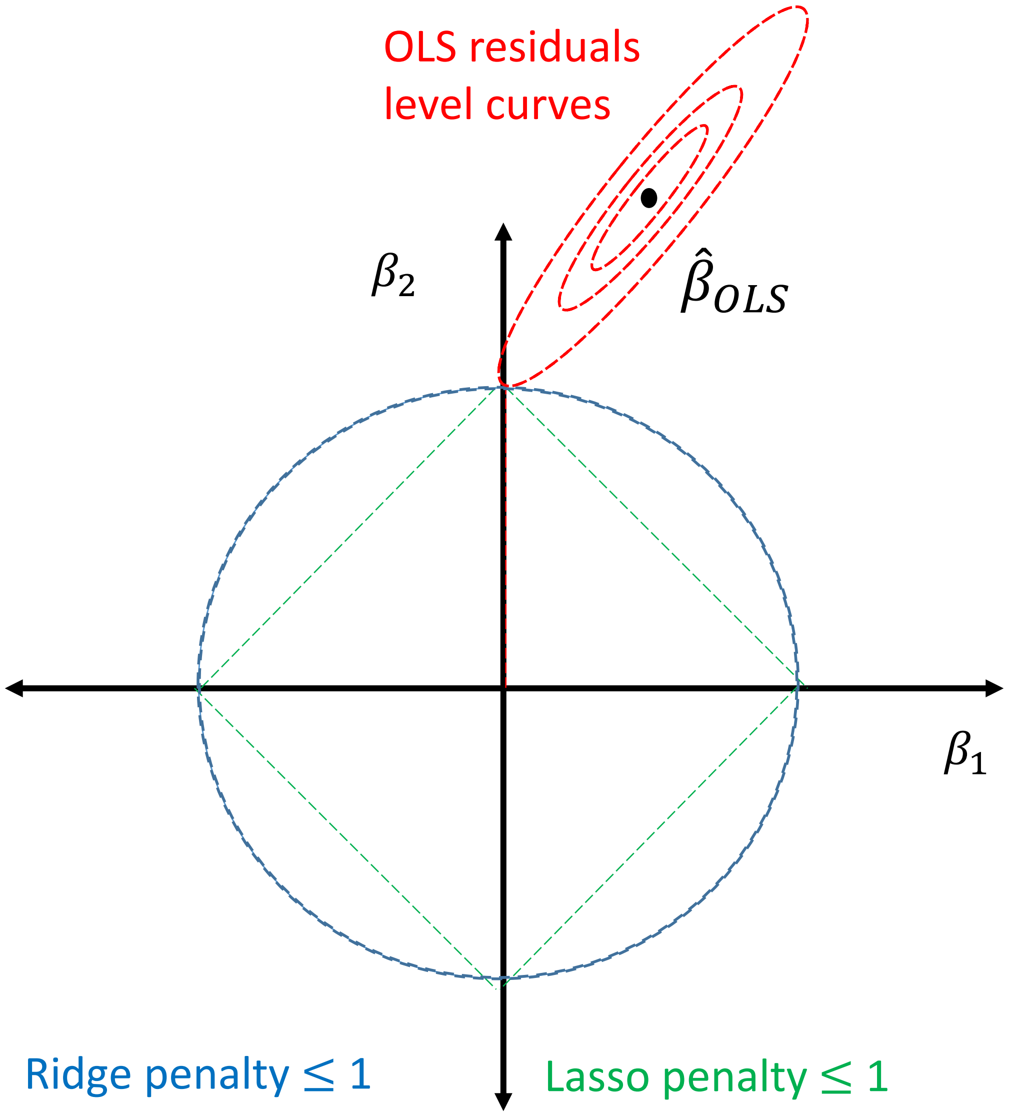
What do these look like graphically?



Ridge penalty ≤ 1

Lasso penalty ≤ 1

Ridge and Lasso Equations Redux



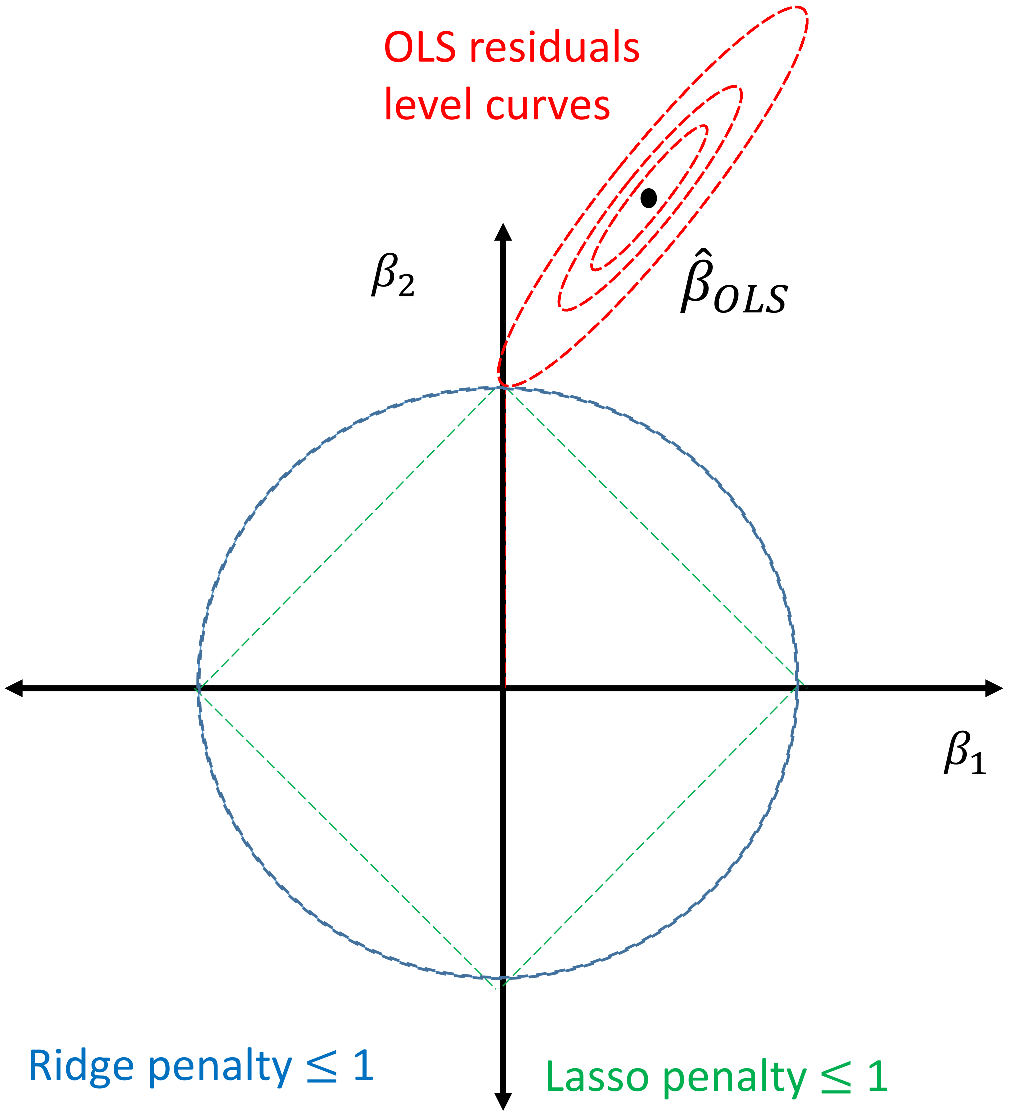
Suppose the optimal OLS beta is this point in black

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



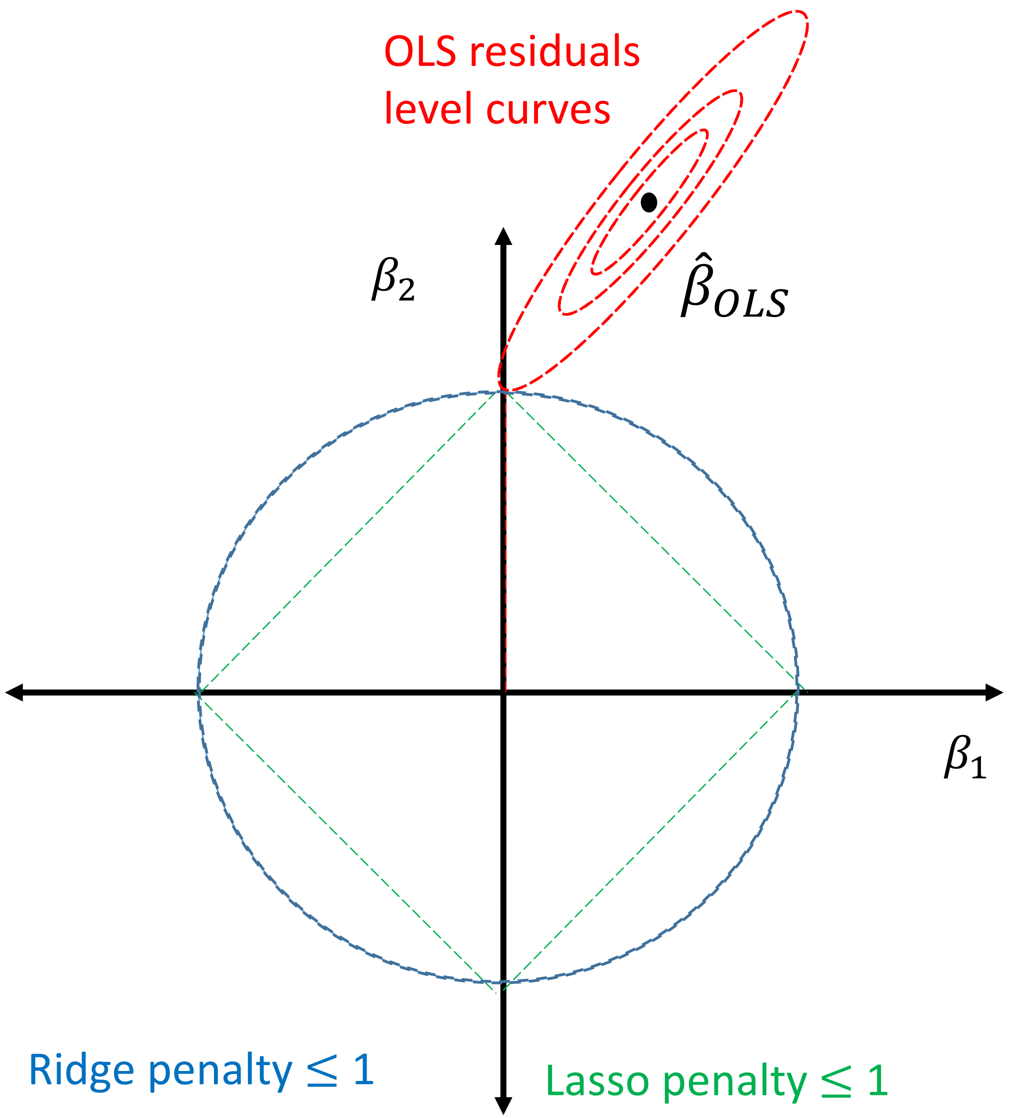
Graphically what the ridge equation is asking is: “find the lowest residual level curve while staying within the blue circle”

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

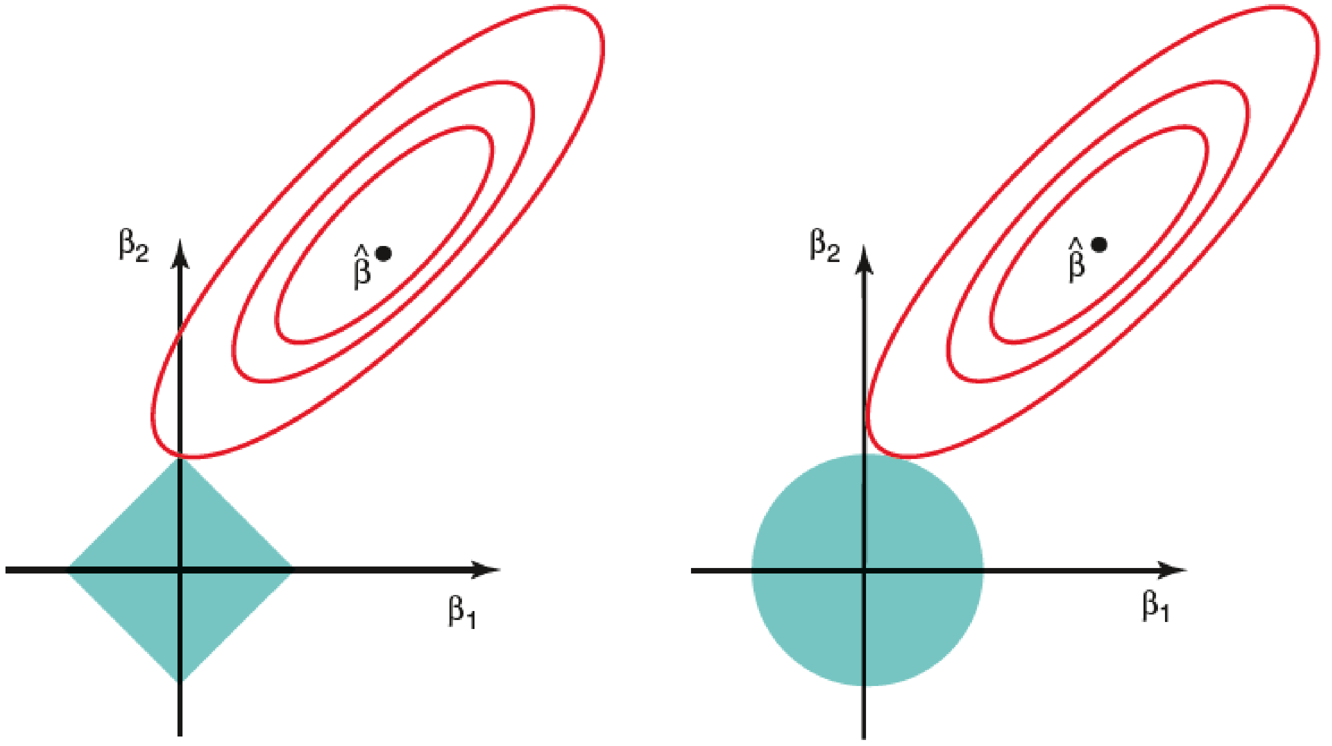


FIGURE 6.7. *Contours of the error and constraint functions for the lasso* (left) *and ridge regression* (right). *The solid blue areas are the constraint regions, $|\beta_1| + |\beta_2| \leq s$ and $\beta_1^2 + \beta_2^2 \leq s$, while the red ellipses are the contours of the RSS.*

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

Class 7 Summary

- Lasso penalizes coefficients both the magnitude and number of coefficients
- This creates a natural variable selection mechanism where the model "selects" certain variables and sets others exactly equal to zero
- A Lasso model (like ridge) is actually many models, each indexed by a value of lambda (the amount of shrinkage penalization desired)
- Two useful values of lambda are lambda.min and lambda.1se, the former being the lambda that minimizes cross-validated error, and the latter being the former plus one standard error
- We estimate a lasso model using cv.glmnet, specifying the option "alpha = 1"
- When to use Ridge vs Lasso? Use Ridge if we think many variables matter a little, Lasso if only a few variables matter a lot
- Next class: ElasticNet with both Ridge and Lasso penalty!