Regularized Regression

In a regression framework the goal is to model the relationship between a variable of interest, y, and a set of covariates $\mathcal{X}$. Using the normal distributional assumptions then the joint distribution of the observed data, given the data $x_1, \ldots, x_n$ along with β and σ^2 can be written as:

$$\begin{aligned} p(y_1, \ldots, y_n | \tilde{x}_1, \ldots, \tilde{x_n}, \tilde{\beta}, \sigma^2) &= \prod_{i=1}^{n} p(y_i | \tilde{x}_i, \tilde{\beta}, \sigma^2) & (1) \\ &= (2\pi\sigma^2)^{-n/2} \exp\left[-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(y_i - \tilde{\beta}^T \tilde{x}_i\right)^2\right]. & (2) \end{aligned}$$

Note this is the same as the sampling, or generative model, that we have seen earlier in class.

The model is often formulated using matrix expressions and a multivariate normal distribution. Let

$$\tilde{y} | X, \tilde{\beta}, \sigma^2 \sim MVN(X\tilde{\beta}, \sigma^2 I), \tag{3}$$

where $\tilde{y}$ is an $n \times 1$ vector of the responses, X is an $n \times p$ matrix of the covariates where the i^{th} row is $\tilde{x}_i$, and I is a $p \times p$ identity matrix.

In a classical setting, typically least squares methods are used to compute the values of the covariates in a regression setting. Note in a normal setting these correspond to maximum likelihood estimates. Specifically, we seek to minimize the sum of squared residuals (SSR), where $SSR(\tilde{\beta}) = \left(\tilde{y} - X\tilde{\beta}\right)^T \left(\tilde{y} - X\tilde{\beta}\right)$.

Thus we will take the derivative of this function with respect to β to minimize this expression.

$$\begin{aligned} \frac{d}{d\tilde{\beta}} SSR(\tilde{\beta}) &= \frac{d}{d\tilde{\beta}} \left(\tilde{y} - X\tilde{\beta}\right)^T \left(\tilde{y} - X\tilde{\beta}\right) & (4) \\ &= \frac{d}{d\tilde{\beta}} \left(\tilde{y}^T \tilde{y} - 2\tilde{\beta}^T X^T \tilde{y} + \tilde{\beta}^T X^T X \tilde{\beta}\right) & (5) \\ &= -2X^T \tilde{y} + 2X^T X \tilde{\beta} & (6) \\ \text{then set} = 0 \quad \text{which implies} \quad & X^T X \beta = X^t \tilde{y} & (7) \\ \text{and} \quad & \tilde{\beta} = (X^T X)^{-1} X^t \tilde{y}. & (8) \end{aligned}$$

This value is the OLS estimate of $\tilde{\beta}_{OLS} = (X^T X)^{-1} X^T \tilde{y}$. Under the flat prior $p(\tilde{\beta}) \propto 1$, $\tilde{\beta}_{OLS}$ is the mean of the posterior distribution.

Bayesian Modeling and Regularization Ordinary Least Squares (OLS) regression can be written as:

$$\hat{\tilde{\beta}}_{OLS} = \text{arg min}_{\hat{\tilde{\beta}}} \; ||\tilde{y} - X\hat{\tilde{\beta}}||_2^2 \rightarrow \hat{\tilde{\beta}} = (X^T X)^{-1} X^T \tilde{y},$$

where $||\tilde{x}||_p = (|x_1|^p + \cdots + |x_m|^p)^{1/p}$ is an LP norm. So the L2 norm is $||\tilde{x}||_2 = \sqrt{x_1^2 + \ldots x_m^2}$.

Recall ridge regression is a form of penalized regression such that:

$$\hat{\tilde{\beta}}_R = \text{arg min}_{\hat{\tilde{\beta}}_R} \; ||\tilde{y} - X\hat{\tilde{\beta}}||_2^2 + \lambda||\hat{\tilde{\beta}}_R||_2^2 \rightarrow \hat{\tilde{\beta}}_R = \left(X^T X + \lambda I\right)^{-1} X^T \tilde{y}, \tag{9}$$

where λ is a tuning parameter that controls the amount of shrinkage.

- As λ gets large all of the values are shrunk toward 0.
- As λ goes to 0, the ridge regression estimator results in the OLS estimator.
- It can be shown that ridge regression results better predictive ability than OLS by reducing variance of the predicted values at the expense of bias. Note that typically the X values are assumed to be standardized, so that the intercept is not necessary.
- **Q:** How do we choose λ?

  An alternative form of penalized regression is known as Least Absolute Shrinkage and Selection Operator (LASSO). The LASSO uses an L1 penalty such that:

  $$\hat{\tilde{\beta}}_L = \text{arg min}_{\hat{\tilde{\beta}}} \; ||\tilde{y} - X\hat{\tilde{\beta}}||_2^2 + \lambda||\hat{\tilde{\beta}}_L||_1, \tag{10}$$

  the L1 penalty results in $||\tilde{x}||_1 = |x_1| + \cdots + |x_m|$, which minimizes the absolute differences.

  The nice feature of LASSO, relative to ridge regression, is that coefficients are shrunk to 0 providing a way to do *variable selection.*

  One challenge with LASSO is coming up with proper distributional assumptions for inference about variables.

  Consider the following prior $p(\tilde{\beta}) = N(0, I_p\tau^2)$. How does this relate to ridge regression? First compute the posterior distribution for $\tilde{\beta}$.

  $$\begin{aligned} p(\tilde{\beta}|-) &\propto \exp\left[-\frac{1}{2}\left(\frac{1}{\sigma^2}\tilde{\beta}^T X^T X\tilde{\beta} - \frac{1}{\sigma^2}\tilde{\beta}^T X^T\tilde{y} + \tilde{\beta}^T\frac{I_p}{\tau^2}\tilde{\beta}\right)\right] \\ &\propto \exp\left[-\frac{1}{2}\left(\frac{1}{\sigma^2}\tilde{\beta}^T\left(X^T X + \frac{\sigma^2}{\tau^2}I_p\right)\tilde{\beta} - \frac{1}{\sigma^2}\tilde{\beta}^T X^T\tilde{y}\right)\right] \end{aligned}$$

  Thus $Var(\tilde{\beta}|-) = \left(X^T X + \frac{\sigma^2}{\tau^2}I_p\right)^{-1}\sigma^2$ and $E(\tilde{\beta}|-) = \left(X^T X + \frac{\sigma^2}{\tau^2}I_p\right)^{-1} X^t\tilde{y}$.

  Q: does this look familiar?

  Define: $\lambda = \frac{\sigma^2}{\tau^2}$. How about now? This is essentially the ridge regression point estimate.

Simulate Regression Data

```r
n <- 50
p <- 10

X <- cbind(matrix(runif(n*p,min = -1, max = 1), n, p))
beta <- rep(0, p )
beta[c(1,2)] <- c(3, 3)

y <- rnorm(n, mean = X %*% beta, sd = 3)
y_center <- y - mean(y)
summary(lm(y_center~X))
```

```
##
## Call:
## lm(formula = y_center ~ X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4111 -1.9258  0.0543  1.6718  9.0089
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.3999     0.5130  -0.780 0.440360
## X1            2.3536     0.8893   2.646 0.011669 *
## X2            3.8748     0.9215   4.205 0.000148 ***
## X3           -0.3988     1.0088  -0.395 0.694780
## X4            0.3629     0.8899   0.408 0.685608
## X5           -0.2413     0.8819  -0.274 0.785876
## X6           -1.2707     0.9899  -1.284 0.206873
## X7            0.8479     0.8304   1.021 0.313485
## X8           -0.7527     1.0737  -0.701 0.487448
## X9           -0.5595     1.0293  -0.544 0.589854
## X10          -0.6095     0.9993  -0.610 0.545485
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.401 on 39 degrees of freedom
## Multiple R-squared:  0.4936, Adjusted R-squared:  0.3638
## F-statistic: 3.801 on 10 and 39 DF,  p-value: 0.001227
```

Ridge Regression

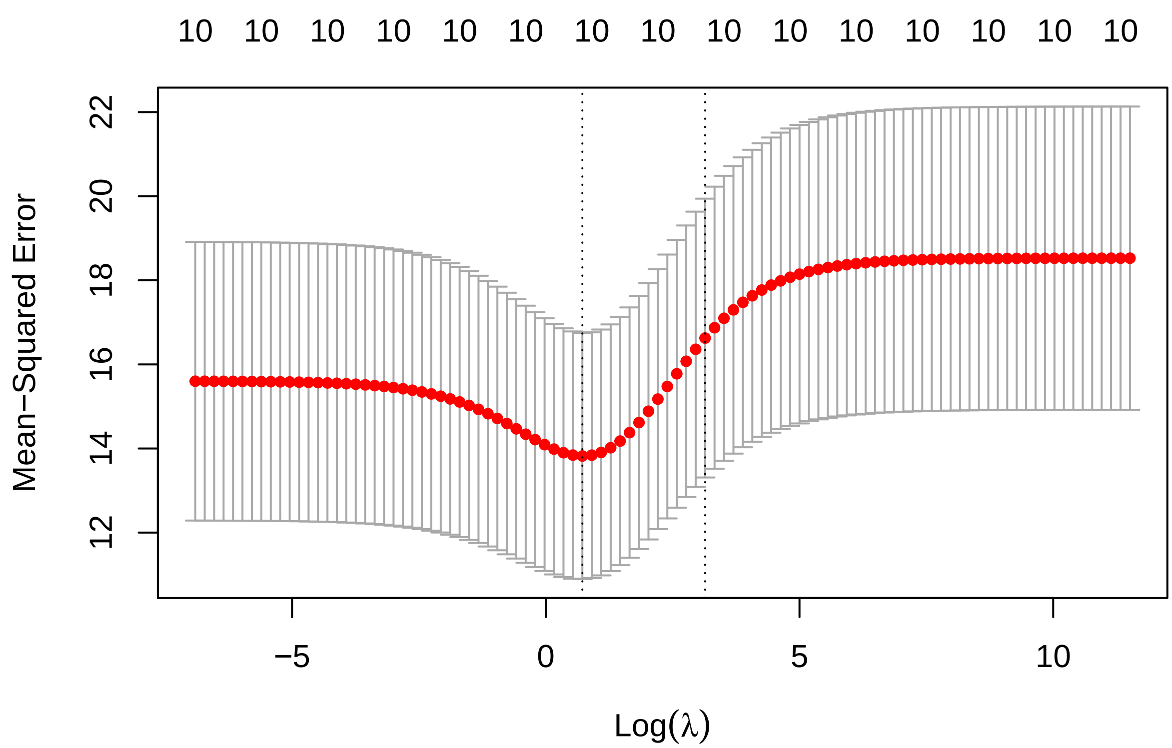
```r
ridge_fit <- glmnet(X, y_center, alpha = 0, lambda = 0)
t(coef(ridge_fit))
```

```
## 1 x 11 sparse Matrix of class "dgCMatrix"

##    [[ suppressing 11 column names '(Intercept)', 'V1', 'V2' ... ]]

##
## s0 -0.3998859 2.354099 3.875362 -0.3979093 0.3623919 -0.2414221 -1.271342
##
## s0 0.8477595 -0.7520009 -0.5592078 -0.6091662
```

```r
t(coef(glmnet(X, y_center, alpha = 0, lambda = 1e6)))
```

```
## 1 x 11 sparse Matrix of class "dgCMatrix"
##    [[ suppressing 11 column names '(Intercept)', 'V1', 'V2' ... ]]
##
## s0 -1.510178e-06 1.101924e-05 1.722992e-05 -7.840096e-06 2.256134e-06
##
## s0 -2.294676e-06 -4.885313e-06 1.179919e-06 -3.950416e-06 -1.836847e-06
##
## s0 -3.01518e-06
```

```r
ridge_fit_cv <- cv.glmnet(X, y_center, alpha = 0)
lambdas_seq <- 10^seq(-3, 5, length.out = 100)
ridge_cv <- cv.glmnet(X, y_center, alpha = 0, lambda = lambdas_seq)
plot(ridge_cv)
```




```r
print(ridge_cv)
```

```
##
## Call:  cv.glmnet(x = X, y = y_center, lambda = lambdas_seq, alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min  2.057    59   13.82 2.928      10
## 1se 23.101    46   16.63 3.315      10
```

```r
ridge_fit <- glmnet(X, y_center, alpha = 0, lambda = ridge_cv$lambda.min)
t(coef(ridge_fit))
```

```
## 1 x 11 sparse Matrix of class "dgCMatrix"
##    [[ suppressing 11 column names '(Intercept)', 'V1', 'V2' ... ]]
##
## s0 -0.2631086 1.562949 2.567943 -0.6287719 0.3019515 -0.1498433 -0.7759721
##
## s0 0.462363 -0.6471148 -0.3728943 -0.4438042
```

Lasso

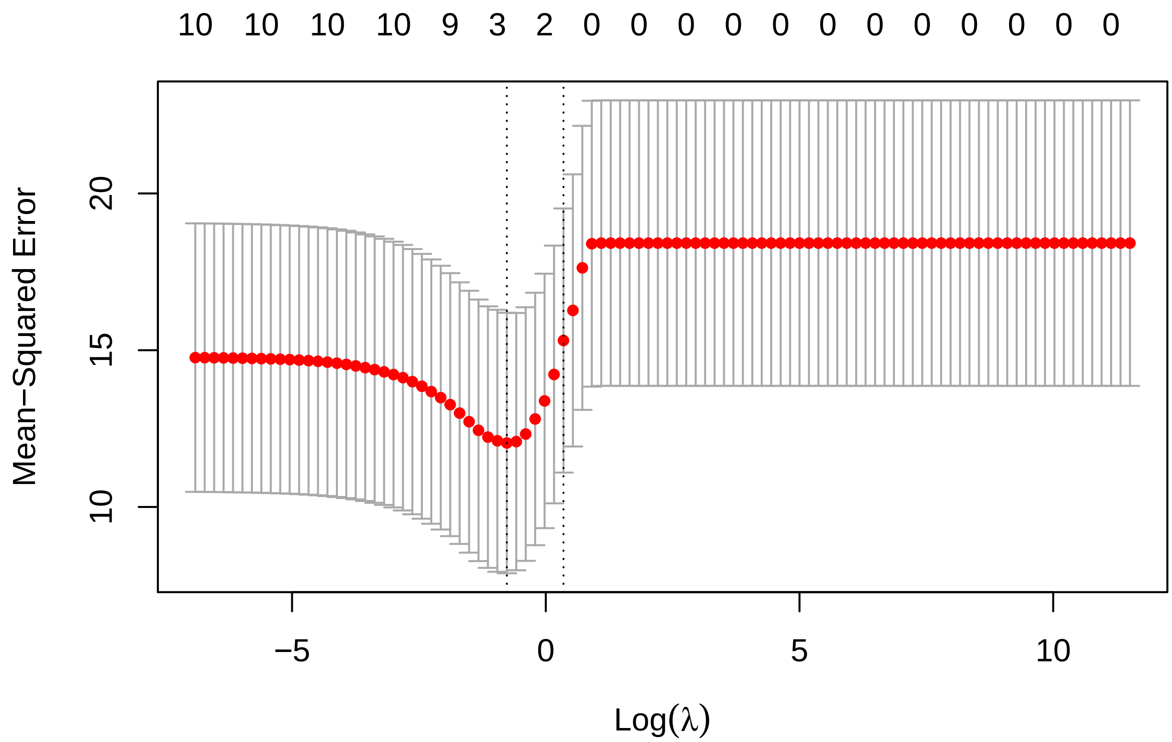
```r
t(coef(glmnet(X, y_center, alpha = 0, lambda = 0)))
```

```
## 1 x 11 sparse Matrix of class "dgCMatrix"
##    [[ suppressing 11 column names '(Intercept)', 'V1', 'V2' ... ]]
##
## s0 -0.3998859 2.354099 3.875362 -0.3979093 0.3623919 -0.2414221 -1.271342
##
## s0 0.8477595 -0.7520009 -0.5592078 -0.6091662
```

```r
t(coef(glmnet(X, y_center, alpha = 1, lambda = 1e6)))
```

```
## 1 x 11 sparse Matrix of class "dgCMatrix"
##    [[ suppressing 11 column names '(Intercept)', 'V1', 'V2' ... ]]
##
## s0 . 0 . . . . . . . . .
```

```r
lasso_cv <- cv.glmnet(X, y_center, alpha = 1, lambda = lambdas_seq)
plot(lasso_cv)
```




```
lasso_fit <- glmnet(X, y_center, alpha = 1, lambda = lasso_cv$lambda.min)
coef(lasso_fit)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                     s0
## (Intercept) -0.1580537
## V1           1.6410982
## V2           3.1995546
## V3           .
## V4           .
## V5           .
## V6          -0.8606924
## V7           .
## V8           .
## V9           .
## V10          .
```