Lecture 10: Regularization

Alan Lee

Department of Statistics
STATS 784 Lecture 10

August 25, 2017

Outline

Today's agenda

In this lecture we discuss the topic of regularization, a way of improving the predictive ability of least squares.

Motivation

For the linear regression model $y = X\beta + \epsilon$:

$$E(\hat{\beta}) = \beta$$

but

$$E(\hat{\beta}^T\hat{\beta}) = \hat{\beta}^T\hat{\beta} + \sigma^2 trace(X^TX)^{-1}.$$

The vector of regression coefficients is too long!

Thus, regularisation: A modification to least squares using a modified criterion that penalizes large coefficients and "shrinks" them towards zero.

Ridge Regression

Instead of minimizing the residual sum of squares, minimize the RSS $||y - Xb||^2$ subject to the constraint

$$\sum_{j=1}^{p} b_j^2 \leq s$$

for a fixed value s. If the least squares estimates $\hat{\beta}_j$ satisfy

$$\sum_{j=1}^{p} \hat{\beta}_j^2 \leq s$$

then this is just least squares, otherwise the fitted coefficients will have shorter length.

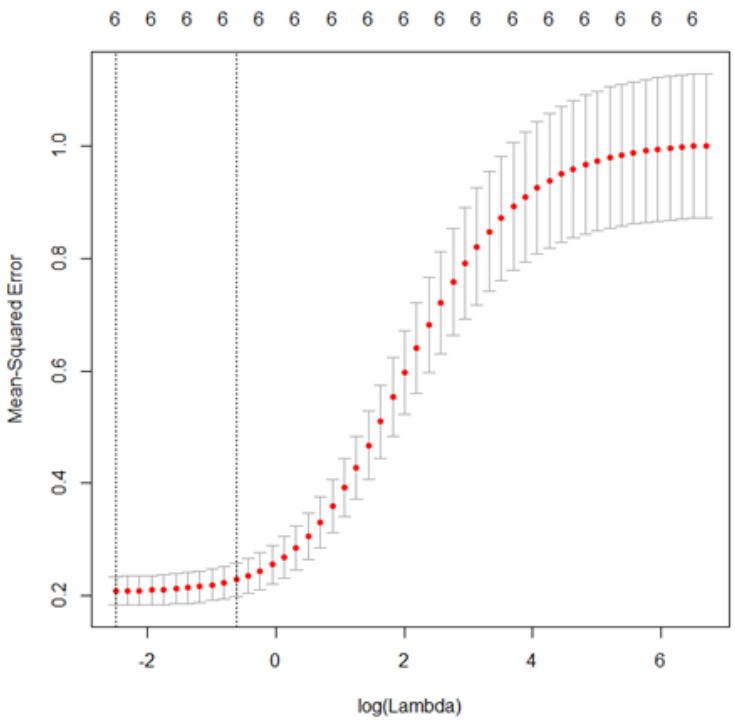
Equivalent formulation

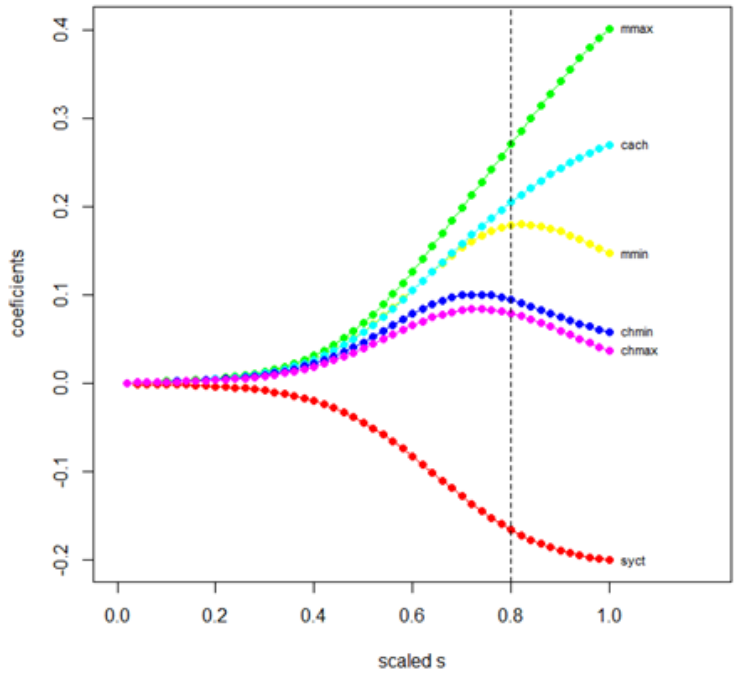
Equivalent to minimizing $||y - Xb||^2 + \lambda||b||^2$ for a fixed value of λ.

The minimizer is

$$\hat{\beta}_{\text{Ridge}} = (X^T X + \lambda I)^{-1} X^T y.$$

Ridge trace


6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
Mean-Squared Error
0.2
0.4
0.6
0.8
1.0
-2
0
2
4
6
log(Lambda)


coeficients
-0.2
-0.1
0.0
0.1
0.2
0.3
0.4
0.0
0.2
0.4
0.6
0.8
1.0
scaled s
mmax
cach
mmin
chmin
chmax
syct

The Lasso

Similar to ridge but we are minimizing the residual sum of squares $||y - Xb||^2$ subject to the (different) constraint

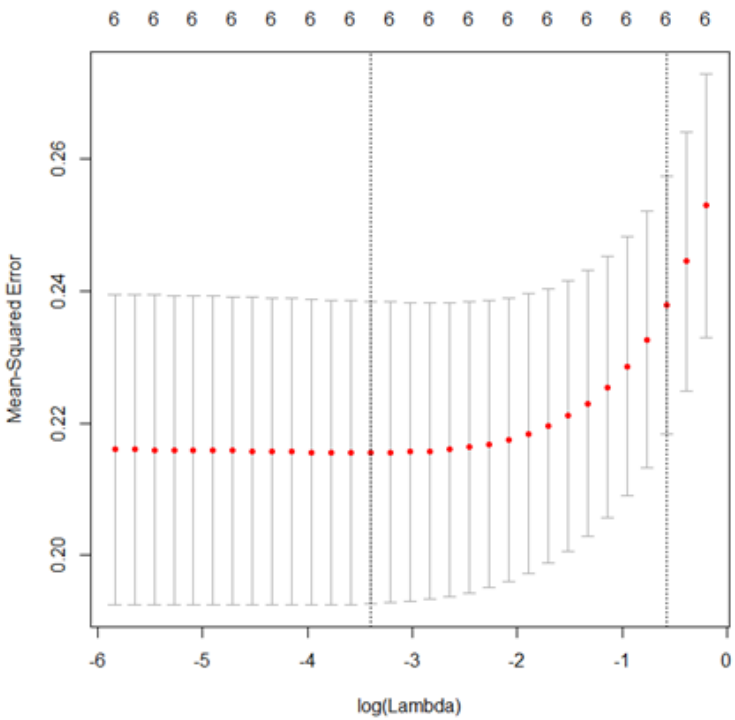
$$\sum_{j=1}^{p} |b_j| \leq s$$

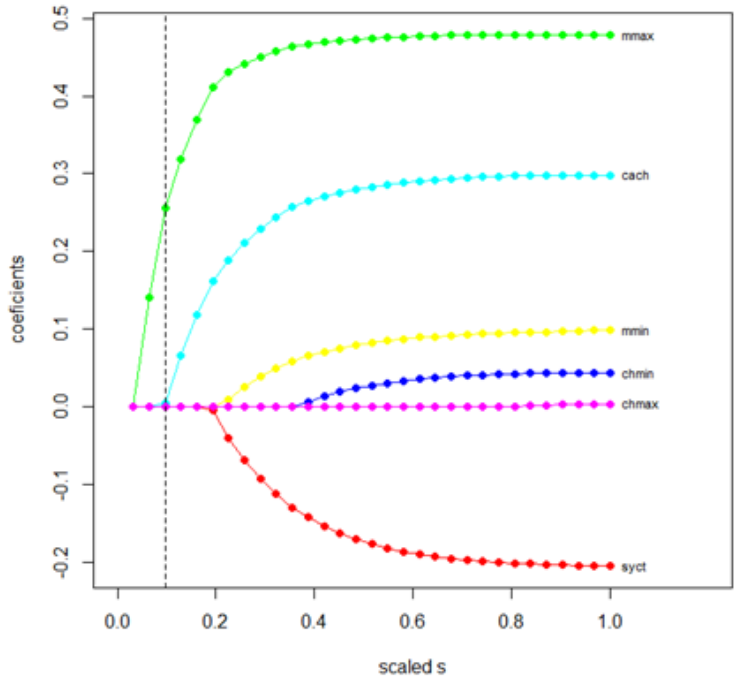
for a fixed value s.

If the LS estimates satisfy this constraint then the lasso is equivalent to LS. Otherwise the lasso coefficient vector has a shorter length.

Some lasso coefficients can be zero. Thus, the lasso is a compromise between ridge (which shrinks but does not set coefficients to 0) and subset selection (which sets some coefficients to 0 but does not shrink).

Lasso trace


6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
Mean-Squared Error
0.20
0.22
0.24
0.26
-6
-5
-4
-3
-2
-1
0
log(Lambda)


coeficients
-0.2
-0.1
0.0
0.1
0.2
0.3
0.4
0.5
mmax
cach
mmin
chmin
chmax
syct
0.0
0.2
0.4
0.6
0.8
1.0
scaled s

Elastic net

Combination of ridge and lasso: minimize

$$||y - Xb||^2 + \lambda_1 \sum_j |b_j| + \lambda_2 \sum_j b_j^2$$

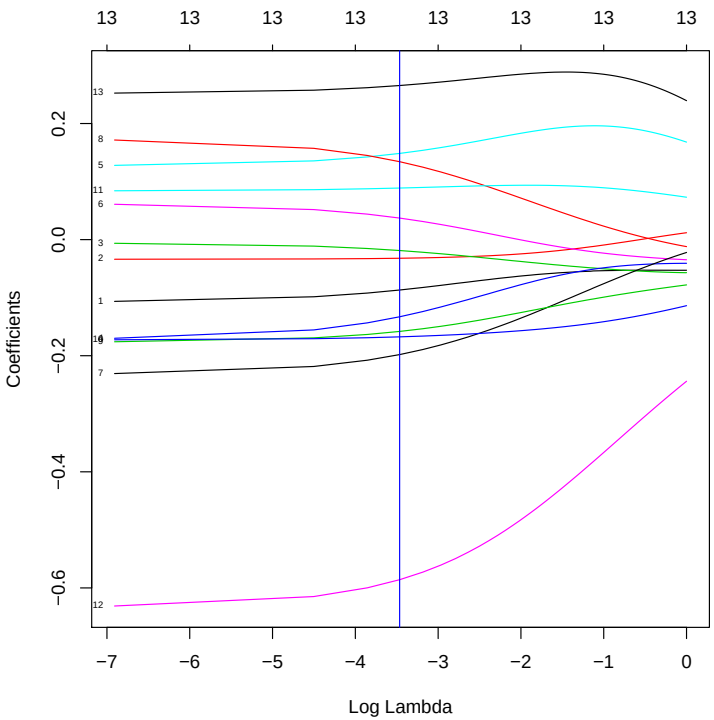
Standardization

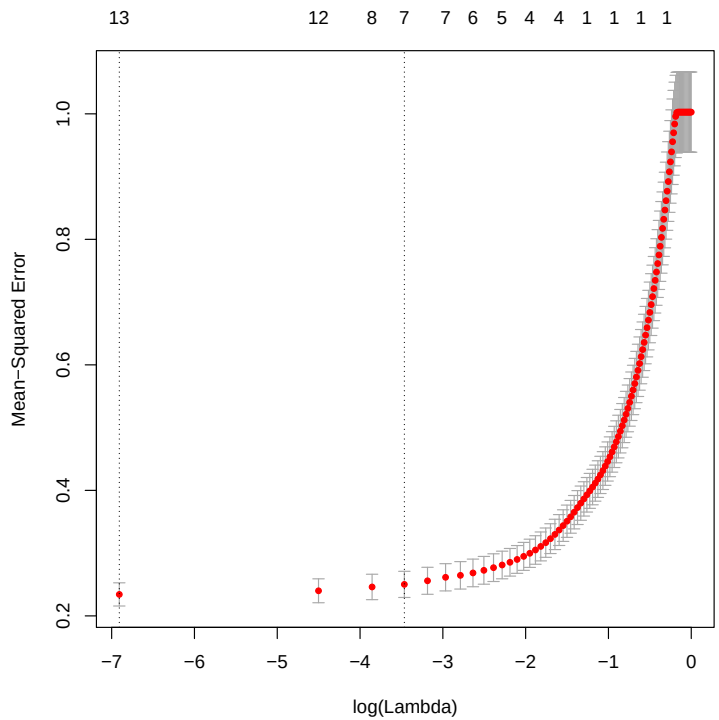
To ensure all variables are shrunk equally, it is usual to standardize all variables to have mean 0 and sd 1. Then we don't need a constant term in the regression.

Boston data, ridge fit

```
> library(glmnet)
# ridge fit
> lambda.seq = seq(1,0.001, length=100)
> ridgefit = glmnet(X,y, alpha=0, lambda = lambda.seq)
> ridgecv = cv.glmnet(X,y,lambda = lambda.seq)
> par(mfrow=c(1,2))
> plot(ridgefit, xvar="lambda", label=TRUE)
> abline(v=log(ridgecv$lambda.1se), col="blue")
> plot(ridgecv)
# prediction error
> Pred.err = ridgecv$cvm[ridgecv$lambda==ridgecv$lambda.min]
> Pred.err
[1] 0.2343135
```

Ridge trace: Boston data





Boston data, lasso fit

```
> # lasso fit
> lambda.seq = seq(1,0.001, length=100)
> lassofit = glmnet(X,y, alpha=1, lambda = lambda.seq)
> lassocv = cv.glmnet(X,y,lambda = lambda.seq)
> par(mfrow=c(1,2))
> plot(lassofit, xvar="lambda", label=TRUE)
> abline(v=log(lassocv$lambda.1se), col="blue")
> plot(lassocv)
> # prediction error
Pred.err = lassocv$cvm[lassocv$lambda==lassocv$lambda.min]
> Pred.err
[1] 0.239059
```

Lasso trace: Boston data

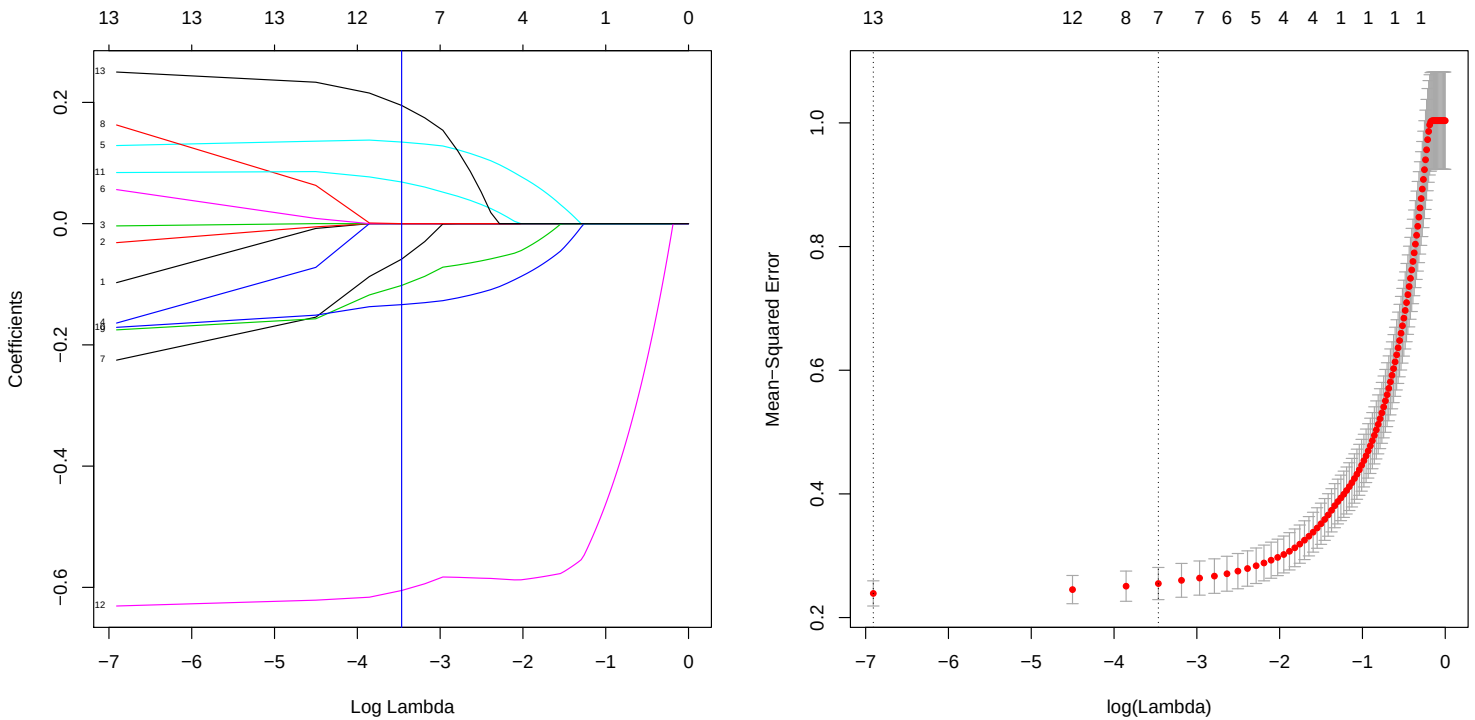


Image data

In the Boston example there are few variables and relatively many cases . In this situation ridge and the lasso won't improve much on ordinary least squares, as we saw in the CV plots. On the other hand, if we have many more variables than cases, least squares won't work, as the normal equations have no unique solution. However, regularisation will work well in this case.

We illustrate with the data from Assignment 2 in 2016, predicting the variable `left_eye_center_x` from the 9216 pixel variables.

Image data (cont)

These data are adapted from the Kaggle competition "Facial Keypoints Detection" described further at \\https://www.kaggle.com/c/facial-keypoints-detection

There are both training and test sets available on the web page. The data refer to a collection of about 1900 images of faces, represented by 96 x 96 pixel arrays. For each pixel, a greyscale value between 0 and 255 is also recorded. In addition, the locations of the left eye in the image are also given.

Image data (cont)

The training and test data sets supplied each have 9218 variables: These are

left_eye_center_x: the x-coordinate of the center of the left eye (from the subject's point of view)

left_eye_center_y: the y-coordinate of the center of the left eye (from the subject's point of view)

V1-V9216: The greyscale values of the 9216 pixels in the image.

Typical image

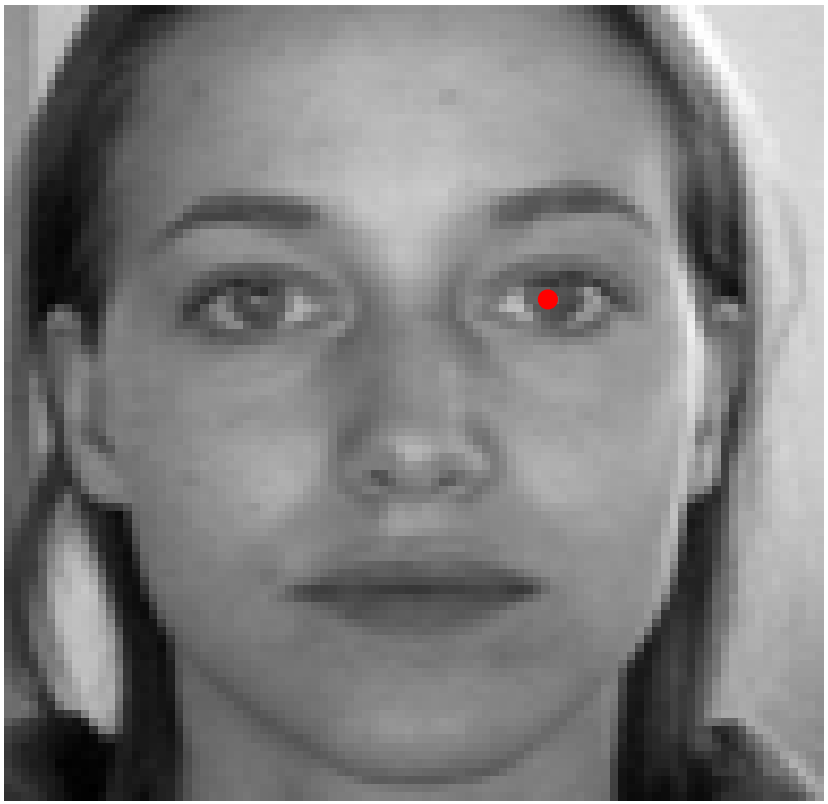
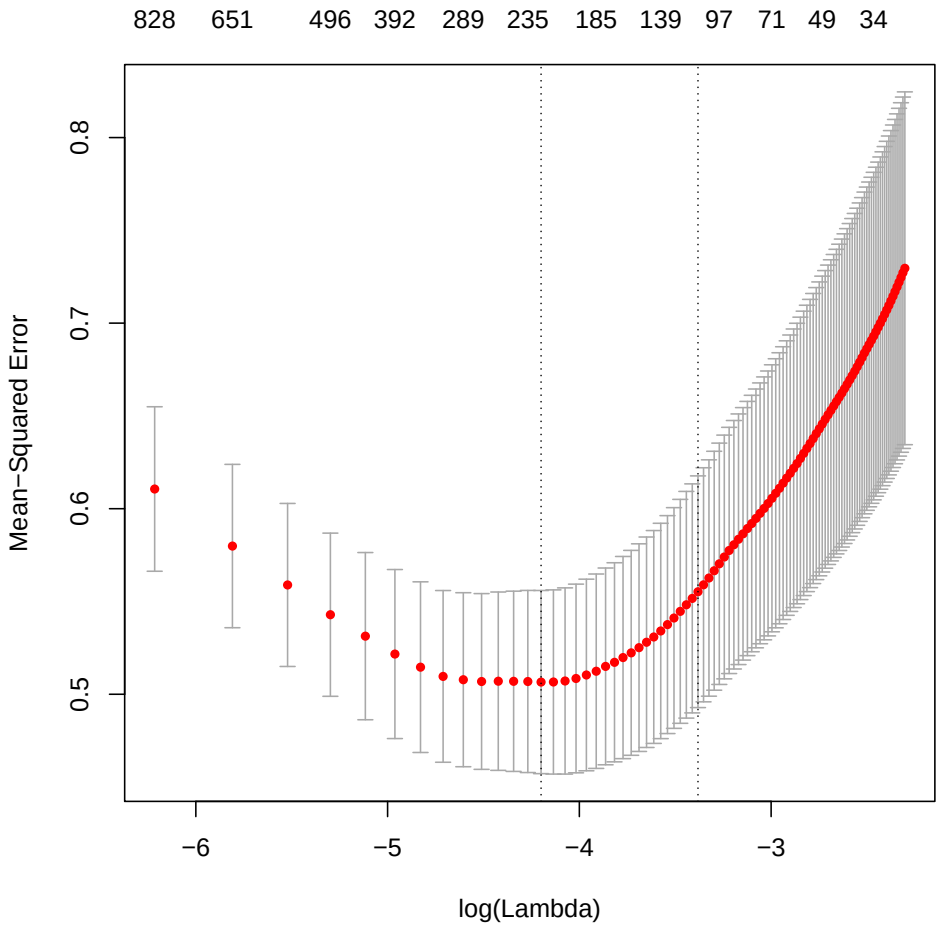


Image data: ridge fit

We work with the standardized data. See the documentation for `glmnet` for the reasons for this. We have a training set of 953 images in a data frame `training.df` with 9218 variables (x and y coordinates of the left eye, plus 9216 pixel variables)

```
# ridge fit
X = as.matrix(training.df)[,-c(1,2)]
y = training.df[,1]
y = as.vector(scale(y))
X = scale(X)
lambda.seq = seq(0.1,0.001, length=100)
ridgefit = glmnet(X,y, alpha=0, lambda = lambda.seq,
       standardize=FALSE)
ridgecv = cv.glmnet(X,y,lambda = lambda.seq)
plot(ridgecv)
```

Example: MSE plot



Errors

```
# CV error
CV = min(ridgecv$cvm)
CV
[1] 0.5065697

CV.1se = ridgecv$cvm[ridgecv$lambda==
           ridgecv$lambda.1se]
CV.1se
[1] 0.5552499
```

Errors

```
# training error
lambdaMin = ridgecv$lambda.min
coefs=coef(ridgefit, s=lambdaMin)
mypred = cbind(1,X)%*%coefs  # no predict function
mean((y-mypred)^2)
[1] 0.00135280  #!!!  very small
#test error
newX = scale(test.df[,-c(1,2)])
testpred = predict(ridgefit, newx=newX, s=lambdaMin)
newY = scale(test.df[,1])
mean((newY-testpred)^2)
[1] 0.7029947
```

Points to note

- With so many variables, ridge regression is seriously overfitting.
- Exercise: is the lasso doing the same?
- Scaling issues can be tricky
- Shrinkage helps in this example: reduces MSE from about 0.6 to about 0.5.

Regularization and classification

We can apply regularization to logistic regression as well: We fit the model by penalizing the log-likelihood for logistic regression. If $l(\beta)$is the log-likelihood, we estimate the parameters by minimizing

$$-l(\beta) + \lambda \sum_j \beta_j^2$$

for ridge, and

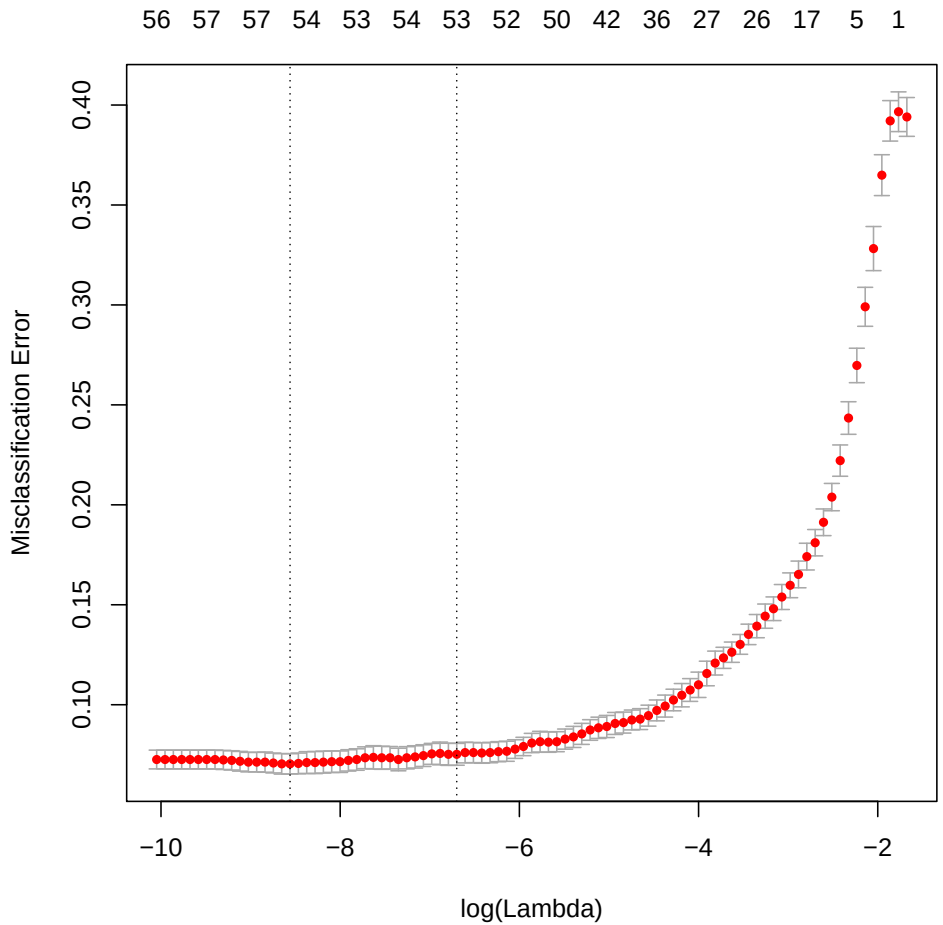
$$-l(\beta) + \lambda \sum_j |\beta_j|$$

for the lasso.

Example: Spam data

```
X =as.matrix(spam.df[,-58])
y = factor(spam.df[,58])
lassofit = glmnet(X,y, alpha=1, family="binomial")
lassocv = cv.glmnet(X,y,lambda = lassofit$lambda,
     family="binomial",type.measure="class")
plot(lassocv)
MissClass = lassocv$cvm[lassocv$lambda==
                        lassocv$lambda.min]
[1] 0.07215823
```

Example: Spam data



References

1. Hastie, T., Tibshirani, R.J. and Friedman, J. (2009). *The Elements of Statistical Learning, 2nd Ed*. Springer.
2. Hoerl, A E. and Kennard, R.W. (1970) Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12, 55-67.
3. James, G., Witten, D., Hastie. and Tibshirani, R.J. (2013). *An Introduction to Statistical Learning*. Springer.
4. Kuhn, M. and Johnson, K. (2013). *Applied Predictive Modelling*. Springer.
5. Tibshirani, R. (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58, 267-288.