Ridge Regression and Lasso

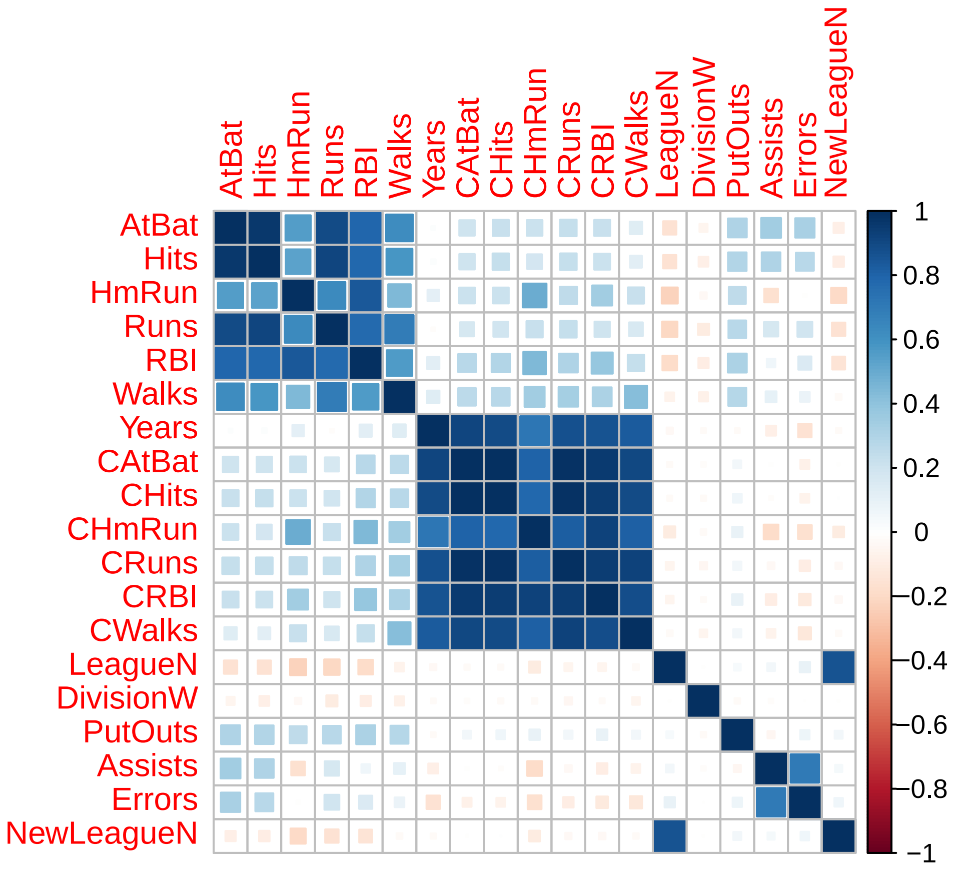
Yifei Sun

Contents

```r
library(ISLR)
library(glmnet)
library(caret)
library(corrplot)
library(plotmo)
```

Predict a baseball player's salary on the basis of various statistics associated with performance in the previous year. Use `?Hitters` for more details.

```r
data(Hitters)
# delete rows containing the missing data
Hitters <- na.omit(Hitters)
# matrix of predictors (glmnet uses input matrix)
x <- model.matrix(Salary~.,Hitters)[,-1]
# vector of response
y <- Hitters$Salary

corrplot(cor(x), method = "square", type = "full")
```



Ridge regression using `glmnet()`

`alpha` is the elasticnet mixing parameter. `alpha=1` is the lasso penalty, and `alpha=0` the ridge penalty. `glmnet()` function standardizes the variables by default. `ridge.mod` contains the coefficient estimates for a set of lambda values. The grid for lambda is in `ridge.mod$lambda`.

```r
# fit the ridge regression (alpha = 0) with a sequence of lambdas
ridge.mod <- glmnet(x, y, standardize=TRUE,
                    alpha = 0,
                    lambda = exp(seq(-1, 10, length=100)))
```

`coef(ridge.mod)` gives the coefficient matrix. Each column is the fit corresponding to one lambda value.

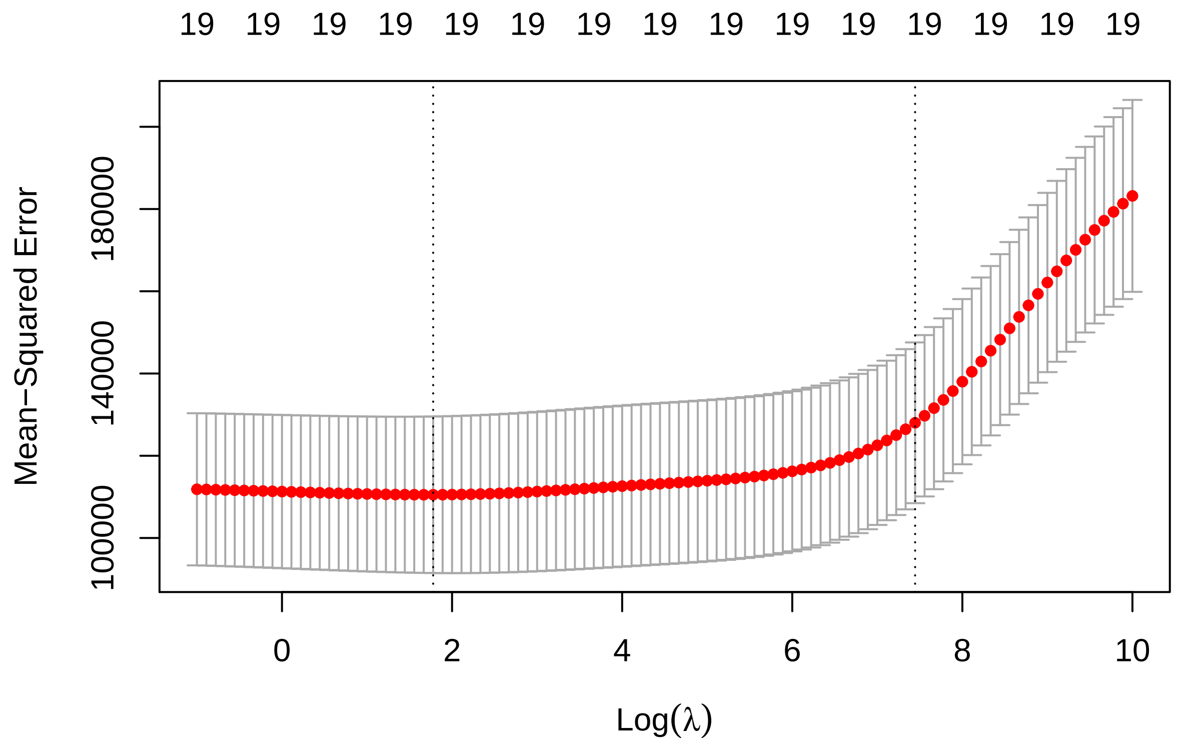
```r
mat.coef <- coef(ridge.mod)
dim(mat.coef)
```

```
## [1]  20 100
```

Cross-validation

We use cross-validation to determine the optimal value of lambda. The two vertical lines are the for minimal MSE and 1SE rule. The 1SE rule gives the model with fewest coefficients that's less than one SE away from the sub-model with the lowest error.

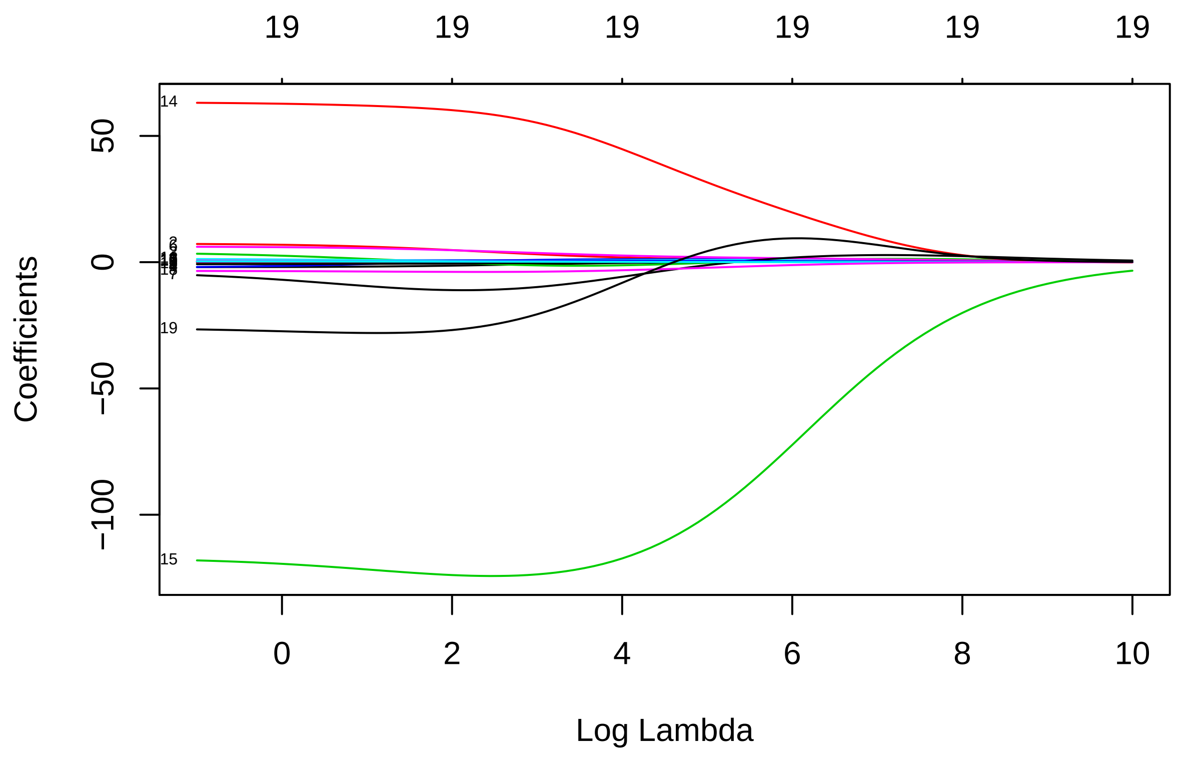
```r
set.seed(2)
cv.ridge <- cv.glmnet(x, y, type.measure = "mse",
                      alpha = 0,
                      lambda = exp(seq(-1, 10, length=100)))

plot(cv.ridge)
```



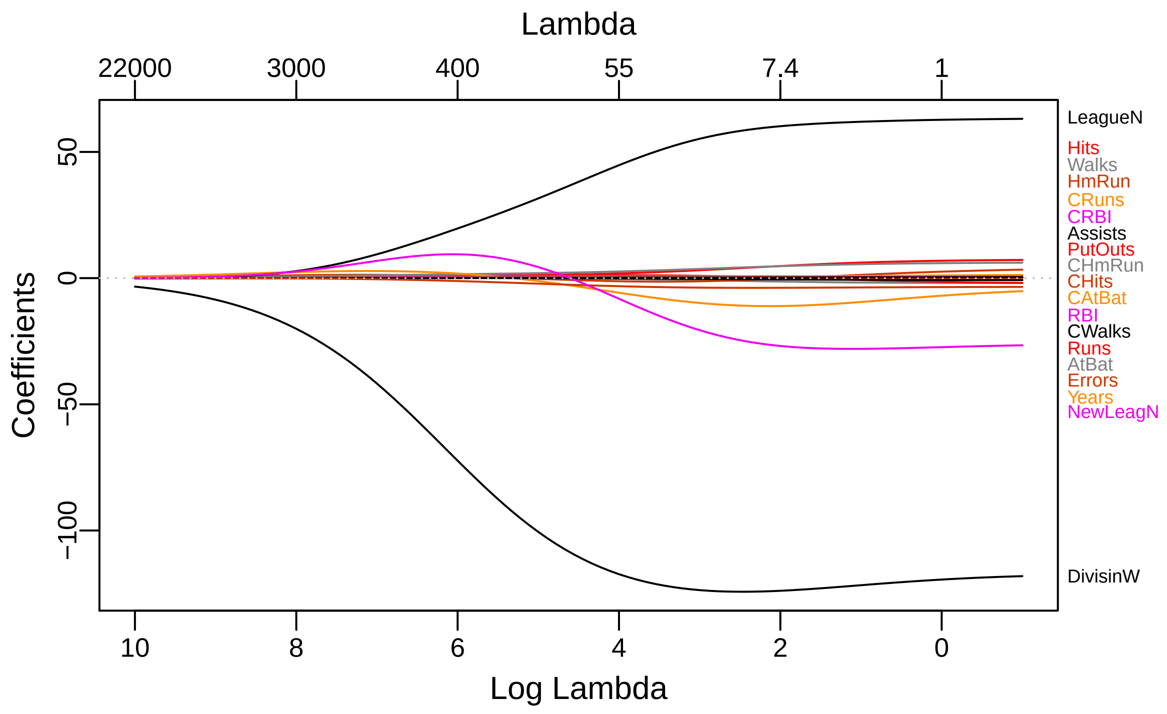
Trace plot

There are two functions for generating the trace plot.

```
plot(ridge.mod, xvar = "lambda", label = TRUE)
```



```
plot_glmnet(ridge.mod, xvar = "rlambda", label = 19)
```



Coefficients of the final model

Get the coefficients of the optimal model. `s` is value of the penalty parameter `lambda` at which predictions are required.

```
best.lambda <- cv.ridge$lambda.min
best.lambda
```

```
## [1] 5.916694
```

```
predict(ridge.mod, s = best.lambda, type="coefficients")
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  1.412660e+02
## AtBat       -1.473964e+00
## Hits         5.137966e+00
## HmRun        1.367440e-01
## Runs        -5.326394e-03
## RBI          2.101496e-01
## Walks        4.944991e+00
## Years       -1.090354e+01
## CAtBat      -4.131265e-02
## CHits        1.945238e-01
## CHmRun       7.502141e-01
## CRuns        6.010338e-01
## CRBI         3.389147e-01
## CWalks      -5.573029e-01
## LeagueN      6.075254e+01
## DivisionW   -1.235100e+02
## PutOuts      2.779354e-01
## Assists      2.713712e-01
## Errors      -3.816287e+00
## NewLeagueN  -2.745355e+01
```

```
# predict(cv.ridge, s = best.lambda, type="coefficients")
# predict(cv.ridge, s = "lambda.min", type="coefficients")
# predict(cv.ridge, s = "lambda.1se", type="coefficients")
```

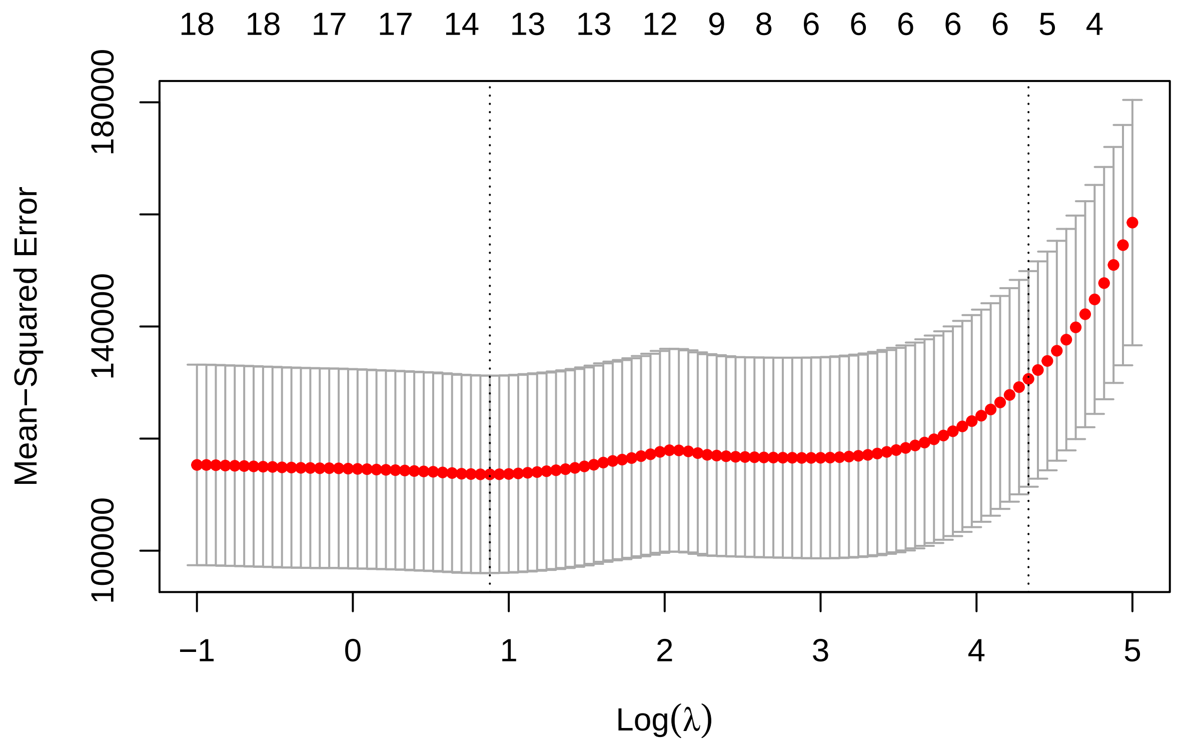
Lasso using `glmnet()`

The syntax is along the same line as ridge regression. Now we use `alpha = 1`.

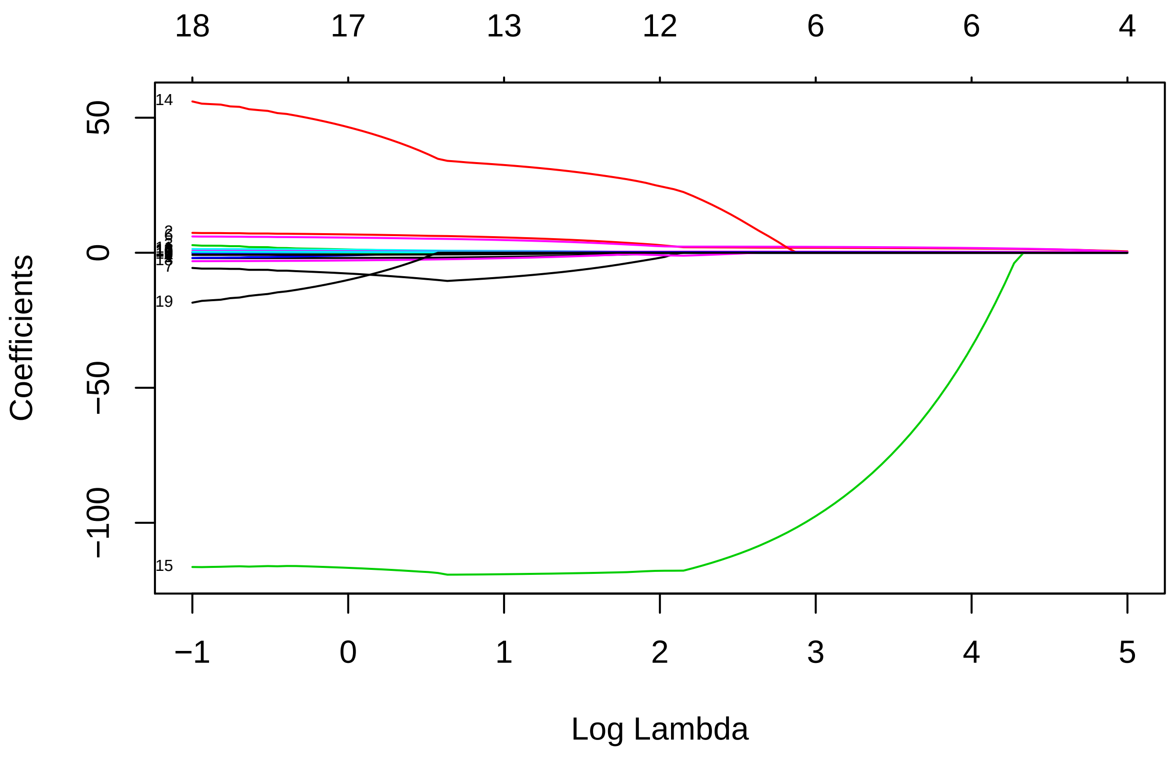
```
cv.lasso <- cv.glmnet(x,y, alpha = 1, lambda = exp(seq(-1, 5, length=100)))
cv.lasso$lambda.min
```

```
## [1] 2.407979
```

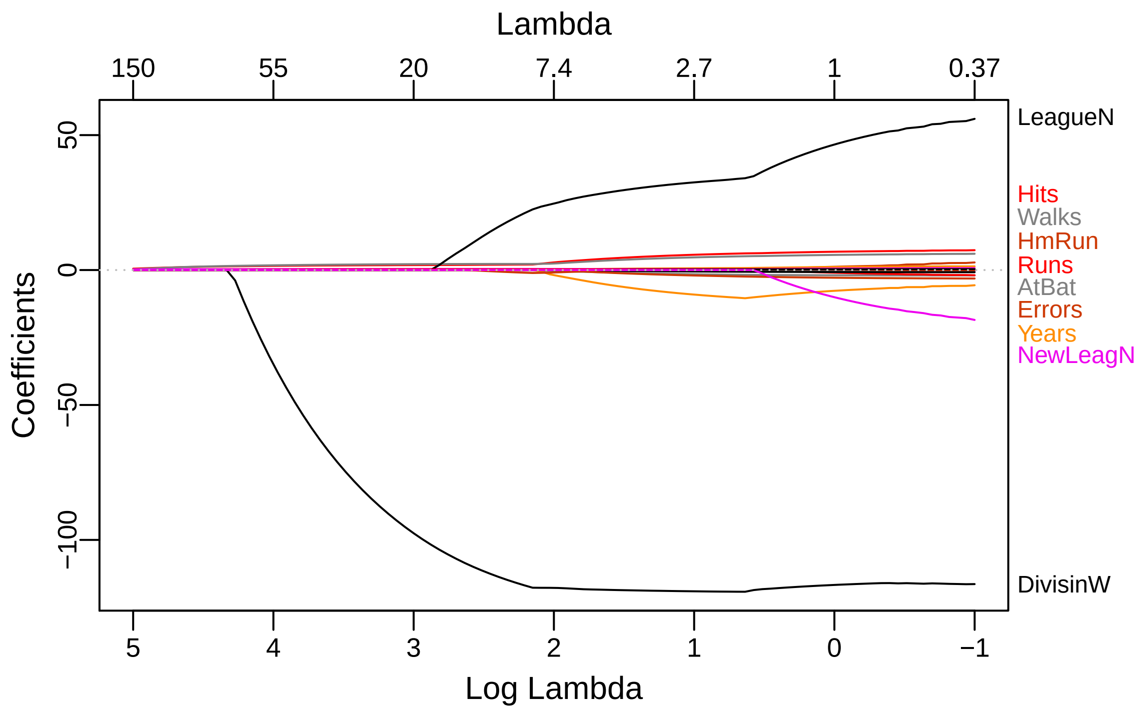
```
plot(cv.lasso)
```



```
# cv.lasso$glmnet.fit is a fitted glmnet object for the full data
# You can also plot the result obtained from glmnet()
plot(cv.lasso$glmnet.fit, xvar = "lambda", label=TRUE)
```



```
plot_glmnet(cv.lasso$glmnet.fit)
```

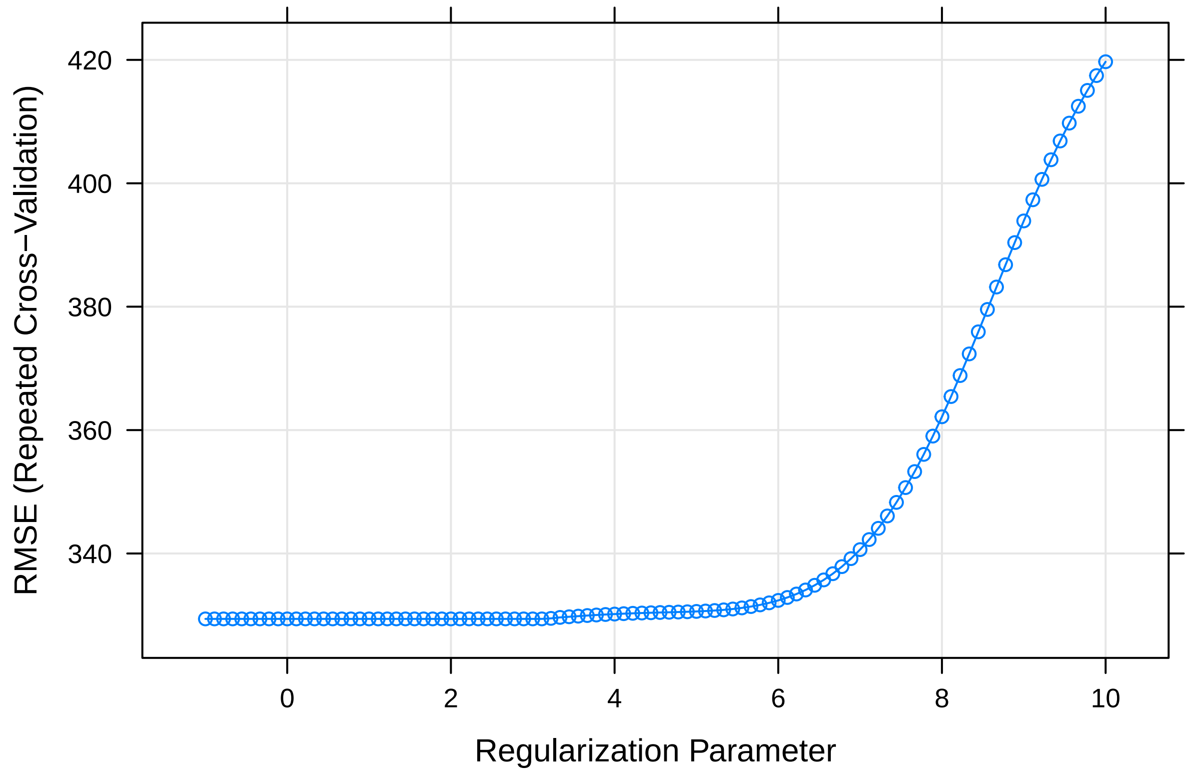


```
predict(cv.lasso, s="lambda.min", type="coefficients")
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  128.0081008
## AtBat         -1.6277701
## Hits           5.8545159
## HmRun          .
## Runs           .
## RBI            .
## Walks          4.8544747
## Years         -9.5792176
## CAtBat         .
## CHits          .
## CHmRun         0.5883697
## CRuns          0.6886035
## CRBI           0.3689474
## CWalks        -0.5603354
## LeagueN       32.9975204
## DivisionW   -119.1215157
## PutOuts        0.2749755
## Assists        0.1877294
## Errors        -2.1553836
## NewLeagueN     .
```

Ridge and lasso using caret

```
ctrl1 <- trainControl(method = "repeatedcv", number = 10, repeats = 5)
# you can try other options

set.seed(2)
ridge.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 0,
                                          lambda = exp(seq(-1, 10, length=100))),
                   # preProc = c("center", "scale"),
                   trControl = ctrl1)

plot(ridge.fit, xTrans = function(x) log(x))
```



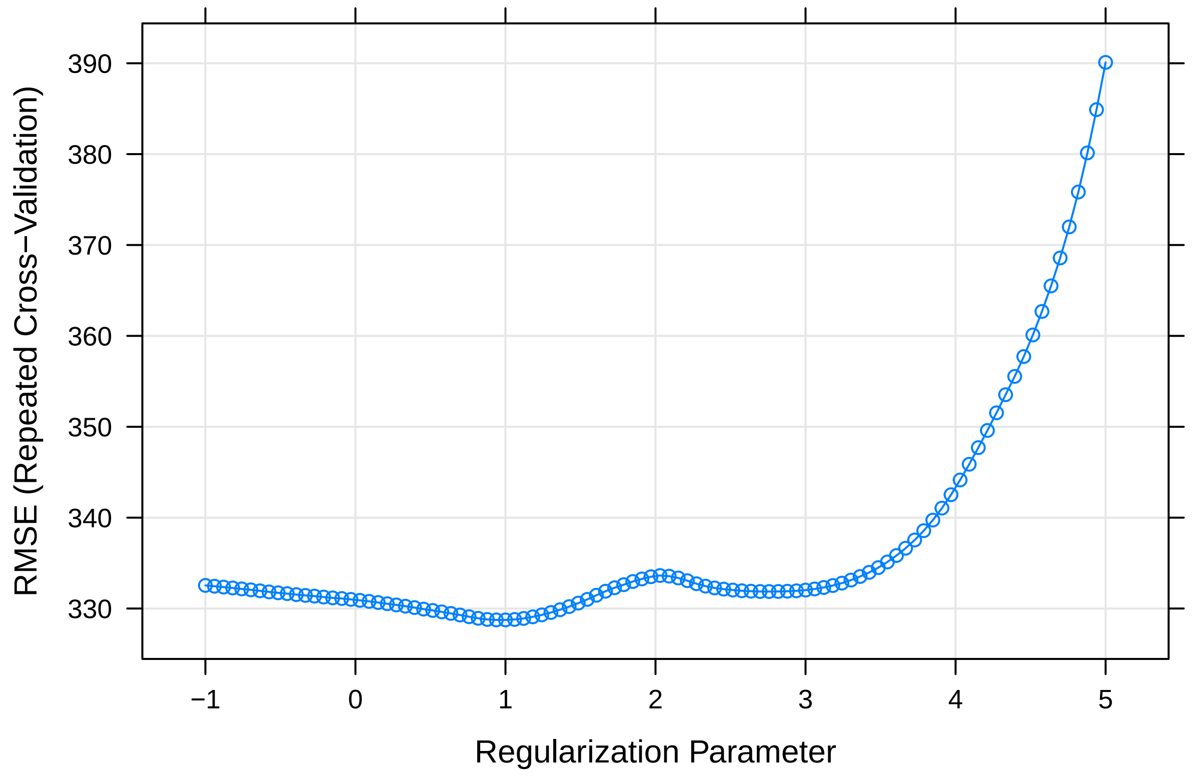
```
ridge.fit$bestTune
```

```
##    alpha   lambda
## 38     0 22.44597
```

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

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  8.112693e+01
## AtBat       -6.815959e-01
## Hits         2.772312e+00
## HmRun       -1.365680e+00
## Runs         1.014826e+00
## RBI          7.130225e-01
## Walks        3.378558e+00
## Years       -9.066800e+00
## CAtBat      -1.199478e-03
## CHits        1.361029e-01
## CHmRun       6.979958e-01
## CRuns        2.958896e-01
## CRBI         2.570711e-01
## CWalks      -2.789666e-01
## LeagueN      5.321272e+01
## DivisionW   -1.228345e+02
## PutOuts      2.638876e-01
## Assists      1.698796e-01
## Errors      -3.685645e+00
## NewLeagueN  -1.810510e+01
```

```r
set.seed(2)
lasso.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 1,
                                          lambda = exp(seq(-1, 5, length=100))),
                   # preProc = c("center", "scale"),
                   trControl = ctrl1)
plot(lasso.fit, xTrans = function(x) log(x))
```



```
lasso.fit$bestTune
```

```
##    alpha  lambda
## 33     1 2.55843
```

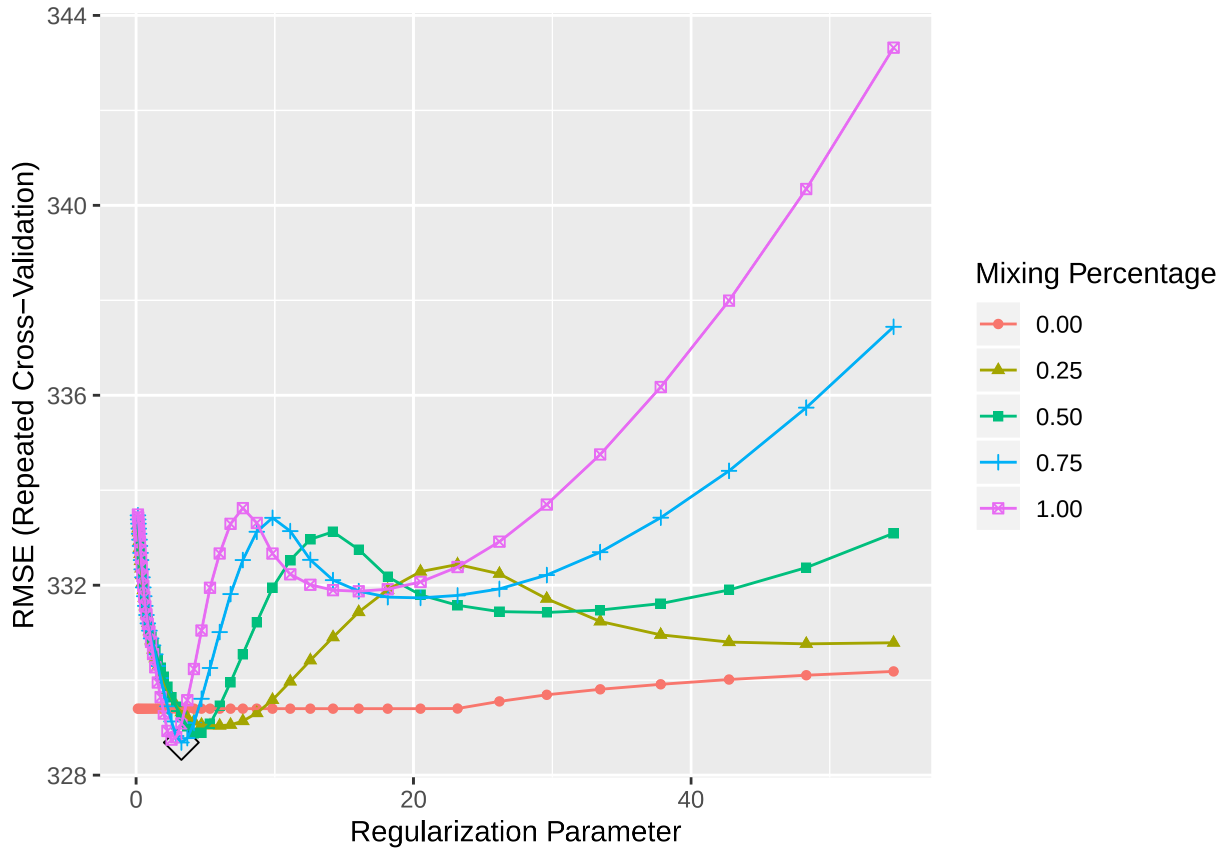
```
coef(lasso.fit$finalModel,lasso.fit$bestTune$lambda)
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)  126.5159502
## AtBat         -1.5793921
## Hits           5.7316569
## HmRun          .
## Runs           .
## RBI            .
## Walks          4.7869075
## Years         -9.7796085
## CAtBat         .
## CHits          .
## CHmRun         0.5238187
## CRuns          0.6700796
## CRBI           0.3915821
## CWalks        -0.5422646
## LeagueN       32.2600576
## DivisionW   -119.3229627
## PutOuts         0.2732756
## Assists         0.1792516
## Errors         -2.1105189
## NewLeagueN      .
```

```
set.seed(2)
enet.fit <- train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = seq(0, 1, length = 5),
                                         lambda = exp(seq(-2, 4, length = 50))),
                  # preProc = c("center", "scale"),
                  trControl = ctrl1)
enet.fit$bestTune
```

```
##     alpha   lambda
## 177  0.75 3.266351
```

```
ggplot(enet.fit, highlight = TRUE)
```

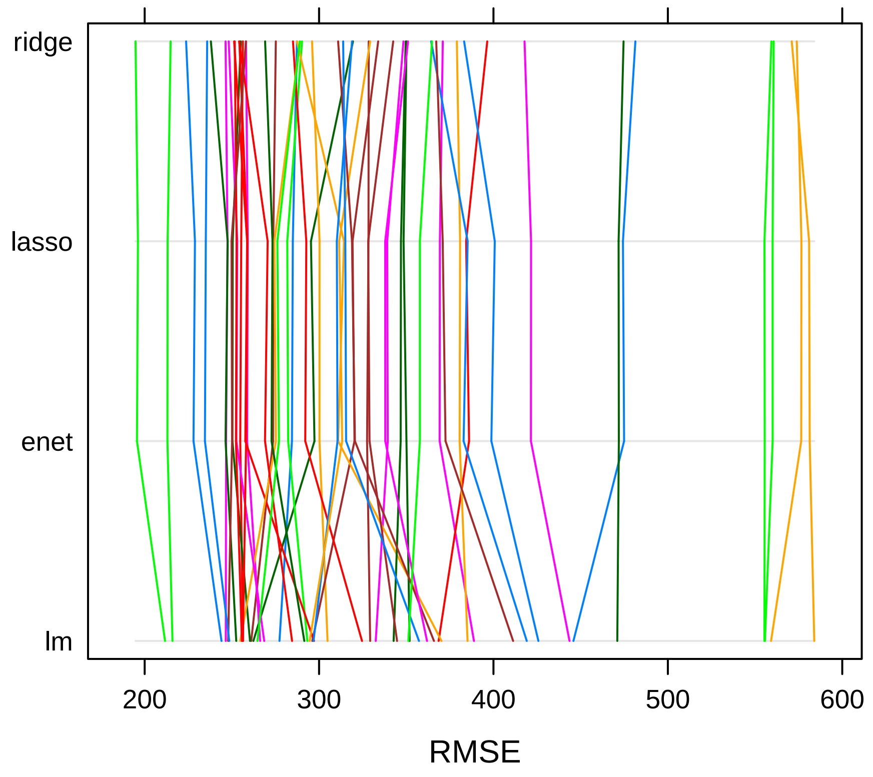


```
set.seed(2)
lm.fit <- train(x, y,
                method = "lm",
                trControl = ctrl1)
```

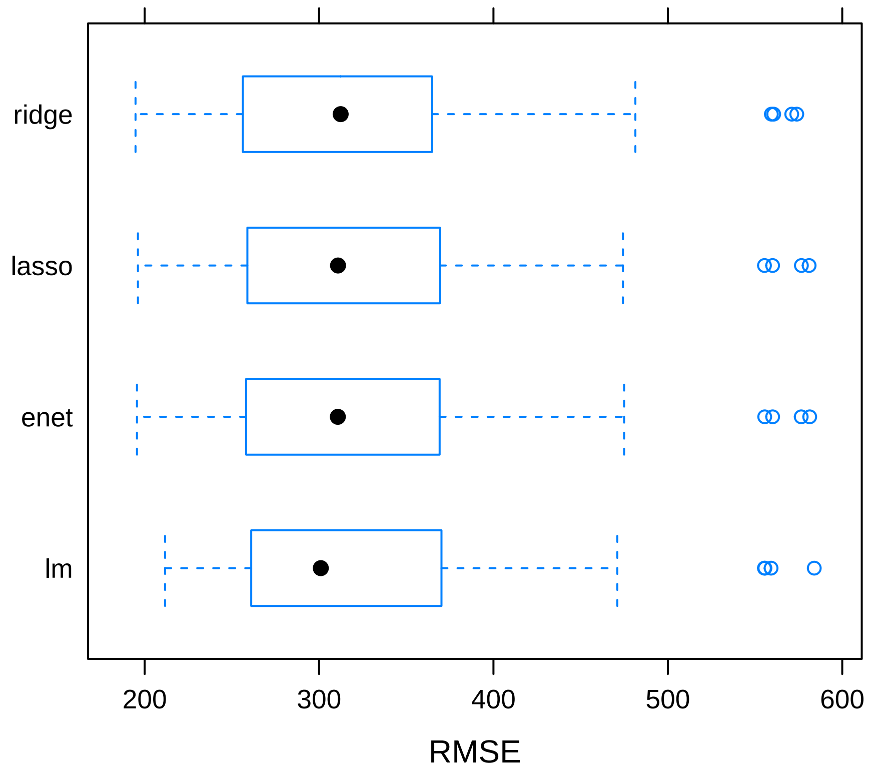
```r
resamp <- resamples(list(enet = enet.fit, lasso = lasso.fit, ridge = ridge.fit, lm = lm.fit))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: enet, lasso, ridge, lm
## Number of resamples: 50
##
## MAE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## enet  159.9571 202.4090 219.9305 233.4855 258.2937 365.5725    0
## lasso 160.4923 202.9029 219.2534 233.5877 258.3526 365.9717    0
## ridge 159.1346 200.5880 224.7497 233.3368 259.0923 359.2888    0
## lm    164.2278 203.9309 229.5114 238.3943 257.1248 372.3725    0
##
## RMSE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## enet  195.5780 258.3066 310.7614 328.6871 366.3068 581.3332    0
## lasso 196.1341 258.9435 310.8612 328.7470 366.4100 581.0099    0
## ridge 194.7566 256.7452 312.3703 329.3997 364.6214 573.9334    0
## lm    211.6756 261.3999 300.9936 335.1071 369.7822 583.9501    0
##
## Rsquared
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## enet  0.06236945 0.4090767 0.5414844 0.4952418 0.6120019 0.7853878    0
## lasso 0.06302265 0.4097392 0.5432945 0.4949083 0.6116662 0.7852193    0
## ridge 0.06158103 0.4052727 0.5283642 0.4924478 0.6149772 0.7757865    0
## lm    0.06951446 0.3417336 0.5189533 0.4774099 0.6159995 0.7892321    0
```

```r
parallelplot(resamp, metric = "RMSE")
```



```r
bwplot(resamp, metric = "RMSE")
```


ridge
lasso
enet
lm
200
300
400
500
600
RMSE