statLearn R LAB.6(b) RIDGE REGRESSION AND LASSO

First, need to install and load the 'glmnet' package, glmnet does not use model formula language, so we must set up an 'x' and 'y'

```
library(ISLR)
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.3.2
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-10
```

```
Hitters=na.omit(Hitters)
with(Hitters,sum(is.na(Salary)))
```

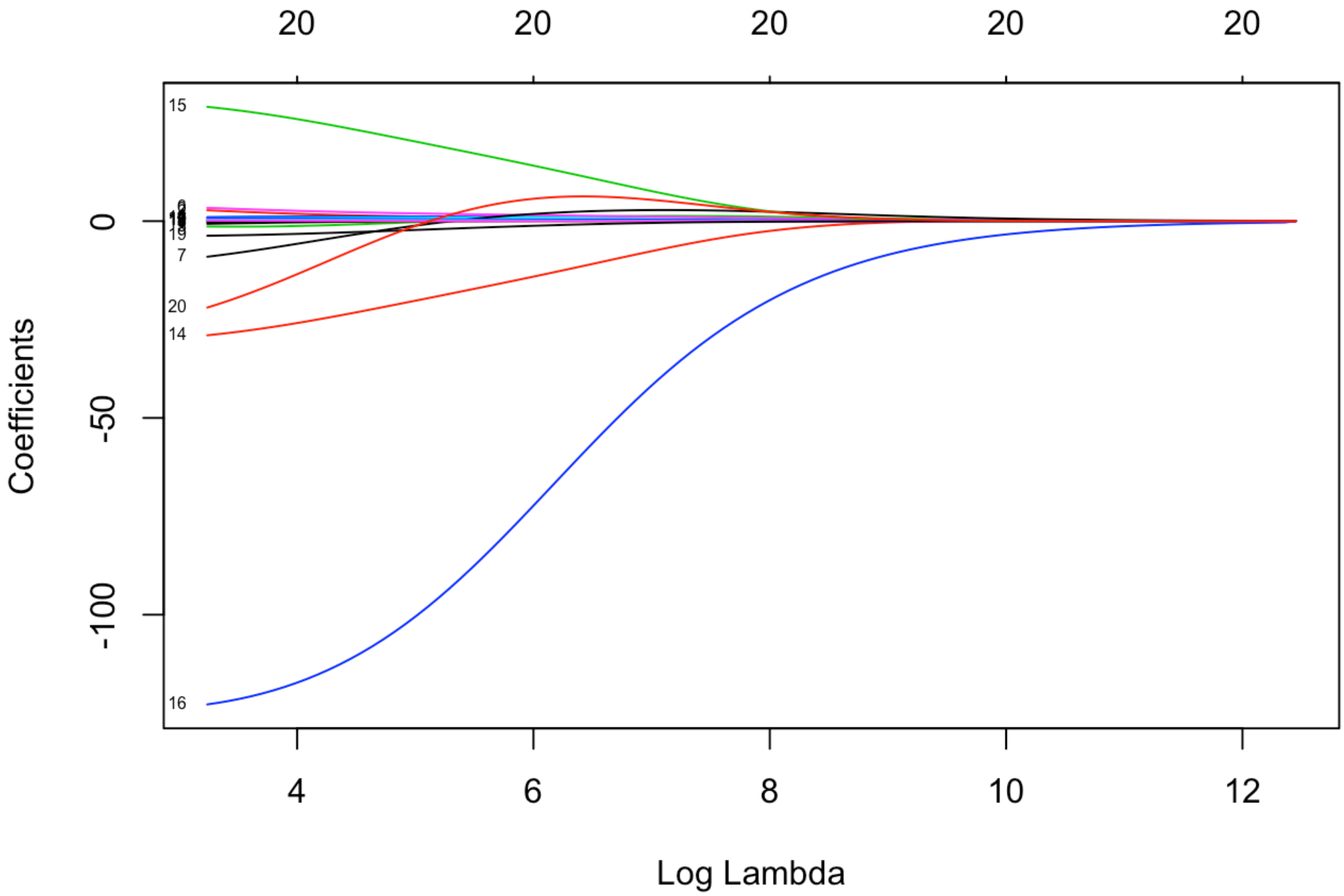
```
## [1] 0
```

```
x = model.matrix(Salary~.-1, data=Hitters)
y = Hitters$Salary
```

Ridge Regression: Shrinkage of Coefficients

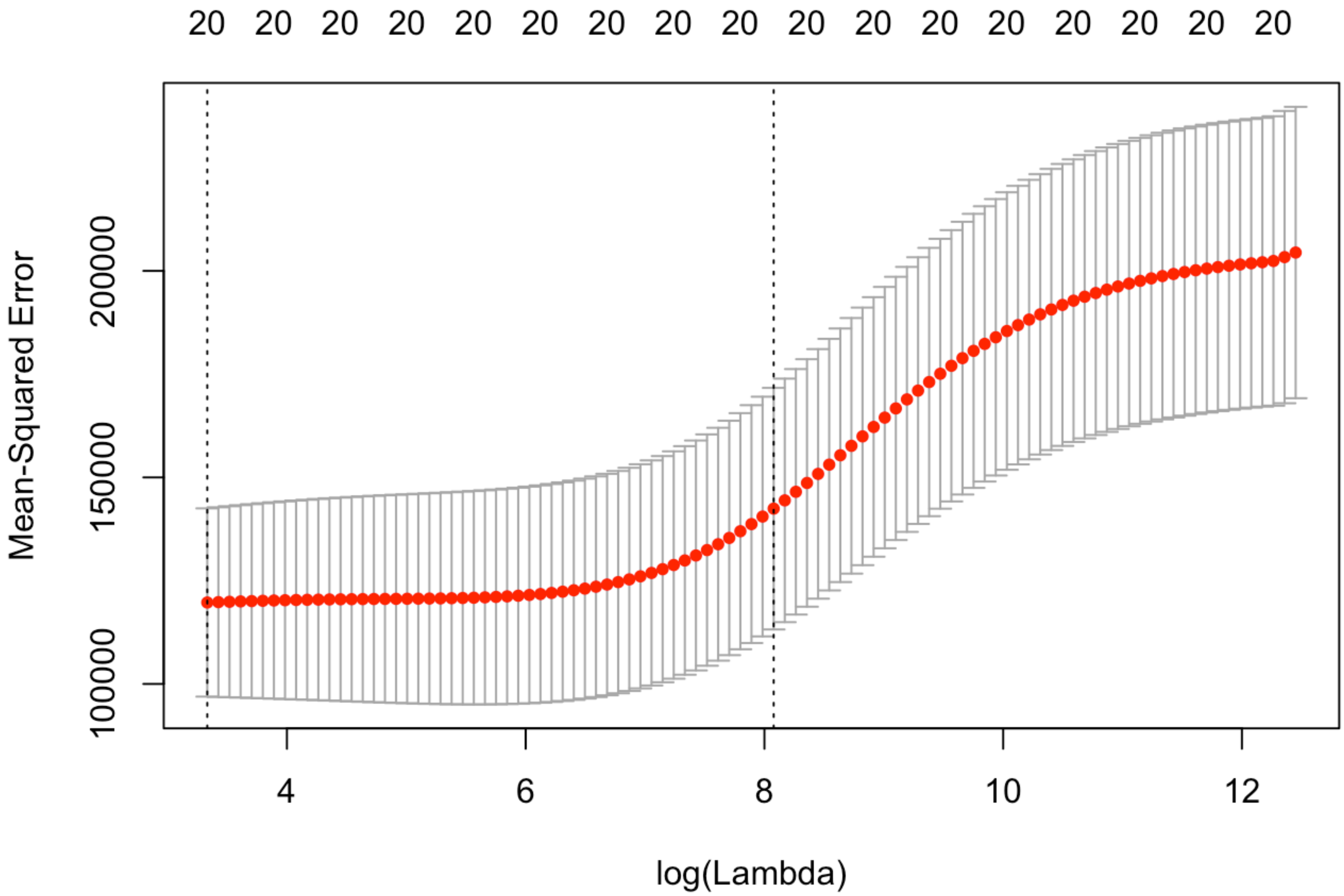
To fit Ridge-Regression model we calling 'glmnet' with 'alpha=0' (see helpfile; to fit Lasso model, alpha is set to 1). 'cv.glmnet' function will do cross-validation for us

```
fit.ridge = glmnet(x, y, alpha=0)
plot(fit.ridge, xvar="lambda", label=TRUE)
```



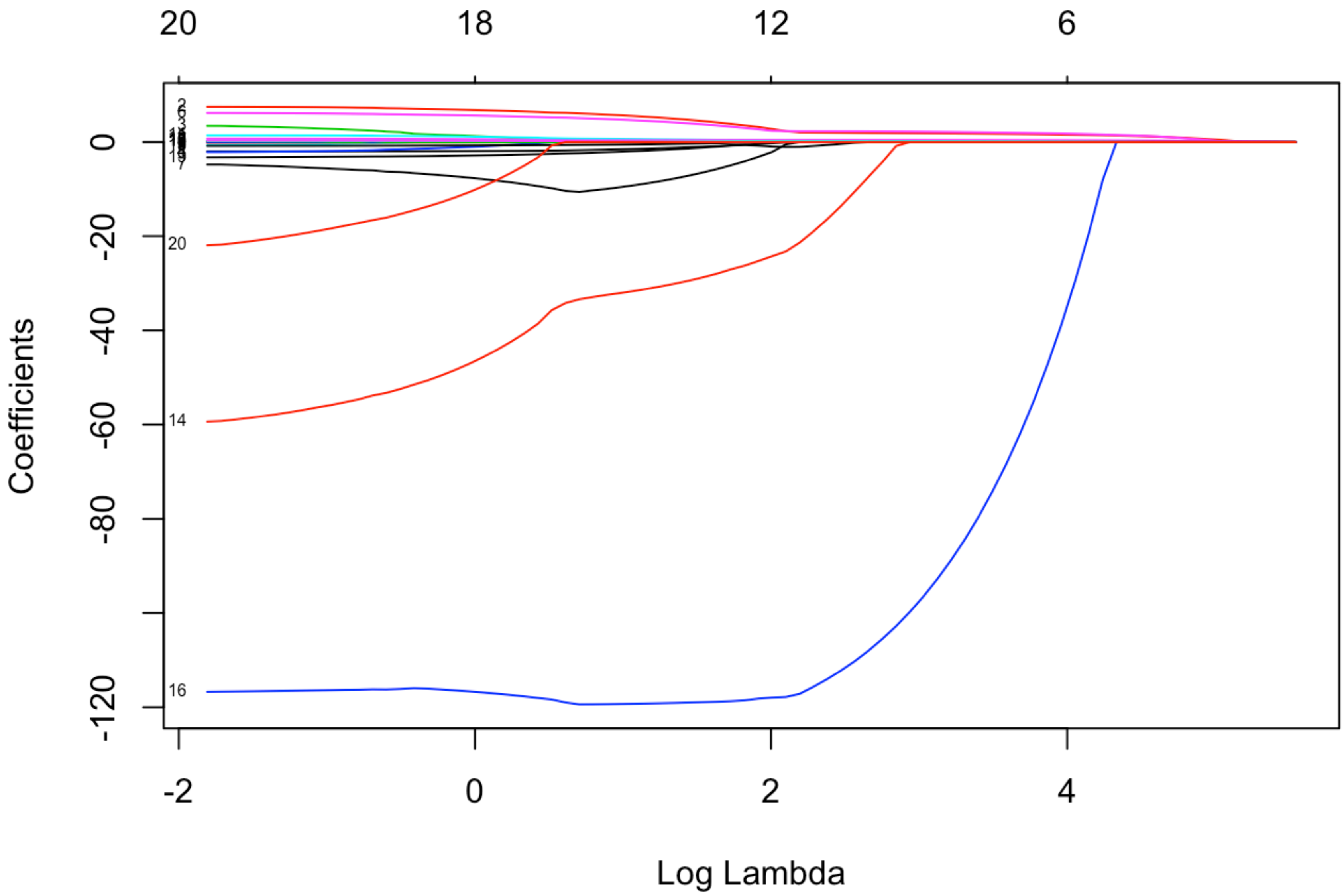
Coefficients are plotted as a function of log lambda; RR models are penalized by the SumSquares of Coefficients controlled by parameter lambda: RSS+lambda*penalty Coefficients shrink towards zero, as lambda increases

```
cv.ridge = cv.glmnet(x, y, alpha=0)
plot(cv.ridge)
```

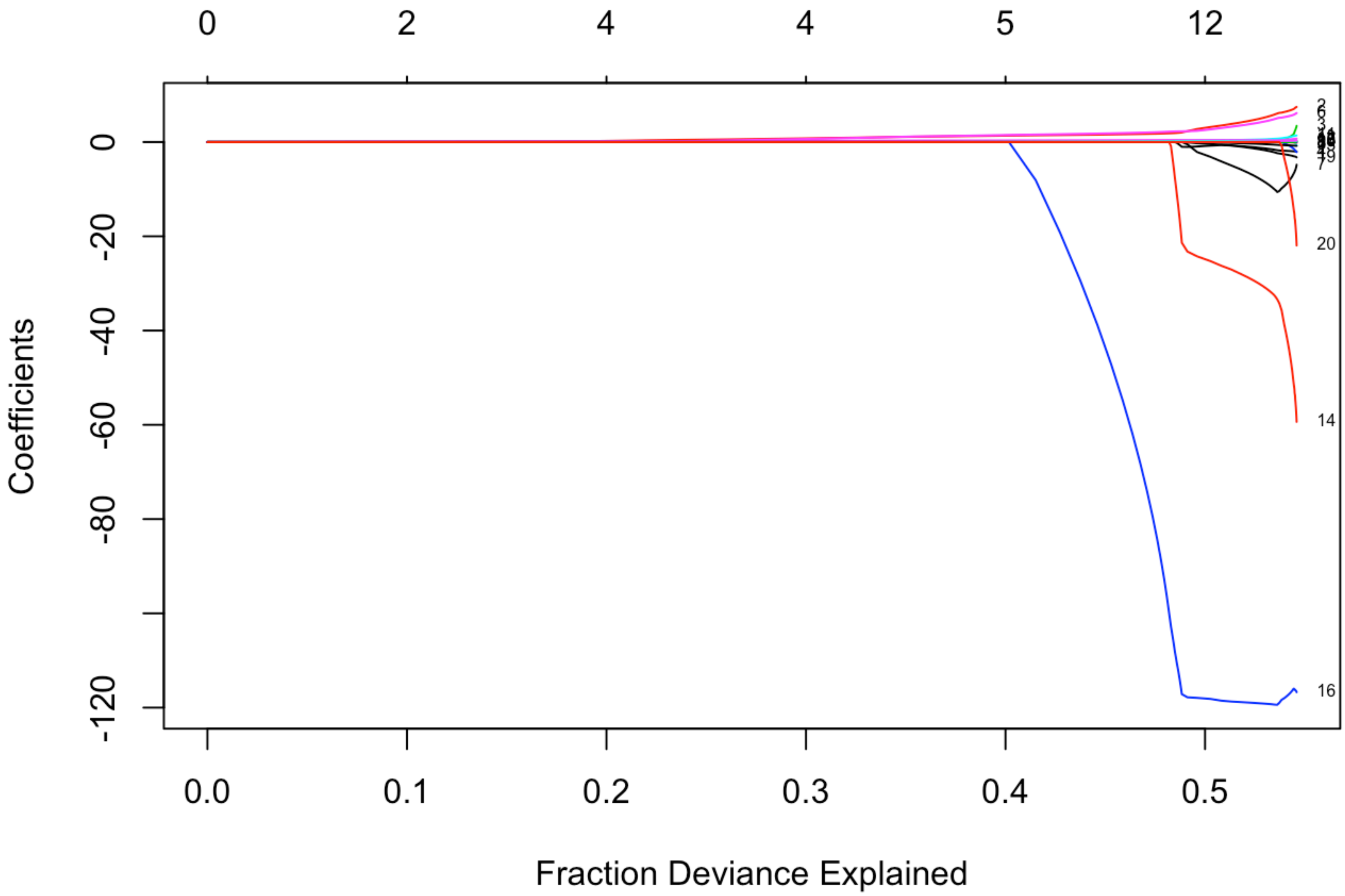


The Lasso: does Shrinkage and Variable Selection Now, we fit lasso model; using the default 'alpha=1'
Lasso is similar to RR, but has slightly different penalty
uses absolute value of coefficiebts, shrinking some to zero

```
fit.lasso = glmnet(x, y, alpha=1)
plot(fit.lasso, xvar="lambda", label=TRUE)
```



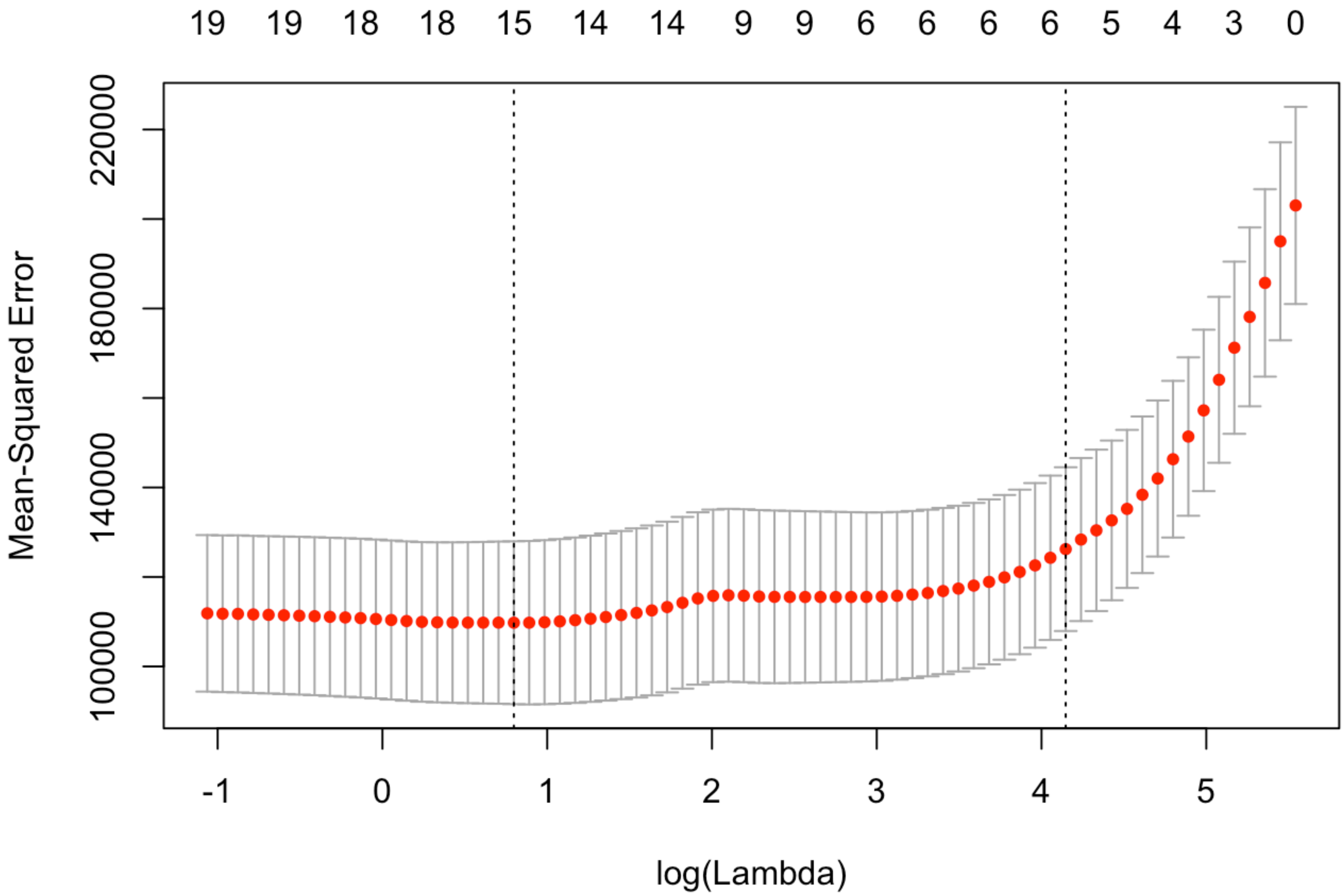

```
plot(fit.lasso, xvar="dev", label=TRUE)
```



Plot shows that initially all features are included, and then as lambda increases, only a few features are retained and the coefficients for unrelated features move to zero Plot xvar="dev" shows R^2

Use Cross-Validation to get MSE Estimates for lasso model

```
cv.lasso = cv.glmnet(x, y, alpha=1)
plot(cv.lasso)
```




```
coef(cv.lasso)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) 115.3773590
## AtBat         .
## Hits          1.4753071
## HmRun         .
## Runs          .
## RBI           .
## Walks         1.6566947
## Years         .
## CAtBat        .
## CHits         .
## CHmRun        .
## CRuns         0.1660465
## CRBI          0.3453397
## CWalks        .
## LeagueA       .
## LeagueN       .
## DivisionW   -19.2435216
## PutOuts       0.1000068
## Assists       .
## Errors        .
## NewLeagueN    .
```

Use earlier TRAIN/VALIDATION to select 'lambda' for the lasso: It tries to fit 100 values of lambda, and if nothing changes, then it stops. Use best lambda and fit to whole data set

```
set.seed(1)
train = sample(seq(263), 180, replace=FALSE)
lasso.tr = glmnet(x[train,], y[train])
lasso.tr
```

```
##
## Call:  glmnet(x = x[train, ], y = y[train])
##
##       Df    %Dev    Lambda
##  [1,]  0 0.00000 246.40000
##  [2,]  1 0.05013 224.50000
##  [3,]  1 0.09175 204.60000
##  [4,]  2 0.13840 186.40000
##  [5,]  2 0.18000 169.80000
##  [6,]  3 0.21570 154.80000
##  [7,]  3 0.24710 141.00000
##  [8,]  3 0.27320 128.50000
##  [9,]  4 0.30010 117.10000
## [10,]  4 0.32360 106.70000
## [11,]  4 0.34310  97.19000
## [12,]  4 0.35920  88.56000
## [13,]  5 0.37360  80.69000
```

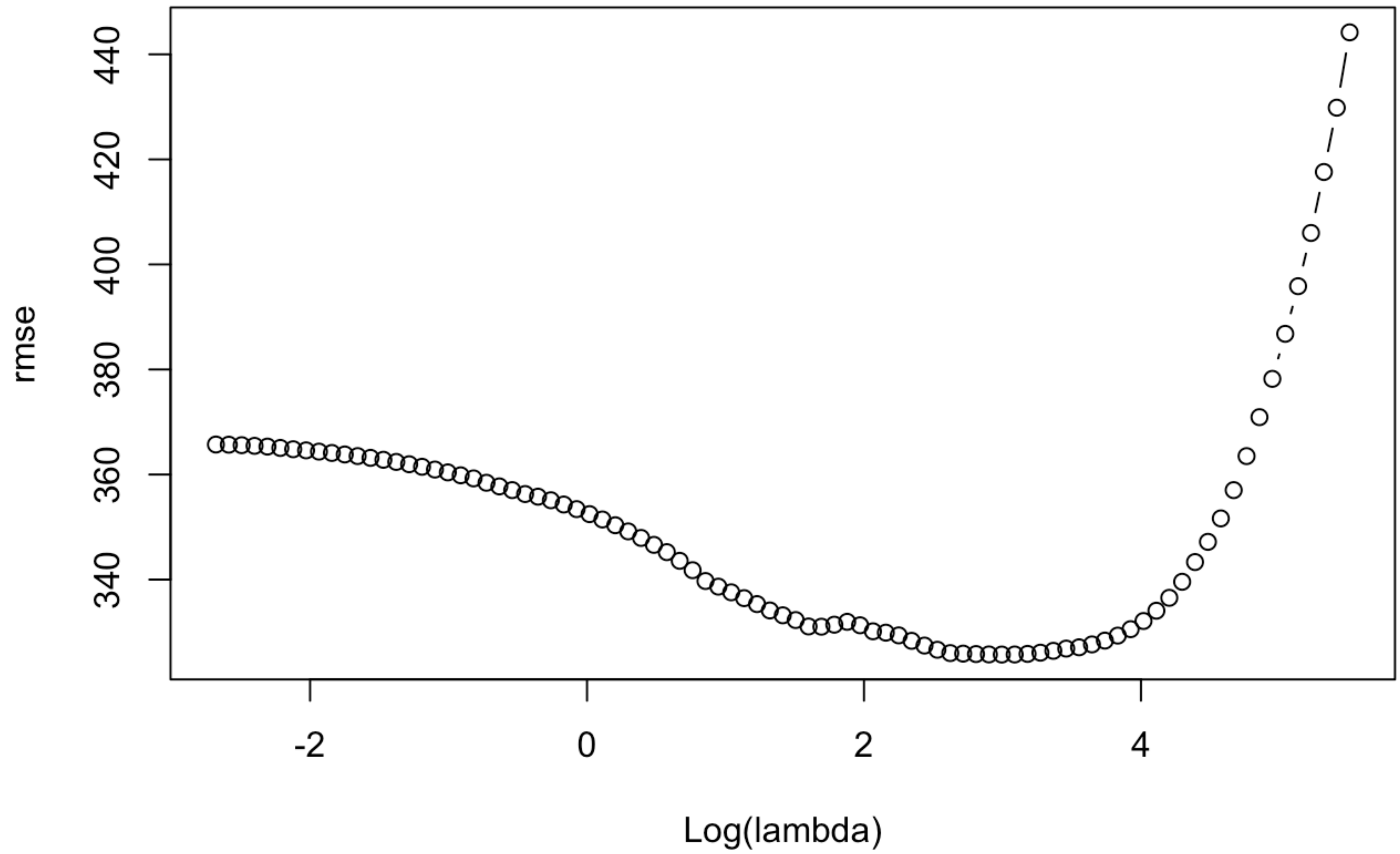
```
## [14,]  5 0.38900  73.52000
## [15,]  5 0.40190  66.99000
## [16,]  5 0.41260  61.04000
## [17,]  5 0.42140  55.62000
## [18,]  5 0.42880  50.67000
## [19,]  5 0.43490  46.17000
## [20,]  5 0.43990  42.07000
## [21,]  5 0.44410  38.33000
## [22,]  5 0.44760  34.93000
## [23,]  6 0.45140  31.83000
## [24,]  7 0.45480  29.00000
## [25,]  7 0.45770  26.42000
## [26,]  7 0.46010  24.07000
## [27,]  8 0.46220  21.94000
## [28,]  8 0.46380  19.99000
## [29,]  8 0.46520  18.21000
## [30,]  8 0.46630  16.59000
## [31,]  8 0.46730  15.12000
## [32,]  8 0.46810  13.78000
## [33,]  9 0.47110  12.55000
## [34,]  9 0.47380  11.44000
## [35,]  9 0.47620  10.42000
## [36,] 10 0.48050   9.49500
## [37,]  9 0.48450   8.65200
## [38,] 10 0.48770   7.88300
## [39,] 10 0.49360   7.18300
## [40,] 11 0.49890   6.54500
## [41,] 12 0.50450   5.96300
## [42,] 12 0.51010   5.43400
## [43,] 13 0.51470   4.95100
## [44,] 13 0.51850   4.51100
## [45,] 13 0.52170   4.11000
## [46,] 14 0.52440   3.74500
## [47,] 14 0.52670   3.41200
## [48,] 15 0.52870   3.10900
## [49,] 15 0.53030   2.83300
## [50,] 15 0.53160   2.58100
## [51,] 16 0.53280   2.35200
## [52,] 17 0.53420   2.14300
## [53,] 18 0.53580   1.95300
## [54,] 18 0.53760   1.77900
## [55,] 18 0.53890   1.62100
## [56,] 18 0.54000   1.47700
## [57,] 18 0.54090   1.34600
## [58,] 18 0.54160   1.22600
## [59,] 18 0.54220   1.11700
## [60,] 18 0.54280   1.01800
## [61,] 18 0.54320   0.92770
## [62,] 18 0.54360   0.84530
## [63,] 18 0.54380   0.77020
```

```
## [64,] 19 0.54410   0.70180
## [65,] 19 0.54430   0.63940
## [66,] 19 0.54450   0.58260
## [67,] 19 0.54470   0.53090
## [68,] 19 0.54490   0.48370
## [69,] 20 0.54510   0.44070
## [70,] 20 0.54520   0.40160
## [71,] 20 0.54530   0.36590
## [72,] 20 0.54540   0.33340
## [73,] 20 0.54550   0.30380
## [74,] 20 0.54560   0.27680
## [75,] 20 0.54570   0.25220
## [76,] 20 0.54570   0.22980
## [77,] 20 0.54580   0.20940
## [78,] 20 0.54580   0.19080
## [79,] 20 0.54590   0.17380
## [80,] 20 0.54590   0.15840
## [81,] 20 0.54590   0.14430
## [82,] 20 0.54590   0.13150
## [83,] 20 0.54600   0.11980
## [84,] 19 0.54600   0.10920
## [85,] 19 0.54600   0.09948
## [86,] 19 0.54600   0.09064
## [87,] 19 0.54600   0.08259
## [88,] 20 0.54600   0.07525
## [89,] 20 0.54600   0.06856
```

```
pred = predict(lasso.tr, x[-train,])
dim(pred)
```

```
## [1] 83 89
```

```
rmse = sqrt(apply((y[-train]-pred)^2, 2, mean))
plot(log(lasso.tr$lambda), rmse, type="b", xlab="Log(lambda)")
```




```
lam.best = lasso.tr$lambda[order(rmse)[1]]
lam.best
```

```
## [1] 19.98706
```

```
coef(lasso.tr, s=lam.best)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  107.9416686
## AtBat          .
## Hits           0.1591252
## HmRun          .
## Runs           .
## RBI            1.7340039
## Walks          3.4657091
## Years          .
## CAtBat         .
## CHits          .
## CHmRun         .
## CRuns          0.5386855
## CRBI           .
## CWalks         .
## LeagueA      -30.0493021
## LeagueN        .
## DivisionW   -113.8317016
## PutOuts        0.2915409
## Assists        .
## Errors         .
## NewLeagueN     2.0367518
```

==