ANLY 590 Brody Vogel HW 0

Brody Vogel

9/14/2018

Here I load packages and data (ISLR has 'Hitters').

```
library("glmnet")
```

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

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

```
## Loaded glmnet 2.0-13
```

```
library("ISLR")
library("plotmo")
```

```
## Warning: package 'plotmo' was built under R version 3.4.4
```

```
## Loading required package: plotrix
```

```
## Warning: package 'plotrix' was built under R version 3.4.4
```

```
## Loading required package: TeachingDemos
```

This builds a dataframe without missing values or categorical variables. It appears that Salary is the only variable with missing values. When observations with missing Salary values are removed, a fair number of rows (59) are lost.

```
                  # throw out the NaNs                    # only keep numeric columns
Hitters_numeric <- na.omit(Hitters[, unlist(lapply(Hitters, is.numeric))])
```

1.1

This first trains the LASSO on the numeric variables of the Hitters dataset, using Salary as the target variable. It should be noted that the default LASSO algorithm from glmnet - the one used below - normalizes the input dataframe. Plotted, then, are the coefficient trajectories for each variable; as the visual kind of shows (I omitted the numeric coefficient trajectories), the final three predictors that remain in the model are: Hits, CRBI, and CRuns. These seem suitable, although I'm surprised HmRun didn't make the cut.

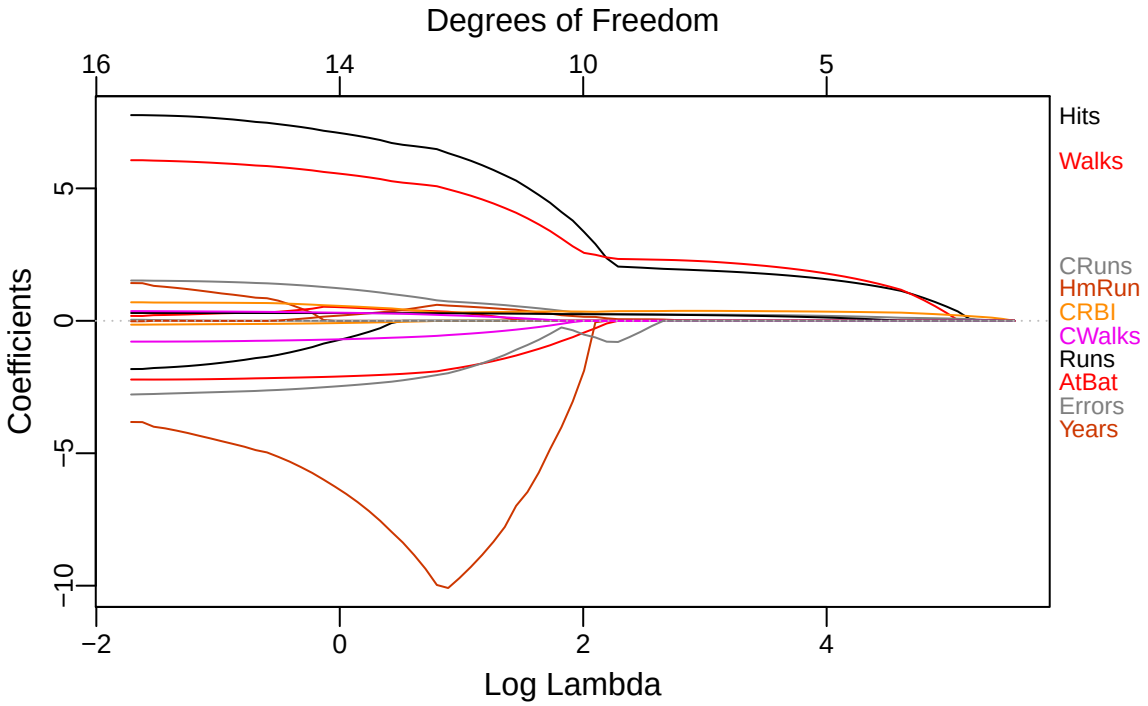
Next, a cross-validated LASSO model is fit to the data. The algorithm chooses $\lambda \approx 2.22$ as the optimal regularization penalty value. With $\lambda \approx 2.22$, then, there are 5 variables left in the model: Hits, Walks, CRuns, CRBI, and PutOuts. It's interesting that Years doesn't make the cut, here, as one would think that'd be an important factor in Salary.

```
# for reproducibility
set.seed(36)

# train the LASSO
lasso.hitters <- glmnet(as.matrix(Hitters_numeric[, -length(Hitters_numeric)]), Hitters_numeric$Salary,
```

```
#coef(lasso.hitters)

# coefficient trajectory plot
plot_glmnet(lasso.hitters, label = 10, xvar = 'lambda')
```




```
# cross-validated LASSO
lasso.hitters.cv <- cv.glmnet(as.matrix(Hitters_numeric[, -length(Hitters_numeric)]), Hitters_numeric$Sa

# optimized lambda value
best.lam <- lasso.hitters.cv$lambda.min
best.lam
```

```
## [1] 2.220313
```

1.2

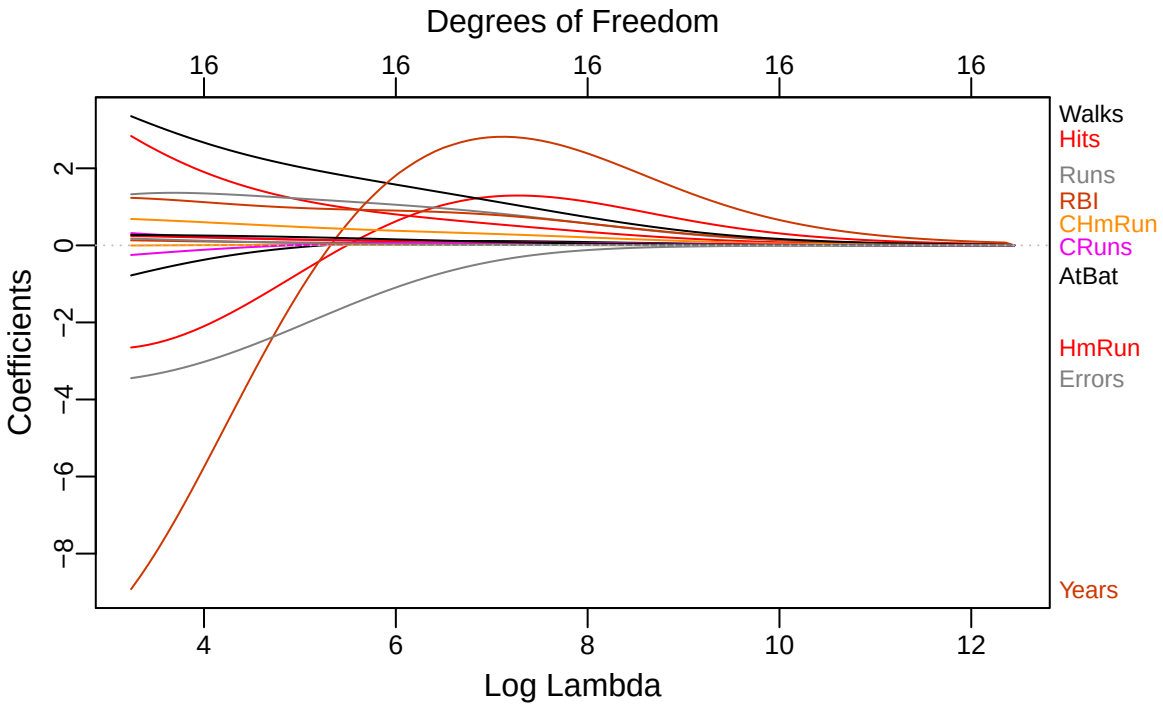
Finally, Ridge Regression is implemented on the same dataset. As expected, the coefficient trajectories look much different. Interestingly, though, it seems that different variables are being chosen as "most important", based on coefficient sizes.

The cross-validated model chooses $\lambda \approx 28.017$ as the optimal regularization penalty value. Again based on coefficient sizes, it looks like Ridge Regression picks out Hits, HmRun, Runs, RBI, Walks, Years, CHmRun, and Errors as important variables.

```
# train a Ridge Regression model
ridge.hitters <- glmnet(as.matrix(Hitters_numeric[, -length(Hitters_numeric)]), Hitters_numeric$Salary,

# coefficient trajectory plot
```

```r
plot_glmnet(ridge.hitters, xvar = 'lambda')
```




```r
# cross-validated Ridge model
ridge.hitters.cv <- cv.glmnet(as.matrix(Hitters_numeric[, -length(Hitters_numeric)]), Hitters_numeric$Sa

# optimal lambda value
best.lam.ridge <- ridge.hitters.cv$lambda.min
best.lam.ridge
```

```
## [1] 28.01718
```

```r
# coefficients from cross-validated Ridge model
coef(ridge.hitters.cv, lambda = best.lam.ridge)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                            1
## (Intercept) 204.187152675
## AtBat         0.090089537
## Hits          0.372342169
## HmRun         1.172627192
## Runs          0.598953622
## RBI           0.596440693
## Walks         0.775959444
## Years         2.475202324
## CAtBat        0.007600761
## CHits         0.029298212
## CHmRun        0.217084304
## CRuns         0.058824003
## CRBI          0.060704823
## CWalks        0.058907683
## PutOuts       0.052855985
## Assists       0.007741647
```

```
## Errors       -0.141046138
```

2

Any model can be trained with varying levels of flexibility. If a model is less flexible, it will stick more to its broad structure and learn less from the training data. On the other hand, if a model is more flexible, it will better learn the intricacies of the training data and let them shape its structure. In the first case, the model is biased: it makes strong assumptions about the data's structure and only slightly modifies those assumptions based on the training data. In the second case, the model has a lot of variance: because it learns so much of its structure from the training data, that structure would change dramatically with different training data. Depending on how closely unseen (test) data resembles the training data, then, the two models would have very different performance. This is the bias-variance tradeoff.

As demonstrated above, regularization plays a large role in this tradeoff. Speaking first to the LASSO model, when we used regularization to get exactly three features, we consequently built a biased model; we shrunk the coefficients to a degree that greatly decreased the variability of the models predictions for players' salaries. When we let cross-validation find an optimal regularization penalty value, though, the outputted model had more variability; it kept five variables, thus increasing the effect of the features on predictions and decreasing the model's bias. The implementation of Ridge Regression decreased the bias even moreso: because none of the coefficients can shrink to 0, the features will have a greater impact on the model's predictions for player salary, which makes the model more variable and less biased.