Group Name: The Dream Team

Team Members: Sanjay Hariharan Azeem Zaman Lin Xiao Sunith Suresh

Part 1

```r
library(gamlr) # loads Matrix as well
```

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

```r
help(hockey) # describes the hockey data and shows an example regression

data(hockey) # load the data

# Combine the covariates all together
x <- cBind(config,team,player) # cBind binds together two sparse matrices

# build 'y': home vs away, binary response
y <- goal$homegoal

nhlreg <- gamlr(x, y,
    free=1:(ncol(config)+ncol(team)), ## free denotes unpenalized columns
    family="binomial", standardize=FALSE)

## coefficients (grab only the players)
# AICc selection
Baicc <- coef(nhlreg)[colnames(player),]

#Which AICc is the minimum? Aka what is the value of that line on the plot##
log(nhlreg$lambda[which.min(AICc(nhlreg))])
```

```
##     seg55
## -9.165555
```

Number of Players with Significant Effects: 646

So out of 2439 total players, about 26.4% players had a significant effect contributing to a goal. This is one of the benefits of using the lasso; there are many coefficients that are selected to be exactly zero. If we had used ridge regression, there would be many coefficients that were small in absolute value, but not zero.

The 10 players with largest Beta coefficient values are: PETER_FORSBERG, TYLER_TOFFOLI, ONDREJ_PALAT, ZIGMUND_PALFFY, SIDNEY_CROSBY, JOE_THORNTON, PAVEL_DATSYUK, LOGAN_COUTURE, ERIC_FEHR, MARTIN_GELINAS

We know Logan Couture is a popular and good player, and we can see that the odds that his team scored when he is on the ice is multiplied by 1.4451459, or increases by 44.5%.

Also, Mario Lemieux, who I thought was a very good player, actually has no effect on the odds of his team scoring, since he is not included in the players with significant effects.

The intercept is 0.0791386.

This means that regardless of the player or special effects, the odds that the home team scored is multiplied by 1.0823543, or increases by around 8%, which means that the home team always has a small advantage.

Part 2

Standardizing the coefficients makes them all have a standard deviation of 1, or multiplying the L1 penalty by each coefficient standard deviation.

Keep in mind that we want to find which players that have a positive or negative effect on their team scoring a goal, so we do not want to penalize players who play for long periods of time, since they are likely better and have a greater positive effect. The players who play for a long time will have a larger standard deviation, and those who player for a short amount of time will have a small standard deviation.

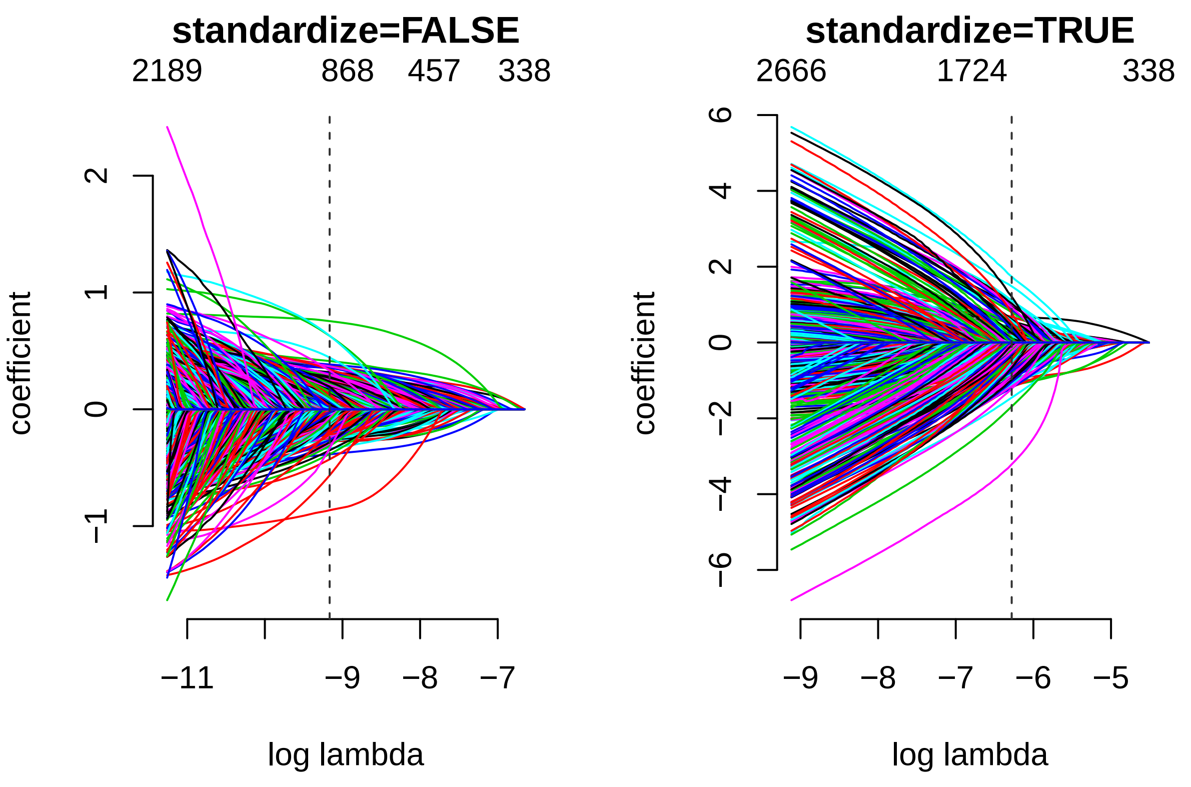
So we would be giving a bigger penalty for those who spend more time on the ice, and a smaller penalty for those who play the same, essentially making them all equal. But keeping in mind our end goal, this is not appropriate.

Now we consider the results of standardizing the data:

```r
#Run GAMLR with Standardize = TRUE#
nhlreg.std <- gamlr(x, y,
    free=1:(ncol(config)+ncol(team)), ## free denotes unpenalized columns
    family="binomial",
    standardize=TRUE) # standardize the data

#Pull coefficients for all players#
Baicc.std <- coef(nhlreg.std)[colnames(player),]

#Plot both standardized and un-standardized lasso plots#
par(mfrow=c(1,2))
plot(nhlreg, main="standardize=FALSE")
plot(nhlreg.std, main = "standardize=TRUE")
```



Number of Players with Significant Effects: 646

Number of Players with Significant Effects after Standardization: 865

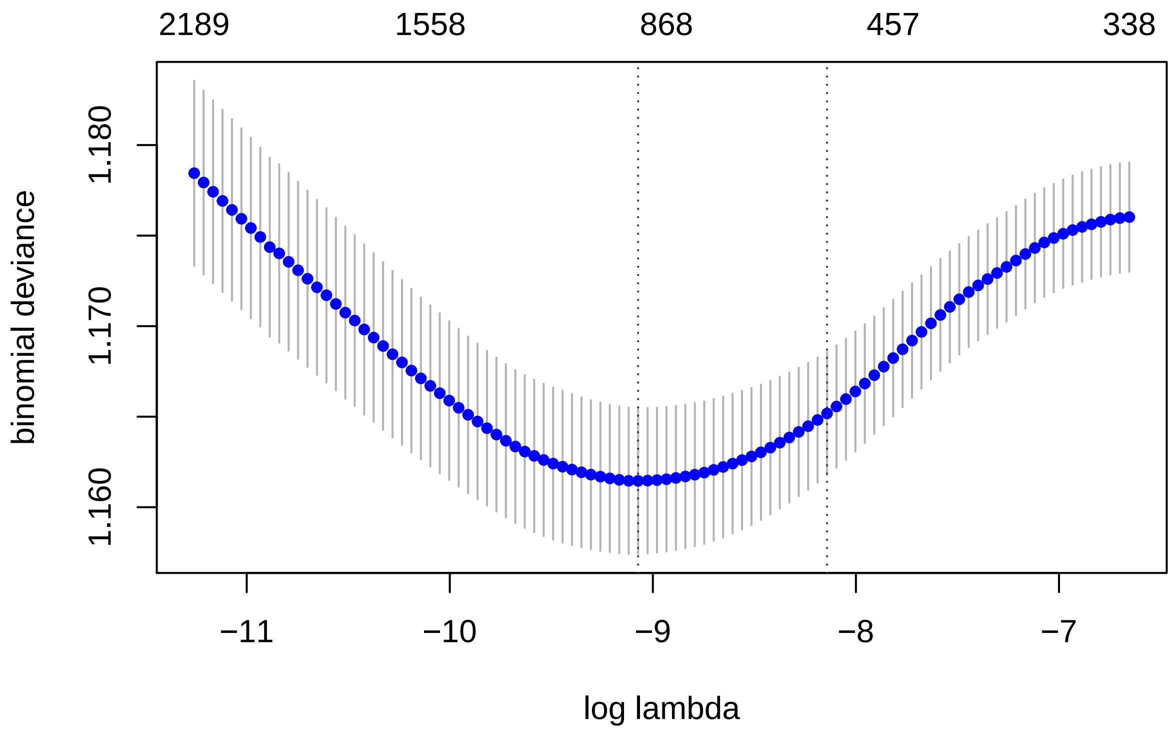
A comparison of the graphs reveals that when standardizing the data, more coefficients are non-zero. Furthermore, the estimated parameters themselves are generally larger, based on the wider spread of the coefficient lines. We also lose our interpretation of the effect relationship between β_k and the odds, since our data will no longer be binary.

For the non-standardized data, the magnitude of the largest estimated parameter at the lowest value of lambda appears to be around 3, while for the standardized there are estimates as large as 6 in absolute value.

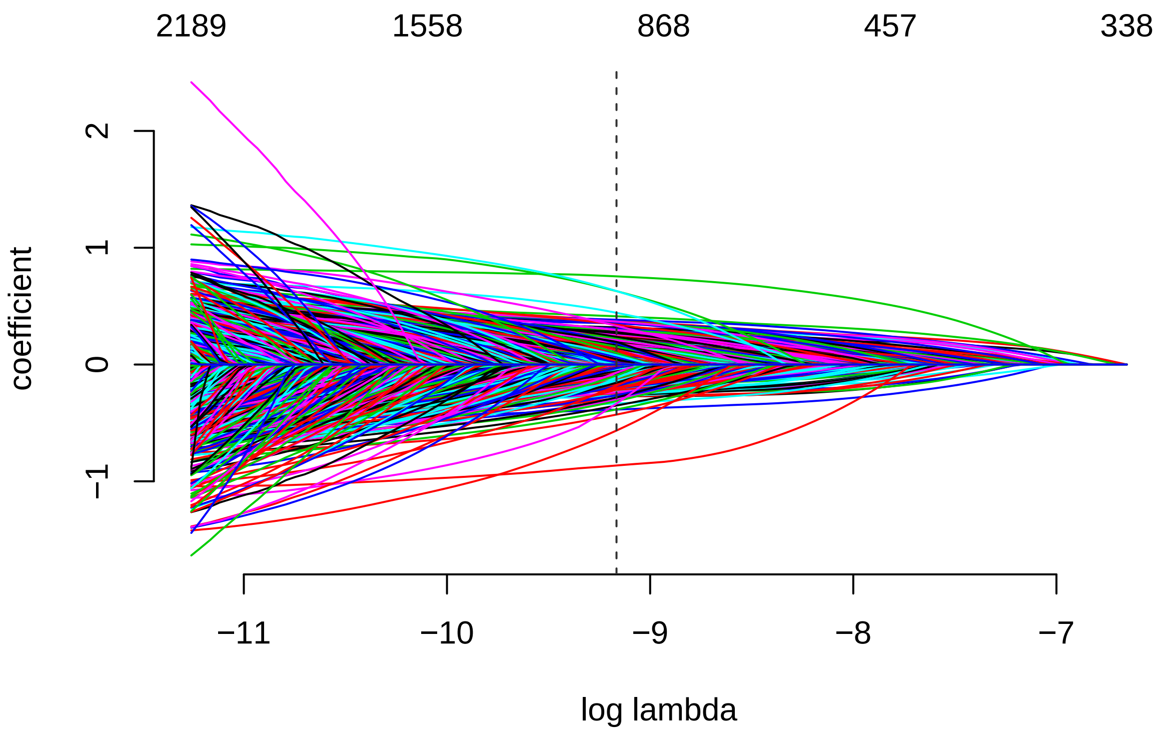
Part 3

```r
#Run Cross-Validation GAMLR#
cv.nhlreg <- cv.gamlr(x, y, free = 1:(ncol(config) + ncol(team)),
                      family= "binomial",
                      standardize = FALSE)

#Plot CV GAMLR result#
plot(cv.nhlreg)
```



```r
#Plot normal GAMLR#
plot(nhlreg)
```



```r
#What are the log lambdas under several different criteria?#
log(nhlreg$lambda[which.min(AICc(nhlreg))])
```

```
##     seg55
## -9.165555
```

```r
log(nhlreg$lambda[which.min(BIC(nhlreg))])
```

```
##      seg1
## -6.653644
```

```r
log(cv.nhlreg$lambda.min)
```

```
## [1] -9.072521
```

```r
log(cv.nhlreg$lambda.1se)
```

```
## [1] -8.142184
```

We can see that AICc is pretty close to the CV minimum value.

AICc attempts to approximates the OOS deviance, and the lambda value is an estimate of the lambda that minimizes the OOS error. The CV Minimum value is trying to minimize the OOS error as well. Since n is quite large with respect to p, AICc will be quite close to AIC, so either one of those criteria will work well.

```r
#Pull Coefficients for players under CV Min#
cv.lambda.min <- coef(cv.nhlreg, select = "min")[colnames(player),]

#How many players have a significant effect?#
sum(cv.lambda.min != 0)
```

```
## [1] 601
```

```r
#What are the names of the top 10 players?#
names(sort(cv.lambda.min, decreasing = TRUE))[1:10]
```

```
##  [1] "PETER_FORSBERG" "ONDREJ_PALAT"   "TYLER_TOFFOLI"  "ZIGMUND_PALFFY"
##  [5] "SIDNEY_CROSBY"  "JOE_THORNTON"   "PAVEL_DATSYUK"  "LOGAN_COUTURE"
##  [9] "ERIC_FEHR"      "MATT_MOULSON"
```

```r
#Pull Coefficients for players under CV 1 SE#
cv.lambda.1se <- coef(cv.nhlreg, select = "1se")[colnames(player),]

#How many players have a significant effect?#
sum(cv.lambda.1se != 0)
```

```
## [1] 207
```

```r
#What are the names of the top 10 players?#
names(sort(cv.lambda.1se, decreasing = TRUE))[1:10]
```

```
##  [1] "PETER_FORSBERG"    "SIDNEY_CROSBY"     "JOE_THORNTON"
##  [4] "PAVEL_DATSYUK"     "LUBOMIR_VISNOVSKY" "ALEXANDER_SEMIN"
##  [7] "ALEX_OVECHKIN"     "RYAN_GETZLAF"      "MARIAN_GABORIK"
## [10] "HENRIK_SEDIN"
```

The CV Lambda 1se model balances prediction against "False discovery", and thus accounts for uncertainty about our OOS error. It will choose a simpler model, which is why the names of our top 10 players differs slightly.

```r
#Pull coefficients for players under lowest BIC value#
bic.nhlreg <- coef(nhlreg, select = which.min(BIC(nhlreg)))[colnames(player),]

#How many players have significant effects#
sum(bic.nhlreg != 0)
```

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

BIC is telling us that no players have any significant effects at all.

The Interpretation of BIC is different. It approximates the probability that the Lasso Regression Model is true. We find the lambda value that gives us the smallest BIC, which is the highest probabiity that the model is true, or that the OOS error is at a minimum.

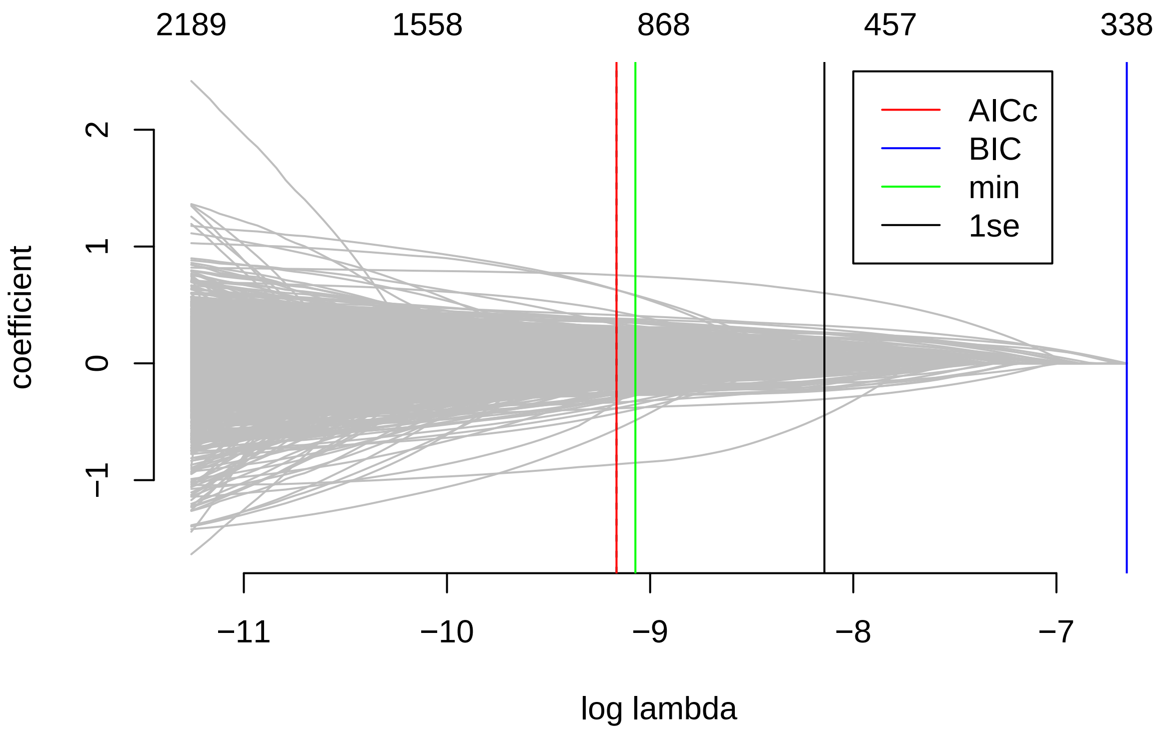
Since BIC chooses a model with all player coefficients at zero, it means that it is sufficient to predict a home goal based on the fixed special effects in the model. It also means that there is a lot of uncertainty in the lasso model, unable to distinguish a player's level of player in a season from average.

All of these values can be plotted on the same graph.

```r
#Plot full lasso object#
plot(nhlreg, col = "gray")

#Add colored lines for each IC lambda value (AICc, BIC, CVmin, CV1se)#
abline(v = log(nhlreg$lambda[which.min(AICc(nhlreg))]), col = "red")
abline(v = log(nhlreg$lambda[which.min(BIC(nhlreg))]), col = "blue")
abline(v = log(cv.nhlreg$lambda.min), col = "green")
abline(v = log(cv.nhlreg$lambda.1se), col = "black")

#Add legend#
legend(-8, 2.5, c("AICc","BIC","min","1se"),
       lty = c(1,1,1,1),
       col = c("red","blue","green","black"))
```

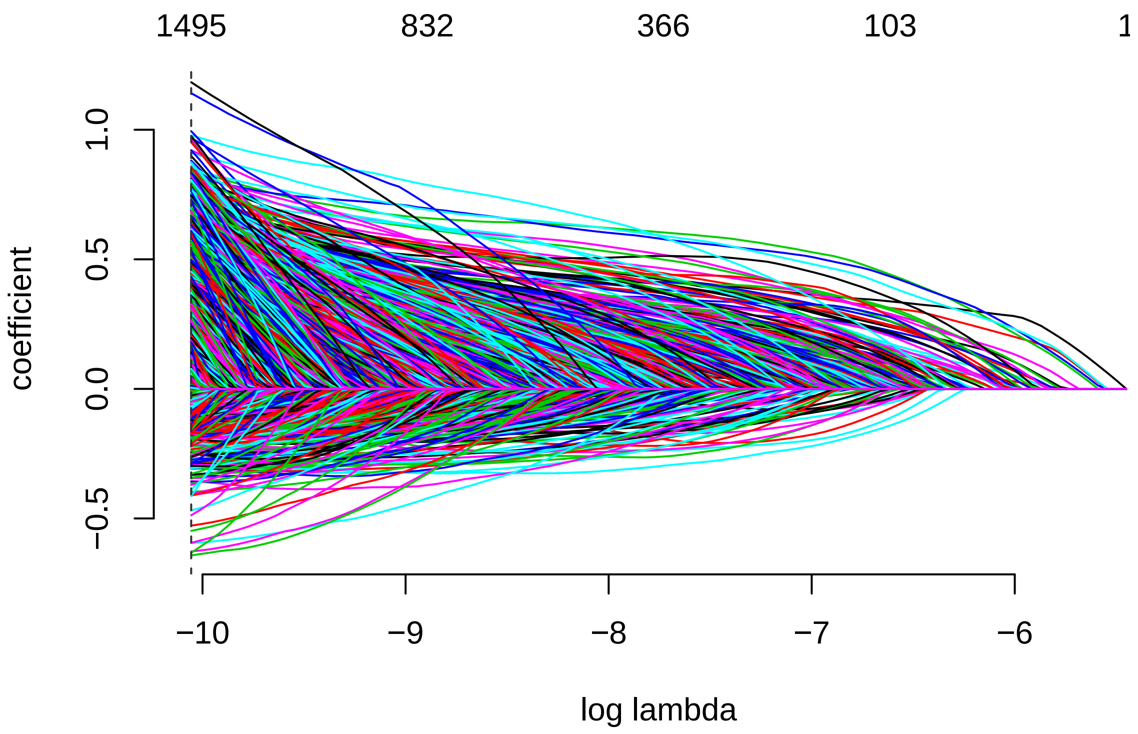


In the graph the similarity of AICc and the minimum OOS deviance determined by cross validation is clear. The extremity of the BIC is also clear; no parameters are greater than zero at this level of λ.

Part 4

```r
#Re-run GAMLR with only player models#
nhlreg.player <- gamlr(player, y, family = "binomial", standardize = FALSE)

#Plot the result#
plot(nhlreg.player)
```



```r
#Which AICc is the minimum? Aka what is the value of that line on the plot##
log(nhlreg$lambda[which.min(AICc(nhlreg.player))])
```

```
##    seg100
## -11.25881
```

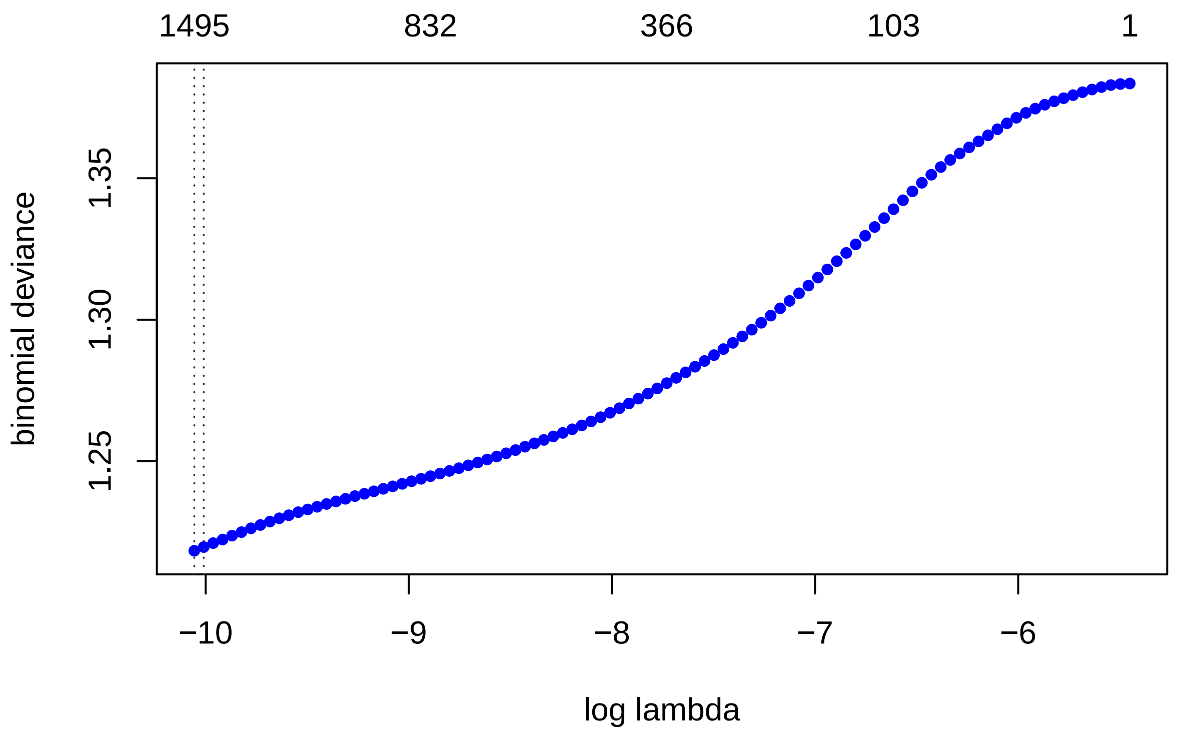
It looks like without our controlling variables, our AIC value is at the farthest left of the plot, which means we are choosing the most complex model. Put a different way, we are selecting a model that is an approximation of the model we would obtain without use of the lasso (the logit model).

It is useful to think of this result in terms of the bias-variance tradeoff. In the regular (logit) model without the lasso, there is low bias (the parameters are unbiased estimators of the true values under certain conditions) but high variance. The most extreme case of the lasso occurs when the parameter λ is so large that all of the parameters are estimated to be zero except for the constant term β_0. In this case, our prediction is a constant, which has low variance and high bias (a constant is not an unbiased estimator).

When we omit the controls we are likely introducing bias into the model (assuming that these factors actually do impact the response varaible, which appears to be the case). This bias increases the OOS deviance. The OOS deviance, however, decreases as the model becomes more complex. The effect of omitting the controls on the bias (and thus the OOS deviance) is so large that the method is forced to pick the most complex model to minimize the deviance.

```r
#Run CV Gamlr#
cv.nhlreg.player <- cv.gamlr(player, y, family = "binomial", standardize = FALSE)

#Plot result#
plot(cv.nhlreg.player)
```



```r
#Pull coefficients for players under lowest CV lambda value#
lambda.min.cv <- coef(cv.nhlreg.player, select = "min")[-1,]

#How many players are significant?#
sum(lambda.min.cv != 0)
```

```
## [1] 1494
```

There are now 1494 significant players.

Even with cross validation, our model is still quite complicated. 1494 players are significant, which is a large amount!

Let us take a look at the actual OOS Deviances and AICc values for models with and without controls:

```r
#OOS Deviance with controls#
min(cv.nhlreg$cvm)
```

```
## [1] 1.161454
```

```r
#OOS Deviance with just players#
min(cv.nhlreg.player$cvm)
```

```
## [1] 1.218285
```

```r
#Minimum AICc with controls#
min(AICc(cv.nhlreg$gamlr))
```

```
## [1] 80480.25
```

```r
#Minimum AICc with just players#
min(AICc(cv.nhlreg.player$gamlr))
```

```
## [1] 84395.8
```

It is clear that the model with controls has a lower OOS Deviance and AICc value and thus performs better.

Extra Credit

When player β_k is on the ice:

$P(\text{Home Team Scored}|\text{Goal was scored and player k was on the ice}) = e^{\beta_k}/(1+e^{\beta_k})$

$P(\text{Away Team Scored}|\text{Goal was scored and player k was on the ice}) = 1 - P(\text{Home Team Scored}|\text{Goal was scored})$

To approximate difference in Probability for that specific player k, we can use:

$P(\text{Home Team Scored}|\text{Goal was scored and player k was on the ice})$ -
$(1 - P(\text{Home Team Scored}|\text{Goal was scored and player k was on the ice})) =$
$2 * P(\text{Home Team Scored}|\text{Goal was scored and player k was on the ice}) - 1$

If we know the number of total goals scored while that player was on the ice, call it **G**, and multiply it by the difference in Probability, we get the estimated number of home goals scored **minus** the estimated number of away goals scored, which is analogous to the classic plus-minus.

Estimated # of Goals = G * Difference in Probability, and we define it as the new Plus-Minus(NPM), compared to the class Plus Minus(CPM).

```r
P <- exp(Baicc)/(1+exp(Baicc))

# We turn all the non-zero values into 1
# So we can calculate the column sums as the number of
# total goals scored while that player was on the ice.
G <- colSums(abs(player))

# The way we defined new Plus Minus
NPM <- G*(2*P-1)

# And we calculate the classic Plus Minus
# Since home players originally get an +1, and -1 for away players
# We need to use homegoal column to adjust it before calculating the CPM
CPM <- colSums(player*c(-1,1)[y+1])

# Compare the top 10 players
names(sort(CPM, decreasing = T)[1:10])
```

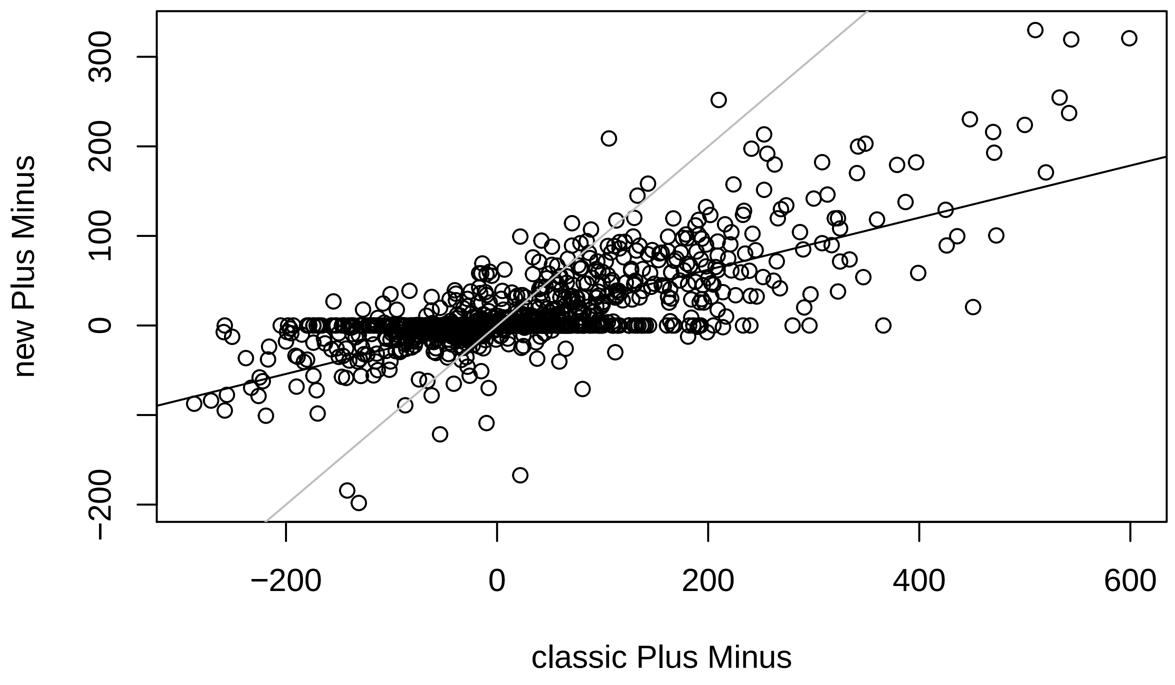
```
##  [1] "PAVEL_DATSYUK"     "SIDNEY_CROSBY"     "HENRIK_SEDIN"
##  [4] "ALEX_OVECHKIN"     "DANIEL_SEDIN"      "JOE_THORNTON"
##  [7] "NICKLAS_LIDSTROM"  "EVGENI_MALKIN"     "HENRIK_ZETTERBERG"
## [10] "DANIEL_ALFREDSSON"
```

```r
names(sort(NPM, decreasing = T)[1:10])
```

```
##  [1] "JOE_THORNTON"      "PAVEL_DATSYUK"     "SIDNEY_CROSBY"
##  [4] "ALEX_OVECHKIN"     "HENRIK_LUNDQVIST"  "HENRIK_SEDIN"
##  [7] "MARIAN_HOSSA"      "NICKLAS_LIDSTROM"  "DANIEL_ALFREDSSON"
## [10] "ANDREI_MARKOV"
```

```
# We plot the points and fit a line
plot(CPM, NPM, xlim=range(CPM), xlab="classic Plus Minus",
     ylab="new Plus Minus")
abline(lm(NPM~CPM))

# Also draw th y=x line
abline(a=0,b=1,col=8)
```



```
# Finally we try to find the correlation
cor(NPM, CPM)
```

```
## [1] 0.7407613
```

We can see from the top 10 players' names, as well as the plot and the correlation that Plus Minus values give similar results. Also, by drawing the y=x line, since new Plus Minus considers more factors when evaluating a player, we can tell that all the points that are below y=x are the ones that have been overrated by classic Plus Minus, and the ones that are above the line are the ones that are underrated by classic Plus Minus.