Assessing popular prediction models on baseball strikeouts-SAMPLE

Akhil Jonnalagadda

Baseball

Baseball is a sport that has an abundance of metrics that can be assessed from a data mining perspective. As such, modeling the data is a unique and intriguing challenge—is it possible to find a model that most accurately describes the most pertinent metrics for baseball's pitchers and batters? Further, is it possible to find the traits of the best players by employing statistical modeling techniques?

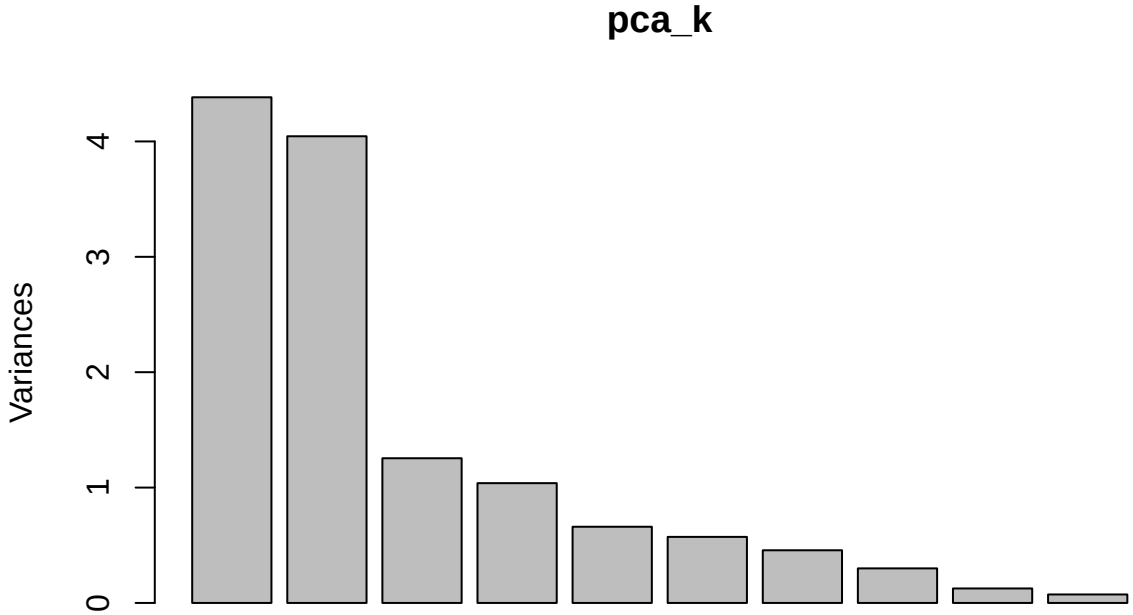
By breaking down baseball's two most widely discussed positions, we are able to better breakdown trends through analysis of a pitcher's strikeouts (K's) and walks and hits per inning (whip) and a batter's homerun counts.

Pitching

There are many metrics that define a pitcher. Pitchers need to limit earned runs—this means getting many strikeouts or preventing hits and walks. As such, pitchers with a high count of strikeouts are perceived as better and pitchers with low whips are seen as especially elite. To begin, examination of PCA was utilized to see if it was possible to sort on features to find the best pitchers.

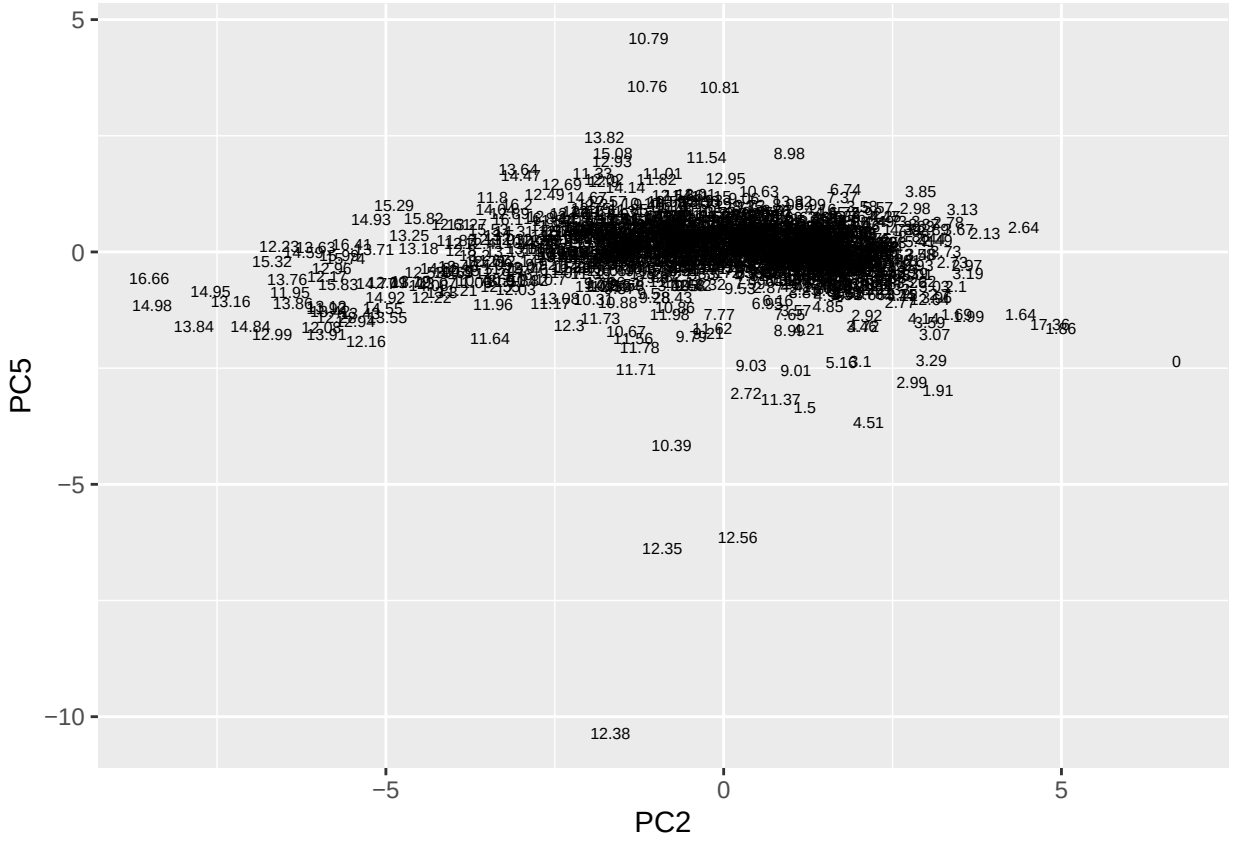
Principal Components Analysis

PCA on strikeouts per nine innings was the first variable examined. Ideally, it will be possible to distinguish between high and low strikeout counts—if so, there is confidence that principal component analysis works to sort bad pitchers from great pitchers.



```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.0935 2.0111 1.11989 1.01895 0.81258 0.75666 0.67560
## Proportion of Variance 0.3371 0.3111 0.09647 0.07987 0.05079 0.04404 0.03511
## Cumulative Proportion  0.3371 0.6483 0.74473 0.82460 0.87539 0.91943 0.95454
##                           PC8     PC9    PC10    PC11    PC12    PC13
## Standard deviation     0.54739 0.35414 0.27132 0.24248 0.18138 0.02425
## Proportion of Variance 0.02305 0.00965 0.00566 0.00452 0.00253 0.00005
## Cumulative Proportion  0.97759 0.98724 0.99290 0.99742 0.99995 1.00000
```

```
##        PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8
## k_9   0.02 -0.41  0.09 -0.37  0.14  0.28 -0.23  0.39
## wpct  0.16 -0.16 -0.51 -0.41  0.02 -0.68  0.20  0.07
## era  -0.18  0.39  0.19 -0.07 -0.12 -0.20 -0.02  0.72
## sv   -0.01 -0.40  0.37  0.26 -0.27 -0.30  0.12  0.06
## ip    0.45  0.12  0.10  0.06  0.00 -0.15 -0.18 -0.17
## hr    0.38  0.21  0.15  0.14  0.11 -0.14 -0.33  0.18
## bb_9 -0.22  0.06  0.45 -0.64 -0.03 -0.10 -0.18 -0.42
## so    0.38 -0.22  0.13 -0.26  0.04  0.08 -0.19  0.21
## tbf   0.44  0.15  0.12  0.07  0.01 -0.16 -0.17 -0.19
## obp  -0.17  0.42  0.24 -0.10 -0.12 -0.23  0.09 -0.02
## sho   0.29  0.09 -0.11 -0.21 -0.83  0.31  0.19  0.02
## hb    0.32  0.09  0.31 -0.13  0.37  0.16  0.77  0.06
## gf   -0.03 -0.42  0.36  0.20 -0.19 -0.27  0.10  0.01
```

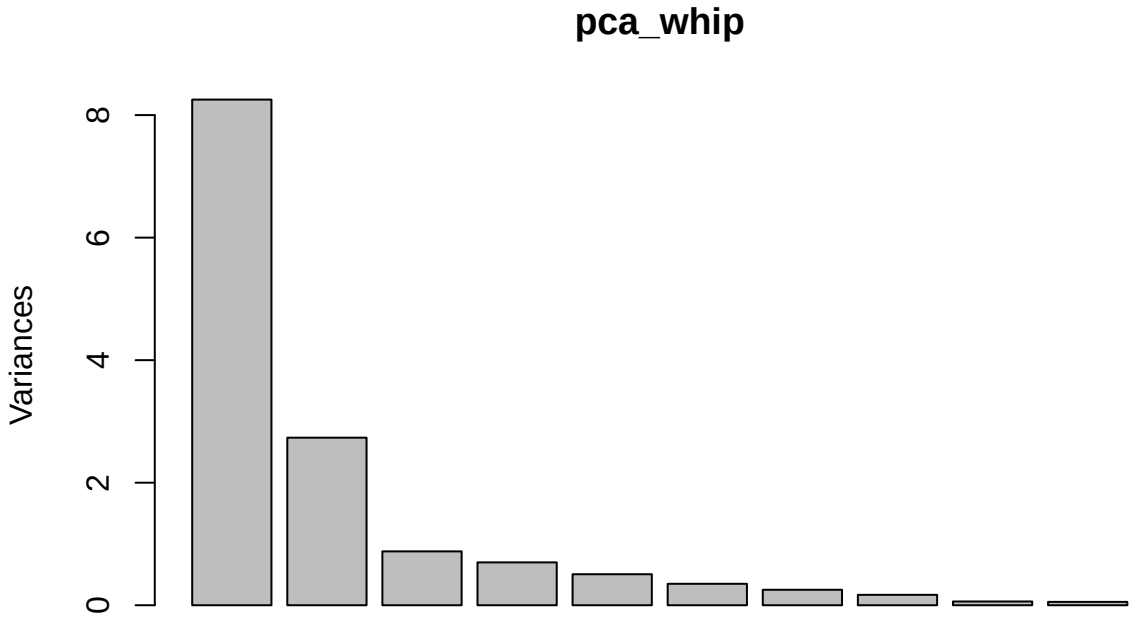


The plotted PCA's showcase the graphical significance of each principal component in the whole. Numerically, the summarized principal components describe how much of the data is explained by each, both in cumulative and individually. Following the summary of the significance of each principal component, a generalized

breakdown of the impact on each variable is listed.

The PCA of K's per nine innings showcases a high degree of clustering around (0,0) in the axis of (PC2, PC5); however, there is a trend that the best pitchers have PC2 values between -5 and -7.5 while the worst pitchers have PC2 values greater than zero. This isn't a great model since it results in having the highest strikeout count per nine innings (17.36) close to the lowest (0). Despite this, the average great pitcher contains many of the same traits, according to PC2.

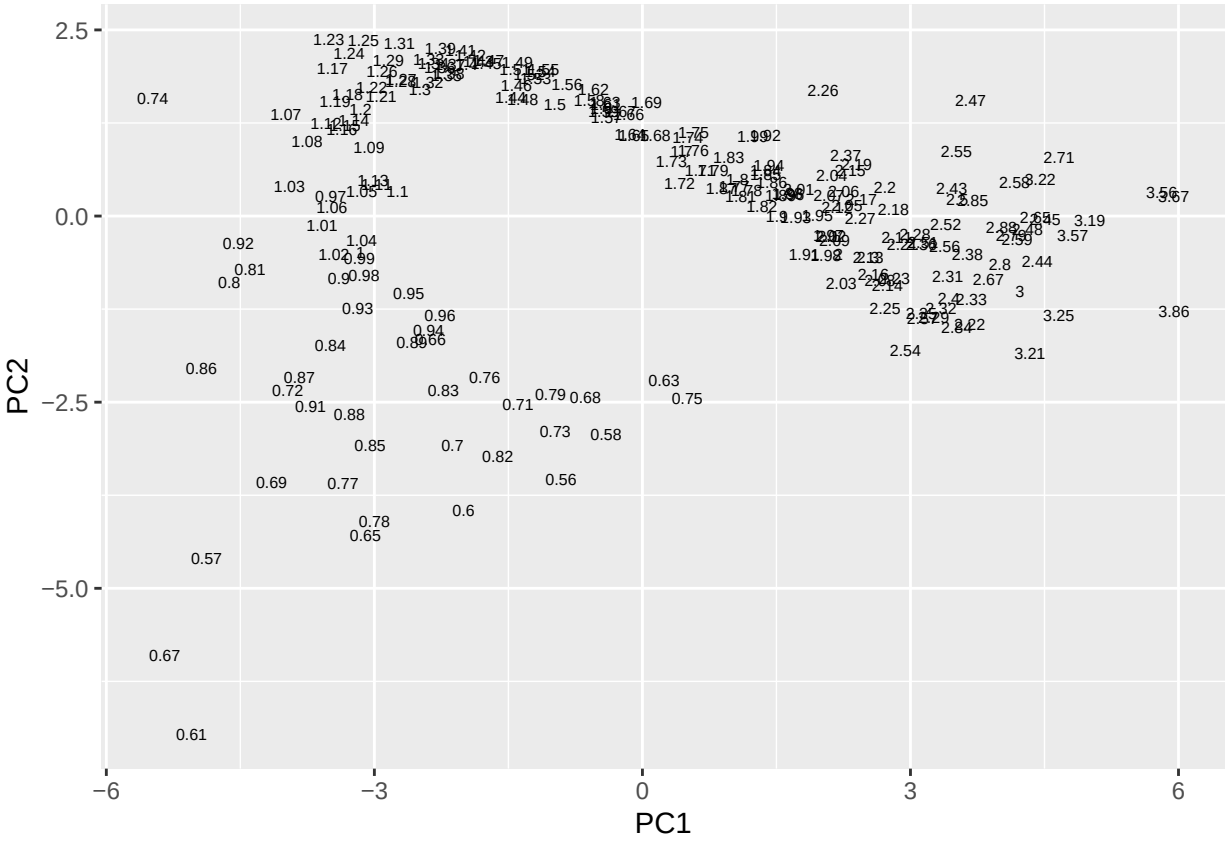
Although it is possible to sort (between good and bad pitchers, at least) on strikeouts per nine innings, it isn't very easy to follow due to the cluster in the center, as well as the similar mappings of the best and worst characteristics of pitchers. To get a better feel for whether this is always the case, a pitcher's whip was also assessed using PCA.




```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.8728 1.6538 0.93756 0.83620 0.71153 0.59173 0.50236
## Proportion of Variance 0.5895 0.1954 0.06279 0.04995 0.03616 0.02501 0.01803
## Cumulative Proportion  0.5895 0.7849 0.84767 0.89761 0.93378 0.95879 0.97681
##                           PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.41218 0.24782 0.23406 0.13780 0.10956 0.08394 0.02232
## Proportion of Variance 0.01214 0.00439 0.00391 0.00136 0.00086 0.00050 0.00004
## Cumulative Proportion  0.98895 0.99333 0.99725 0.99860 0.99946 0.99996 1.00000
```

```
##        PC1   PC2   PC3   PC4   PC5   PC6   PC7
## whip  0.32  0.11 -0.08  0.35 -0.12  0.09 -0.09
## wpct -0.23 -0.24  0.35  0.13 -0.50  0.65 -0.04
## sv   -0.20 -0.38 -0.44  0.26 -0.19 -0.24  0.00
## ip   -0.31  0.26 -0.09  0.08 -0.08  0.13  0.09
## hr   -0.21  0.45 -0.21 -0.04  0.02  0.06 -0.05
## k_9  -0.20 -0.30  0.13  0.34  0.74  0.24 -0.24
## bb_9  0.29  0.06 -0.09  0.43  0.16  0.20  0.73
## so   -0.33  0.09 -0.03  0.21 -0.04  0.07  0.04
## tbf  -0.29  0.32 -0.13  0.05 -0.06  0.14  0.08
## obp   0.32  0.18 -0.12  0.26 -0.10  0.05 -0.09
## sho  -0.22  0.12  0.60  0.43 -0.13 -0.58  0.10
## hb   -0.23  0.38 -0.15  0.17  0.17  0.03 -0.21
```

```
## gf   -0.24 -0.34 -0.43  0.18 -0.15 -0.15  0.04
## era   0.31  0.08 -0.06  0.36 -0.17  0.01 -0.57
```



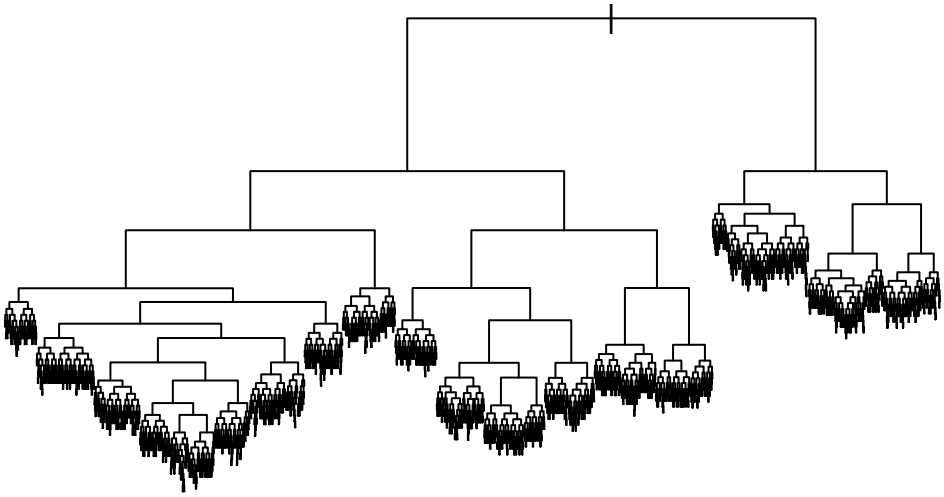
This graph very clearly lays out the best and worst features from the first two principal components (which comprise 72.68% of the variation in whip). The best—nay, elite—pitchers have whips with values less than one. We clearly see that under these principal components, these kinds of pitchers tend to have PC1 and PC2 values that are negative. Specifically, PC1 on average seems to be around -3 while PC2 is around -2.5. For great pitchers (with whip values less than 1.5), PC1 is in the range [-6, -1.5] and PC2 values are in the range of [-7, 2.5]. When these happen simulataneously, great pitchers are found.

Clearly it is possible to sort and filter the best traits using PCA, but is it possible to find an accurate model of strikeouts per nine innings predictions? To do this, rather than running a simple OLS linear model, random forests allow for the aggregation of important traits to predicting outcomes.

Random Forest

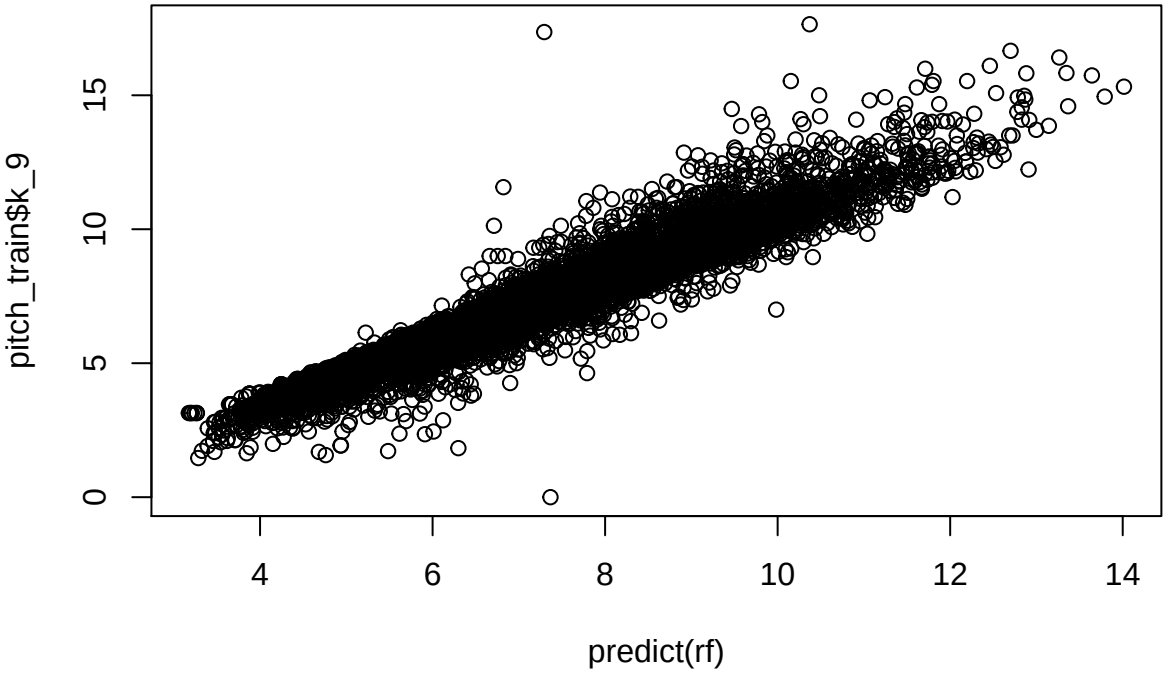
Random forests are resourceful for aggregating many trees to develop a best model. To best understand how this is done, an initial tree can be examined.

Tree



```
## [1] 0.4532665
```

This tree is just one possibility of how the data of pitchers could be presented and sorted. When aggregated 500 times, the best outcomes are given more significance which results in a best model. This process creates a 'random forest' wherein the model is able to be estimated through every tree produced. The RMSE for the single tree is around 0.45. Pruning the tree could produce an even better RMSE; however, a random forest should generate a better model.




```
## [1] 0.7152127
```

Interestingly, the RMSE does not lower from the single tree but rather increases. This is disappointing, but because the model accounts for more variation, the argument can be made that it accounts for more randomness and thus is the better model. Unfortuantely, this is a weak argument.

The point stands that a very good model can be generated from the tree and random forest method—one even better than standard OLS regression.

Lasso Selection and OLS

```
##   (Intercept)          wins        losses           era             g
##  1.717302e+01  9.700478e-01 -5.090247e-01 -1.075347e+00  1.630968e-01
##            gs            sv           svo            ip             h
##  1.128852e-01  9.121043e-02  1.847734e-01 -5.742635e-03 -1.213602e+00
##             r            er            hr            bb           avg
## -2.292875e-01  2.693539e-01  1.544339e-01 -4.211006e-01 -2.733828e+02
##          whip            cg           sho            hb           ibb
##  1.146247e+01 -6.454562e-01 -6.550737e-01 -2.675453e-01 -9.274587e-01
##            gf            hl          gidp            go            ao
## -1.215327e-01 -3.778492e-02  2.107533e-01 -1.615675e-01 -1.180141e-01
##            wp            sb            cs           tbf            np
##  9.139027e-01  5.714235e-01  1.171125e+00  4.694020e-01  1.662468e-02
##         go_ao           obp           slg           ops           h_9
## -3.725240e-01 -6.047193e+01 -7.819536e+00  3.097011e+01  6.813388e+00
##          p_ip          wpct
## -9.930583e-01 -8.894681e+00
```

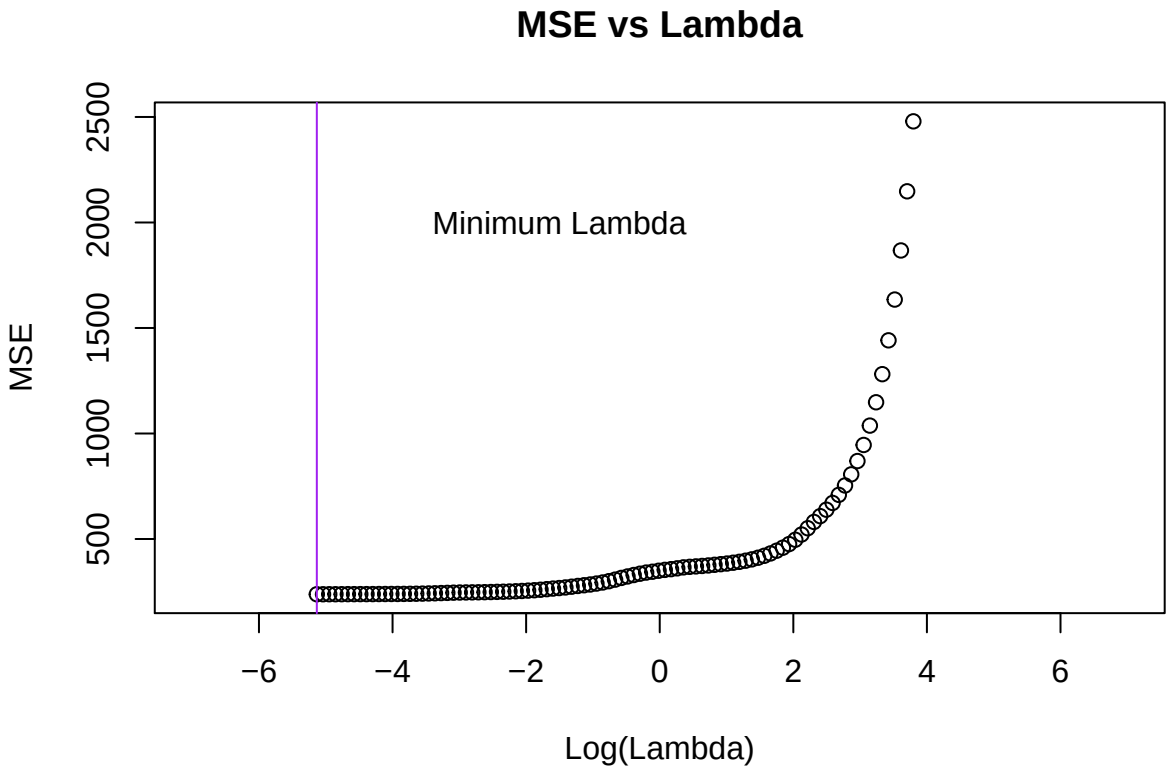
```
## [1] -5.133875
```

```
## [1] 0.005893678
```

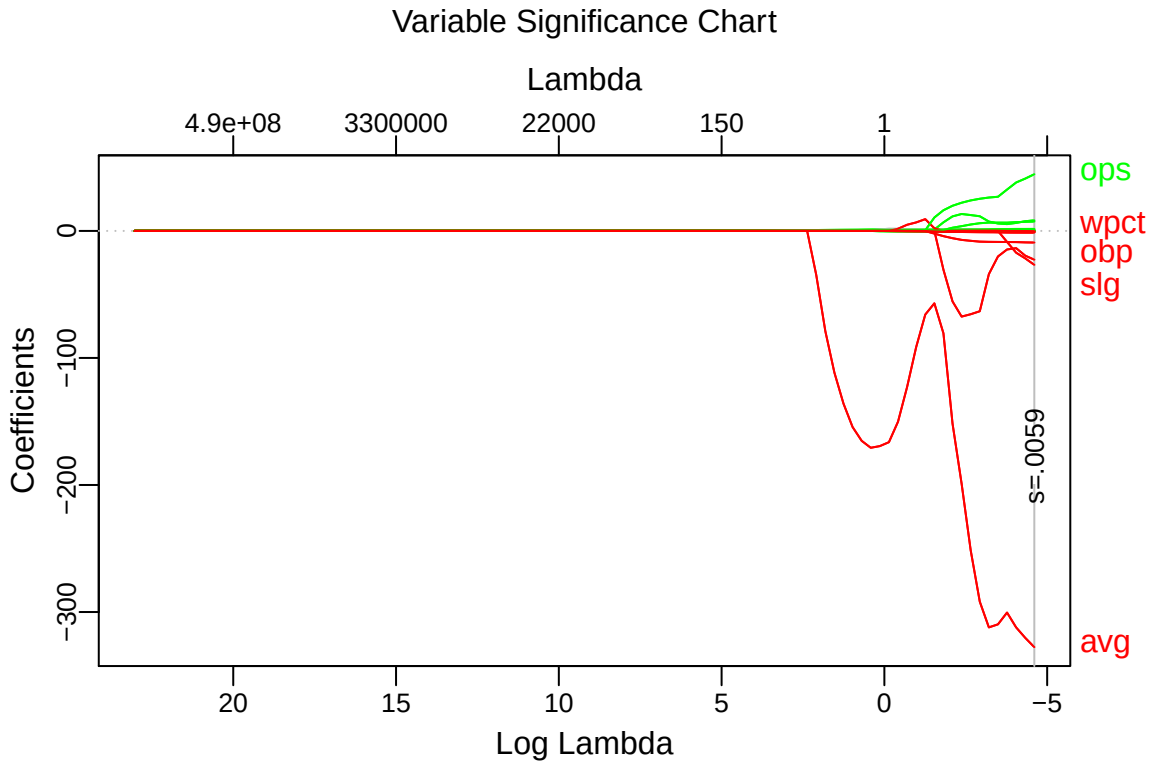
Here we are looking at creating a model to predict strikeouts. Simply put the goal of a pitcher is to strike the batter out in baseball. Modeling this is useful for teams to understand a pitchers ability to do their job.

Before any analysis we had to clean and address the data. This included removing all categorical variables and non trend specific data such as ID numbers. After this all metrics using strikeouts in their calculations were removed. The remaining variables for these models included those such as home runs, sv(pitcher ending a wining game), and an assortment of others.

The first method used was a LASSO(least absolute shrinkage and selection operator) to predict strikeouts. First we had to create training data and testing data. Using an 80/20 split on our sample that was done. The next part of this method including creating a matrix of all variable combinations and using a feature section method to choose the most significant variables to use in our analysis. This part also included finding the optimal "lambda" our penalty vector in this regression.



In this graph we see across these lambda values we plot the variables we selected and their respective coefficients. As not all metrics are the same we highlighted the 5 most significant according to our section model. The red being for negative impact and green for postie. OPS, the players on base and slugging average is the most significant positive variable and batting average being the most negative significant variable.



The next part of this method was to lock down the lambda. This was done by plotting the mean squared error and a series of lambda values to find the lowest mean squared error. As seen about the lambda was .0044845 or 5.407 in absolute value logarithmic form.

Using these variables and lambda we can compute our LASSO regression. Using this model and the test data we compare our predictions to our actual outcomes to find out error rate. The root squared mean error(RSME) was 15.46 for this model. Considering that the min/max of strikeouts is 0/372 this is an understandable error rate.

For a comparison of another regression technique we used our selected variables in an ordinary least squares(OLS). Using the same prediction procedure we calculated a 15.10375 RSME. Again a fair error rate for the model and very similar to our LASSO regression error.

While these regressions seem to be doing a decent job at estimating strikeouts we need to compare against some standard to understand that its a "better" method in some sense. The simplest way was to run OLS with all variables(post data clean). OLS is the simplest regression technique and the go to in most economists toolbox. Running the prediction and comparing to our test sample we had a RSME of 69.4515. Which is far worse than our past error rates. Now this begs the question of why OLS is so much worse without feature section. Simply put is that OLS without section control will over fit the model. Our in sample error rate was 14.92575 which is the best error rate of this group of models. In practice(test data) we see this does not hold as we over fit our model. Some variables had more influence than they should have creating increased error rates. The section method in our LASSO then OLS avoids this issue. In summary, these methods are "better" than our dry cut OLS.