Final Project

Benji Gold and Sam Alkalay

Introduction

Baseball, America's pastime, has a long and storied tradition that dates back well over 100 years. Since the 1850's, some form of statistics measuring how good a player is has been tracked. This began through the use of the box score, which tracked basic statistics, such as hits, runs, and errors, from which a player's batting average can be constructed. Over one hundred years later, a pioneering statistician by the name of Bill James introduced new statistical concepts, such as on-base percentage and runs created, in his annual Baseball Abstract (Lee 2018). As technology has improved, the statistics being tracked became more and more sophisticated. Then, in 2015 analytics in baseball took a giant leap. With the introduction of Statcast, teams were able to track novel metrics, such as a batter's exit velocity (the speed of the baseball as it comes off the bat, immediately after a batter makes contact) and barrel percentage (the percentage of baseballs hit off of the player's barrel) ("Statcast Search"). Around the league, teams adopted these new statistics to try and gain a competitive advantage, through which they would be able to better predict a player's potential. However, is this actually the case? While these new statistics are widely used, it is unclear whether they actually provide any useful information for predicting a player's potential. This research project intends to explore that idea through the use of a logistic regression model to predict whether a player is an all-star. The research question of interest is:

Do old or new wave statistics do a better job at predicting whether a player is selected as an all-star?

The response variables of interest are: All.Star: Whether a player is selected as an all-star. Salary: How much money a player makes.

For our analysis, we have selected two datasets. The first is from Baseball Reference, which consists of standard statistics that offer a broad view of a player's performance in a particular season. The second is from Statcast, which consists of each player's primary position. ADD MORE ABOUT WHAT WE DID WITH THE DATA HERE

Methodology

Results

Discussion

Packages and Data

```
library(tidyverse)
library(tidymodels)
library(glmnet)
library(caret)
library(MASS)
stats <- read.csv("data/stats.csv")


stats <- replace(stats, stats =="", NA)
stats <- stats %>%
  drop_na() %>%
  mutate(AVG300 = case_when(batting_avg >= .3 ~ "Greater than 300", TRUE ~ "Less than 300"
         HR40 = case_when(b_home_run >= 40 ~ "Greater than 40", TRUE ~ "Less than 40"))
view(stats)
```

Lassos for Variable Selection

```
# LASSO Variable Selection Basic Stats
y <- stats$All.Star
x <- model.matrix(All.Star ~ player_age + b_ab + b_total_pa + b_total_hits +
                    b_double + b_triple + b_home_run + b_strikeout + b_walk +
                    batting_avg + slg_percent + on_base_percent + Position, data = stats)
m_lasso_cv <- cv.glmnet(x, y, alpha = 1)
best_lambda <- m_lasso_cv$lambda.min
best_lambda
```

```
[1] 0.0005971366
```

```
m_best <- glmnet(x, y, alpha = 1, lambda = best_lambda)
m_best$beta
```

```
25 x 1 sparse Matrix of class "dgCMatrix"
                          s0
(Intercept)      .
player_age      -0.003176761
b_ab            -0.002263051
b_total_pa       .
b_total_hits     0.006359722
b_double         0.002503274
b_triple         0.004067213
b_home_run       0.013346863
b_strikeout     -0.000643750
b_walk           0.003362868
batting_avg      0.093129338
slg_percent      .
on_base_percent -0.338829822
Position2B       0.045439507
Position3B      -0.024930933
PositionC        0.065479851
PositionCF       0.061661930
PositionCH       0.018044570
PositionDH      -0.022177327
PositionDNP     -0.002194886
PositionLF      -0.048285422
PositionPH       0.001872678
PositionRF       0.004971593
PositionSP       0.152741692
PositionSS       0.045662118
```

```r
# LASSO Variable Selection Advanced Stats
y <- stats$All.Star
x <- model.matrix(All.Star ~ player_age + launch_angle_avg + sweet_spot_percent +
                    barrel + solidcontact_percent + flareburner_percent +
                    hard_hit_percent + avg_hyper_speed + z_swing_percent +
                    oz_swing_percent + meatball_swing_percent, data = stats)
m_lasso_cv <- cv.glmnet(x, y, alpha = 1)
best_lambda <- m_lasso_cv$lambda.min
best_lambda
```

```
[1] 0.005820234
```

```
m_best <- glmnet(x, y, alpha = 1, lambda = best_lambda)
m_best$beta
```

```
12 x 1 sparse Matrix of class "dgCMatrix"
                                  s0
(Intercept)                .
player_age               -0.0018096273
launch_angle_avg         -0.0001599061
sweet_spot_percent         .
barrel                    0.0080466625
solidcontact_percent     -0.0030825651
flareburner_percent      -0.0018259472
hard_hit_percent         -0.0018130937
avg_hyper_speed            .
z_swing_percent            .
oz_swing_percent           .
meatball_swing_percent -0.0021930005
```

Regressions

```
#Basic model
m1 <- glm(All.Star ~ player_age + b_ab +  b_total_hits +
                     b_double + b_triple + HR40 + b_strikeout +
                     b_bb_percent + AVG300 + slg_percent +
                     on_base_percent + Position,
  data = stats,
  family = "binomial"
)
tidy(m1)
```

```
# A tibble: 24 x 5
   term                  estimate std.error statistic p.value
   <chr>                    <dbl>     <dbl>     <dbl>   <dbl>
 1 (Intercept)           -4.05      2.66       -1.52   0.128
 2 player_age            -0.0544    0.0563     -0.967  0.333
 3 b_ab                  -0.0149    0.00880    -1.70   0.0899
 4 b_total_hits           0.0752    0.0292      2.58   0.00996
 5 b_double               0.00356   0.0382      0.0932 0.926
 6 b_triple              -0.117     0.108      -1.08   0.279
 7 HR40Less than 40       0.0579    0.940       0.0616 0.951
 8 b_strikeout          0.00288   0.00881    0.327  0.744
 9 b_bb_percent         0.250     0.0812     3.08   0.00210
10 AVG300Less than 300 -0.371     0.749     -0.496  0.620
# ... with 14 more rows
```

```
m1_aug <- augment(m1) %>%
  mutate(prob = exp(.fitted)/(1 + exp(.fitted)),
         pred_leg = ifelse(prob > 0.5, "All-Star", "Not All-Star"))
table(m1_aug$pred_leg, m1_aug$All.Star)
```

```
                 0   1
  All-Star       7  20
  Not All-Star 425  34
```

```
#Advanced model
m2 <- glm(All.Star ~ player_age + launch_angle_avg +
                    barrel + solidcontact_percent + flareburner_percent +
                    hard_hit_percent + meatball_swing_percent,
  data = stats,
  family = "binomial"
)
tidy(m2)
```

```
# A tibble: 8 x 5
  term                   estimate std.error statistic  p.value
  <chr>                     <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)             1.38       1.76       0.785 4.32e- 1
2 player_age             -0.0361     0.0468    -0.772 4.40e- 1
3 launch_angle_avg       -0.00881    0.0283    -0.312 7.55e- 1
4 barrel                  0.0852     0.0136     6.27  3.56e-10
5 solidcontact_percent   -0.0805     0.0943    -0.854 3.93e- 1
6 flareburner_percent    -0.0129     0.0379    -0.341 7.33e- 1
7 hard_hit_percent       -0.0256     0.0263    -0.974 3.30e- 1
8 meatball_swing_percent -0.0358     0.0162    -2.21  2.72e- 2
```

```
m2_aug <- augment(m2) %>%
  mutate(prob = exp(.fitted)/(1 + exp(.fitted)),
         pred_leg = ifelse(prob > 0.5, "All-Star", "Not All-Star"))
```

```r
table(m2_aug$pred_leg, m2_aug$All.Star)
```

```
               0   1
All-Star       6  12
Not All-Star 426  42
```