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

```r
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

```r
# 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] 0.000866343
```

```r
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      -0.0032526019
b_ab            -0.0020846949
b_total_pa       .
b_total_hits     0.0058564190
b_double         0.0024827743
b_triple         0.0042533649
b_home_run       0.0134683743
b_strikeout     -0.0007402728
b_walk           0.0030997147
batting_avg      0.0291414236
slg_percent      .
on_base_percent -0.2377737907
Position2B       0.0404320332
Position3B      -0.0269939426
PositionC        0.0604933165
PositionCF       0.0570205562
PositionCH       0.0026634929
PositionDH      -0.0214856377
PositionDNP     -0.0021127547
PositionLF      -0.0499863104
PositionPH       .
PositionRF       0.0015718234
PositionSP       0.1569408560
PositionSS       0.0418296629
```

```
# 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
```

```r
# obp percentage lasso
y <- stats$on_base_percent
x <- model.matrix(on_base_percent ~ launch_angle_avg + sweet_spot_percent +
                    barrel + solidcontact_percent + flareburner_percent +
                    hard_hit_percent + avg_hyper_speed + z_swing_percent +
                    oz_swing_percent + meatball_swing_percent, data = stats)
m_lasso_cv <- cv.glmnet(x, y, alpha = 1)
best_lambda <- m_lasso_cv$lambda.min
best_lambda
```

```
[1] 6.588831e-05
```

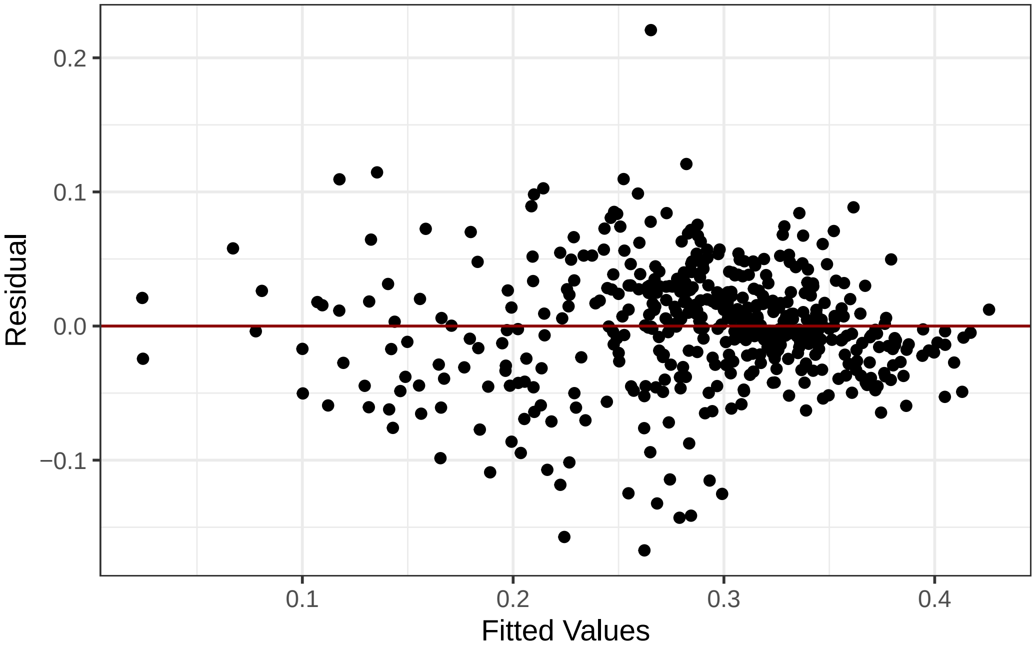
```r
m_best <- glmnet(x, y, alpha = 1, lambda = best_lambda)
m_best$beta
```

```
11 x 1 sparse Matrix of class "dgCMatrix"
                                  s0
(Intercept)             .           
launch_angle_avg       -0.0004177699
sweet_spot_percent      0.0020967297
barrel                  0.0013805557
solidcontact_percent    0.0022122425
flareburner_percent     0.0036068089
hard_hit_percent        0.0017835578
avg_hyper_speed        -0.0062577021
z_swing_percent         0.0006378467
oz_swing_percent       -0.0029185729
meatball_swing_percent  0.0007305436
```

```r
# obp percentage prediction
m3 <- lm(on_base_percent ~ sweet_spot_percent +
                    barrel + solidcontact_percent + flareburner_percent +
                    hard_hit_percent + z_swing_percent +
                    oz_swing_percent + meatball_swing_percent,
  data = stats)

summary(m3)
```

```

Call:
lm(formula = on_base_percent ~ sweet_spot_percent + barrel + 
    solidcontact_percent + flareburner_percent + hard_hit_percent + 
    z_swing_percent + oz_swing_percent + meatball_swing_percent, 
    data = stats)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.167277 -0.023263 -0.000016  0.026078  0.220647 

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)    
(Intercept)             0.0700061  0.0194563   3.598 0.000354 ***
sweet_spot_percent      0.0019772  0.0004004   4.938 1.09e-06 ***
barrel                  0.0013301  0.0001686   7.891 2.07e-14 ***
solidcontact_percent    0.0022739  0.0008601   2.644 0.008470 ** 
flareburner_percent     0.0038182  0.0004997   7.641 1.19e-13 ***
hard_hit_percent        0.0008075  0.0002987   2.703 0.007115 ** 
z_swing_percent         0.0006259  0.0004381   1.428 0.153809    
oz_swing_percent       -0.0029453  0.0003171  -9.288  < 2e-16 ***
meatball_swing_percent  0.0007211  0.0002533   2.847 0.004610 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.04441 on 477 degrees of freedom
Multiple R-squared:  0.6983,	Adjusted R-squared:  0.6933 
F-statistic:   138 on 8 and 477 DF,  p-value: < 2.2e-16
```

```
m3_aug <- augment(m3)
m3_aug |>
ggplot(aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0, color = "darkred") +
  labs(x = "Fitted Values",
       y = "Residual") +
  theme_bw()
```



```
# slugging percentage lasso
y <- stats$slg_percent
x <- model.matrix(slg_percent ~ launch_angle_avg + sweet_spot_percent +
                    barrel + solidcontact_percent + flareburner_percent +
                    hard_hit_percent + avg_hyper_speed + z_swing_percent +
                    oz_swing_percent + meatball_swing_percent, data = stats)
m_lasso_cv <- cv.glmnet(x, y, alpha = 1)
best_lambda <- m_lasso_cv$lambda.min
best_lambda
```

```
[1] 0.00320379
```

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

```
11 x 1 sparse Matrix of class "dgCMatrix"
```

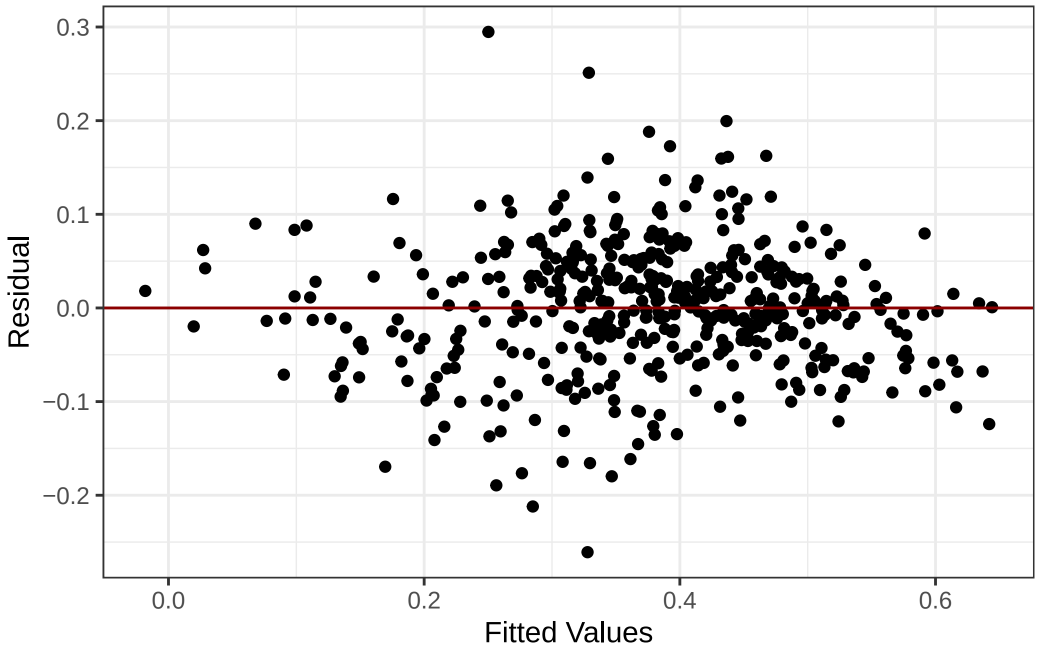
```
                                s0
(Intercept)             .
launch_angle_avg        0.0004311878
sweet_spot_percent      0.0040637028
barrel                  0.0034700241
solidcontact_percent    0.0013063252
flareburner_percent     0.0023459342
hard_hit_percent        0.0019696927
avg_hyper_speed         0.0021344093
z_swing_percent         0.0008773026
oz_swing_percent       -0.0002807259
meatball_swing_percent  0.0006261759
```

```
# slugging percentage prediction
m4 <- lm(slg_percent ~ 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)

summary(m4)
```

```
Call:
lm(formula = slg_percent ~ 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)

Residuals:
      Min        1Q    Median        3Q       Max
-0.260804 -0.041484  0.002206  0.040864  0.294753

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)            -0.0807292  0.0316715  -2.549   0.0111 *
launch_angle_avg        0.0005148  0.0005433   0.948   0.3438
sweet_spot_percent      0.0038654  0.0006637   5.824 1.06e-08 ***
barrel                  0.0035645  0.0002677  13.318  < 2e-16 ***
solidcontact_percent    0.0022226  0.0013590   1.635   0.1026
flareburner_percent     0.0031461  0.0007964   3.950 9.00e-05 ***
```

```
hard_hit_percent        0.0011213  0.0010244   1.095   0.2742
avg_hyper_speed         0.0060747  0.0058969   1.030   0.3035
z_swing_percent         0.0013843  0.0006861   2.018   0.0442 *
oz_swing_percent       -0.0008703  0.0005029  -1.731   0.0842 .
meatball_swing_percent  0.0006949  0.0003958   1.756   0.0798 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.06915 on 475 degrees of freedom
Multiple R-squared:  0.7432,    Adjusted R-squared:  0.7378
F-statistic: 137.5 on 10 and 475 DF,  p-value: < 2.2e-16
```

```r
m4_aug <- augment(m4)
m4_aug |>
ggplot(aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0, color = "darkred") +
  labs(x = "Fitted Values",
       y = "Residual") +
  theme_bw()
```


Residual
Fitted Values