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.

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. The final data file we have was compiled from baseballsavant.com with a mix of more traditional stats and statcast stats. This complete file can be found in the stats.csv file. Once that was done, we entered a players position, salary, and team from baseball prospectus. we used wikipedia to find rosters for the 2019 all-star game and created a categorical variable column with that information. ## Methodology

Results

Discussion

Packages and Data

```
library(tidyverse)
library(tidymodels)
library(glmnet)
library(caret)
library(MASS)
library(lme4)
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"), pi
view(stats)
```

Lassos for Variable Selection

```
# LASSO Variable Selection Basic Stats
y <- stats$All.Star
x <- model.matrix(All.Star ~ player_age + b_ab + b_total_pa + b_total_hits + b_home_run +
                    b_double + b_triple + b_home_run * HR40 + b_strikeout + b_walk +
                    batting_avg + slg_percent + on_base_percent + Position, data = stats)
m_lasso_cv <- cv.glmnet(x, y, alpha = 1)
best_lambda <- m_lasso_cv$lambda.min
best_lambda
```

```
[1] 0.0007192532
```

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

```
29 x 1 sparse Matrix of class "dgCMatrix"
                                        s0
(Intercept)                      .
player_age                      -2.999163e-03
b_ab                            -1.526707e-03
b_total_pa                       .
b_total_hits                     4.002210e-03
b_home_run                       1.204542e-02
AVG300Less than 300             -1.277885e-01
batting_avg                      .
b_double                         2.582686e-03
b_triple                         4.503069e-03
HR40Less than 40                -1.005933e-01
b_strikeout                     -7.937712e-04
b_walk                           3.198065e-03
slg_percent                      5.306527e-02
on_base_percent                 -2.675252e-01
Position2B                       3.664957e-02
Position3B                      -3.611155e-02
PositionC                        6.239333e-02
PositionCF                       5.118330e-02
PositionCH                       3.848824e-05
PositionDH                      -3.906953e-02
PositionDNP                     -7.096175e-03
PositionLF                      -5.230209e-02
PositionPH                       9.388429e-04
PositionRF                       .
PositionSP                       1.719706e-01
PositionSS                       4.371010e-02
AVG300Less than 300:batting_avg  .
b_home_run:HR40Less than 40      .
```

```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] 0.00483206
```

```
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              -0.0020095320
launch_angle_avg        -0.0002126235
sweet_spot_percent         .
barrel                   0.0082201264
solidcontact_percent    -0.0032632615
flareburner_percent     -0.0019236121
hard_hit_percent        -0.0019900029
avg_hyper_speed            .
z_swing_percent            .
oz_swing_percent           .
meatball_swing_percent -0.0022706870
```

Regressions

```
#Basic model
m1 <- glm(All.Star ~ player_age + b_ab +  b_total_hits +
                    b_double + b_triple + b_home_run + 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)         -5.96      2.78       -2.15   0.0319
 2 player_age          -0.0369    0.0570     -0.647  0.518
 3 b_ab                -0.00925   0.00925    -1.00   0.317
 4 b_total_hits         0.0450    0.0300      1.50   0.133
```

```
 5 b_double             0.0182    0.0411      0.443  0.657
 6 b_triple            -0.0469    0.116      -0.404  0.686
 7 b_home_run           0.0719    0.0524      1.37   0.170
 8 b_strikeout         -0.00188   0.00930    -0.203  0.839
 9 b_bb_percent         0.155     0.105       1.48   0.140
10 AVG300Less than 300 -0.617     0.768      -0.804  0.421
# ... with 14 more rows
```

```r
m1_aug <- augment(m1) %>%
  mutate(prob = exp(.fitted)/(1 + exp(.fitted)),
         pred_leg = ifelse(prob > 0.32, "All-Star", "Not All-Star"))
table(m1_aug$pred_leg, m1_aug$All.Star)
```

```
               0   1
All-Star      22  30
Not All-Star 410  24
```

```r
#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
```

```r
m2_aug <- augment(m2) %>%
  mutate(prob = exp(.fitted)/(1 + exp(.fitted)),
         pred_leg = ifelse(prob > 0.32, "All-Star", "Not All-Star"))
table(m2_aug$pred_leg, m2_aug$All.Star)
```

```
                0   1
All-Star       22  20
Not All-Star 410  34
```

```r
# Remove Nationals from Data
rol_stats <- stats |>
  filter(Team != "WAS")
```

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

```
[1] 0.00066814
```

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

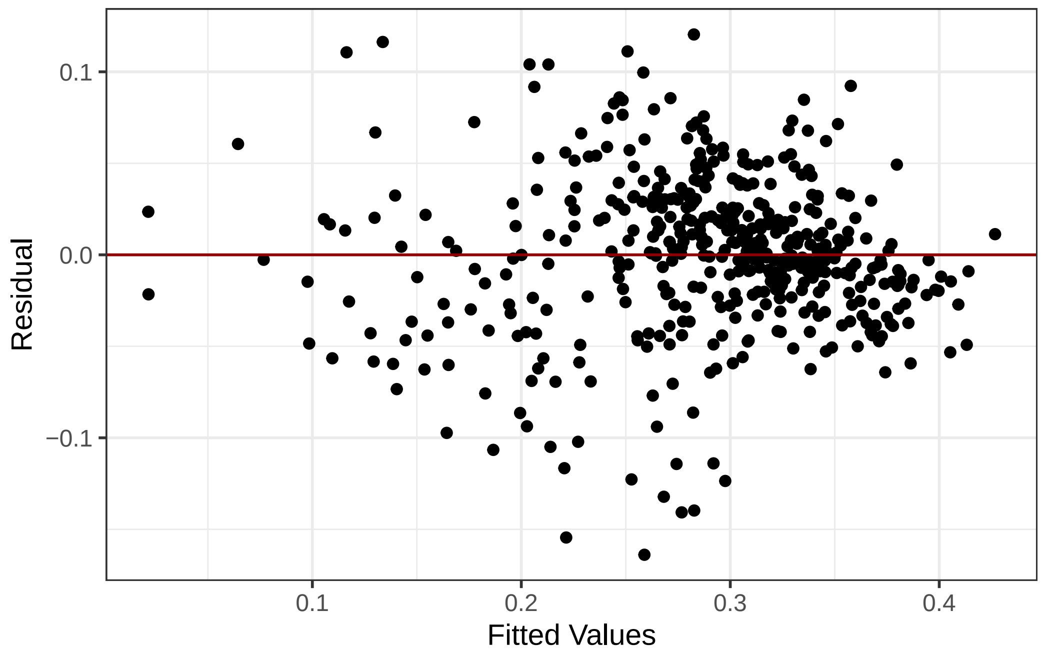
```
11 x 1 sparse Matrix of class "dgCMatrix"
                                  s0
(Intercept)                .
launch_angle_avg          -0.0001757362
sweet_spot_percent         0.0021098426
barrel                     0.0013163299
solidcontact_percent       0.0019774564
flareburner_percent        0.0036596286
hard_hit_percent           0.0009201660
avg_hyper_speed            .
```

```
z_swing_percent         0.0005104550
oz_swing_percent       -0.0028668542
meatball_swing_percent  0.0007049818
```

```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 = rol_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 = rol_stats)

Residuals:
      Min        1Q    Median        3Q       Max
-0.163904 -0.022970  0.000785  0.025770  0.120400

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            0.0706000  0.0195729   3.607 0.000344 ***
sweet_spot_percent     0.0019988  0.0004026   4.965 9.74e-07 ***
barrel                 0.0013448  0.0001692   7.946 1.51e-14 ***
solidcontact_percent   0.0020952  0.0008502   2.464 0.014091 *
flareburner_percent    0.0038121  0.0004983   7.650 1.20e-13 ***
hard_hit_percent       0.0008692  0.0003016   2.882 0.004141 **
z_swing_percent        0.0005975  0.0004419   1.352 0.177044
oz_swing_percent      -0.0029670  0.0003208  -9.248  < 2e-16 ***
meatball_swing_percent  0.0007096  0.0002542   2.791 0.005470 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.04356 on 457 degrees of freedom
Multiple R-squared:  0.7042,    Adjusted R-squared:  0.699
F-statistic:   136 on 8 and 457 DF,  p-value: < 2.2e-16
```

```r
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()
```




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

```
[1] 0.006083572
```

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

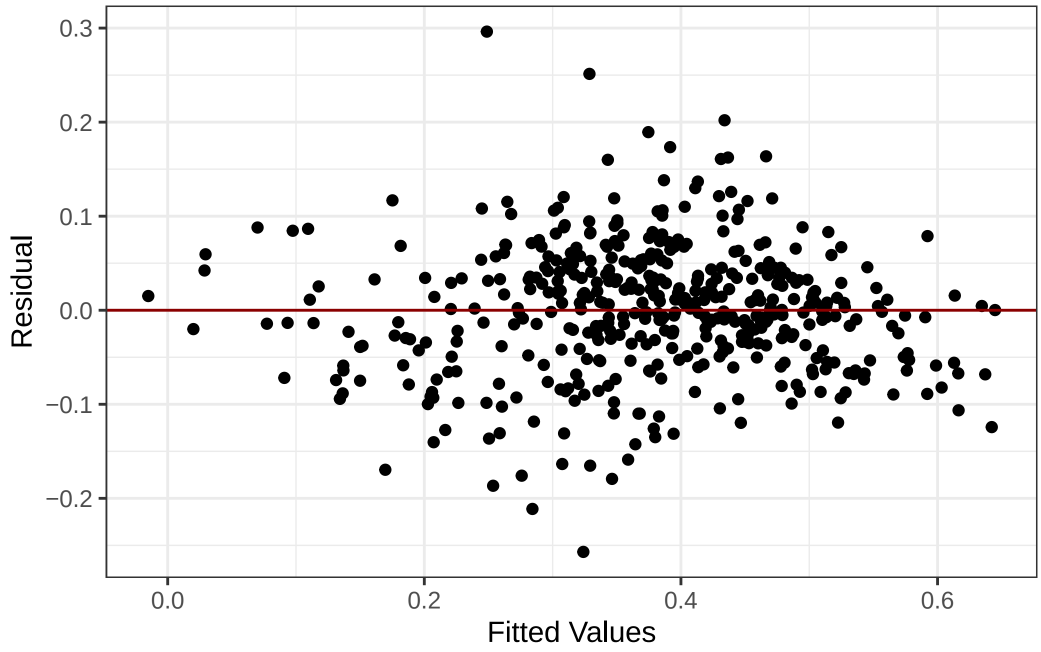
```
11 x 1 sparse Matrix of class "dgCMatrix"
                                s0
(Intercept)           .
launch_angle_avg      2.400294e-04
sweet_spot_percent    4.283953e-03
barrel                3.391195e-03
solidcontact_percent  4.099652e-04
flareburner_percent   1.641752e-03
hard_hit_percent      2.438334e-03
avg_hyper_speed       4.788239e-06
z_swing_percent       5.378204e-04
oz_swing_percent      .
meatball_swing_percent 5.341360e-04
```

```r
# 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 = rol_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 = rol_stats)

Residuals:
      Min        1Q    Median        3Q       Max
-0.256940 -0.041724  0.001751  0.040773  0.296249

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)            -0.0790676  0.0327432  -2.415  0.01614 *
```

```
launch_angle_avg         0.0004505  0.0005610   0.803  0.42241    
sweet_spot_percent       0.0038602  0.0006829   5.653 2.79e-08 ***
barrel                   0.0035814  0.0002773  12.916  < 2e-16 ***
solidcontact_percent     0.0021829  0.0013866   1.574  0.11612    
flareburner_percent      0.0031794  0.0008195   3.880  0.00012 ***
hard_hit_percent         0.0010022  0.0010543   0.951  0.34233    
avg_hyper_speed          0.0067277  0.0060541   1.111  0.26704    
z_swing_percent          0.0013267  0.0007133   1.860  0.06351 .  
oz_swing_percent        -0.0008821  0.0005243  -1.683  0.09314 .  
meatball_swing_percent   0.0007281  0.0004096   1.777  0.07618 .  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.06994 on 455 degrees of freedom
Multiple R-squared:  0.7345,	Adjusted R-squared:  0.7287 
F-statistic: 125.9 on 10 and 455 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()
```




```
# Subset for nationals
nationals_stats <- stats |>
  filter(Team == "WAS")

# Predict
pred_obp <- predict(m3, nationals_stats)
pred_slg <- predict(m4, nationals_stats)

# Add to DF
nationals_stats <- nationals_stats |>
  mutate(Predicted_OBP = pred_obp,
         Predicted_SLG = pred_slg)

# Display
nationals_pred_stats <- nationals_stats[ , c("last_name", "first_name", "Position",
                                             "Predicted_OBP", "Predicted_SLG")]
```

ARTICLE ABOUT BATTING ORDER STRATEGY: https://www.sportsbettingdime.com/guides/strategy/ba
order-sabermetrics/