Technical Report

Group 6

Abstract

The competitive balance tax (implemented in 2002) is a rule implementing salary caps for baseball players in a given team. This paper attempts to uncover if there is any underlying difference in the number of lost potential players (called "couldabeens") in the pre-rule (1968-2002) and post-rule eras (2002-2018).

Introduction

Major League Baseball's "competitive balance tax," first implemented in 2002, has become a favorite subject of outrage among players and fans alike in recent years. On paper, the policy was meant to make the sport more competitive and boost salaries for players on lower-revenue teams. The owners and the players' union would negotiate the largest reasonable amount a team could spend on its roster, and any team that wanted to spend beyond that cap would pay a "tax" (a share of their excess payroll) to be redistributed to the poorer teams. In practice the policy has effectively become a salary cap, particularly after the 2016 renegotiations. In 2018 only two MLB teams out of thirty went over the salary threshold, and only one of those by any significant margin (the team that did so, the Boston Red Sox, unsurprisingly went on to win the World Series). With leaguewide revenue growth far outpacing growth in the revenue cap and most younger players effectively locked out of salary negotiations by free agency rules, players have understandably chafed at the limitation on their salaries, with mutterings of a player strike unless the rule is altered.

But while the depressive effect on players' salaries is not in dispute, the question of whether the rule *hurts the game* (a far graver sin among baseball fans than merely conspiring to lower salaries) is more open. At least in the conventional wisdom, the tax encourages teams to cultivate a massive pool of recruits and then pay the best of them the minimum permitted salary for up to seven years before they age into free agency. Once these players enter free agency, the narrative goes, the team dumps them for younger players whom they can pay the minimum salary. The remainder of their salary cap goes towards retaining a few older superstars with the name recognition to bring in fans, with many good-but-not-Mike-Trout players pushed into early retirement. The implication, then, is that the salary policy "hurts the game" because the MLB is less likely to be putting the best 30 shortstops in the world on the field at any given time. I would like to test this anecdotal observation empirically.

So, the working hypothesis states that since the advent of the competitive balance tax, the number of better players forced into retirement annually will have increased. A "better player forced into retirement" will be defined as a player under the age of 32 who a) left baseball due to their contract not being renewed, rather than injury or personal circumstances, and b) was outperforming the median rookie in their position at the time of their retirement. In other words, we might expect to find the number of couldabeens to increase after the implementation of the rule.

Methods

Our data comes from https://stathead.com/baseball/ and we mainly have four sets of data.

1. Rookie pitchers
2. Rookie position players
3. Retired pitchers
4. Retired position players

The research question in this paper hinges on classifying retired players of lost potential. To motivate this, we will first define our classifier for a "couldabeen".

For a given year Y, we first compute the median rookie's WAR, call it m_Y. Additionally, compute the standard deviation of that data and call it σ_Y. Then, given a threshold $t \in \mathbb{R}$, we construct the corrosponding classifier for "couldabeen" status C of a given retired player p (from the year Y) to be as follows:

$$C(p) = \begin{cases} True, \text{ WAR}_p \geq m_Y + t\sigma_Y \\ False, \text{ WAR}_p < m_Y + t\sigma_Y \end{cases}$$

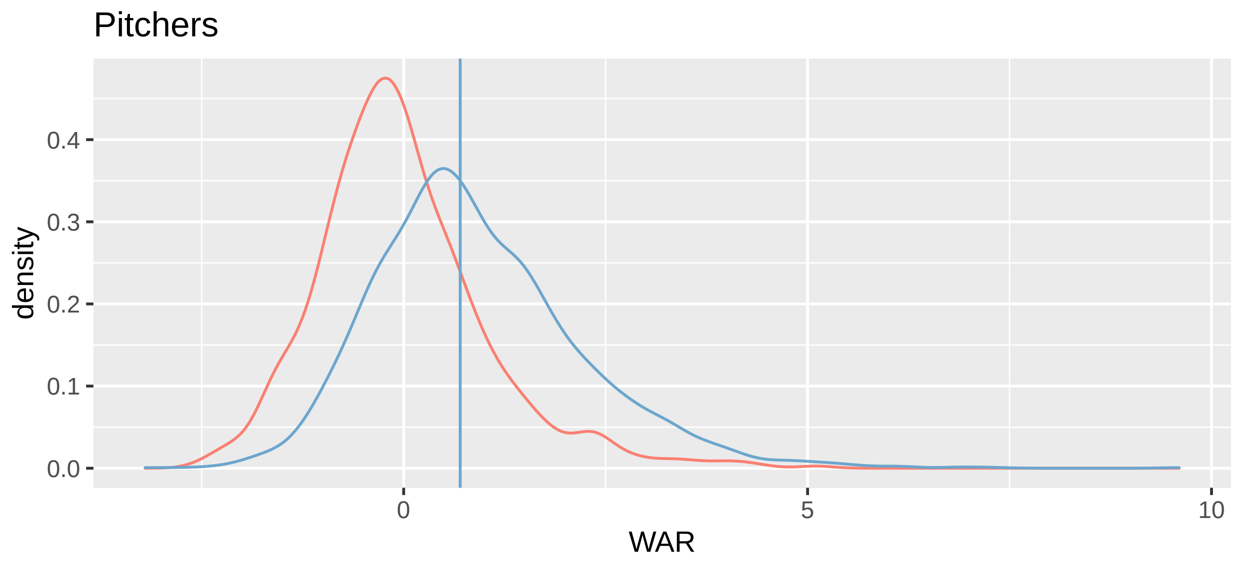
Once every retired player is classified appopriately, we run two primary models:

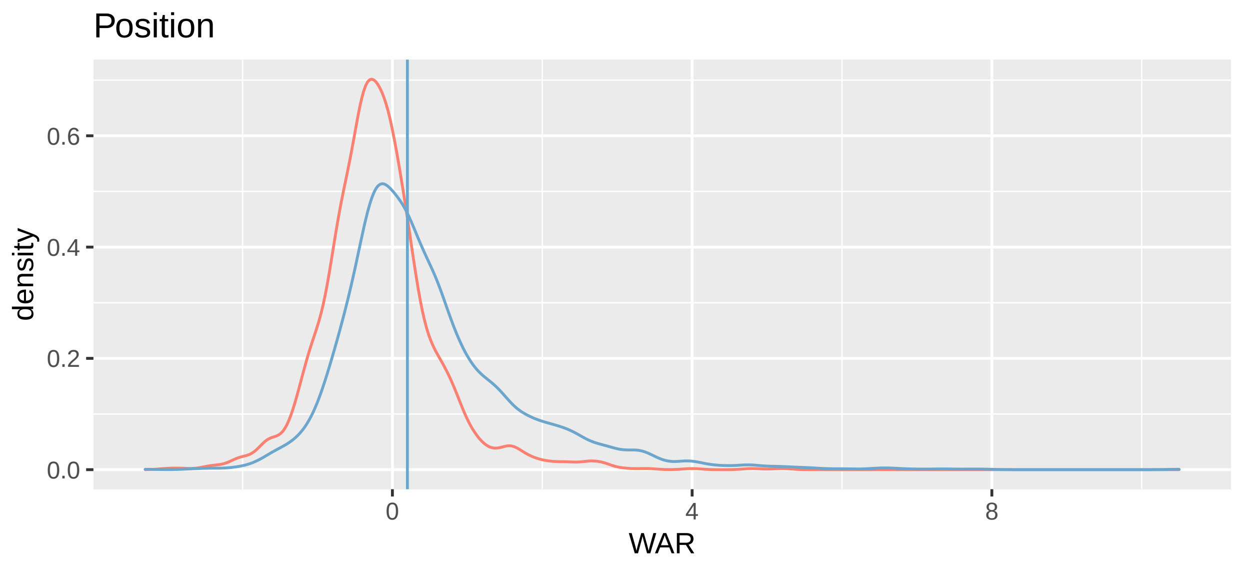
(i) Logisitic Model: On the retired players dataset, each player has a new column, `exceeds_threshold` which is the result of the classifier $C(p)$. This classifier is a boolean, so we run two models on the pre-rule era and the post-rule era and see the effects `Year` has on `exceeds_threshold`. Because there will always be "couldabeens", we do not expect a large effect size and hence a very significant result, however, the **sign** of our coefficient will be essential for our inference. If our research hypothesis is correct (that there is an effect), we expect to see a positive coefficient for β_{Year}.

(ii) Linear Model: After aggregating all the retired players and their classifications, we summarize the data by year to obtain the proportion of retired players that were couldabeens that year, called `prop`. As such, we now have 50 data points (for each year), and a response variable being the proportion of couldabeens. As such, we run a linear model fitting `Year ~ prop` for the pre-rule era and the post-rule era. Because there will always be "couldabeens", we do not expect a large effect size and hence a very significant result, however, the **sign** of our coefficient will be essential for our inference. If our research hypothesis is correct (that there is an effect), we expect to see a positive coefficient for β_{Year}.

Exploratory Data Analysis

Below, we can visualize the densities of the WAR statistic for each type of player. In blue, we have the rookie players (with the median line denoted), and in red, we have the corrosponding retired players. As such, we can visualize the "couldabeens" as the retired players to the left of the median line (if the threshold $t = 0$), and we corrospondingly count the number of couldabeens based on year, from which we make our inference on the impact of Year.

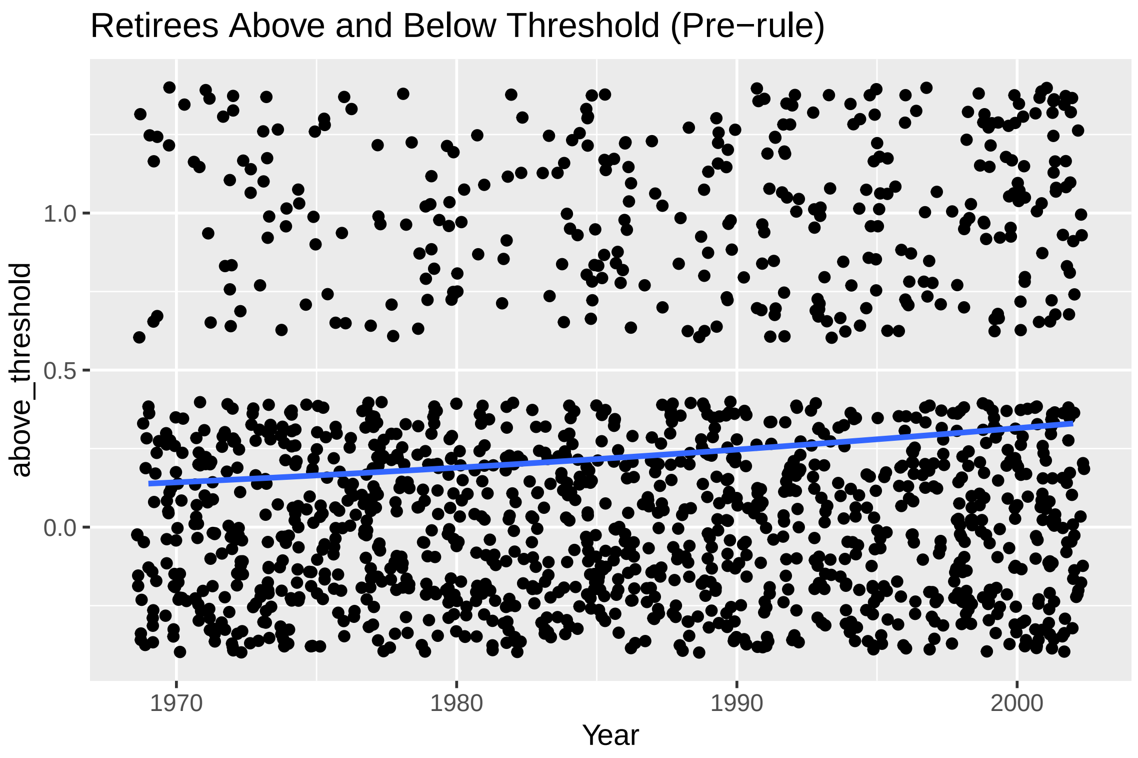




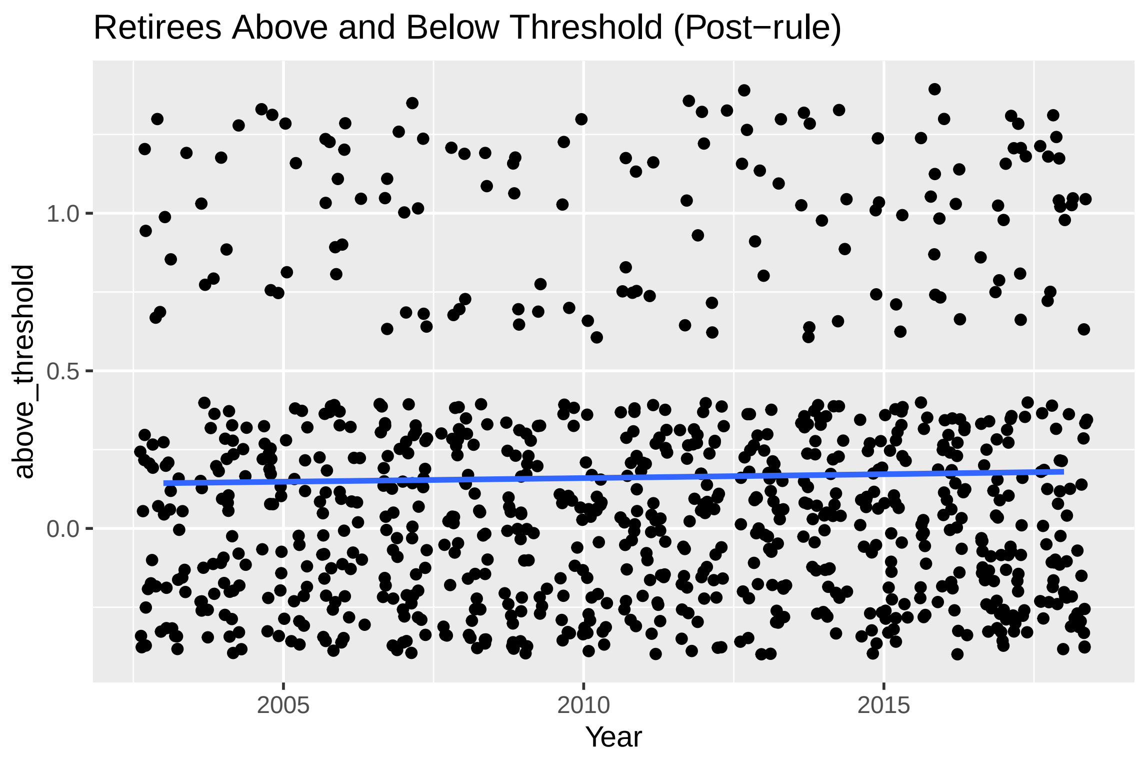
Results

Logistic Models

```r
# Partition retirees into pre-rule and post-rule era
retirees_pre <- prerule(retirees)
retirees_post <- postrule(retirees)
```



```
##
## Call:
## glm(formula = above_threshold ~ Year, family = "binomial", data = dataset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8949  -0.7642  -0.6479  -0.5547   1.9883
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -68.617412  12.853177  -5.339 9.37e-08 ***
## Year          0.033921   0.006465   5.247 1.55e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1583.3  on 1470  degrees of freedom
## Residual deviance: 1554.9  on 1469  degrees of freedom
## AIC: 1558.9
##
## Number of Fisher Scoring iterations: 4
```



```
##
## Call:
## glm(formula = above_threshold ~ Year, family = "binomial", data = dataset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6297  -0.6096  -0.5900  -0.5617   1.9697
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -37.46198   40.21840  -0.931    0.352
## Year          0.01781    0.02000   0.891    0.373
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 781.43  on 880  degrees of freedom
## Residual deviance: 780.63  on 879  degrees of freedom
## AIC: 784.63
##
## Number of Fisher Scoring iterations: 4
```

Linear Models

```
# Partition dataset into years before and after rule
couldabeens_pre <- prerule(couldabeens)
couldabeens_post <- postrule(couldabeens)
```

Couldabeens: Pre-rule Era (1969-2002)

```
# Obtain linear model for pre-rule years
model_pre <- lm(formula = prop ~ I(Year), data = couldabeens_pre)
coefs_pre <- model_pre$coefficients
```

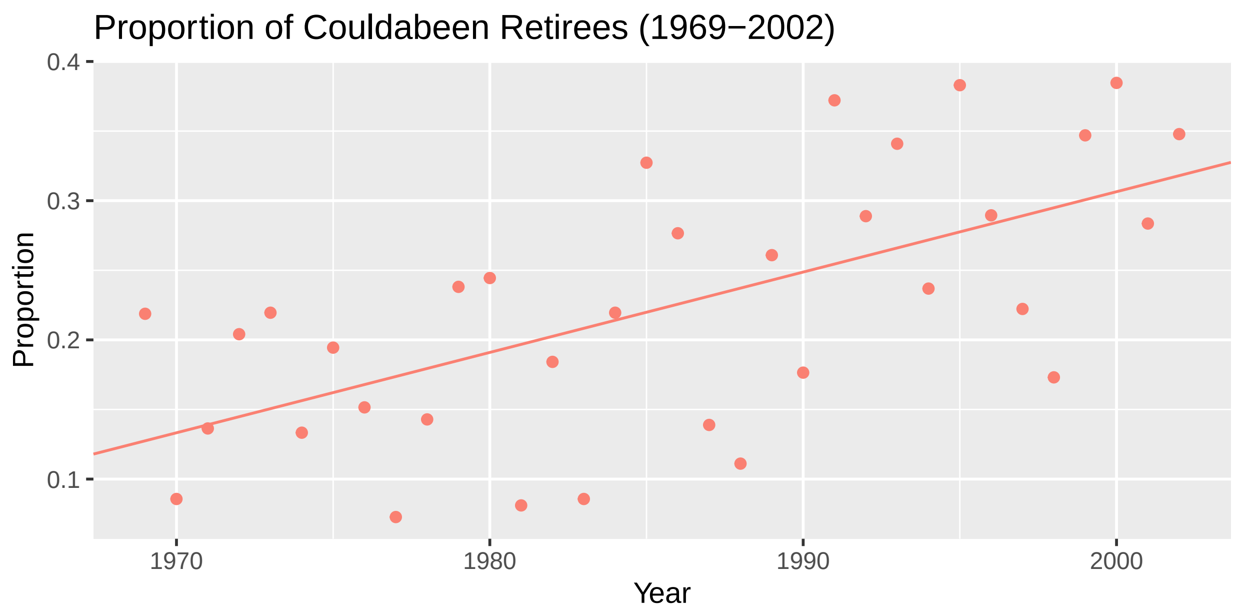


Figure 1: Proportion of Retirees who were Coulabeens prior to the implementation of the Luxury Tax

```
##
## Call:
## lm(formula = prop ~ I(Year), data = couldabeens_pre)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.126056 -0.044806  0.005781  0.053314  0.117608
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -11.238735   2.549750  -4.408  0.00011 ***
## I(Year)       0.005773   0.001284   4.495 8.56e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07346 on 32 degrees of freedom
## Multiple R-squared:  0.3871, Adjusted R-squared:  0.3679
## F-statistic: 20.21 on 1 and 32 DF,  p-value: 8.56e-05
```

Couldabeens: Post-rule Era (2003-2018)

```r
# Obtain linear model for post-rule years
model_post <- lm(formula = prop ~ I(Year), data = couldabeens_post)
coefs_post <- model_post$coefficients
```

```
##
## Call:
## lm(formula = prop ~ I(Year), data = couldabeens_post)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.049689 -0.024453  0.001825  0.018410  0.049237
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.987717   3.481582  -1.145    0.271
## I(Year)      0.002064   0.001732   1.192    0.253
##
## Residual standard error: 0.03193 on 14 degrees of freedom
## Multiple R-squared:  0.09209,    Adjusted R-squared:  0.02724
## F-statistic:  1.42 on 1 and 14 DF,  p-value: 0.2532
```

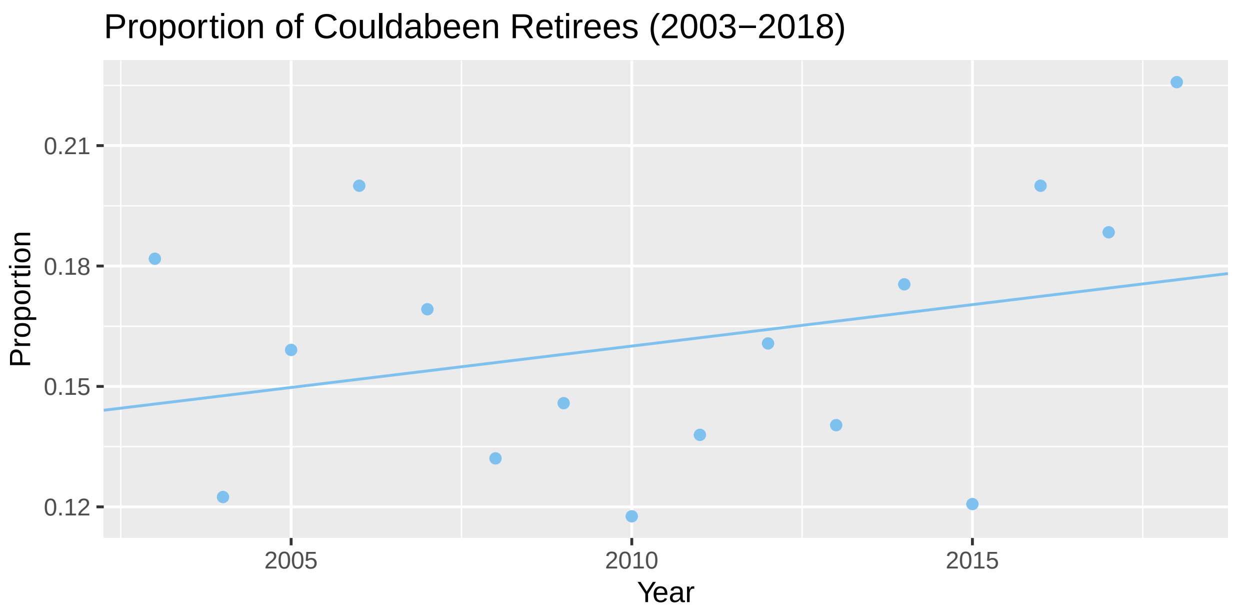


Figure 2: Proportion of Retirees who were Coulabeens after the implementation of the Luxury Tax

Simpson's Paradox

Interestingly, when we choose not to partition the dataset into the post-rule and pre-rule eras and fit a linear model `Year ~ prop`, we find that a very resounding instance of Simpson's Paradox. In both partitions, we find that the parameter $\beta_{Year} > 0$. However, if we do not make the partition, we find that $\beta_{Year} \approx 0$. This raises some questions regarding the role our partition plays in our inference and modeling choices.

Firstly, this begs the question: should we partition the data? For the sake of valid statistical inference, if we assume `Year` is a valid predictor, it initially seems as though it would be errenous practice to do so. If we partition the data, we essentially say that our inferential result is **conditioned** on only that era's data (since our parameter is fit to only that data). As such, (assuming no confounding variable) it would be better statistical practice to not partition the data and only make inference on a β_{Year} parameter trained on the entire dataset.

However, as we can see, if we do not partition the data, our models suffer from Simpson's paradox and there seemingly is no trend along the years. As it suggests, there is indeed a possible explanation for why this partition is necessary. One confounding variable that the model failed to include (we did not find and import the data yet) is the salary data (possibly adjusted for inflation) for the given year itself

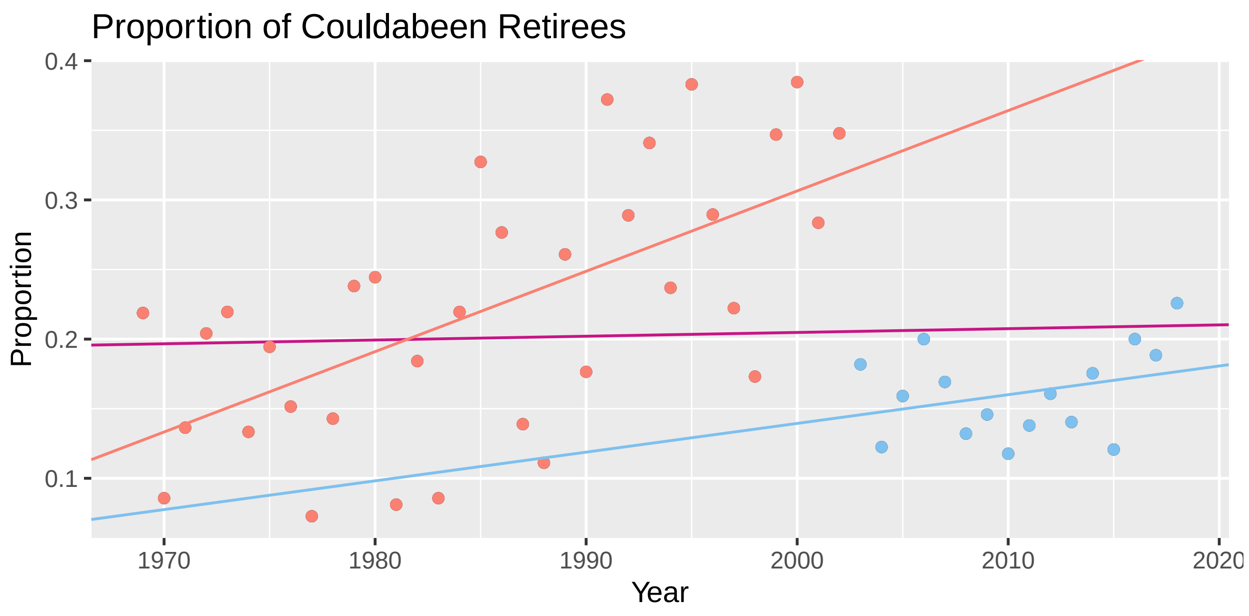


Figure 3: Proportion of Retirees who were Coulabeens

Discussion

Overall Results and Considerations

Threshold Stability

Recall, that our couldabeen classifier was constructed as follows:

For a given year Y, we first compute the median rookie's WAR, call it m_Y. Additionally, compute the standard deviation of that data and call it σ_Y. Then, given a threshold $t \in \mathbb{R}$, we construct the corrosponding classifier for "couldabeen" status C of a given retired player p (from the year Y) to be as follows:

$$C(p) = \begin{cases} True, \text{ WAR}_p \geq m_Y + t\sigma_Y \\ False, \text{ WAR}_p < m_Y + t\sigma_Y \end{cases}$$

All in all, we can see there are quite a few problems with our model. Although we have obtained a positive result on the slope of β_{Year} for a particular threshold in the post-rule era, varying the threshold tells another story. In particular, if we run a linear model on the proportion of couldabeens against year i.e. `Year ~ prop` against a varying threshold (in standard deviations of rookie WAR), we found that β_{Year} is quite unstable. Consider the graph below:

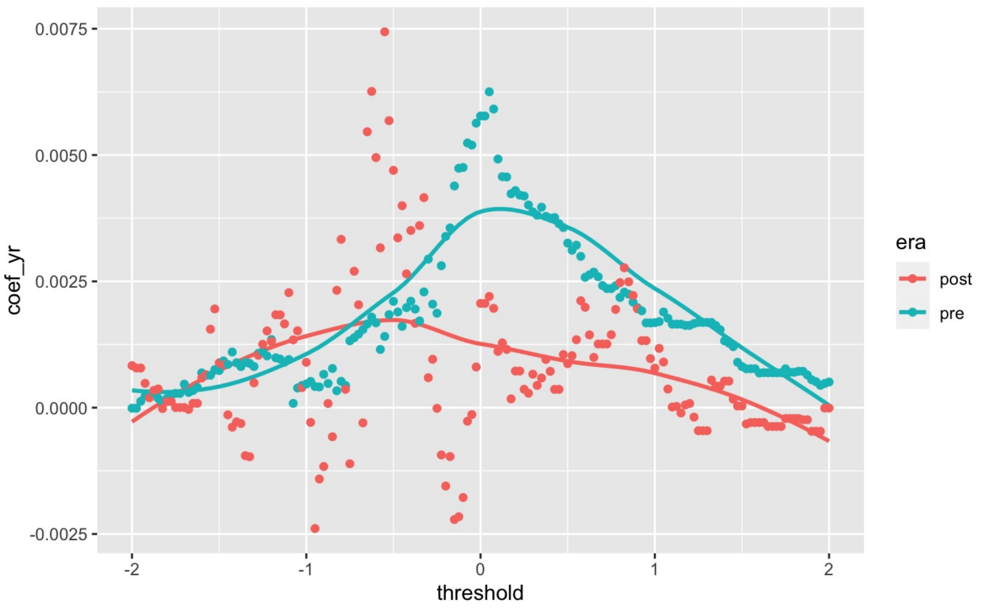


Figure 4: Performance of the Linear Model against varying threshold for couldabeen status

One reason this instability may arise could be due to the fact that our thresholds for every year were calculated as a function of *only* that year's data. As such, the sample from which we obtain the threshold by each level (corrosponding to each year) is very small compared to the whole dataset; as such, small changes to the threshold cause high levels of noise in our final model.

To treat this problem, we may reconsider our method of computing the threshold. Instead of strictly using just the data for a given year, it may be beneficial to "smooth" the threshold out by taking the data of that year and adjacent years (like a window). For instance, instead of considering just 2018 data, we may take 2017-2019 data for a "window length of 1". Hopefully, once this adjustment is made, our results will be more stable and less noisy with respect to the threshold, which should be just an arbitrary value rather than a value that carries so much weight in the entire inferential process.

Furthermore, the addition of some supplementary salary data may be useful as another predictor since `Year` alone does not seem to have good explanatory power.

Code Appendix

Importing the Datasets

```
# Load rookies datasets
df_pit_rkes <- read_csv("data/rookie-pitcher.csv")
df_pos_rkes <- read_csv("data/rookie-position.csv")
# Load retirees datasets
df_pit_ret <- read_csv("data/retirees-pitcher.csv")
df_pos_ret <- read_csv("data/retirees-position.csv")
```

Wrangling the Datasets

```
# select appopriate columns from player datasets
wrangle_init <- function(dataset){
  colnames(dataset)[3] <- "WAR"
  dataset %>% select(WAR, Year)
}
```

```
# Obtain wrangled datasets
pit_rkes <- wrangle_init(df_pit_rkes)
pit_ret <- wrangle_init(df_pit_ret)
pos_rkes <- wrangle_init(df_pos_rkes)
pos_ret <- wrangle_init(df_pos_ret)
```

Finding the Couldabeen Thresholds

```
# obtain summary of WAR: median and variance
find_thresholds <- function(dataset, sds = 0){
  dataset %>%
    group_by(Year) %>%
    summarize(mean_WAR = median(WAR), sd_WAR = sqrt(var(WAR))) %>%
    mutate(threshold = mean_WAR + sds*sd_WAR) %>%
    select(Year, threshold)
}
```

```
# Get thresholds in each year
pit_thresholds <- find_thresholds(pit_rkes)
pos_thresholds <- find_thresholds(pos_rkes)
```

```
head(pit_thresholds)
```

```
## # A tibble: 6 x 2
##    Year threshold
##   <dbl>     <dbl>
## 1  1969     1
## 2  1970     0.8
## 3  1971     0.650
## 4  1972     1.1
## 5  1973     0.8
## 6  1974     1
```

Classifying Retiree Couldabeens

```r
# Appends above_threshold column to retirees dataset (checks if a retiree exceeds a threshold)
compare_thresholds <- function(dataset, summary_dataset){
  above_threshold <- rep(NA, nrow(dataset))
  for(i in 1:nrow(dataset)){
    year <- as.numeric(dataset[i,2])
    above_threshold[i] <- (dataset[i,1] > summary_dataset[year - 1968, 2])
  }
  dataset <- cbind(dataset,above_threshold)
  colnames(dataset)[3] <- "above_threshold"
  dataset
}
```

```r
# See and record which players cross that year's adjusted threshold from rookie players
pit_ret <- compare_thresholds(pit_ret, pit_thresholds)
pos_ret <- compare_thresholds(pos_ret, pos_thresholds)
# Get all retired couldabeens by combining the rows
retirees <- rbind(pit_ret,pos_ret)
```

```r
head(retirees)
```

```
##    WAR Year above_threshold
## 1  1.8 1972            TRUE
## 2  0.1 1974           FALSE
## 3  0.3 1976           FALSE
## 4 -0.5 1977           FALSE
## 5  0.4 1977           FALSE
## 6 -1.8 1974           FALSE
```

Counting Couldabeens by Year

```r
# Counts couldabeens in a given year
count_cbns <- function(dataset){
  dataset %>%
    group_by(Year) %>%
    summarize(cbns = sum(above_threshold))
}
```

```r
# Count the retiree couldabeens by year
couldabeens <- count_cbns(retirees)
```

```r
head(couldabeens)
```

```
## # A tibble: 6 x 2
##    Year  cbns
##   <dbl> <int>
## 1  1969     7
## 2  1970     3
## 3  1971     6
## 4  1972    10
## 5  1973     9
## 6  1974     6
```

Counting Retirees in a Given Year

```
# Yields the sum of retirees in two datasets (pitchers, position commonly)
total_retirees_by_yr <- function(pitchers, position){
  year_count_pitchers <- retirees_by_yr(pitchers)$retirees
  year_count_position <- retirees_by_yr(position)$retirees
  data.frame(Year = 1969:2018, retirees = year_count_position + year_count_pitchers)
}
# Counts the retirees in a single dataset
retirees_by_yr <- function(dataset){
  year_count <- dataset %>%
    group_by(Year) %>%
    summarize(retirees = n())
}
```

```
# Find number of retirees by year
num_retirees <- total_retirees_by_yr(df_pit_ret, df_pos_ret)
num_retirees <- data.frame(retirees = num_retirees)
```

```
head(num_retirees)
```

```
##   retirees.Year retirees.retirees
## 1          1969                32
## 2          1970                35
## 3          1971                44
## 4          1972                49
## 5          1973                41
## 6          1974                45
```

Retiree Proportion of Couldabeens

```
# Append number of retirees that year
couldabeens <- cbind(couldabeens, num_retirees)
# Find and append proportion of couldabeens : retirees
couldabeens <- couldabeens %>% mutate(prop = cbns/retirees)
```

```
head(couldabeens)
```

```
##   Year cbns retirees       prop
## 1 1969    7       32 0.21875000
## 2 1970    3       35 0.08571429
## 3 1971    6       44 0.13636364
## 4 1972   10       49 0.20408163
## 5 1973    9       41 0.21951220
## 6 1974    6       45 0.13333333
```

Modeling Helper Functions

```r
# Filters year to be in the post-rule era
postrule <- function(dataset, rule_year = 2002){
  dataset %>% filter(Year > rule_year)
}
# Filters year to be in the pre-rule era
prerule <- function(dataset, rule_year = 2002){
  dataset %>% filter(Year <= rule_year)
}
# Prep the boolean data for a logistic model
prep_booleans <- function(dataset){
  bools <- as.numeric(dataset$above_threshold)
  dataset[,3] <- bools
  dataset
}
```

Model Functions

```r
# Logisitic regression model on exceeding the threshold
logistic_model <- function(dataset){
  logistic_model <- glm(formula = above_threshold ~ Year, data = dataset, family = "binomial")
  logistic_model
}
# Linear model predicting proportion of couldabeens based on the year
linear_model <- function(dataset){
  linear_model <- lm(formula = prop ~ I(Year), data = dataset)
  linear_model
}
```

Logistic Models

```r
# Partition dataset into years before and after rule
retirees_pre <- prerule(retirees)
retirees_post <- postrule(retirees)
# On the retirees pre-rule dataset
model_log_pre <- logistic_model(retirees_pre)
model_log_post <- logistic_model(retirees_post)
```

Linear Models

```r
# Partition dataset into years before and after rule
couldabeens_pre <- prerule(couldabeens)
couldabeens_post <- postrule(couldabeens)
# Obtain linear model for pre-rule years
model_pre <- lm(formula = prop ~ I(Year), data = couldabeens_pre)
coefs_pre <- model_pre$coefficients
# Obtain linear model for post-rule years
model_post <- lm(formula = prop ~ I(Year), data = couldabeens_post)
coefs_post <- model_post$coefficients
```

References

(1) https://stathead.com/baseball/