MSD 2019 Final Project

A replication and extension of Wage disparity and team productivity: evidence from Major League Baseball by Craig A. Depken II, 1999

Forrest Hofmann (fhh2112) & Sagar Lal (sl3946)

2019-05-08 11:56:31

Contents

Introduction

Problem Description

Motivation

Data Source

Reproduction

Reproduction Code and Analysis

```
teams <- read_csv(here('data/teams.csv'))
salaries <- read_csv(here('data/salaries.csv'))
```

We clean the data by calculating the win percentage and removing an incomplete data point. The Lahman database is clearly missing some data for the 1987 Texas Rangers. For full data, see https://www.baseball-reference.com/teams/TEX/1987.shtml.

```
teams$WSWin <- as.logical(teams$WSWin == 'Y')
teams <- teams %>%
  filter(1985 <= yearID & yearID <= 2016) %>%
  mutate(winPercentage = W / (W + L) * 1000) %>%
  filter(yearID != 1987 & teamID != 'TEX')
```

```r
salaries <- salaries %>%
  filter(1985 <= yearID & yearID <= 2016) %>%
  mutate(salaryMil = salary / 1000000) %>%
  filter(yearID != 1987 & teamID != 'TEX')
```

```r
teams <- teams %>%
  inner_join(salaries) %>%
  group_by(yearID, teamID, G, W, L, WSWin, winPercentage) %>%
  summarize(totalSalaryMil = sum(salaryMil))
```

```r
salaries <- salaries %>%
  inner_join(teams) %>%
  mutate(salaryShare = salaryMil / totalSalaryMil * 100) %>%
  mutate(salaryShareSquared = salaryShare ^ 2) %>%
  select(yearID, teamID, playerID, salary, salaryShare, salaryShareSquared)
```

```r
teams <- teams %>%
  inner_join(salaries) %>%
  group_by(yearID, teamID, G, W, L, winPercentage, WSWin, totalSalaryMil) %>%
  summarize(HHI = sum(salaryShareSquared))
```

```r
teams_old <- teams %>%
  filter(1985 <= yearID & yearID <= 1998) %>%
  mutate(normalizedYear = yearID - 1985)
```

```r
summary(teams_old$winPercentage)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   327.2   456.8   496.9   500.1   543.2   703.7
```

```r
sd(teams_old$winPercentage)
```

```
## [1] 67.34053
```

```r
summary(teams_old$totalSalaryMil)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.613  14.217  23.655  26.220  37.022  72.356
```

```r
sd(teams_old$totalSalaryMil)
```

```
## [1] 14.00647
```

```r
summary(teams_old$HHI)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   427.5   666.8   754.7   801.5   876.3  2158.3
```

```r
sd(teams_old$HHI)
```

```
## [1] 220.2525
```

```r
hhi_fixed_old <- lm(formula = winPercentage ~ totalSalaryMil + HHI + normalizedYear +
                                              teamID + 0,
                    data = teams_old)
summary(hhi_fixed_old)$coefficients[1:3,]
```

```
##                   Estimate Std. Error   t value     Pr(>|t|)
## totalSalaryMil  1.96115149 0.48091620  4.077948 5.803436e-05
```

```
## HHI            -0.04611478 0.02028827 -2.272977 2.372293e-02
## normalizedYear -4.08667575 1.76297250 -2.318060 2.110738e-02
```

```
hhi_random_old <- lm(formula = winPercentage ~ totalSalaryMil + HHI + normalizedYear,
                     data = teams_old)
summary(hhi_random_old)$coefficients[1:4,]
```

```
##                   Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept)    515.9993052 15.18582553 33.979009 1.416422e-110
## totalSalaryMil   2.1067707  0.41153306  5.119323  5.180395e-07
## HHI             -0.0469241  0.01853488 -2.531665  1.180894e-02
## normalizedYear  -4.7782865  1.57044613 -3.042630  2.530509e-03
```

Reproduction Notes

- original author did not describe how time fixed effects are accounted for (across expansion periods or every year)
- no discussion about limiting to 25 man roster vs 40 man roster
- no discussion of cut players, traded players
- no discussion of signing bonuses

Extension

Extension Code and Analysis

```
teams_new <- teams %>%
  filter(1999 <= yearID & yearID <= 2016) %>%
  mutate(normalizedYear = yearID - 1999)

salaries_new <- salaries %>%
  filter(1999 <= yearID & yearID <= 2016)
```

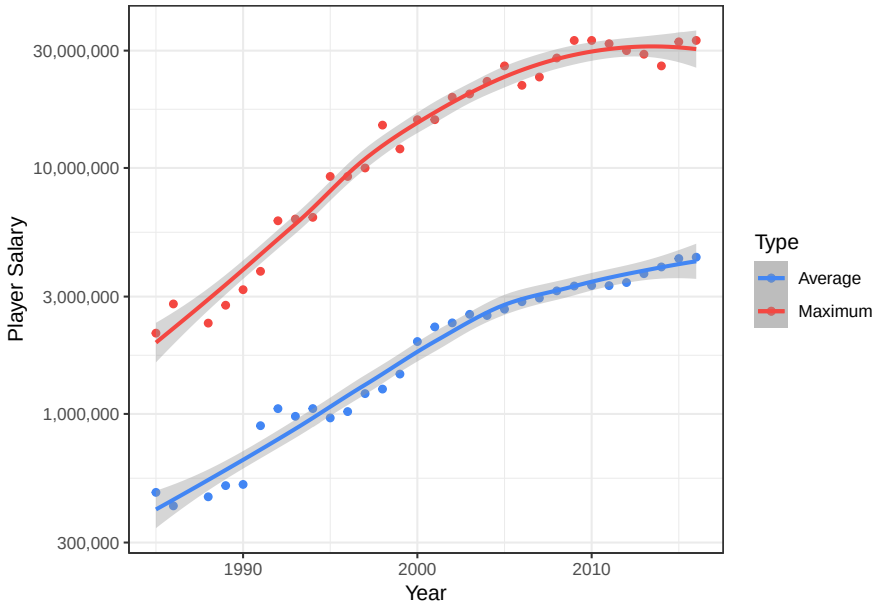
```
hhi_fixed_new <- lm(formula = winPercentage ~ totalSalaryMil + HHI + normalizedYear +
                                              teamID + 0,
                    data = teams_new)
summary(hhi_fixed_new)$coefficients[1:3,]
```

```
##                   Estimate Std. Error   t value     Pr(>|t|)
## totalSalaryMil  0.49949048 0.13266465  3.765061 0.0001868427
## HHI            -0.05358433 0.01442365 -3.715033 0.0002267173
## normalizedYear -2.00506563 0.73422847 -2.730847 0.0065462996
```

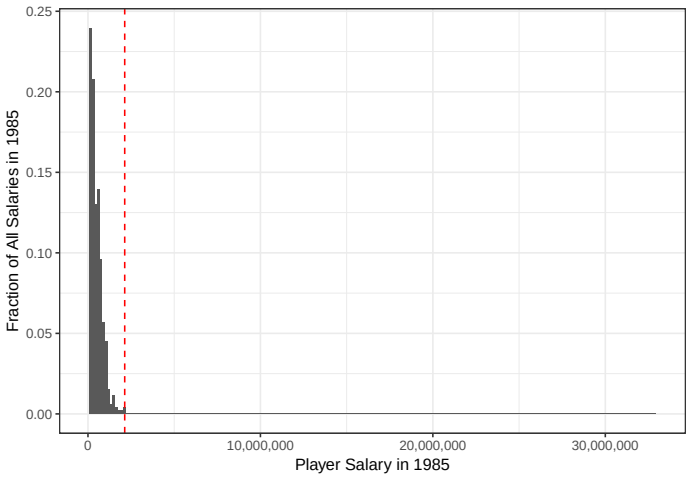
```
hhi_random_new <- lm(formula = winPercentage ~ totalSalaryMil + HHI + normalizedYear,
                     data = teams_new)
summary(hhi_random_new)$coefficients[1:4,]
```

```
##                    Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept)    503.61973192 15.73186679 32.012713 7.433408e-125
## totalSalaryMil   0.70690261  0.08558449  8.259705  1.226558e-15
## HHI             -0.04403327  0.01403935 -3.136419  1.807431e-03
## normalizedYear  -2.75511624  0.63636357 -4.329469  1.794894e-05
```

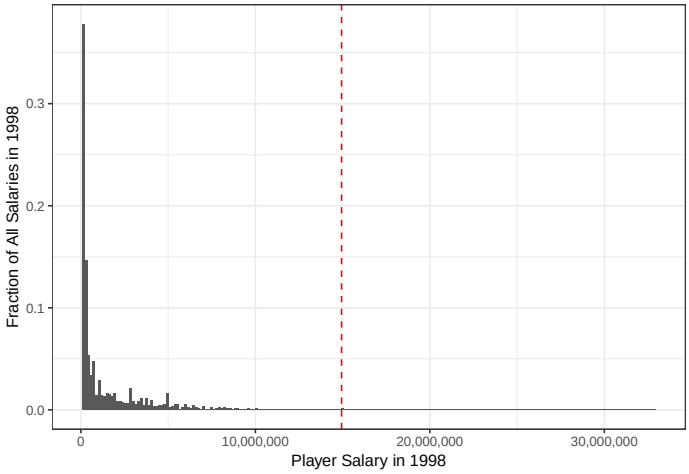
```r
salary_vs_time <- salaries %>%
  group_by(yearID) %>%
  summarize(avg = mean(salary), max = max(salary))

ggplot(data = salary_vs_time) +
  geom_point(aes(x = yearID, y = avg, color = 'Average')) +
  geom_smooth(aes(x = yearID, y = avg, color = 'Average')) +
  geom_point(aes(x = yearID, y = max, color = 'Maximum')) +
  geom_smooth(aes(x = yearID, y = max, color = 'Maximum')) +
  scale_color_manual(values = c('#4286f4', '#f44741')) +
  scale_y_log10(labels = comma) +
  labs(color = 'Type') +
  xlab('Year') +
  ylab('Player Salary')
```



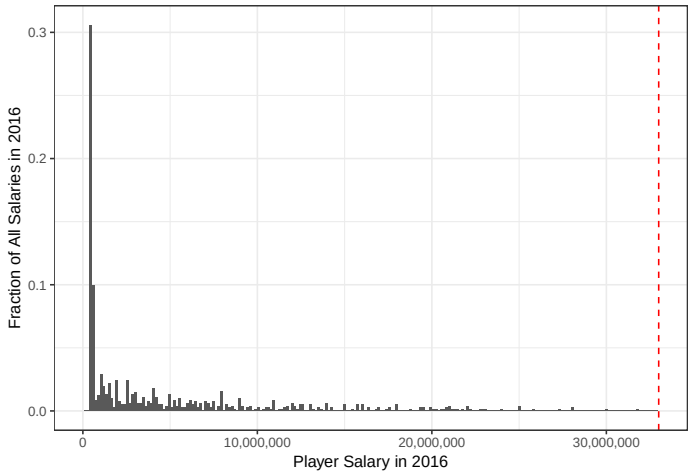

```r
salaries_1985 <- filter(salaries, yearID == 1985)
salaries_1998 <- filter(salaries, yearID == 1998)
salaries_2016 <- filter(salaries, yearID == 2016)

ggplot(data = salaries_1985) +
  geom_histogram(aes(x = salary, y = (..count..) / sum(..count..)), binwidth = 150000) +
  geom_vline(xintercept = max(salaries_1985$salary), color = 'red', linetype = 'dashed') +
  scale_x_continuous(limits = c(0, max(salaries$salary)), labels = comma) +  xlab('Player Salary in 198
  ylab('Fraction of All Salaries in 1985')
```



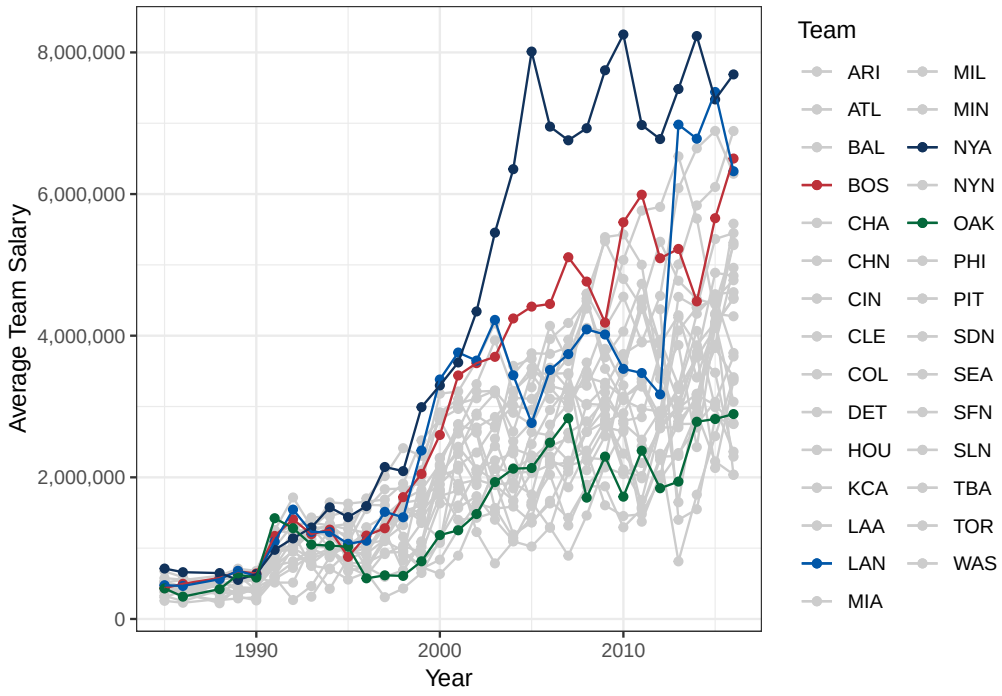

```
ggplot(data = salaries_1998) +
  geom_histogram(aes(x = salary, y = (..count..) / sum(..count..)), binwidth = 150000) +
  geom_vline(xintercept = max(salaries_1998$salary), color = 'red', linetype = 'dashed') +
  xlim(0, max(salaries$salary)) +
  scale_x_continuous(limits = c(0, max(salaries$salary)), labels = comma) +
  xlab('Player Salary in 1998') +
  ylab('Fraction of All Salaries in 1998')
```



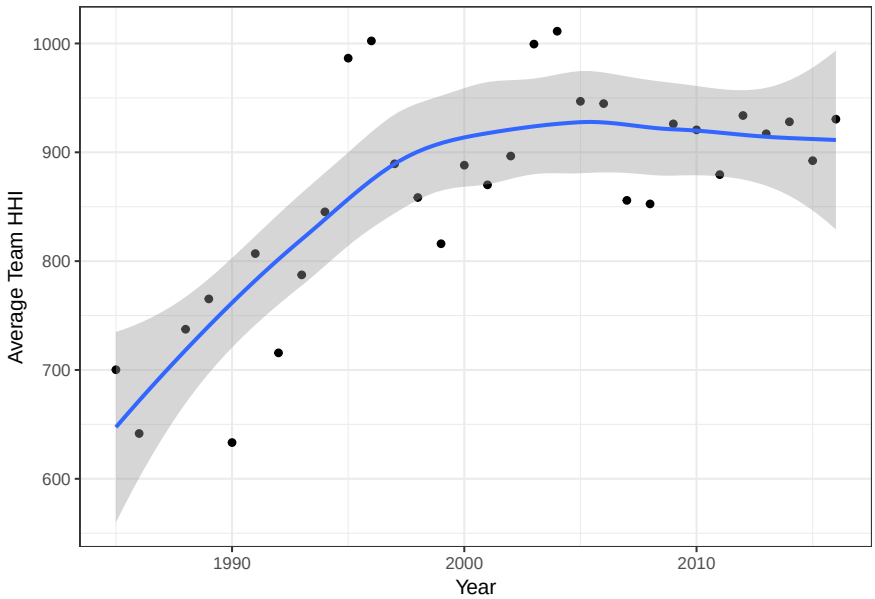

```
ggplot(data = salaries_2016) +
  geom_histogram(aes(x = salary, y = (..count..) / sum(..count..)), binwidth = 150000) +
  geom_vline(xintercept = max(salaries_2016$salary), color = 'red', linetype = 'dashed') +
  scale_x_continuous(limits = c(0, max(salaries$salary)), labels = comma) +
  xlab('Player Salary in 2016') +
  ylab('Fraction of All Salaries in 2016')
```




```
current_teamIDs <- c('ARI', 'ATL', 'BAL', 'BOS', 'CHA', 'CHN', 'CIN', 'CLE', 'COL', 'DET',
                     'HOU', 'KCA', 'LAA', 'LAN', 'MIA', 'MIL', 'MIN', 'NYA', 'NYN', 'OAK',
                     'PHI', 'PIT', 'SDN', 'SEA', 'SFN', 'SLN', 'TBA', 'TEX', 'TOR', 'WAS')
team_colors <- c('#cccccc', '#cccccc', '#cccccc', '#BD3039', '#cccccc',
                 '#cccccc', '#cccccc', '#cccccc', '#cccccc', '#cccccc',
                 '#cccccc', '#cccccc', '#cccccc', '#0157a8', '#cccccc',
                 '#cccccc', '#cccccc', '#11325b', '#cccccc', '#04683b',
                 '#cccccc', '#cccccc', '#cccccc', '#cccccc', '#cccccc',
                 '#cccccc', '#cccccc', '#cccccc', '#cccccc', '#cccccc')
colored_teamIDs <- c('BOS', 'LAN', 'NYA', 'OAK')

team_salary_vs_time <- salaries %>%
  filter(teamID %in% current_teamIDs) %>%
  group_by(yearID, teamID) %>%
  summarize(avg = mean(salary)) %>%
  mutate(flag = teamID %in% colored_teamIDs)

underlay_data <- filter(team_salary_vs_time, !flag)
overlay_data <- filter(team_salary_vs_time, flag)

ggplot() +
  geom_point(data = underlay_data, aes(x = yearID, y = avg, color = teamID)) +
  geom_line(data = underlay_data, aes(x = yearID, y = avg, color = teamID)) +
  geom_point(data = overlay_data, aes(x = yearID, y = avg, color = teamID)) +
  geom_line(data = overlay_data, aes(x = yearID, y = avg, color = teamID)) +
  scale_y_continuous(labels = comma) +
  scale_color_manual(values = team_colors) +
  labs(color = 'Team') +
  xlab('Year') +
  ylab('Average Team Salary')
```

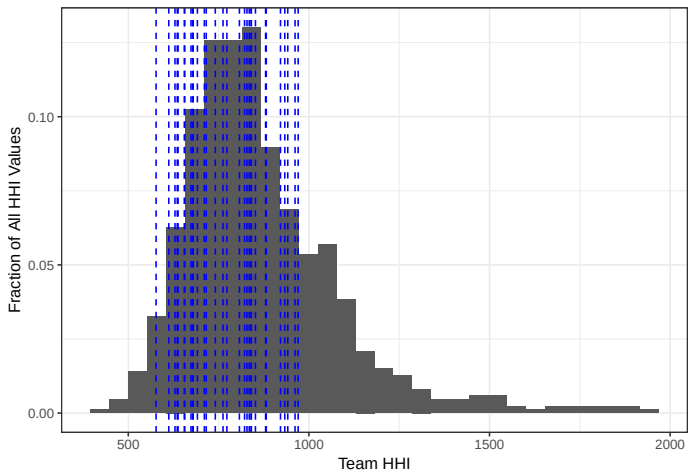



```
hhi_vs_time <- teams %>%
  group_by(yearID) %>%
  summarize(avg = mean(HHI))

ggplot(data = hhi_vs_time) +
  geom_point(aes(x = yearID, y = avg)) +
  geom_smooth(aes(x = yearID, y = avg)) +
  xlab('Year') +
  ylab('Average Team HHI')
```

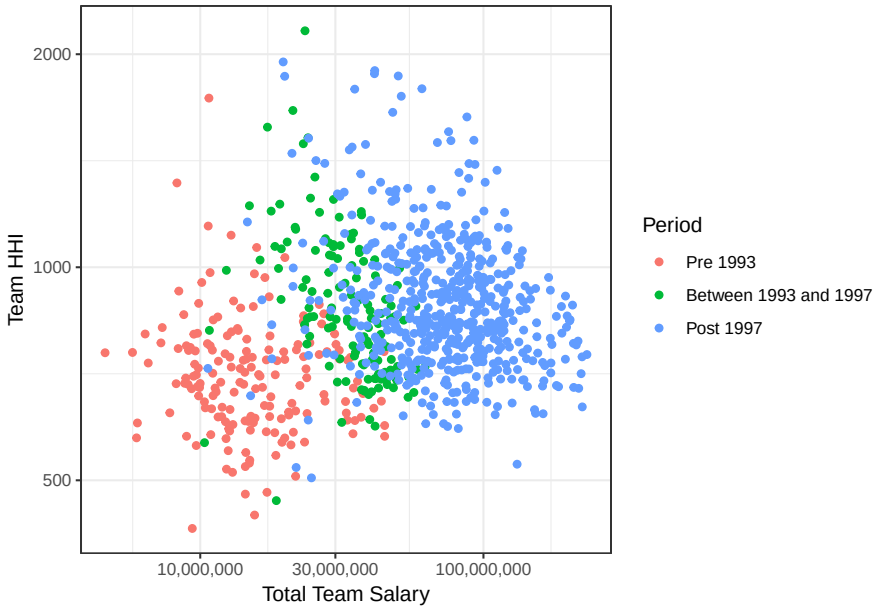



```
ggplot(data = filter(teams, mean(teams$HHI) - 5 * sd(teams$HHI) <= HHI &
                            HHI <= mean(teams$HHI) + 5 * sd(teams$HHI))) +
  geom_histogram(aes(x = HHI, y = (..count..) / sum(..count..))) +
```

```
  geom_vline(data = filter(teams, WSWin), aes(xintercept = HHI), color = 'blue', linetype = 'dashed') +
  xlab('Team HHI') +
  ylab('Fraction of All HHI Values')
```




```
year_to_period <- function(year) {
  if (year <= 1992)
    return('Pre 1993')
  else if (1993 <= year & year <= 1997)
    return('Between 1993 and 1997')
  else
    return('Post 1997')
}

hhi_vs_total_salary <- mutate(teams, period = year_to_period(yearID))
hhi_vs_total_salary$period <- factor(hhi_vs_total_salary$period,
                                     levels = c('Pre 1993', 'Between 1993 and 1997', 'Post 1997'))

ggplot(data = hhi_vs_total_salary) +
  geom_point(aes(x = totalSalaryMil * 1000000, y = HHI, color = period)) +
  scale_x_log10(labels = comma) +
  scale_y_log10() +
  labs(color = 'Period') +
  xlab('Total Team Salary') +
  ylab('Team HHI')
```



We compute the Gini coefficient for each team's salary. For more information, see https://en.wikipedia.org/wiki/Gini_coefficient.

```
gini <- salaries %>%
  group_by(yearID, teamID) %>%
  summarize(gini = Gini(salary))
teams <- inner_join(teams, gini)
```

We compute the Atkinson coefficient for each team's salary. For more information, see https://en.wikipedia.org/wiki/Atkinson_index.

```
atkinson <- salaries %>%
  group_by(yearID, teamID) %>%
  summarize(atk = Atkinson(salary))
teams <- inner_join(teams, atkinson)
```

```
teams_old <- teams %>%
  filter(1985 <= yearID & yearID <= 1998) %>%
  mutate(normalizedYear = yearID - 1985)

teams_new <- teams %>%
  filter(1999 <= yearID & yearID <= 2016) %>%
  mutate(normalizedYear = yearID - 1999)
```

```
gini_fixed_old <- lm(formula = winPercentage ~ totalSalaryMil + gini + normalizedYear +
                                               teamID + 0,
                     data = teams_old)
summary(gini_fixed_old)$coefficients[1:3,]
```

```
##                   Estimate Std. Error   t value     Pr(>|t|)
## totalSalaryMil    2.312489  0.4521101  5.114879 5.559085e-07
## gini           -129.827687 58.3962938 -2.223218 2.693371e-02
## normalizedYear   -3.793613  1.8442237 -2.057025 4.053398e-02
```

```
gini_random_old <- lm(formula = winPercentage ~ totalSalaryMil + gini + normalizedYear,
                      data = teams_old)
summary(gini_random_old)$coefficients[1:4,]
```

```
##                   Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)     539.115522 24.3427437 22.146868 1.404711e-67
## totalSalaryMil    2.495248  0.3775998  6.608182 1.532621e-10
## gini           -133.160525 54.5170461 -2.442548 1.510025e-02
## normalizedYear   -4.559438  1.6394568 -2.781066 5.724596e-03
```

```
gini_fixed_new <- lm(formula = winPercentage ~ totalSalaryMil + gini + normalizedYear +
                                               teamID + 0,
                     data = teams_new)
summary(gini_fixed_new)$coefficients[1:3,]
```

```
##                    Estimate Std. Error   t value     Pr(>|t|)
## totalSalaryMil    0.6624573  0.1276041  5.191504 3.069387e-07
## gini           -218.5193970 58.0041403 -3.767307 1.852177e-04
## normalizedYear   -2.7481628  0.7167617 -3.834137 1.425403e-04
```

```
gini_random_new <- lm(formula = winPercentage ~ totalSalaryMil + gini + normalizedYear,
                      data = teams_new)
summary(gini_random_new)$coefficients[1:4,]
```

```
##                    Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)     550.1960765 32.40207090 16.980275 9.795202e-52
## totalSalaryMil    0.8147846  0.08086385 10.076006 6.326317e-22
## gini           -158.9308612 55.31835762 -2.873022 4.232464e-03
## normalizedYear   -3.2653945  0.62553010 -5.220204 2.591533e-07
```

```
atk_fixed_old <- lm(formula = winPercentage ~ totalSalaryMil + atk + normalizedYear +
                                              teamID + 0,
                    data = teams_old)
summary(atk_fixed_old)$coefficients[1:3,]
```

```
##                   Estimate Std. Error   t value     Pr(>|t|)
## totalSalaryMil    2.376013  0.4504696  5.274524 2.527538e-07
## atk            -179.252498 65.9775739 -2.716870 6.966615e-03
## normalizedYear   -3.413854  1.8123452 -1.883667 6.056128e-02
```

```
atk_random_old <- lm(formula = winPercentage ~ totalSalaryMil + atk + normalizedYear,
                     data = teams_old)
summary(atk_random_old)$coefficients[1:4,]
```

```
##                   Estimate Std. Error   t value      Pr(>|t|)
## (Intercept)     509.553006 11.7544621 43.349751 1.975329e-139
## totalSalaryMil    2.555104  0.3759984  6.795518  4.948012e-11
## atk            -181.545622 61.0995554 -2.971308  3.179705e-03
## normalizedYear   -4.188912  1.6071160 -2.606478  9.556548e-03
```

```
atk_fixed_new <- lm(formula = winPercentage ~ totalSalaryMil + atk + normalizedYear +
                                              teamID + 0,
                    data = teams_new)
summary(atk_fixed_new)$coefficients[1:3,]
```

```
##                    Estimate Std. Error   t value     Pr(>|t|)
## totalSalaryMil    0.6943805  0.1274283  5.449185 8.049458e-08
## atk            -267.2160401 60.5183651 -4.415454 1.242698e-05
## normalizedYear   -2.8583544  0.7142924 -4.001659 7.268982e-05
```

```
atk_random_new <- lm(formula = winPercentage ~ totalSalaryMil + atk + normalizedYear,
                     data = teams_new)
summary(atk_random_new)$coefficients[1:4,]
```

```
##                    Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept)     512.1999193 17.31125767 29.587678 2.179218e-113
## totalSalaryMil    0.8479434  0.08182447 10.362956  5.387417e-23
## atk            -193.0102061 57.91250941 -3.332790  9.212586e-04
## normalizedYear   -3.3980916  0.62691299 -5.420356  9.131764e-08
```

```
num_folds <- 5
num_rows <- nrow(teams)
shuffle_idx <- sample(1:num_rows, num_rows, replace = FALSE)

teams_k_fold <- teams[shuffle_idx, ] %>%
  ungroup() %>%
  mutate(fold = (row_number() %% num_folds) + 1) %>%
  mutate(normalizedYear = yearID - 1985)

validate_err <- c()
train_err <- c()
for (f in 1:num_folds) {
  curr_train <- filter(teams_k_fold, fold != f)
  model <- lm(formula = winPercentage ~ totalSalaryMil + HHI + normalizedYear + teamID + 0,
              data = curr_train)
  train_err[f] <- sqrt(mean((predict(model, curr_train) - curr_train$winPercentage) ^ 2))

  curr_validate <- filter(teams_k_fold, fold == f)
  validate_err[f] <- sqrt(mean((predict(model, curr_validate) - curr_validate$winPercentage) ^ 2))
}

avg_validate_err <- mean(validate_err)
se_validate_err <- sd(validate_err) / sqrt(num_folds)

avg_train_err <- mean(train_err)
se_train_err <- sd(train_err) / sqrt(num_folds)
```

```
teams_pre_2011 <- teams %>%
  filter(yearID <= 2011) %>%
  mutate(normalizedYear = yearID - 1985)
teams_post_2012 <- teams %>%
  filter(yearID >= 2012) %>%
  mutate(normalizedYear = yearID - 1985)

time_model <- lm(formula = winPercentage ~ totalSalaryMil + HHI + normalizedYear,
                 data = teams_pre_2011)
time_train_err <- sqrt(mean((predict(model, teams_pre_2011) - teams_pre_2011$winPercentage) ^ 2))
time_validate_err <- sqrt(mean((predict(model, teams_post_2012) - teams_post_2012$winPercentage) ^ 2))
```

Extension Notes

- note that minimum salary has increased over time: https://www.baseball-reference.com/bullpen/Minimum_salary

- Coefficients of different inequality indexes are the same sign and have the same predictive impact (magnitudes are different because HHI has a wider range than the other two which are between 0 and 1), so we just pick hhi in this case to show
- Used fixed effects model when split data independent of time, ie teams from 1985 and 2016 are in the training set. Meanwhile used random effects model when split data based on time, ie teams from 1985-2011 were in training set and 2012-2016 were test set. Note that data is shuffled in this case. The use of the random effects model for the latter was to account for the fact that teams might be dominant in early years but not so much in more recent years ie dont want to worry about team being good in 80s/90s and bad in 2010s. Also handles the issue of teams not existing in training set but potentially in validation set (since teams come and go, move cities, rebrand, etc)
- 6.4%-6.6% error in win percentage predictions for both kinds of regressions. This isn't amazing given that the range of reasonable values is roughly 40%-70% win percentage

Postface

The following is a list of all packages used to generate these results.

```
sessionInfo()
```

```
## R version 3.5.2 (2018-12-20)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.4
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] bindrcpp_0.2.2  forcats_0.3.0   stringr_1.3.1   dplyr_0.7.8
##  [5] purrr_0.3.0     readr_1.3.1     tidyr_0.8.2     tibble_2.0.1
##  [9] ggplot2_3.1.0   tidyverse_1.2.1 scales_1.0.0    ineq_0.2-13
## [13] here_0.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.5 xfun_0.4         haven_2.0.0      lattice_0.20-38
##  [5] colorspace_1.4-0 generics_0.0.2   htmltools_0.3.6  yaml_2.2.0
##  [9] rlang_0.3.1      pillar_1.3.1     glue_1.3.0       withr_2.1.2
## [13] modelr_0.1.2     readxl_1.2.0     bindr_0.1.1      plyr_1.8.4
## [17] munsell_0.5.0    gtable_0.2.0     cellranger_1.1.0 rvest_0.3.2
## [21] evaluate_0.12    labeling_0.3     knitr_1.21       broom_0.5.1
## [25] Rcpp_1.0.0       backports_1.1.3  jsonlite_1.6     hms_0.4.2
## [29] digest_0.6.18    stringi_1.2.4    grid_3.5.2       rprojroot_1.3-2
## [33] cli_1.0.1        tools_3.5.2      magrittr_1.5     lazyeval_0.2.1
## [37] crayon_1.3.4     pkgconfig_2.0.2  xml2_1.2.0       lubridate_1.7.4
## [41] assertthat_0.2.0 rmarkdown_1.11   httr_1.4.0       rstudioapi_0.9.0
## [45] R6_2.3.0         nlme_3.1-137     compiler_3.5.2
```