Baseball Salary Analysis

Edgar Hernandez, Ryan Sims, Jessica Tomas

2022-08-05

Contents

Introduction

We will be examining what statistics related to a baseball player's performance have a significant effect on the salary of a player. This is particularly interesting as it is sort of the flip-side of the coin to the traditional Sabermetrics employed by the Oakland Athletics in the 1990's to analyze player statistics to try to assemble a winning team. We will be instead be leveraging the dataset to instead work for the players, hopefully determining which statistic(s) a player should focus on improving in order to increase their compensation.

Before we begin our analysis, we must first load the data using the `Lahman` library:

```
library(Lahman)
library(knitr)
library(lmtest)
```

We then join the Salaries, Fielding, and Batting tables:

Additionally, we will increment all of the numeric variables by one to facilitate log transformations for the predictors:

```
salary_data[ , ! colnames(salary_data) %in% c("yearID", "lgID", "POS") ] =
  salary_data[ , ! colnames(salary_data) %in% c("yearID", "lgID", "POS") ] + 1
```

Methods

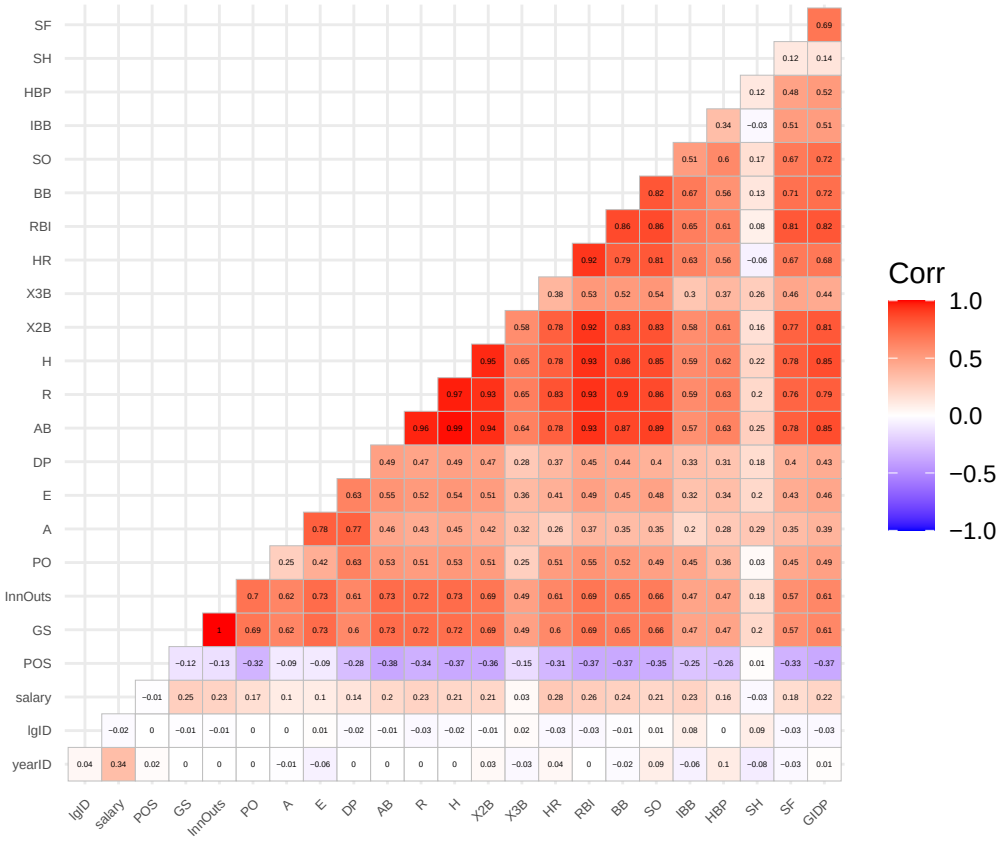
First, we visualize pairwise correlations between `Salary` and other variables:

- Calculating the correlations:

```
salary_data_plot = as.data.frame(lapply(salary_data, as.integer))
salary_data_cor = cor(salary_data_plot)
```

- Plotting the relationships:

```
library(ggcorrplot)
my_plt = ggcorrplot(salary_data_cor, lab = TRUE, lab_size = 1, type = "lower")
my_plt + theme(axis.text.x = element_text(size = 5), axis.text.y = element_text(size = 5))
```



- As expected, certain batting predictors such as `Doubles (X2B)` and `Triples (X3B)` have a very high correlation. We will keep these correlations under consideration as we refine our model.Additionally, we will remove `InnOuts` from our model since it is perfectly correlated with `Games Started (GS)`.

Before generating models, we begin by splitting the data into a test and train dataset. For this project we will split 60% of the data into a training set and 40% of the data into a testing set.

```
salary_data = subset(salary_data, select = -c(InnOuts))
set.seed(20220805)
salary_trn_idx = sample(nrow(salary_data), size = trunc(0.60 * nrow(salary_data)))
salary_trn = salary_data[salary_trn_idx, ]
salary_tst = salary_data[-salary_trn_idx, ]
```

Next, we create a simple additive model with all of the predictors and a model with no predictors:

```
simple_add = lm(salary ~ ., data = salary_trn)
simple_fit = lm(salary ~ 1, data = salary_trn)
```

We then use AIC and BIC stepwise procedures to refine our model (See appendix):

```
biggest = formula(lm(salary ~ ., data = salary_trn))
back_aic = step(simple_add, direction = "backward", trace = 0)

n = length(resid(simple_add))
back_bic = step(simple_add, direction = "backward", k = log(n), trace = 0)
```

- Now we can compare the LOOCV RMSE and Adjusted R Squared for both models:

```
calc_loocv_rmse = function(model) {
  sqrt(mean((resid(model) / (1 - hatvalues(model))) ^ 2))
}

rsquared = c(summary(back_aic)$adj.r.squared, summary(back_bic)$adj.r.squared)
rmse = c(calc_loocv_rmse(back_aic), calc_loocv_rmse(back_bic))
aic_results = data.frame(Model = c("Backward AIC", "Backward BIC"),
                         AdjR = rsquared, RMSE = rmse)

kable(aic_results,
      col.names = c("Selection Procedure", "Adjusted R. Squared", "LOOCV RMSE"))
```

Selection Procedure	Adjusted R. Squared	LOOCV RMSE
Backward AIC	0.2707598	2793220
Backward BIC	0.2704224	2793664

- Although both models have similar Adjusted R^2 and LOOCV RMSE values, the BIC model uses fewer predictors:

```
attr(back_aic$terms, "term.labels")
```

```
##  [1] "yearID" "lgID"   "POS"    "GS"     "PO"     "A"      "E"      "DP"
##  [9] "AB"     "X2B"    "X3B"    "HR"     "BB"     "SO"     "IBB"    "SH"
## [17] "GIDP"
```

```
attr(back_bic$terms, "term.labels")
```

```
##  [1] "yearID" "POS"    "GS"     "PO"     "A"      "E"      "DP"     "X2B"
##  [9] "X3B"    "HR"     "BB"     "SO"     "IBB"    "SH"     "GIDP"
```

Next we want to look at a additive model that takes into account all 2-way interactions and then perform backwards stepwise AIC against that fitted model to arrive at a set of significant coefficients

- Now we can compare the LOOCV RMSE and Adjusted R Squared for both models:

Because salary does not increase linearly in baseball and can range in orders of magnitude, we will create a model with a log transformation applied to our predictor. To do so, we will exclude the two data points where `salary` $= 0$.

```
log_fit = lm(log(salary) ~ ., salary > 0, data = salary_trn)
```

Next, we will perform the stepwise procedures:

```
biggest = formula(lm(log(salary) ~ ., salary > 0, data = salary_trn))
back_aic_log = step(log_fit, direction = "backward", trace = 0)

n = length(resid(log_fit))
back_bic_log = step(log_fit, direction = "backward", k = log(n), trace = 0)
```

- Now we can compare the LOOCV RMSE and Adjusted R Squared for both models:

```
calc_loocv_rmse_log = function(model) {
  sqrt(mean((exp(resid(model)) / (1 - exp(hatvalues(model)))) ^ 2))
}

rsquared = c(summary(back_aic_log)$adj.r.squared, summary(back_bic_log)$adj.r.squared)
rmse = c(calc_loocv_rmse_log(back_aic_log), calc_loocv_rmse_log(back_bic_log))
```

```
aic_results = data.frame(Model = c("Log Backward AIC", "Log Backward BIC"),
                         AdjR = rsquared, RMSE = rmse)

kable(aic_results,
      col.names = c("Selection Procedure", "Adjusted R. Squared", "LOOCV RMSE"))
```

Selection Procedure	Adjusted R. Squared	LOOCV RMSE
Log Backward AIC	0.3877737	8243.581
Log Backward BIC	0.3871234	9501.515

- Both models have a similar Adjusted R^2. Even though the AIC model has a lower LOOCV RMSE, the BIC model is significantly smaller. Going forward we will determine whether one model is better for predicting vs explaining.

```
attr(back_aic_log$terms, "term.labels")
```

```
##  [1] "yearID" "POS"    "GS"     "PO"     "A"      "E"      "DP"     "AB"
##  [9] "H"      "X2B"    "X3B"    "HR"     "RBI"    "BB"     "SO"     "IBB"
## [17] "HBP"    "SH"     "SF"     "GIDP"
```

```
attr(back_bic_log$terms, "term.labels")
```

```
##  [1] "yearID" "POS"    "GS"     "PO"     "E"      "AB"     "H"      "X2B"
##  [9] "X3B"    "HR"     "RBI"    "BB"     "SO"     "IBB"    "GIDP"
```

We can then run some diagnostics to ensure that our model does not violate normality and equal variance assumptions.

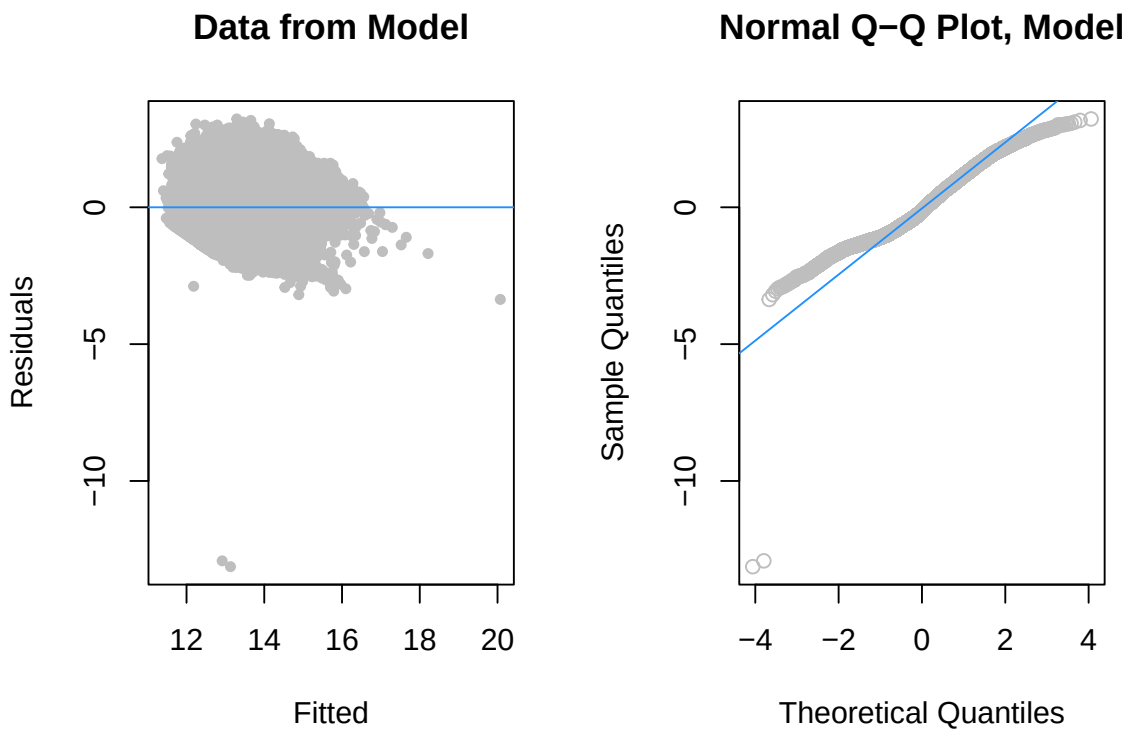
```
diagnostics = function(model, pcol = 'grey', lcol = 'dodgerblue',
                       alpha = .05, plotit = TRUE, testit = TRUE){

  if (plotit){
    par(mfrow = c(1, 2))
    plot(fitted(model), resid(model), col = pcol, pch = 20,
         xlab = "Fitted", ylab = "Residuals", main = "Data from Model")
    abline(h = 0, col = lcol, lwd = 1)
    qqnorm(resid(model), main = "Normal Q-Q Plot, Model", col = pcol)
    qqline(resid(model), col = lcol, lwd = 1)

  }
  if (testit){
    p_val = shapiro.test(resid(model))$p.value
    decision = ifelse(p_val < alpha, "Reject", "Fail to Reject")
    return(list("p_val" = p_val, "decision" = decision))
  }
}
```

- Based on the plots below below, we can observe that our model currently violates both normality and equal variance assumptions:

```
diagnostics(back_bic_log, testit = FALSE)
```

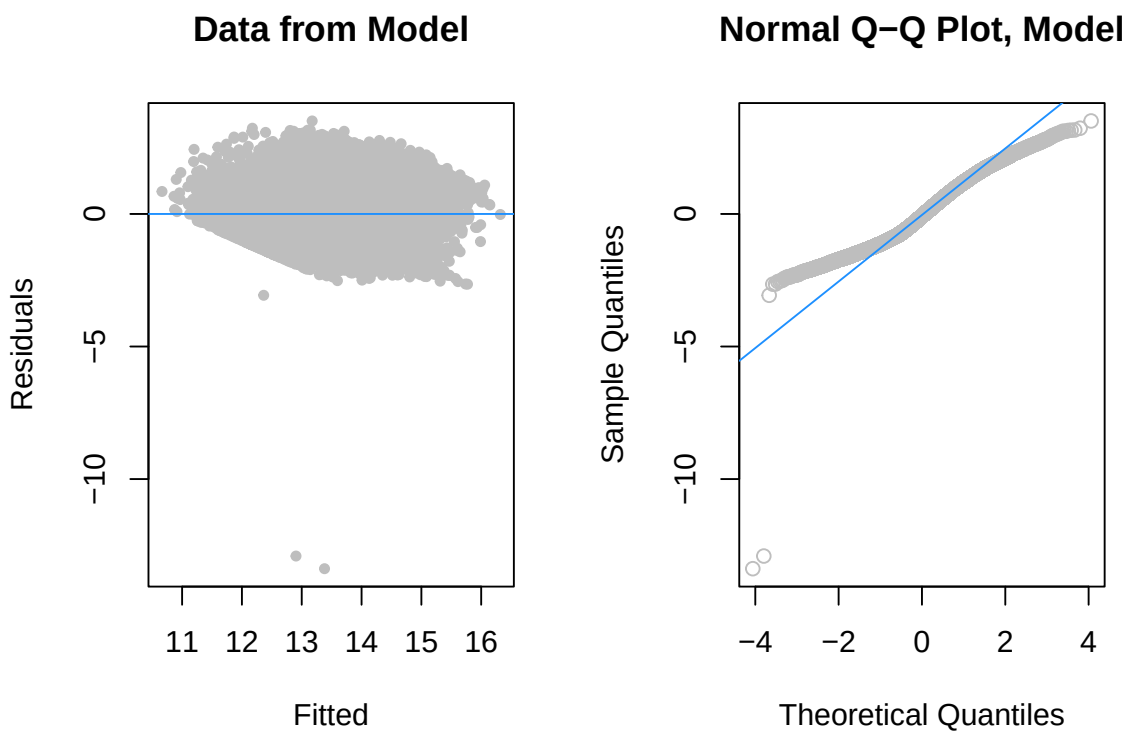


- We can attempt to stabilize the variance by applying log transformations to some of the predictors:

```
selected_fit = lm(log(salary) ~ yearID + POS + log(GS) + log(PO) + log(E) + log(AB) +
                    log(H) + log(X2B) + log(X3B) + log(HR) + log(RBI) + log(BB) +
                    log(SO) + log(IBB) + log(GIDP), salary > 0, data = salary_trn)
```

- After running the diagnostics again, we can see that the variance looks better, but we still have quite a few issues with normality:

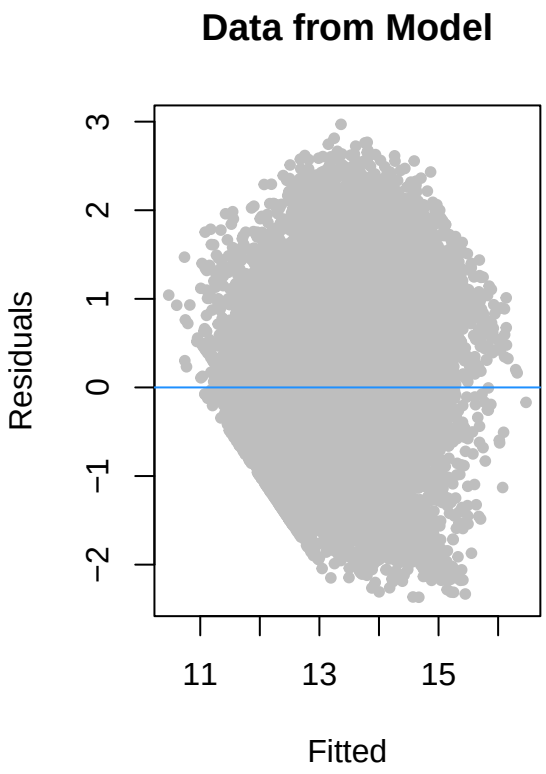
```
diagnostics(selected_fit, testit = FALSE)
```

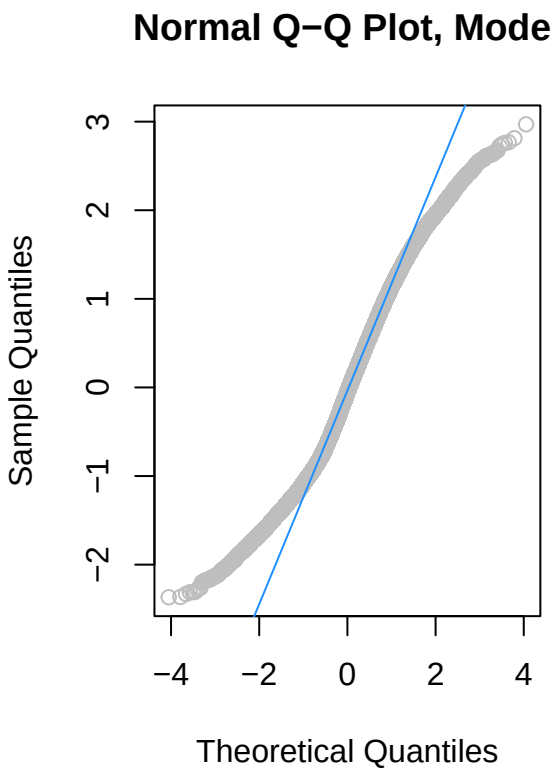


- We can further refine our model by removing influential outliers. We will use the $D_i > \frac{4}{n}$ Cook's distance heuristic to identify them:

```
cd_selected = cooks.distance(selected_fit)
large_cd  = cd_selected > 4 / length(cd_selected)

selected_fit_fix = lm(log(salary) ~ yearID + POS + log(GS) + log(PO) + log(E) + log(AB) +
                    log(H) + log(X2B) + log(X3B) + log(HR) + log(RBI) + log(BB) +
                    log(SO) + log(IBB) + log(GIDP),
                  subset = -which(large_cd | salary < 1), data = salary_trn)
diagnostics(selected_fit_fix, testit = FALSE)
```





Appendix

We also created a forward stepwise AIC model, which was the same size as the backward model:

```
forw_aic = step(simple_fit, direction = "forward", scope = biggest, trace = 0)
summary(forw_aic)$adj.r.squared
```

```
## [1] 0.2707598
```

```
calc_loocv_rmse(forw_aic)
```

```
## [1] 2793220
```

```
all.equal(sort(attr(back_aic$terms, "term.labels")), sort(attr(forw_aic$terms, "term.labels")))
```

```
## [1] TRUE
```