Final Report

Jackson Bandow

3/23/2021

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(plotly)
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```r
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```r
library(Lahman)
library(stringr)
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'carData'
```

```
## The following object is masked from 'package:Lahman':
##
##     Salaries
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

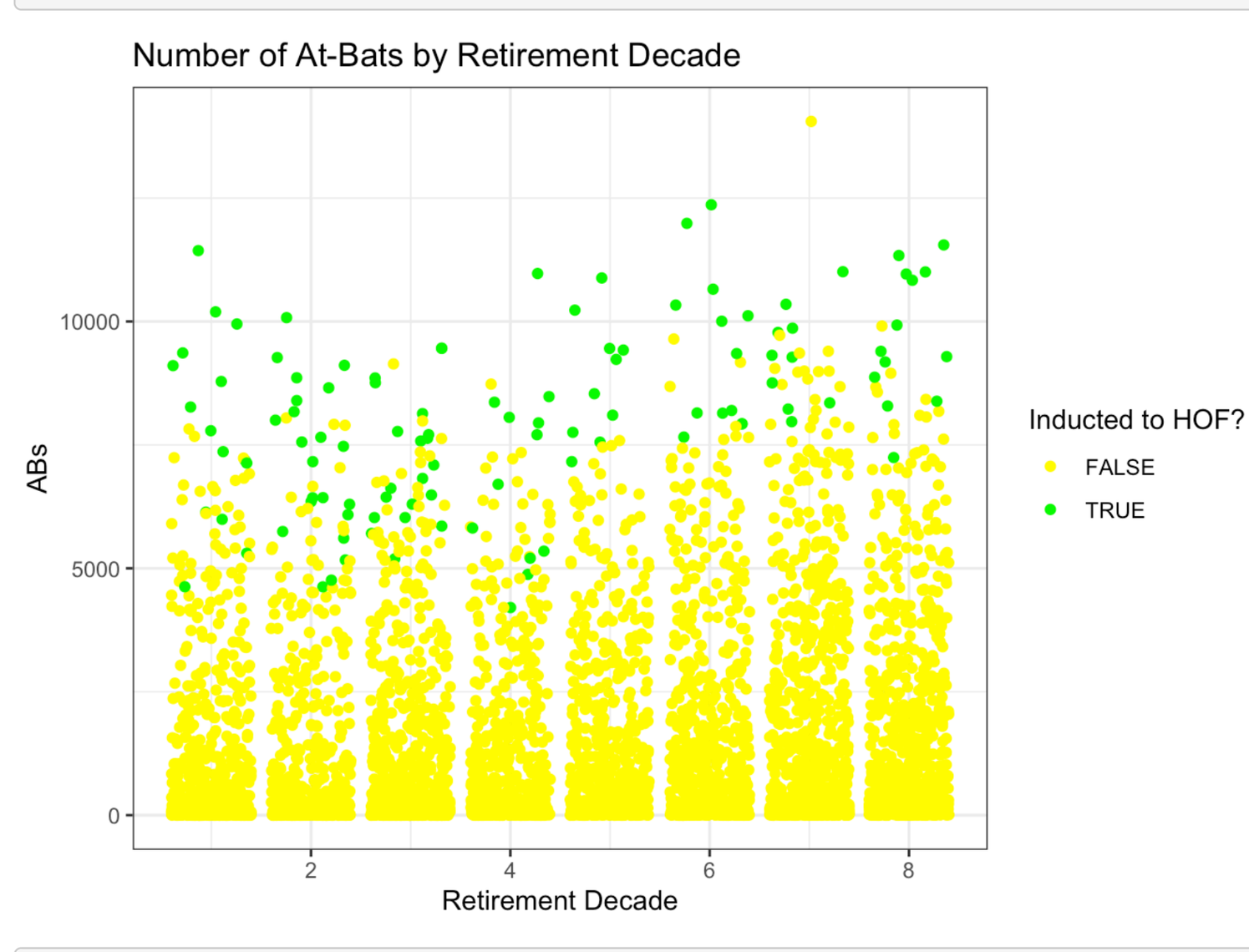
```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
get_prob <- function(input, coefs){
  inp <- t(as.matrix(cbind(rep(1, dim(input)[1]),input)))
  cf <- t(as.matrix(coefs))
  prob_new <- (exp(cf %*% inp)) / (1 + exp(cf %*% inp))
  names(prob_new) <- NULL
  return(as.vector(prob_new))
}

mc_rate <- function(probs, actual) {
  mean(round(probs, 0) != actual)
}

low_out <- function(col){
  stats <- summary(col)
  iqr <- stats[5] - stats[2]
  res <- stats[2] - 1.5*iqr
  names(res) <- NULL
  return(res)
}

standardize <- function(col) {
  return((col - mean(col)) / sd(col))
}
```

```r
baseball <- fread("baseball.csv")
baseball$careerLength <- as.numeric(as.Date(baseball$finalGame) - as.Date(baseball$debut)) / 365.25
baseball$debutDecade <- round(as.numeric(format(baseball$debut, "%Y")), -1)
baseball$retireRank <- floor((as.numeric(format(baseball$finalGame, "%Y")) - 4) / 10) - 191
baseball$FP <- (baseball$PO + baseball$A) / (baseball$PO + baseball$A + baseball$E)

HOF_batters <- baseball %>% filter(inducted & position != "P" & votedBy != "Negro League" & str_sub(playerID, -2,
-1) != "99" & finalGame < as.Date("2004-01-01") & finalGame > as.Date("1924-01-01"))

batters <- baseball %>% filter(position != "P" & votedBy != "Negro League" & str_sub(playerID, -2, -1) != "99" &
finalGame < as.Date("2004-01-01") & finalGame > as.Date("1924-01-01"))
```

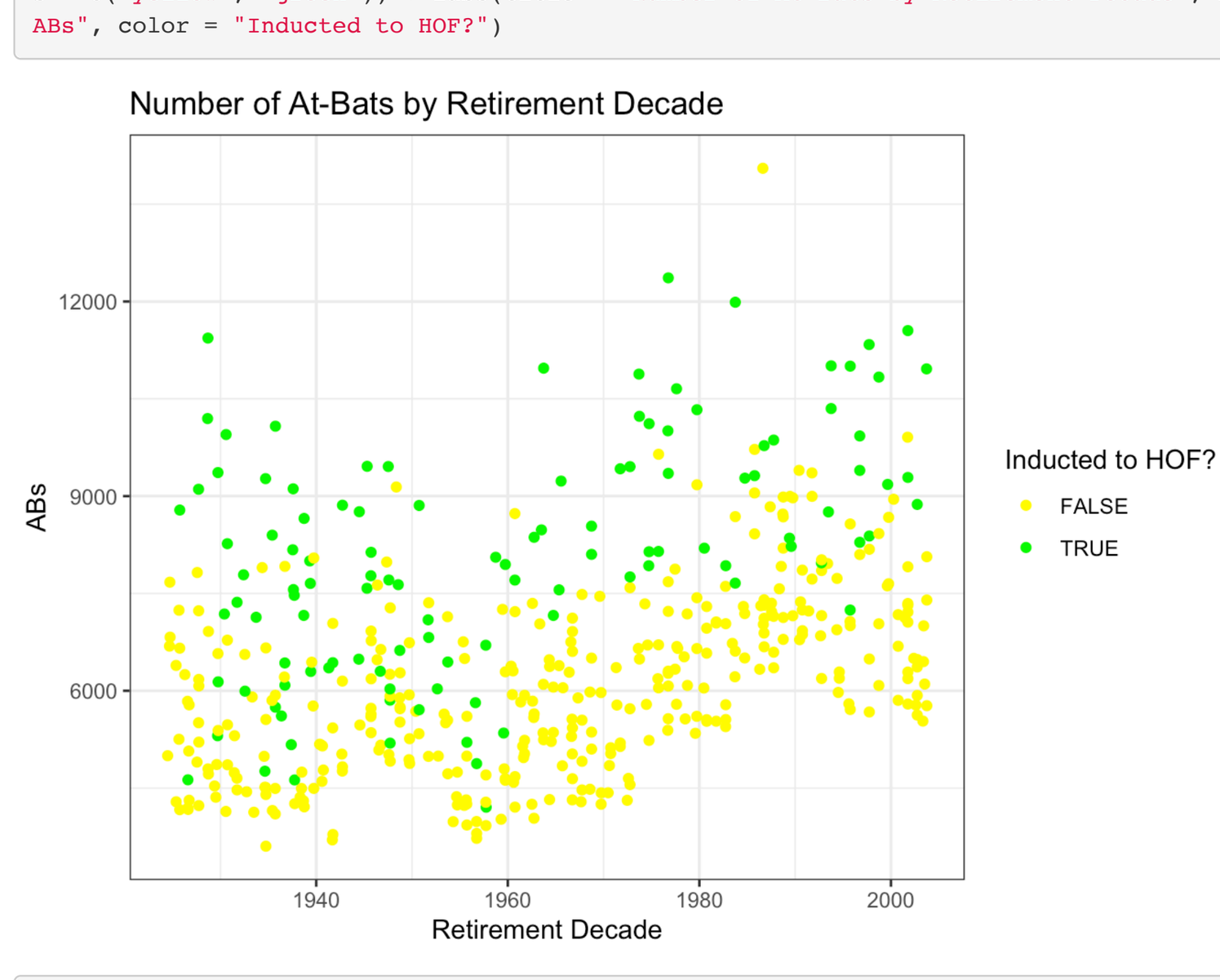
```r
ggplot(batters, aes(retireRank, AB, color = inducted)) + geom_jitter() + theme_bw() + scale_color_manual(values =
c("yellow", "green")) + labs(title = "Number of At-Bats by Retirement Decade", x = "Retirement Decade", y = "ABs"
, color = "Inducted to HOF?")
```




```r
paste0("Composition of HOF batters in data: ", round(mean(batters$inducted)*100, 1), "%.")
```

```
## [1] "Composition of HOF batters in data: 2.2%."
```

```r
# Limit data by ABs by retirement decade
new_batters <- data.frame()
batters <- batters %>% filter(rWAR >= 10)
for (i in unique(batters$retireRank)) {
  temp <- batters %>% filter(retireRank == i)
  new_batters <- rbind(new_batters, temp[order(temp$AB)[(dim(temp)[1]-59):dim(temp)[1]],])
}

ggplot(new_batters, aes(finalGame, AB, color = inducted)) + geom_jitter() + theme_bw() + scale_color_manual(value
s = c("yellow", "green")) + labs(title = "Number of At-Bats by Retirement Decade", x = "Retirement Decade", y = "
ABs", color = "Inducted to HOF?")
```




```r
paste0("Composition of HOF batters in data: ", round(mean(new_batters$inducted)*100, 1), "%.")
```

```
## [1] "Composition of HOF batters in data: 24%."
```

```r
# Make k-fold indeces
rand_ind <- sample(1:dim(new_batters)[1], dim(new_batters)[1], replace = FALSE)
sets <- NULL
fold_size <- dim(new_batters)[1] / 5
for (i in 1:5) {
  sets <- append(sets, list(rand_ind[((i-1)*fold_size + 1):(i*fold_size)]))
}

# Model 1: WAR
mcr1 <- NULL
auc1 <- NULL
for (i in 1:length(sets)) {
  mod <- glm(data = new_batters[!sets[[i]],], formula = inducted ~ rWAR, family = "binomial")
  mcr1 <- c(mcr1, mc_rate(get_prob(new_batters[sets[[i]],c("rWAR")], mod$coefficients), new_batters[sets[[i]],]$i
nducted))
  auc1 <- c(auc1, roc(new_batters[sets[[i]],]$inducted, get_prob(new_batters[sets[[i]],c("rWAR")], mod$coefficien
ts))$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```r
print(paste0("Model 1 -- Misclassification Rate: ", round(mean(mcr1)*100, 1), "%; Area Under the Curve: ", round(
mean(auc1), 3)))
```

```
## [1] "Model 1 -- Misclassification Rate: 13.8%; Area Under the Curve: 0.915"
```

```r
# Model 2: Standard Statistics
mcr2 <- NULL
auc2 <- NULL
for (i in 1:length(sets)) {
  mod <- glm(data = new_batters[!sets[[i]],], formula = inducted ~ G.B + AB + BB + H + HR + R + RBI + SB + FP + O
BP + SLG + BA, family = "binomial")
  mcr2 <- c(mcr2, mc_rate(get_prob(new_batters[sets[[i]],c("G.B", "AB", "BB", "H", "HR", "R", "RBI", "SB", "FP",
"OBP", "SLG", "BA")], mod$coefficients), new_batters[sets[[i]],]$inducted))
  auc2 <- c(auc2, roc(new_batters[sets[[i]],]$inducted, get_prob(new_batters[sets[[i]],c("G.B", "AB", "BB", "H",
"HR", "R", "RBI", "SB", "FP", "OBP", "SLG", "BA")], mod$coefficients))$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```r
print(paste0("Model 2 -- Misclassification Rate: ", round(mean(mcr2)*100, 1), "%; Area Under the Curve: ", round(
mean(auc2), 3)))
```

```
## [1] "Model 2 -- Misclassification Rate: 12.5%; Area Under the Curve: 0.899"
```

```r
# Standardize data by retirement decade
snew_batters <- new_batters %>% select(rWAR, AB, H, HR, BB, G.B, BA, R, RBI, SB, OBP, SLG, FP) %>% log() %>% cbin
d(new_batters[,c("inducted", "steriods", "retireRank")])
for (i in sort(unique(snew_batters$retireRank))) {
  snew_batters[snew_batters$retireRank == i,c("rWAR", "AB", "H", "HR", "BB", "G.B", "BA", "R", "RBI", "SB", "OBP"
, "SLG", "FP")] <- data.frame(apply(snew_batters[snew_batters$retireRank == i,c("rWAR", "AB", "H", "HR", "BB", "G
.B", "BA", "R", "RBI", "SB", "OBP", "SLG", "FP")], 2, standardize))
}

# Standardized Model 1: WAR
mcr3 <- NULL
auc3 <- NULL
for (i in 1:length(sets)) {
  mod <- glm(data = snew_batters[!sets[[i]],], formula = inducted ~ rWAR, family = "binomial")
  mcr3 <- c(mcr3, mc_rate(get_prob(snew_batters[sets[[i]],c("rWAR")], mod$coefficients), snew_batters[sets[[i]],]
$inducted))
  auc3 <- c(auc3, roc(snew_batters[sets[[i]],]$inducted, get_prob(snew_batters[sets[[i]],c("rWAR")], mod$coeffici
ents))$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```r
print(paste0("Standardized Model 1 -- Misclassification Rate: ", round(mean(mcr3)*100, 1), "%; Area Under the Cur
ve: ", round(mean(auc3), 3)))
```

```
## [1] "Standardized Model 1 -- Misclassification Rate: 10.6%; Area Under the Curve: 0.937"
```

```r
# Standardized Model 2: Standard Statistics
mcr4 <- NULL
auc4 <- NULL
for (i in 1:length(sets)) {
  mod <- glm(data = snew_batters[!sets[[i]],], formula = inducted ~ G.B + AB + BB + H + HR + R + RBI + SB + FP +
OBP + SLG + BA, family = "binomial")
  mcr4 <- c(mcr4, mc_rate(get_prob(snew_batters[sets[[i]],c("G.B", "AB", "BB", "H", "HR", "R", "RBI", "SB", "FP",
"OBP", "SLG", "BA")], mod$coefficients), snew_batters[sets[[i]],]$inducted))
  auc4 <- c(auc4, roc(snew_batters[sets[[i]],]$inducted, get_prob(snew_batters[sets[[i]],c("G.B", "AB", "BB", "H"
, "HR", "R", "RBI", "SB", "FP", "OBP", "SLG", "BA")], mod$coefficients))$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```r
print(paste0("Standardized Model 2 -- Misclassification Rate: ", round(mean(mcr4)*100, 1), "%; Area Under the Cur
ve: ", round(mean(auc4), 3)))
```

```
## [1] "Standardized Model 2 -- Misclassification Rate: 14.2%; Area Under the Curve: 0.892"
```

```r
# Standardized Model 1 with Time Interactions: WAR
mcr5 <- NULL
auc5 <- NULL
for (i in 1:length(sets)) {
  mod <- glm(data = snew_batters[!sets[[i]],], formula = inducted ~ retireRank*rWAR, family = "binomial")
  temp <- cbind(snew_batters[sets[[i]],c("retireRank", "rWAR")], snew_batters[sets[[i]],]$retireRank*snew_batters
[sets[[i]],]$rWAR)
  mcr5 <- c(mcr5, mc_rate(get_prob(temp, mod$coefficients), snew_batters[sets[[i]],]$inducted))
  auc5 <- c(auc5, roc(snew_batters[sets[[i]],]$inducted, get_prob(temp, mod$coefficients))$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```r
print(paste0("Standardized Model 1 with Time Interactions -- Misclassification Rate: ", round(mean(mcr5)*100, 1),
"%; Area Under the Curve: ", round(mean(auc5), 3)))
```

```
## [1] "Standardized Model 1 with Time Interactions -- Misclassification Rate: 11.2%; Area Under the Curve: 0.944
"
```

```r
# Standardized Model 2 with Time Interactions: Standard Statistics
mcr6 <- NULL
auc6 <- NULL
for (i in 1:length(sets)) {
  mod <- glm(data = snew_batters[!sets[[i]],], formula = inducted ~ retireRank*G.B + retireRank*AB + retireRank*B
B + retireRank*H + retireRank*HR + retireRank*R + retireRank*RBI + retireRank*SB + retireRank*FP + retireRank*OBP
+ retireRank*SLG + retireRank*BA, family = "binomial")
  temp <- cbind(snew_batters[sets[[i]],c("retireRank", "G.B", "AB", "BB", "H", "HR", "R", "RBI", "SB", "FP", "OBP
", "SLG", "BA")], snew_batters[sets[[i]],c("G.B", "AB", "BB", "H", "HR", "R", "RBI", "SB", "FP", "OBP", "SLG", "B
A")]*snew_batters[sets[[i]],]$retireRank)
  mcr6 <- c(mcr6, mc_rate(get_prob(temp, mod$coefficients), snew_batters[sets[[i]],]$inducted))
  auc6 <- c(auc6, roc(snew_batters[sets[[i]],]$inducted, get_prob(temp, mod$coefficients))$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```r
print(paste0("Standardized Model 2 with Time Interactions -- Misclassification Rate: ", round(mean(mcr6)*100, 1),
"%; Area Under the Curve: ", round(mean(auc6), 3)))
```

```
## [1] "Standardized Model 2 with Time Interactions -- Misclassification Rate: 15.8%; Area Under the Curve: 0.881
"
```

```r
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
# Model 1: WAR
mcr1 <- NULL
auc1 <- NULL
for (i in 1:length(sets)) {
  mod <- randomForest(factor(inducted) ~ rWAR + retireRank, data = new_batters[!sets[[i]],])
  mcr1 <- c(mcr1, mc_rate(predict(mod, new_batters[sets[[i]],c("rWAR", "retireRank")], type = "prob")[,2], new_ba
tters[sets[[i]],]$inducted))
  auc1 <- c(auc1, roc(new_batters[sets[[i]],]$inducted, predict(mod, new_batters[sets[[i]],c("rWAR", "retireRank"
)], type = "prob")[,1])$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```r
print(paste0("Random Forest 1 -- Misclassification Rate: ", round(mean(mcr1)*100, 1), "%; Area Under the Curve: "
, round(mean(auc1), 3)))
```

```
## [1] "Random Forest 1 -- Misclassification Rate: 10.8%; Area Under the Curve: 0.924"
```

```r
# Model 2: Standard Statistics
mcr2 <- NULL
auc2 <- NULL
for (i in 1:length(sets)) {
  mod <- randomForest(factor(inducted) ~ retireRank + G.B + AB + BB + H + HR + R + RBI + SB + FP + OBP + SLG + BA
, data = new_batters[!sets[[i]],])
  mcr2 <- c(mcr2, mc_rate(predict(mod, new_batters[sets[[i]],c("retireRank", "G.B", "AB", "BB", "H", "HR", "R", "
RBI", "SB", "FP", "OBP", "SLG", "BA")], type = "prob")[,2], new_batters[sets[[i]],]$inducted))
  auc2 <- c(auc2, roc(new_batters[sets[[i]],]$inducted, predict(mod, new_batters[sets[[i]],c("retireRank", "G.B",
"AB", "BB", "H", "HR", "R", "RBI", "SB", "FP", "OBP", "SLG", "BA")], type = "prob")[,1])$auc)
}
```

```
## Setting levels: control = FALSE, case = TRUE
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls > cases
```

```r
print(paste0("Random Forest 2 -- Misclassification Rate: ", round(mean(mcr2)*100, 1), "%; Area Under the Curve: "
, round(mean(auc2), 3)))
```

```
## [1] "Random Forest 2 -- Misclassification Rate: 9.8%; Area Under the Curve: 0.924"
```