Mehra_Kai_Assignment-4

Kai Mehra

2023-02-26

Libraries

```
# Importing the {tidyverse}, {ggplot2}, {GGally}, {car}, {caret}, and {rms}
# libraries
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(car)
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(rms)
```

```
## Loading required package: Hmisc
## Loading required package: survival
##
## Attaching package: 'survival'
##
## The following object is masked from 'package:caret':
##
##     cluster
##
## Loading required package: Formula
##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
##
## The following objects are masked from 'package:base':
##
##     format.pval, units
##
## Loading required package: SparseM
##
## Attaching package: 'SparseM'
##
## The following object is masked from 'package:base':
##
##     backsolve
##
##
## Attaching package: 'rms'
##
## The following objects are masked from 'package:car':
##
##     Predict, vif
```

Data

```
# loading in the data sets
hof <- read.csv("../Data/NBA Hall of Famers 2021.csv")
br_hof_preds <- read.csv("../Data/br_hof_preds.csv")

dim(hof)
```

```
## [1] 438   7
```

```
dim(br_hof_preds)
```

```
## [1] 250   3
```

The hof dataset is a Kaggle dataset that contains data on 438 NBA players who have made at least 1 all star game during their career over the course of the entire history of the NBA. The dataset includes data on whether they are a retired player in the hall of fame, a retired player not in the hall of fame, or an active player. Futher, the dataset contains information on the player's position, height, weight, number of all star games made, and birth year.

The br_hof_preds dataset contains the hall of fame predicted probabilities from Basketball Reference (https://www.basketball-reference.com/leaders/hof_prob.html). This dataset contains the player's name and their hall of fame probability according to Basketball Reference's (hence BR) model. The top 250 players in HOF probability are included in the model.

Research Question

Can a player be classified as a hall of famer based on their birth year, height, weight, position, and number of all star appearances?

Most contemporary analyses and discussions of a player's hall of fame likelihood rely on performance statistics (i.e. points per game, total career assists), individual awards (Most Valuable Player awards, All-NBA selections), and team awards (championships, playoff success). I am curious whether a simplistic model taking into account a players era (through birth year), physical characteristics (height, weight, and position), and performance (all star selections) can classify HOF probability well, or are more complicated models like BR's required.

Hypotheses

- H_0 - The hall of fame status of a player is not related to the birth year, height, weight, position, and number of all star appearances.
- H_1 - The hall of fame status of a player is related to at least one of the birth year, height, weight, position, and number of all star appearances.

Variables of Interest

Dependent Variable

hof: hof is a categorical variable with three outcomes:

- 0 - retired player not in the hall of fame
- 1 - retired player in the hall of fame
- 2 - active player

This variable is what we are attempting to classify. We will train the model on the retired players and attempt to apply it to the active players.

Independent Variables

- position: position is a categorical variable displaying what position a player primarily plays/played during their carrer: Guard (G), Forward (F), Center (C).
- all_stars: the number of times a player is selected to an all star game. Fans, players, and the media vote on the 24 best players to be all stars around the midpoint of each season. Players considered in this analysis must have at least 1 all star selection.
- height: the player's height in centimeters.
- weight: the player's weight in kilograms.
- born: the year the player was born

Data Wrangling

```
hof <- hof[-290,] # delete the misinput datapoint
```

There was a data point that had an incorrect measurement for the players height and weight.

```
hof <-
  hof %>%
  rename(
    hof = In_Hall_of_fame,
    all_stars = All_star_selections) # rename to better var names
```

```
hof %>%
  count(hof)
```

```
##   hof   n
## 1   0 244
## 2   1 129
## 3   2  64
```

This data set contains 244 retired players not in the hall of fame, 129 retired players in the hall of fame, and 64 active players.

```
hof$hof <- factor(hof$hof,
                  levels = c(0, 1, 2)) # change hof status to factor

hof$position <- factor(hof$position,
                       levels = c("G","F","C")) # change position to factor
```

```r
# create a dataset with only active players
hof_active <-
  hof %>%
  filter(
    hof == 2
  ) %>%
  dplyr::select(
    -hof
  )

# create a dataset with only retired players
hof_retired <-
  hof %>%
  filter(
    hof == 0 |
      hof == 1
  ) %>%
  dplyr::select(
    -Name
  )
```

Equalizing Frequency of HOF variable

```r
set.seed(2023) # set seed to ensure the same outcome for stochastic processes

hof_retired_not_hof <-
  hof_retired %>%
  filter(hof == 0) # select only non-hof retired players

hof_retired_not_hof <-
  sample_n(hof_retired_not_hof, 130) # randomly sample 130 players

hof_retired <-
  hof_retired %>%
  filter(hof == 1) %>%
  rbind(hof_retired_not_hof) # redefine the retired dataset to be equal
```

Since there were more retired players not in the hall of fame than in the hall of fame, I randomly sampled non-hof retired players to even out the distribution to make the model more accurate.

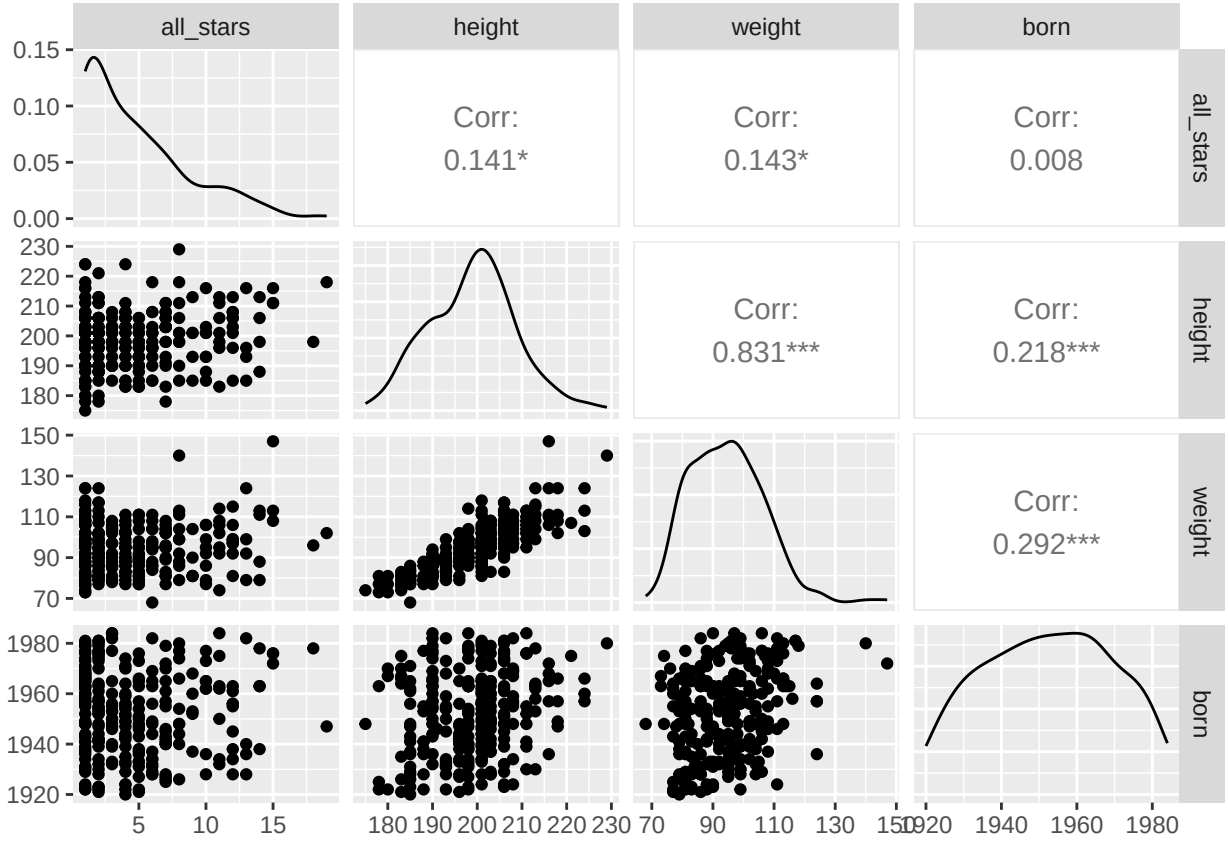
Creating the Model

Checking Distributions

Checking Normality of Variables

```r
hof_retired_num <-
  hof_retired %>%
```

```
  dplyr::select(
    all_stars,
    height,
    weight,
    born
  ) # select only numeric variables

hof_retired_num %>%
  ggpairs()
```



Using the ggpairs function, we can visualize the normality of the numeric variables. all-stars is not normal, but height, weight, and born are relatively normal. height and weight are highly correlated which makes intuitive sense.

Normalizing the Variables with bestNormalize

Normalizing the variables with bestNormalize did not improve the accuracy of the model, so the variables remain untransformed. This does not violate the assumptions of logistic regression, so we will process with untransformed variables.

Setting up the Model

```
logm_hof <- glm(
  formula = hof ~ .,
  data = hof_retired,
  family = "binomial" # logistic model
)
```

```
summary(logm_hof) # extracting the coeffs and values from the regression
```

```
##
## Call:
## glm(formula = hof ~ ., family = "binomial", data = hof_retired)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.4647  -0.4921  -0.1203   0.4159   2.1237
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 84.652332  25.677334   3.297 0.000978 ***
## positionF    1.241918   0.629275   1.974 0.048431 *
## positionC    3.045656   1.108165   2.748 0.005989 **
## all_stars    0.797878   0.102620   7.775 7.54e-15 ***
## height      -0.006705   0.046254  -0.145 0.884738
## weight      -0.052560   0.034057  -1.543 0.122765
## born        -0.042367   0.014166  -2.991 0.002783 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 359.05  on 258  degrees of freedom
## Residual deviance: 170.66  on 252  degrees of freedom
## AIC: 184.66
##
## Number of Fisher Scoring iterations: 6
```

In the first model, height and weight are not significant due to their high correlation with each other. All other variables are significant.

Checking Multicollinearity

```
vif(logm_hof)
```

```
## positionF positionC all_stars    height    weight      born
##  2.594667  4.157272  1.199805  4.960408  3.608829  1.339815
```

Height and weight are strongly multicollinear. Height is also multicollinear with positionC as Centers are generally the tallest players on the team. Thus, I removed height to alleviate both problems.

```
hof_retired_sig <-
  hof_retired %>%
  dplyr::select(-height) # remove height
```

```
logm_hof <- glm(
  formula = hof ~ .,
  data = hof_retired_sig,
  family = "binomial" # logistic
)
```

```
summary(logm_hof) # extracting the coeffs and values from the regression
```

```
##
## Call:
## glm(formula = hof ~ ., family = "binomial", data = hof_retired_sig)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.4705  -0.4907  -0.1204   0.4160   2.1544
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 84.70940   25.68597   3.298 0.000974 ***
## positionF    1.19928    0.55512   2.160 0.030742 *
## positionC    2.95474    0.91124   3.243 0.001185 **
## all_stars    0.79701    0.10233   7.789 6.76e-15 ***
## weight      -0.05500    0.02966  -1.854 0.063674 .
## born        -0.04294    0.01362  -3.153 0.001614 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 359.05  on 258  degrees of freedom
## Residual deviance: 170.68  on 253  degrees of freedom
## AIC: 182.68
##
## Number of Fisher Scoring iterations: 6
```

All variables are now signifcant and the AIC of the model is lower showing that removing height improved the model.

Checking Multicollinearity

```
vif(logm_hof)
```

```
## positionF positionC all_stars    weight      born
##  2.019235  2.814061  1.193794  2.730994  1.237384
```

There are no longer issues with multicollinearity, so we will proceed with this being the final model.

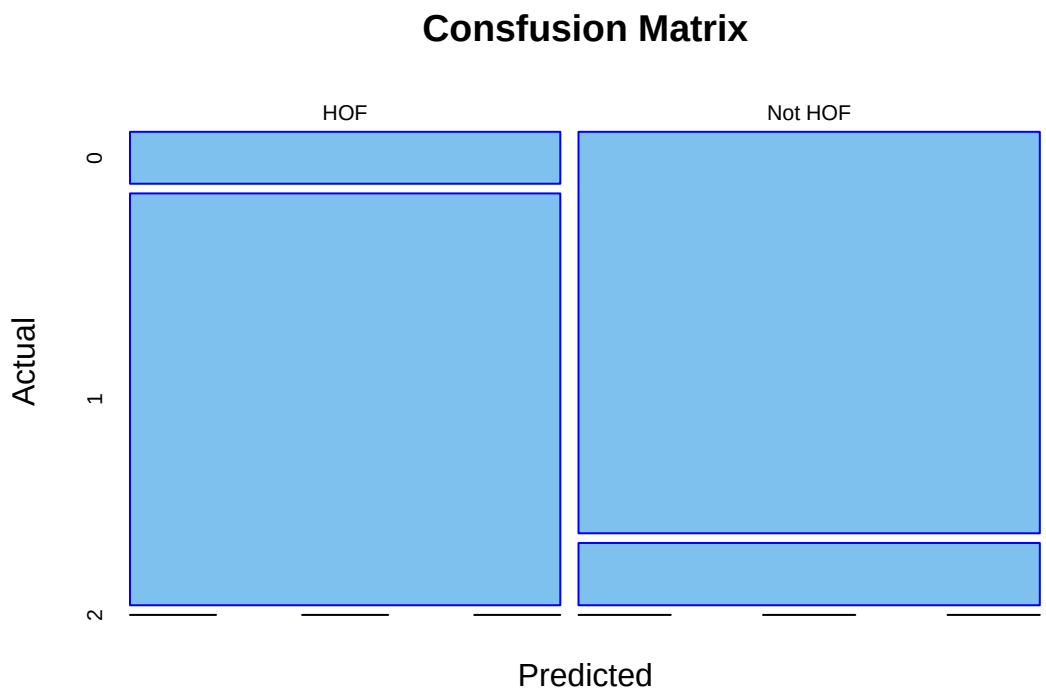
```r
# Obtain probabilities
probs <- predict(
  logm_hof,
  type = "response"
  # needed for probabilities
)
```

```r
# Obtain classes
classes <- factor(
  ifelse(
    probs > 0.50, # 50% cut is standard
    "HOF", # if TRUE
    "Not HOF" # if FALSE
  )
)
```

```r
# Compute confusion matrix
confusionMatrix(
  data = factor(2 - as.numeric(classes)), # predicted
  reference = factor(hof_retired$hof), # actual
  positive = "1" # P(Y = 1)
)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 116  18
##          1  14 111
##
##                Accuracy : 0.8764
##                  95% CI : (0.8301, 0.9139)
##     No Information Rate : 0.5019
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7529
##
##  Mcnemar's Test P-Value : 0.5959
##
##             Sensitivity : 0.8605
##             Specificity : 0.8923
##          Pos Pred Value : 0.8880
##          Neg Pred Value : 0.8657
##              Prevalence : 0.4981
##          Detection Rate : 0.4286
##    Detection Prevalence : 0.4826
##       Balanced Accuracy : 0.8764
##
##        'Positive' Class : 1
##
```

```
mosaic_table <- table(classes, hof_retired$hof)

mosaicplot(mosaic_table,
           main = "Consfusion Matrix",
           xlab = "Predicted",
           ylab = "Actual",
           color = "skyblue2",
           border = "blue")
```



Discussion of Retired Player Model

Overall, the model predicts the retired player data very well. The overall accuracy of the model is 87.64% which is very strong. Further, the sensitivity is 86.05% and specificity is 89.23%. This means that the model is slightly better at correctly predicting when a player does not make the Hall of Fame. However, since overall accuracy, sensitivity, and specificity are all high, we can be confident that the model is predicting the retired player data very well. Additionally, the Kappa of 0.7529 further reinforces the strength of the model. The mosaic plot visually shows the accuracy of the model with the correct prediction boxes being much larger than the incorrect predictions.

In terms of the model itself, it is the following equation:

$$P(hof) = \frac{1}{1 + e^{-(84.7 + 1.19 * positionF + 2.95 * positionC + 0.797 * all_stars - 0.055 * weight - 0.042 * born)}}$$

Model Interpretation: All interpretations are made with the remaining variables being controlled.

- A player playing the forward position is 1.199 times more likely to make the HOF rather than a player playing the guard position.
- A player playing the center position is 2.95 times more likely to make the HOF rather than a player playing the guard position.
- For each all star selection a player receives, they 0.79 times more likely to make the HOF.
- For each additional kilogram of weight a player has, they are 0.055 times less likely to make the HOF.
- For each year later a player was born, they are 0.04 times less likely to make the HOF.

These results make sense as early basketball was dominated by taller players at the forward and center positions. Additionally, making more all star games is generally indicative of a better player, thus a player more likely to make the hall of fame. Finally, players must be at least 5 years past retirement before being elligible for the hall of fame. Thus younger players, with greater birth years, are less likely to make the hall of fame because they have had less time to do so.

Now that this model has been trained on the retired players, I will attempt to validate it using the active players in the data set.

Predicting the HOF status of Active Players

```
all_star_2022 <-
  c("Stephen Curry","DeMar DeRozan", "LeBron James","Giannis Antetokounmpo","Nikola Jokic","Chris Paul"

all_star_2023 <- c("LeBron James","Nikola Jokic","Kyrie Irving","Luka Doncic","Joel Embiid","Giannis An
```

```
for (i in 1:nrow(hof_active)) {
  if(hof_active[i,]$Name %in% all_star_2022){
    hof_active[i, ]$all_stars = hof_active[i, ]$all_stars + 1
  }
  if(hof_active[i,]$Name %in% all_star_2023){
    hof_active[i, ]$all_stars = hof_active[i, ]$all_stars + 1
  }
}
```

This dataset was published before the 2022 and 2023 all star game. Thus, I added 1 or 2 all star appearances to the players who made the 2022 and/or 2023 all star games to make the data up to date. None of the other variables in the data set change over time, so no other adjustment is needed for the active players.

```
hof_active_interest <-
  hof_active %>%
  dplyr::select(
    position,
    all_stars,
    weight,
    born) # selecting variables of interest
```

```
active_probs <-
  predict(logm_hof,
        newdata = hof_active_interest,
        type = "response") # generating probabilities for the active players
```

```
br_hof_preds <-
  br_hof_preds %>%
  rename(Name = Player,
         br_hof_prob = HoF.Prob) %>%
  dplyr::select(-Rank)
```

Basketball Reference publishes their own HOF probabilities which I have created into a new dataframe.

```
# create a new data frame with the HOF probabilities
hof_active_predicted <-
  hof_active %>%
  mutate(
    hof_prob = active_probs
  )
```

```
# combining my HOF model with BR's using left_join
prediction_comparison <-
  left_join(hof_active_predicted, br_hof_preds, by = "Name") %>%
  dplyr::select(
    Name,
    hof_prob,
    br_hof_prob
  )

prediction_comparison_complete <-
  na.omit(prediction_comparison) # removing any players without predictions
```

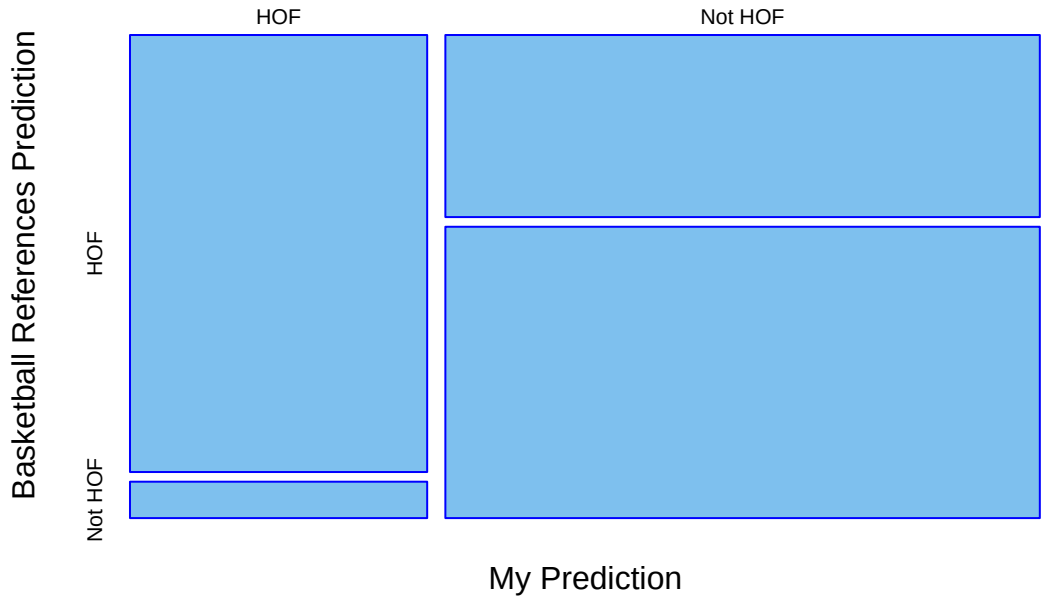
```
active_classes <-
  factor(
    ifelse(
    prediction_comparison_complete$hof_prob > 0.50,
    "HOF", # if TRUE
    "Not HOF" # if FALSE
  )
) # converting the probabilities into classes
```

```
br_hof_classes <-
  factor(
    ifelse(
    prediction_comparison_complete$br_hof_prob > 0.50,
    "HOF", # if TRUE
    "Not HOF" # if FALSE
  )
) # converting the probabilities into classes
```

```
# Compute confusion matrix
confusionMatrix(
  data = active_classes, # predicted
  reference = br_hof_classes, # actual
  positive = "HOF" # P(Y = 1)
) # creating a confusion matrix
```

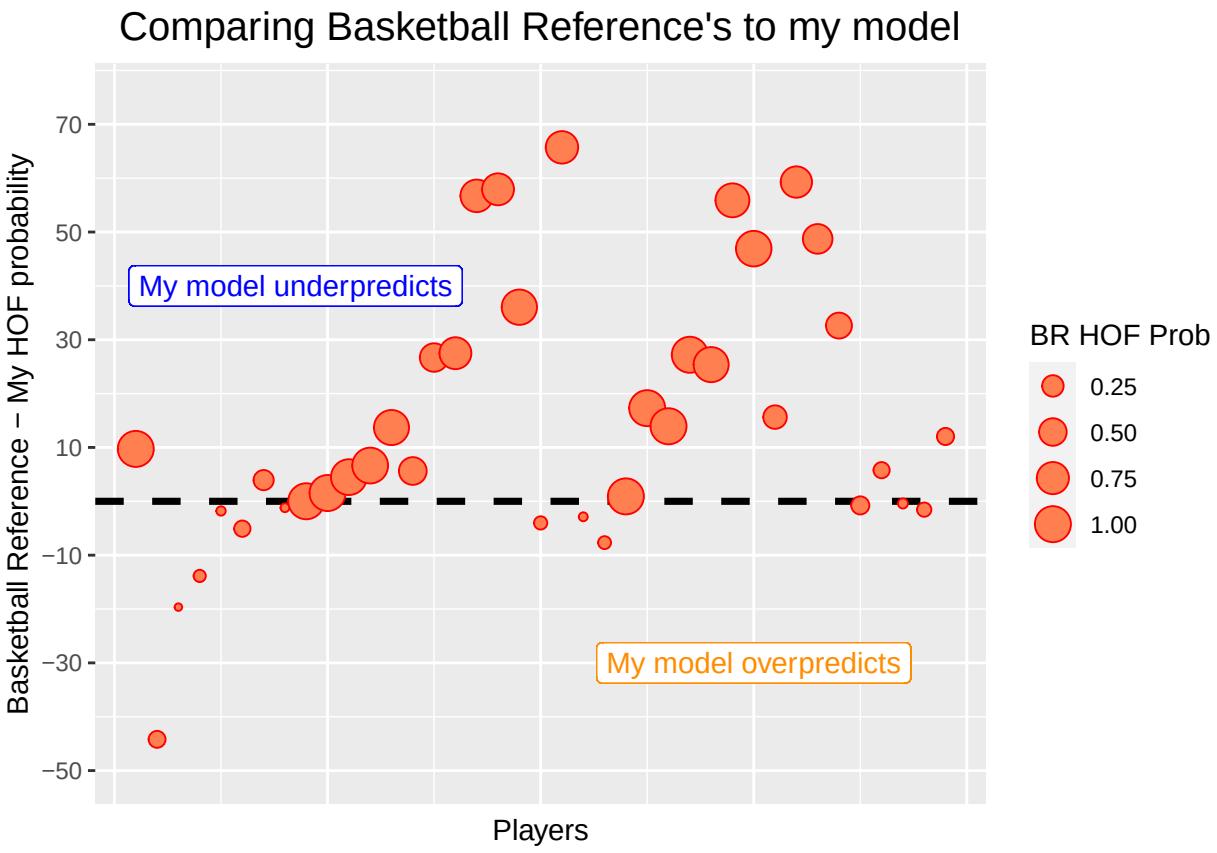
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction HOF Not HOF
##    HOF      12       1
##    Not HOF  10      16
##
##                Accuracy : 0.7179
##                  95% CI : (0.5513, 0.85)
##     No Information Rate : 0.5641
##     P-Value [Acc > NIR] : 0.03588
##
##                   Kappa : 0.459
##
##  Mcnemar's Test P-Value : 0.01586
##
##             Sensitivity : 0.5455
##             Specificity : 0.9412
##          Pos Pred Value : 0.9231
##          Neg Pred Value : 0.6154
##              Prevalence : 0.5641
##          Detection Rate : 0.3077
##    Detection Prevalence : 0.3333
##       Balanced Accuracy : 0.7433
##
##        'Positive' Class : HOF
##
```

```r
mosaic_table <- table(active_classes, br_hof_classes)

mosaicplot(mosaic_table,
           main = "Comparing my model to Basketball Reference",
           xlab = "My Prediction",
           ylab = "Basketball References Prediction",
           color = "skyblue2",
           border = "blue")
```




```
row_num <- as.numeric(nrow(prediction_comparison_complete))

prediction_comparison_complete %>%
  ggplot() +
  geom_hline(
    yintercept = 0,
    linetype = 2,
    color = "black",
    linewidth = 1.25
  ) + # line where models agree
  geom_point(
    aes(
      x = 1:row_num,
      y = (br_hof_prob - hof_prob)*100, # subtracting the probs
      size = br_hof_prob
    ),
    fill = "coral",
    color = "red",
    pch = 21
  ) +
  labs(
    x = "Players",
    y = "Basketball Reference - My HOF probability",
    title = "Comparing Basketball Reference's to my model",
    size = "BR HOF Prob"
  ) + # better labels
```

```
scale_y_continuous(
  limits = c(-50, 75),
  breaks = seq(-50, 75, 20)
) + # better scale
geom_label(
  aes(x=30, y=-30,
      label="My model overpredicts"
      ),
  color="darkorange") +
geom_label(
  aes(x=8.5, y=40,
      label="My model underpredicts"
      ),
  color="blue") + # labeling the parts of the graph
theme(
  plot.title = element_text(size = 15, hjust = 0.5),
  axis.text.x=element_blank(),
  axis.ticks.x=element_blank()
) # centering the title and removing x axis labels
```



Discussion of Active player model

The model did not perform as well on the testing data, if we consider BR's model as the reference. The model only had an accuracy of 71.79% with a Kappa of 0.459. The model did an especially poor job at classifying

which players made the HOF with a sensitivity of 54.55%. The model was systematically underrating the performance of players BR considers as HOF. However, the specificity of the model was very high at 94.12% meaning the model was very good at picking the players who did not make the hall of fame. These results make sense, as the model was trained on retired players who played an entire career. Many of the players in the model are still early in their career and have yet to accumulate all of the all star appearances the eventually will. My model underrates the HOF probability of active players because it relies on an entire career's worth of data. BR's model uses estimates of a player's peak performance and projects the career trajectory of active players to generate a more accurate probability for a player's HOF candidacy.