Final Project

Code ▾

Ethan Tran

2024-12-03

Introduction

The aim of this project is to build a machine learning model that can predict whether or not an NBA player will make the Hall of Fame based on the stats and accolades that they have accumulated throughout their careers. I will be using data from Basketball Reference (https://www.basketball-reference.com), particularly the Stathead Basketball (https://stathead.com/basketball/) database, and implementing multiple techniques to yield the most accurate model for this binary classification problem.

What is the Basketball Hall of Fame?

Only the most prestigious accolade a basketball player can hope to achieve one day. The NBA was founded on June 6th in 1946, and since then only 172 players have accomplished an induction, with about 30 of these inductions a result of post career coaching or off the court contributions to the sport of basketball. This means that about only 2 players are inducted per year, and there have been numerous years where no new players were added to the Hall of Fame. To make this prestigious list of individuals, one must have had an undeniable presence in the sport of basketball and in some capacity have forever stamped their impact on its history. Not only is it an incredible achievement, but the process is extremely slow, with players only being eligible for nomination 5 years after retirement. There is no set criteria for making the Hall of Fame, which can make it difficult to assess whether or not a player has accomplished enough throughout their career to be inducted.

Inspiration

I’ve been watching the NBA with my dad for as long as I can remember, and I’ve seen hundreds of phenomenal players come and go. But I can only name a handful of players that will make the Hall of Fame with certainty. It may be decades before some of my favorite players over the years finally get nominated, if ever. My goal with this model is to be able to accurately predict whether or not an NBA player had a Hall of Fame worthy career based on their total stats and accolades, so that long time fans like myself won’t have to wait years in anticipation.

Data Description

I have created the csv file for the dataset I will be using myself, using statistics recorded on Stathead Basketball as well as the official NBA website (https://www.nba.com/stats). The dataset contains the total regular season stats of all players that began their careers from between the 1976-77 season and the 2002-03 season, and have accumulated over 5000 career points. It is sorted by total points, descending. The reason for these filters is that the NBA merged with the ABA in 1976, and the league became much more competitive as a result. Statistics began to be recorded more rigorously, and it was one of the first seasons where steals and blocks were recorded, which will be used as predictors in my model as they are key statistics in analyzing defensive prowess. Prior to the merger, the NBA was a smaller league, and the criteria for making the Hall of Fame were not nearly as extreme as they are today. I set the minimum points requirement to be 5,000 as this is a low amount, and players with less than this amount of points are not in the conversation of potentially making the Hall of Fame. This means that the 4 Hall of Famers Arvydas Sabonis, Sarunas, Marciulionis, Drazen Petrovic, and Dino Radja were filtered out from the dataset. Their induction to the Hall of Fame was for their historical significance, as they were amongst the first stars from an overseas basketball league to join the NBA. They are special cases which will never arise again, as the NBA has since become an international league containing the best players from all over the world, so I’ve chosen to leave them out of the dataset as they contain significantly lower stats than any of the other Hall of Fame players. Finally, I chose the 2002-03 as the latest NBA debut for a player to be included in my dataset as this was the latest year in which there were no current NBA players in the league. This is important as I do not want current or recently retired players in my dataset since there will be undisputed Hall of Famers amongst this criteria that are not eligible for nomination yet. This also allows there to have been an apt amount of time for Hall of Fame worthy players in the dataset to have received their nomination.

There is no missing data in this dataset as a result.

Loading and Exploring the Data

Hide

```
#Loading the data
NbaStats <- read_csv("C:/Users/Ethan Tran/PSTAT131/Final Project/NbaDataset.csv")
#Renaming some variables for easier application
NbaStats <- NbaStats %>%
  rename(FGP = `FG%`, eFGP = `eFG%`, TSP = `TS%`)
#Factorize the response variable
NbaStats$HOF <- as.factor(NbaStats$HOF)
```

First, since I want the numerical values for my response variable HOF (0 or 1) to be of categorical type, I will factoize it. Next, let’s take a look at the dimensions of the dataset.

Hide

```
dim(NbaStats)
```

```
## [1] 440  38
```

Next, lets take a peek at the first 10 rows of the dataset.

Hide

```
head(NbaStats)
```

```
## # A tibble: 6 × 38
##      Rk Player HOF     MVP  FSTM FSTMD  DPOY   YRS   PTS From  To    Age       G
##   <dbl> <chr>  <fct> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <chr> <chr> <dbl>
## 1     1 Karl … 1         2    11     3     0    18 36928 1985… 2003… 22-40  1476
## 2     2 Kobe … 1         1    11     9     0    19 33643 1996… 2015… 18-37  1346
## 3     3 Micha… 1         5    10     9     1    18 32292 1984… 2002… 21-39  1072
## 4     4 Dirk … 1         1     4     0     0    20 31560 1998… 2018… 20-40  1522
## 5     5 Shaqu… 1         1     8     0     0    18 28596 1992… 2010… 20-38  1207
## 6     6 Hakee… 1         1     6     5     2    17 26946 1984… 2001… 22-39  1238
## # ℹ 25 more variables: GS <dbl>, AS <dbl>, MP <dbl>, FG <dbl>, FGA <dbl>,
## #   `2P` <dbl>, `2PA` <dbl>, `3P` <dbl>, `3PA` <dbl>, FT <dbl>, FTA <dbl>,
## #   ORB <dbl>, DRB <dbl>, TRB <dbl>, AST <dbl>, STL <dbl>, BLK <dbl>,
## #   TOV <dbl>, PF <dbl>, FGP <dbl>, `2P%` <dbl>, `3P%` <dbl>, `FT%` <dbl>,
## #   TSP <dbl>, eFGP <dbl>
```

Variables and Codebook

Selecting variables to keep out of the 33 available was a fairly easy process because I have such a strong familiarity with the NBA and what are important criteria to making the Hall of Fame. In my model, I will be using the following:

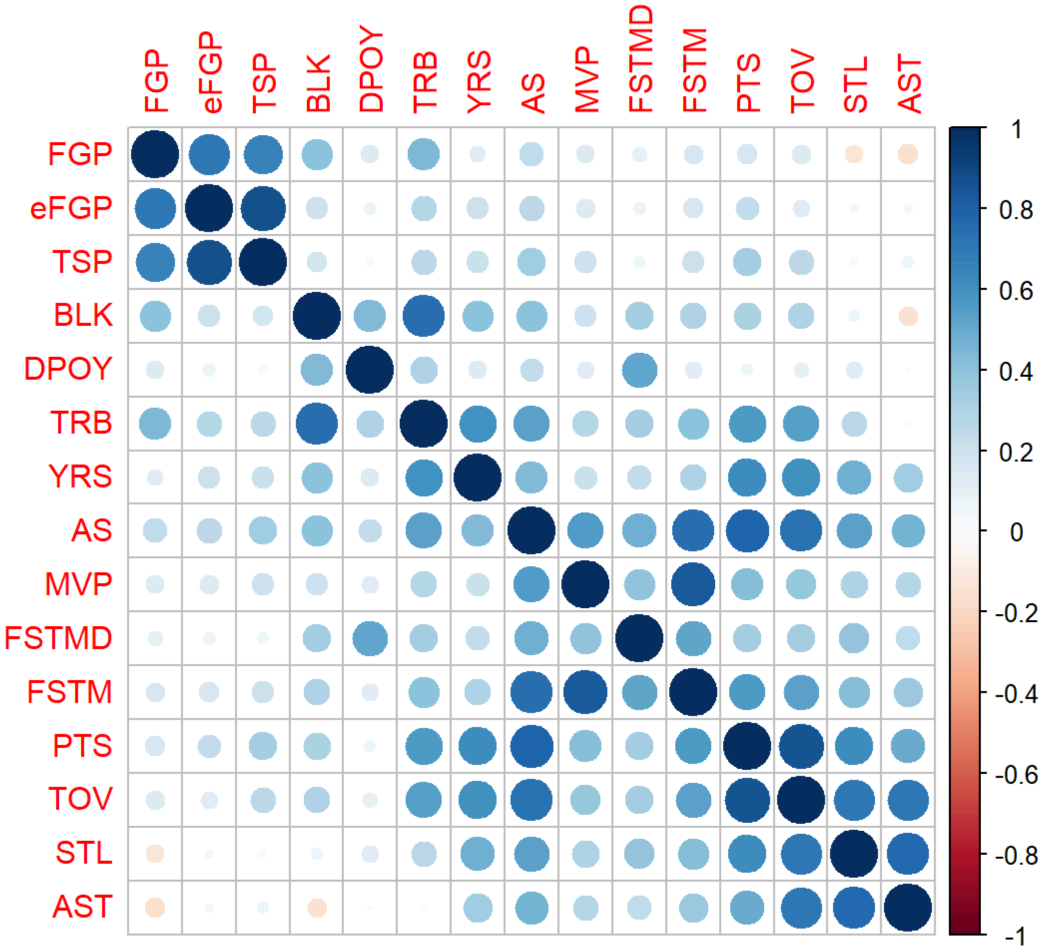
HOF: Gives 1 if a player is in the hall of fame, and 0 if they are not (this is what my model will be predicting). MVP: The number of MVP (Most Valuable Player) awards a player has received FSTM: The number of all NBA first teams a player has made. This is a stat that basically says the number of years a player was the best at their position in the league. FSTMD: The number of all NBA defense first teams a player has made. This is a stat that basically says the number of years a player was the best defender at their position in the league. DPOY: The number of DPOY (Defensive Player of the Year) awards a player has received YRS: The amount of years a player has played in the NBA PTS: The total number of points a player has AS: The total number of All Star appearances a player has TRB: The total number of rebounds a player has AST: The total number of assists a player has STL: The total number of steals a player has BLK: The total number of blocks a player has TOV: The total number of turnovers a player has FGP: A player’s field goal percentage (shots made divided by shots taken) TSP: A player’s true shooting field goal percentage (half the points scored divided by the sum of the field goals attempted and 0.475 times the free throws attempted) This is an advanced statistic that details a player’s efficiency while accounting for free throws and three pointers eFGP: A player’s effective field goal percentage (the sum of field goals made and half the three pointers made divided by field goals attempted) This is an advanced statistic that details a player’s efficiency while accounting for three pointers

Visual Exploratory Data Analysis

The following section will contain plots and visuals that will help us to get a better understanding of the correlation between our predictors, how the predictors affect our response variable, as well as the distribution of our response variable.

Hide

```
#Getting the correlation matrix
cor_mat_vars <- NbaStats[, c("MVP", "FSTM", "FSTMD", "DPOY", "YRS", "PTS", "AS", "TRB", "AST", "STL", "BLK", "TOV", "FGP", "TSP", "eFGP")]
cor_mat <- cor(cor_mat_vars)
#Plotting the correlation matrix
NbaCorMat <- corrplot(cor_mat,
                      order='AOE')
```

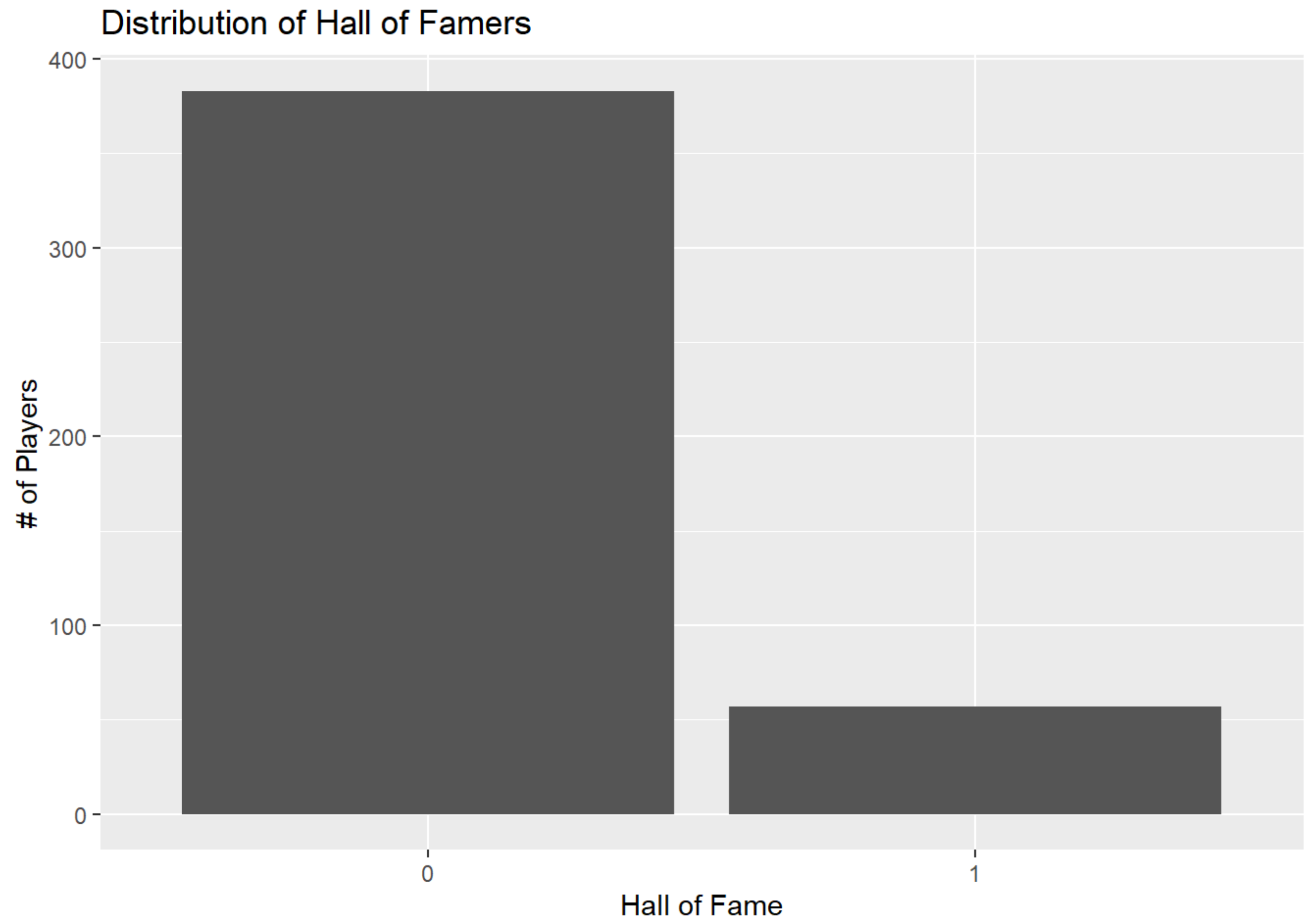


The relationships between the predictor variables make sense. The positive correlation between the three efficiency variables (FGP, eFGP, and TS) make sense. The slight negative correlation between FGP and steals and assists may be because steals and assists are stats that are typically accumulated by guards, which is a position that on average takes more difficult shots than the other positions and has lower field goal percentage as a result. All of the relationships align with intuitive thinking. The predictors that have no correlation such as DPOY and TRB with assists make sense as being the best defender in the league should not affect playmaking or rebounding ability.

How Many Hall of Famers Are in the Dataset

Hide

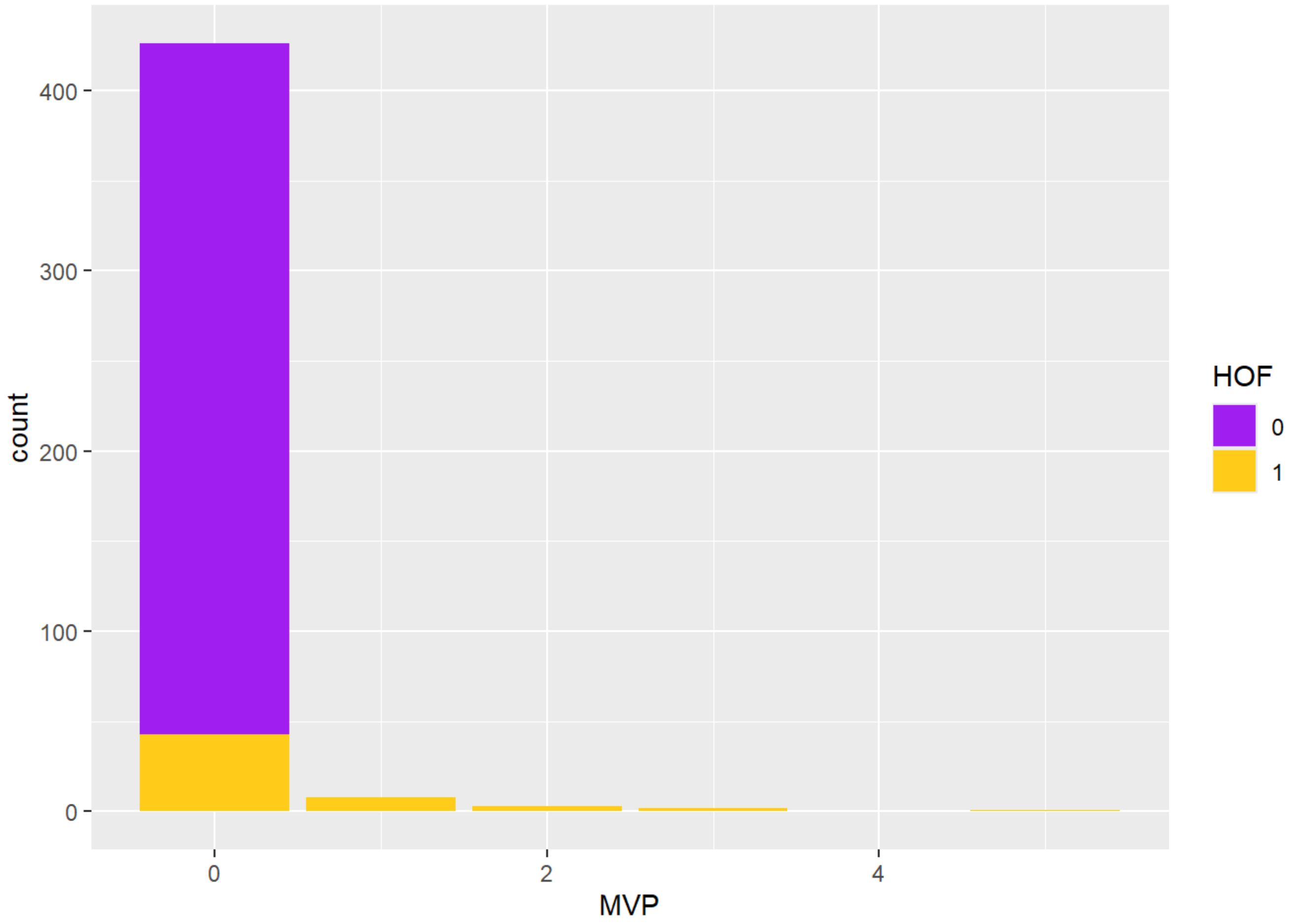
```
NbaStats %>%
  ggplot(aes(x = HOF)) +
  geom_bar() +
  labs(x = 'Hall of Fame', y = '# of Players', title = 'Distribution of Hall of Famers')
```



Of the 440 players in the dataset, around 60 of them are Hall of Famers. That means the remaining players, approximately 380, are not Hall of Famers. Around 14% of the players are Hall of Famers.

Hide

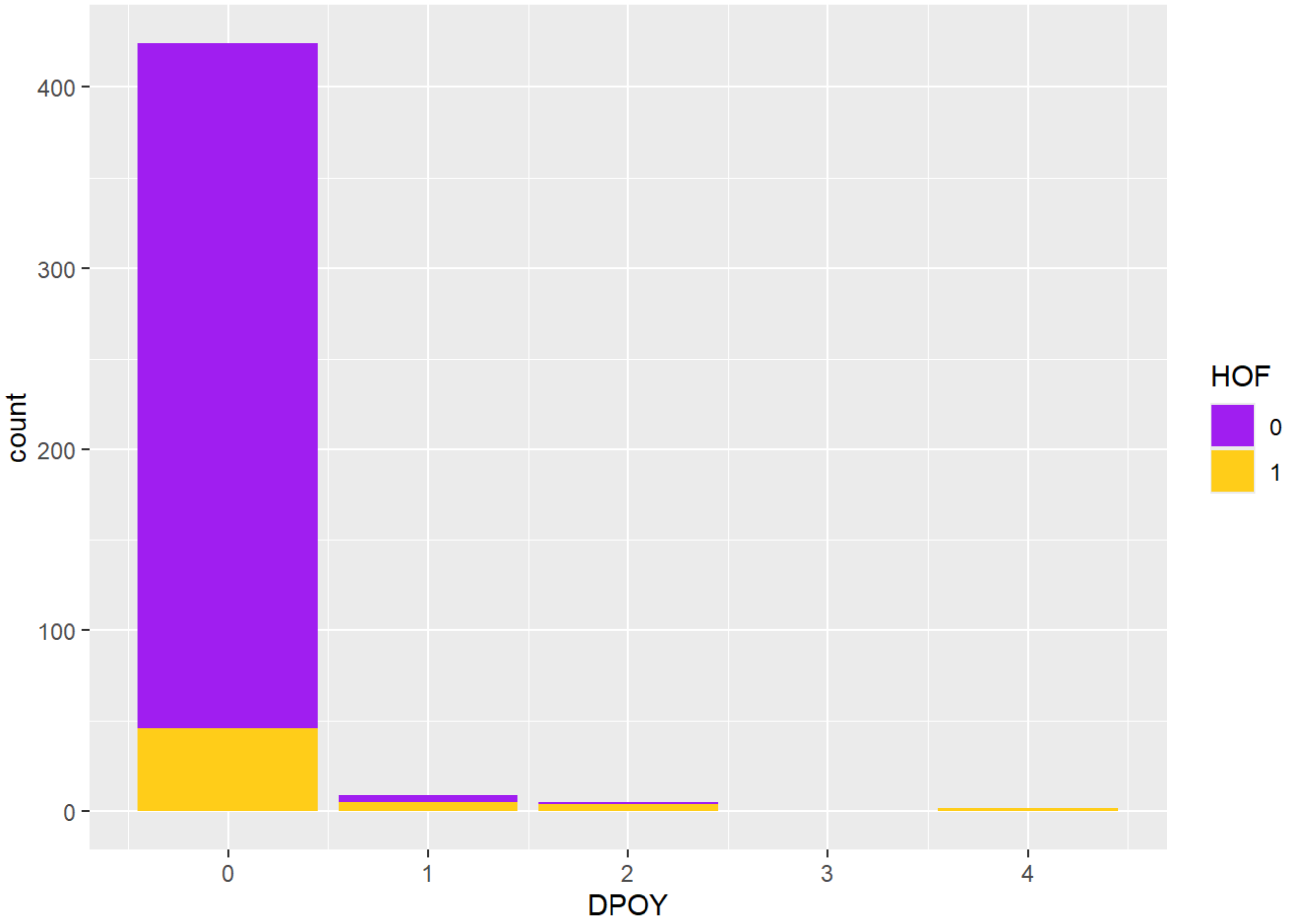
```
ggplot(NbaStats, aes(MVP)) +
  geom_bar(aes(fill = HOF)) +
  scale_fill_manual(values=c("purple", "#FFCE1B"))
```



Every MVP in the NBA’s history has made the Hall of Fame, as highlighted by the plot. Although receiving the award isn’t required to make it, it greatly increases one’s odds. This will be important later as Derrick Rose, the youngest MVP in league history, was injured shortly after winning the 2011 MVP. This means he did not accumulate that many stats in his career, so it will be interesting to see what the model predicts for this specific player.

Hide

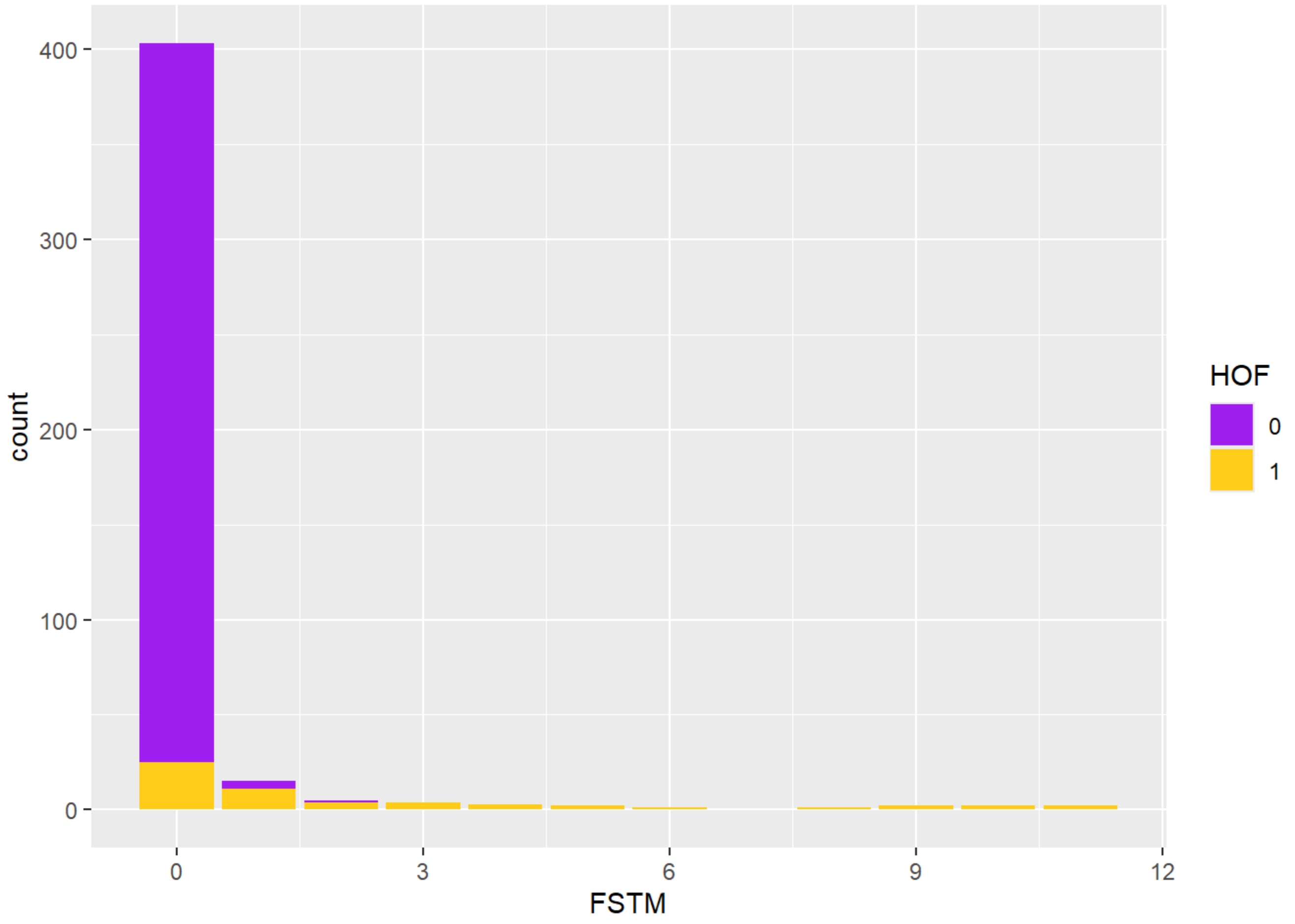
```
ggplot(NbaStats, aes(DPOY)) +
  geom_bar(aes(fill = HOF)) +
  scale_fill_manual(values=c("purple", "#FFCE1B"))
```



Every player with 3 or more Defensive Player of the Year awards has made it to the Hall of Fame. Winning the award doesn't guarantee an induction, but it does greatly increase one's odds.

Hide

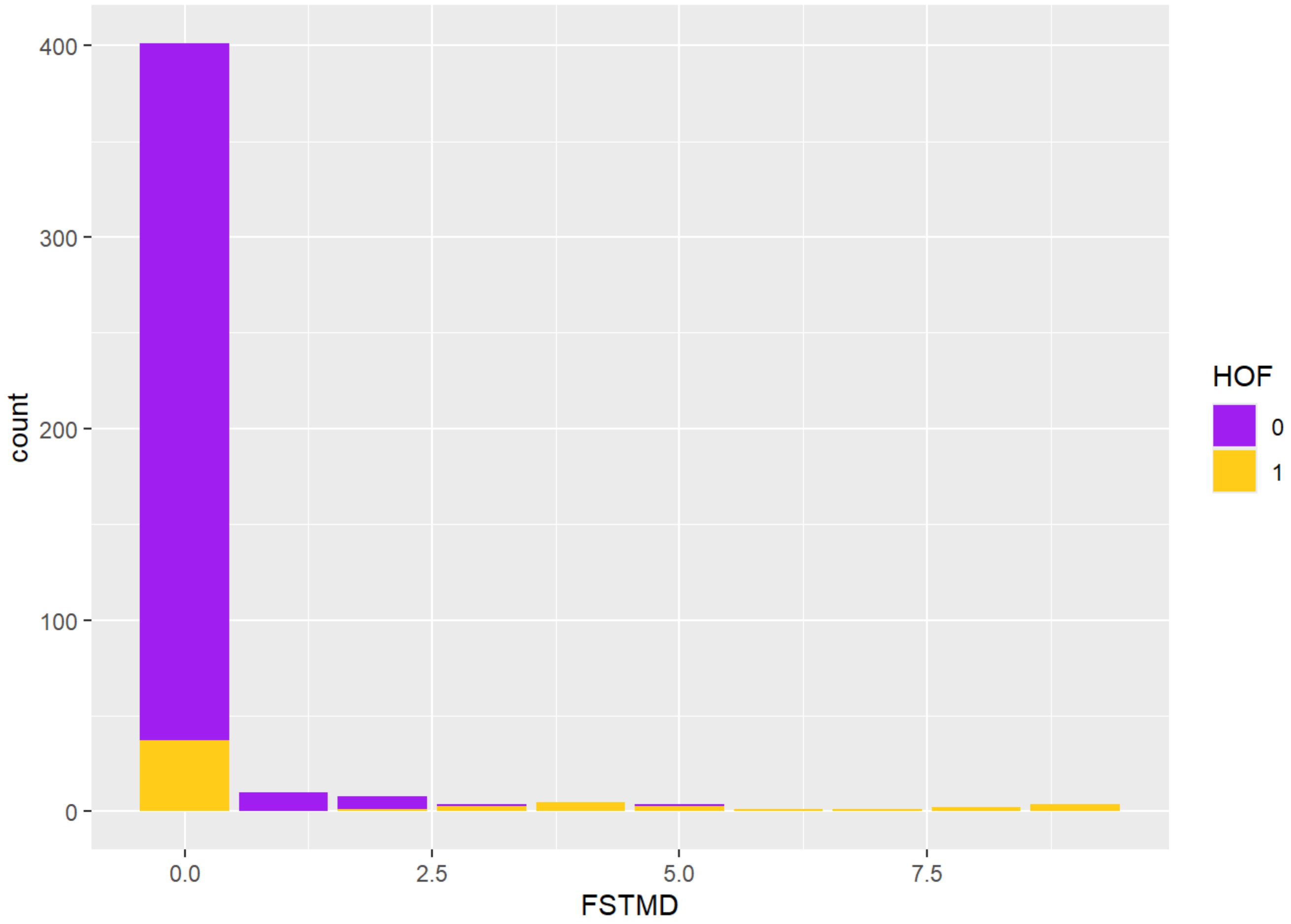
```
ggplot(NbaStats, aes(FSTM)) +
  geom_bar(aes(fill = HOF)) +
  scale_fill_manual(values=c("purple", "#FFCE1B"))
```



The more first team all NBA nominations a player has, the more likely they are to to make the Hall of Fame. Every player with 3 or more has been inducted. This makes sense as having 3 or more years in one’s career as the best player at a position is an accolade only the best of the best have.

Hide

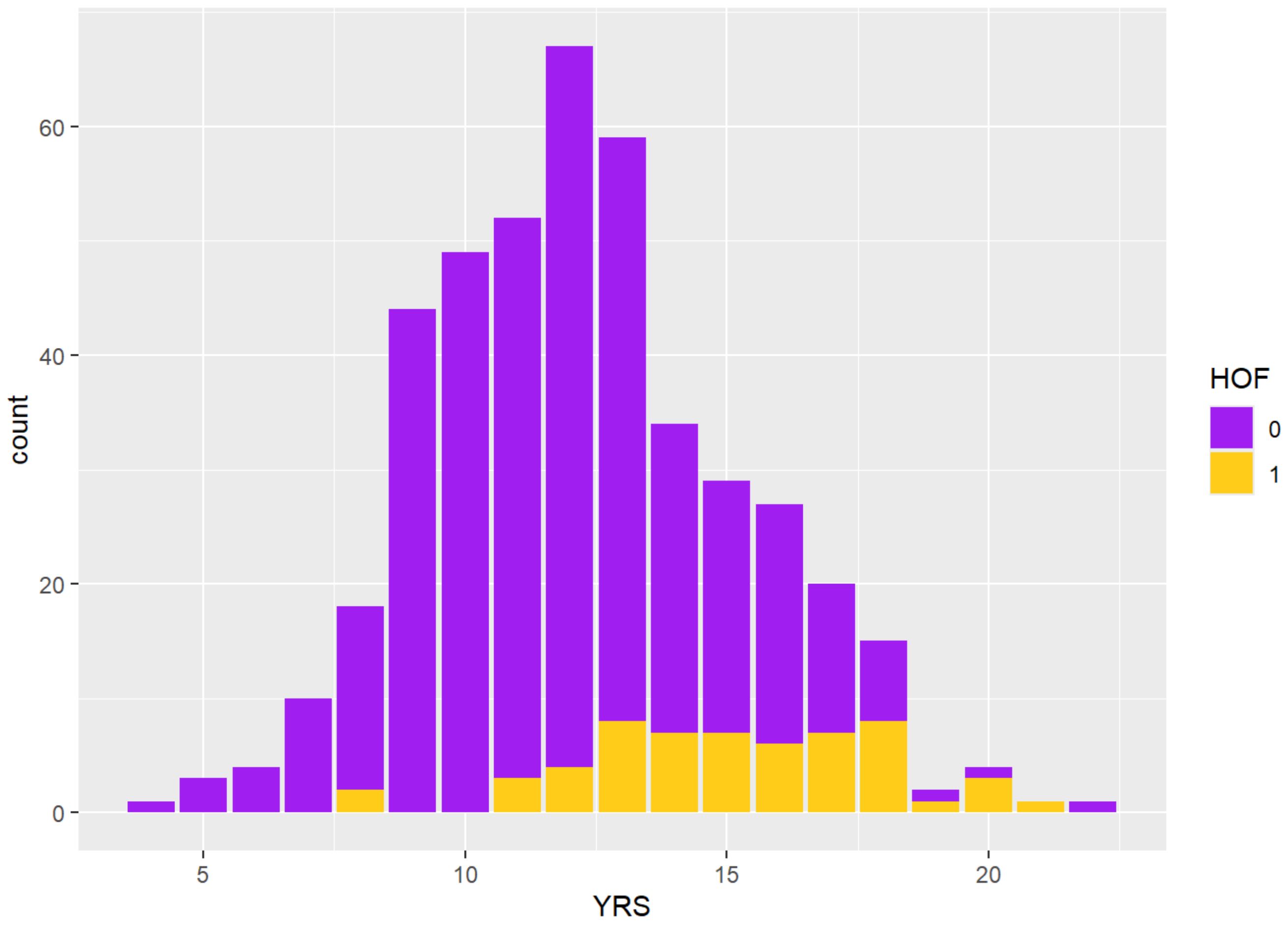
```
ggplot(NbaStats, aes(FSTMD)) +
  geom_bar(aes(fill = HOF)) +
  scale_fill_manual(values=c("purple", "#FFCE1B"))
```



Having 3 or more First Team All Defense selections greatly increases ones odds of induction, and all players with 6 or more are in the Hall of Fame. Having 2 or less doesn’t have an affect on Hall of Fame status, which is probably because this isn’t a large enough amount to indicate someone is an overall good player.

Hide

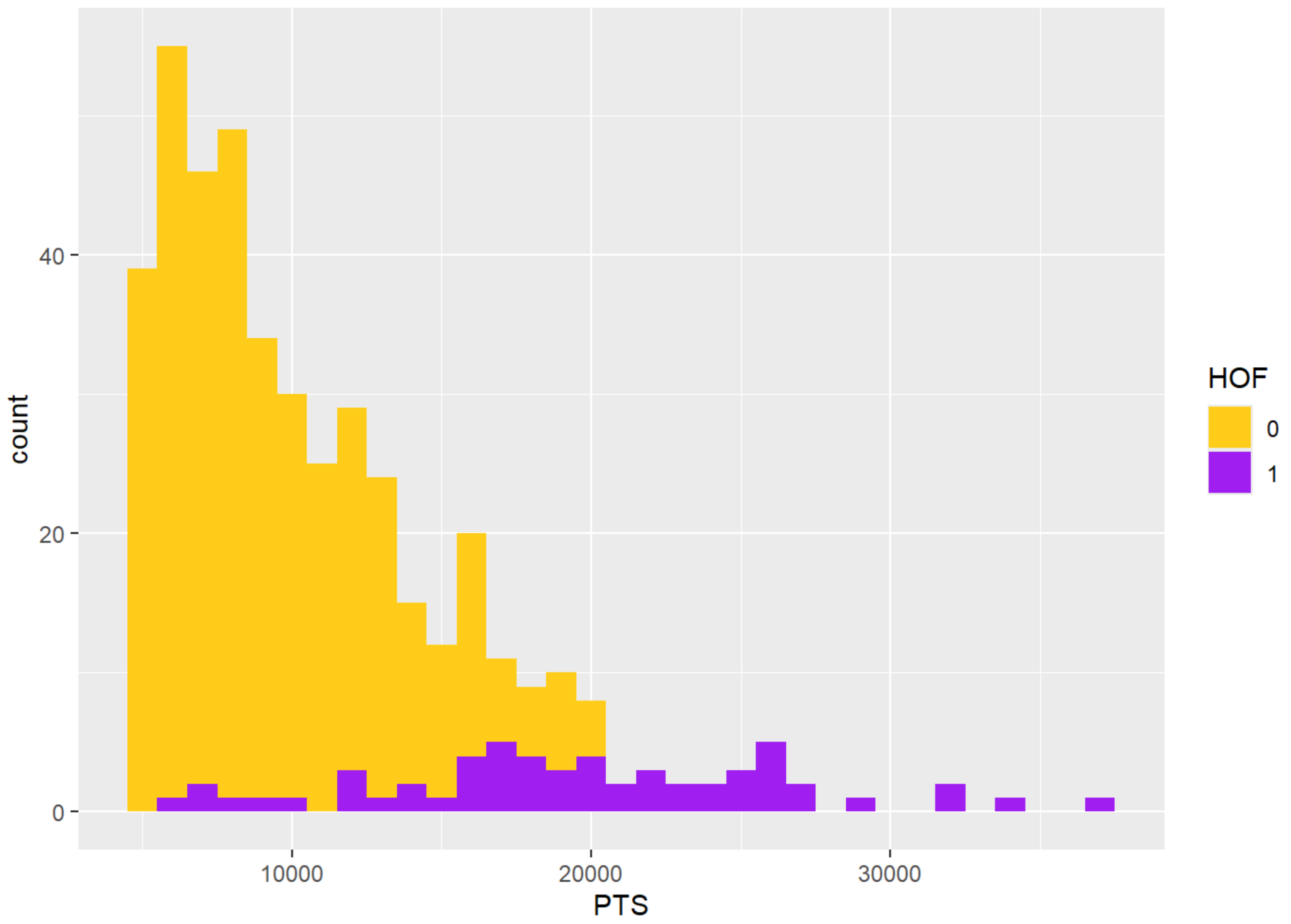
```
ggplot(NbaStats, aes(YRS)) +
  geom_bar(aes(fill = HOF)) +
  scale_fill_manual(values=c("purple", "#FFCE1B"))
```



The more years a player has played in the league, the higher their odds of making the Hall of Fame. This makes sense as the longer one’s career, the more stats and accolades they will accumulate. Moreover, being able to play in the NBA for a long amount of time is indicative of skill.

Hide

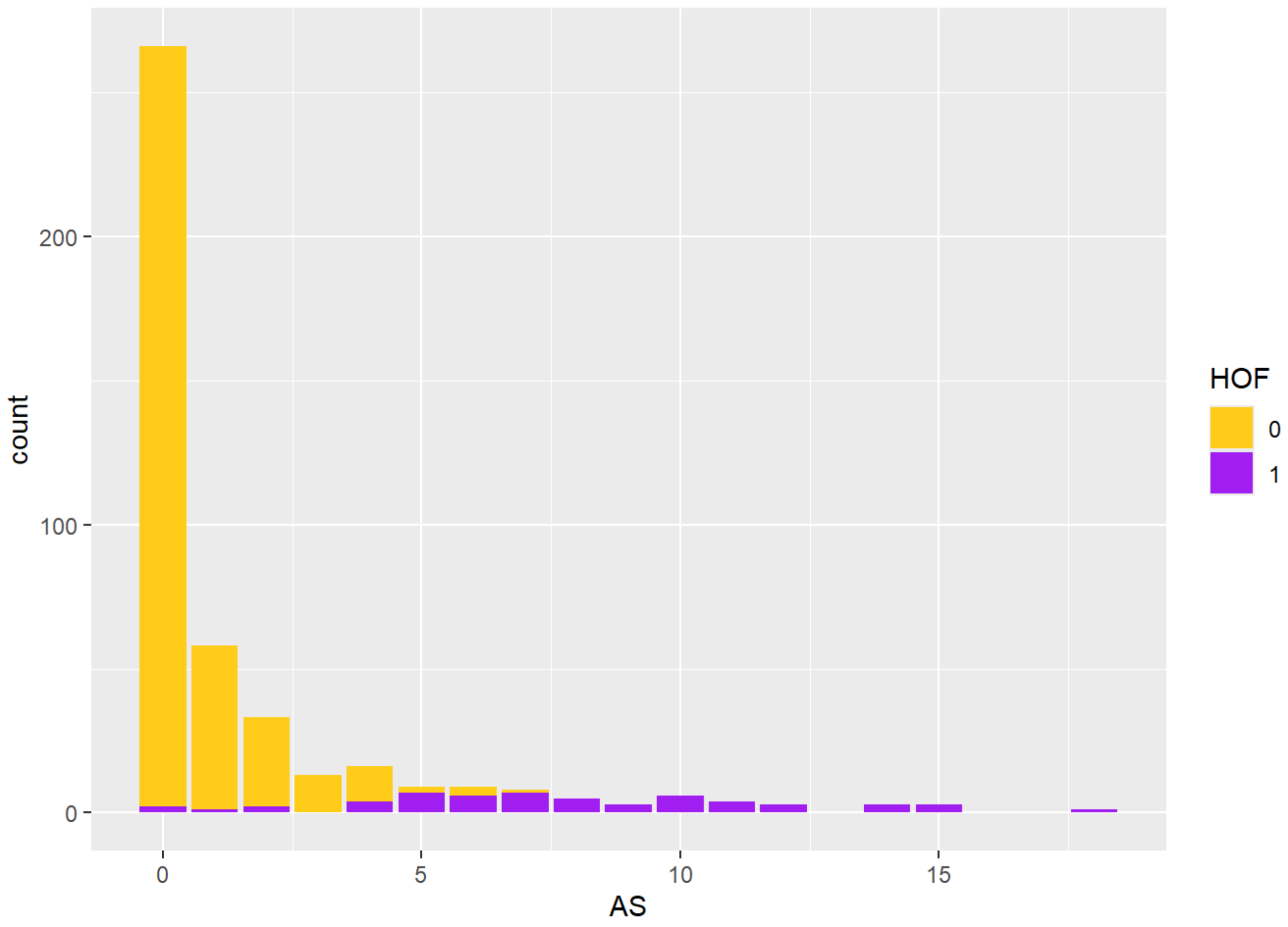
```
ggplot(NbaStats, aes(PTS)) +
  geom_histogram(binwidth = 1000, aes(fill = HOF)) +
           scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



As shown by the plot, having more points is beneficial to one’s chances of making the Hall of Fame. All eligible players with 21000 or more points have been inducted.

Hide

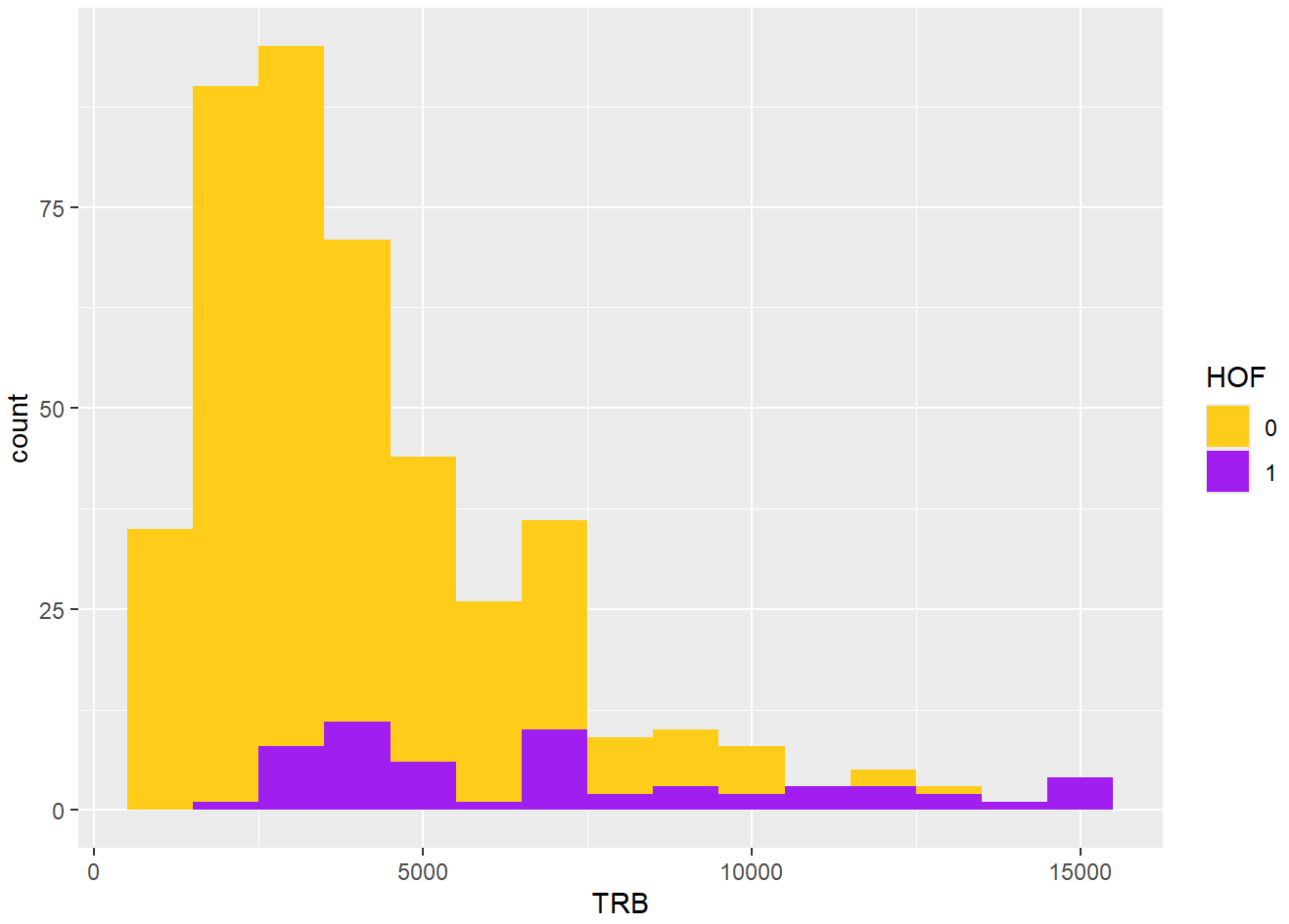
```
ggplot(NbaStats, aes(AS)) +
  geom_bar(aes(fill = HOF)) +
          scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



As shown by the plot, having more All Star appearances increases Hall of Fame likelihood, with all players with 8 or more All Star appearances inducted.

Hide

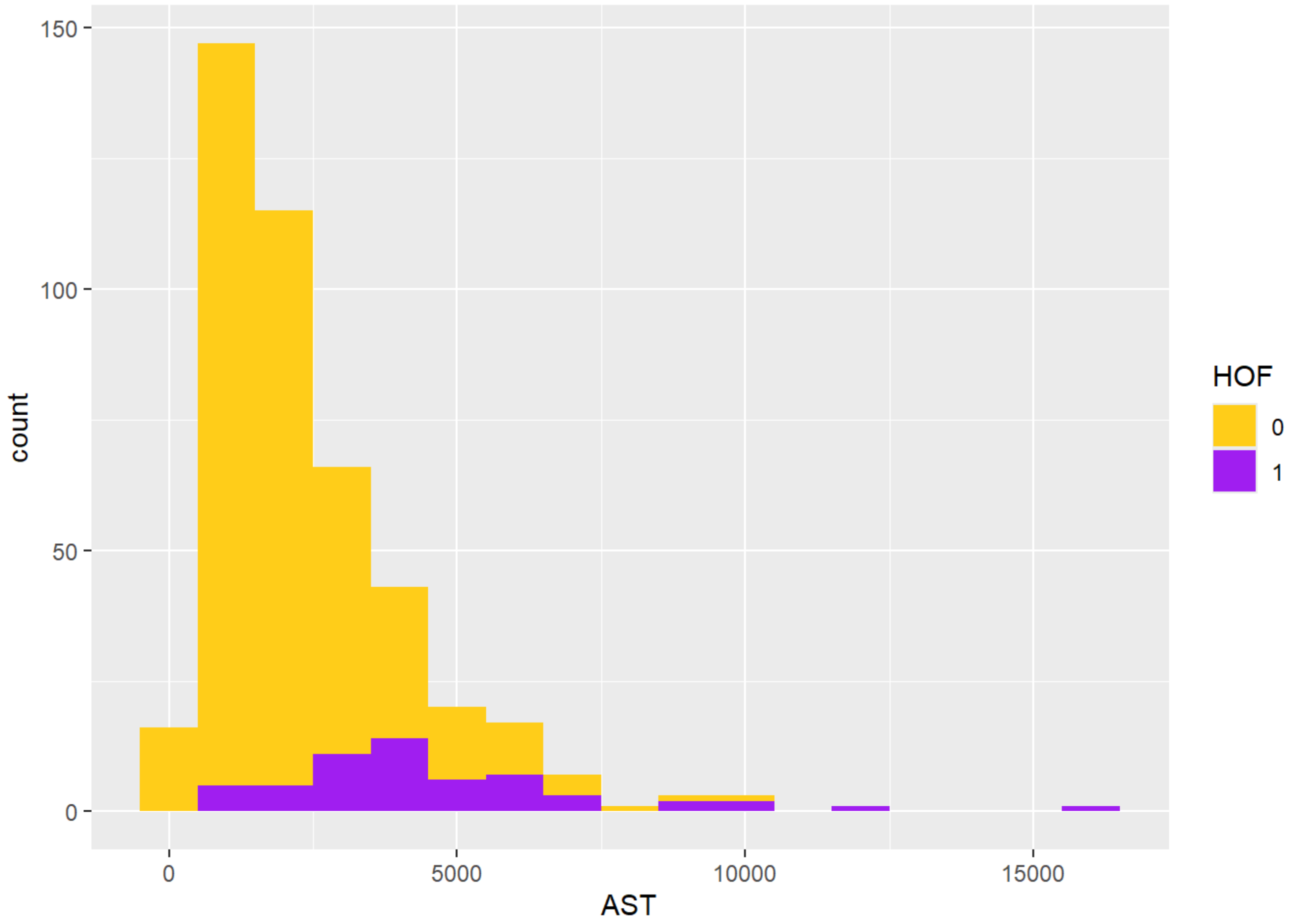
```
ggplot(NbaStats, aes(TRB)) +
  geom_histogram(binwidth = 1000, aes(fill = HOF)) +
          scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



There is a slight correlation between having more rebounds and making the Hall of Fame, which becomes much more evident after passing the 10000 career rebounds milestone.

Hide

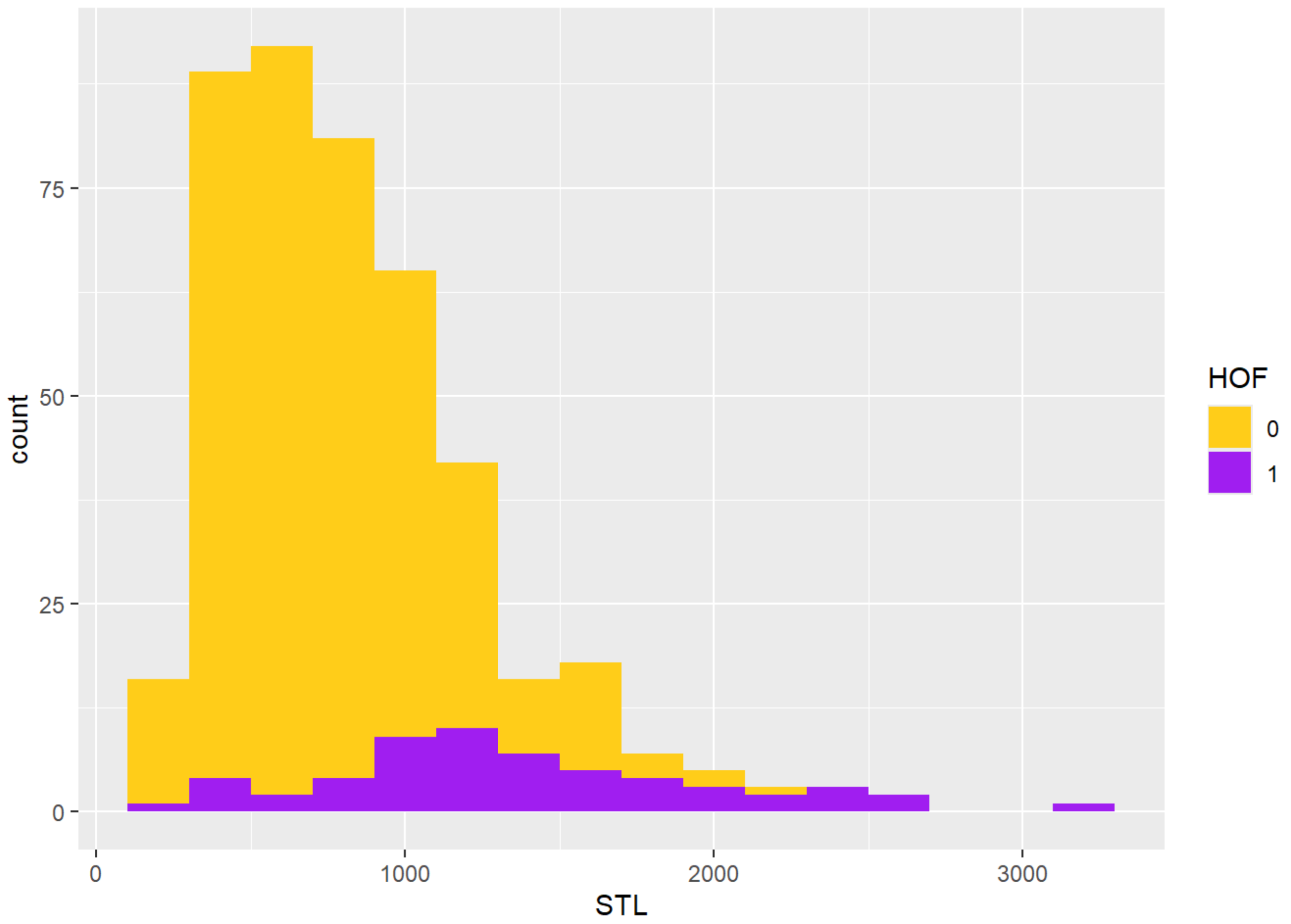
```
ggplot(NbaStats, aes(AST)) +
  geom_histogram(binwidth = 1000, aes(fill = HOF)) +
          scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



The more assists a player has, the greater their likelihood of making the Hall of Fame. The few players with over 11000 assists have all made the Hall of Fame.

Hide

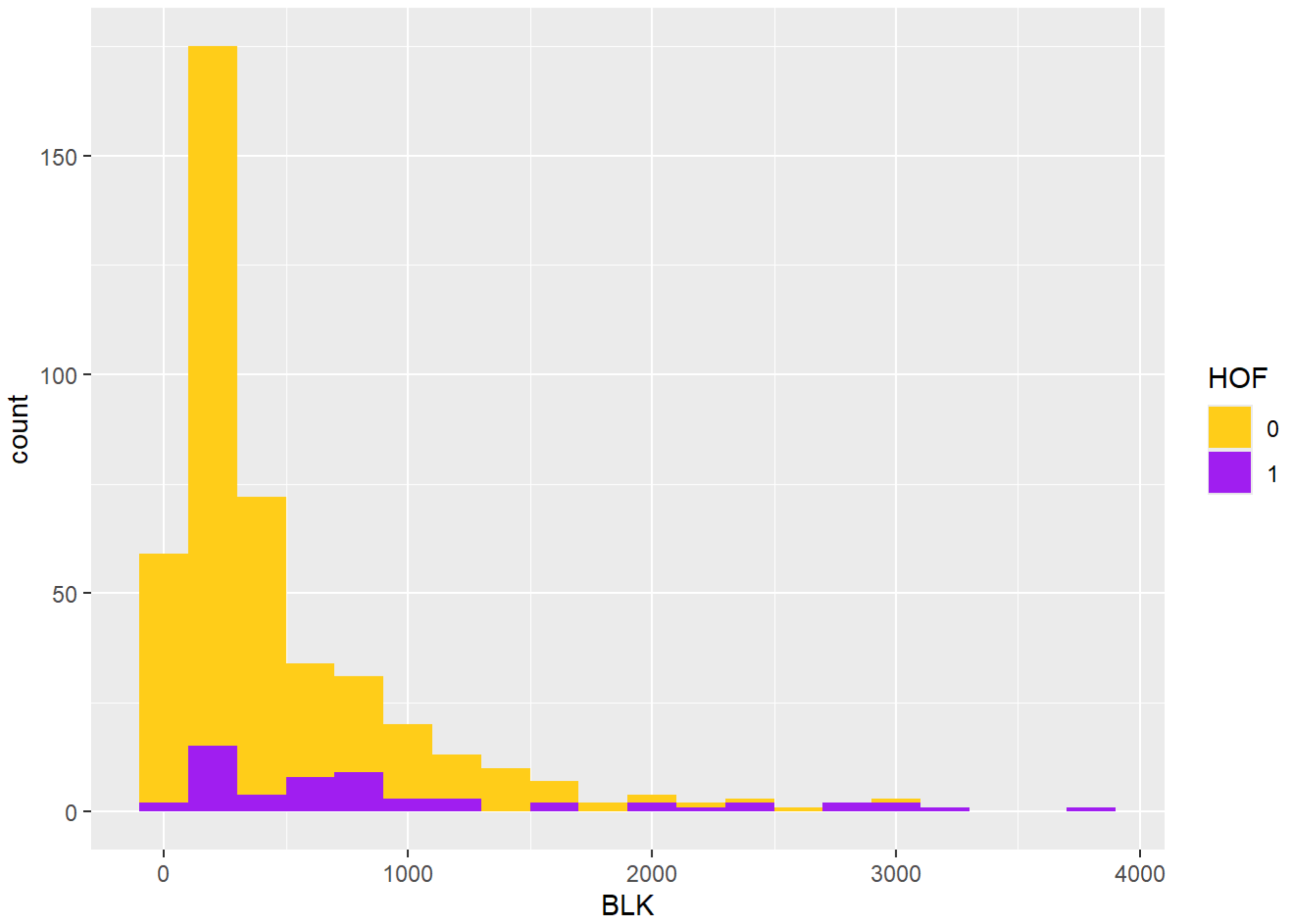
```
ggplot(NbaStats, aes(STL)) +
  geom_histogram(binwidth = 200, aes(fill = HOF)) +
          scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



The more steals a player has, typically the higher their chances of making the Hall of Fame. All players with 2200 or more steals have been inducted.

Hide

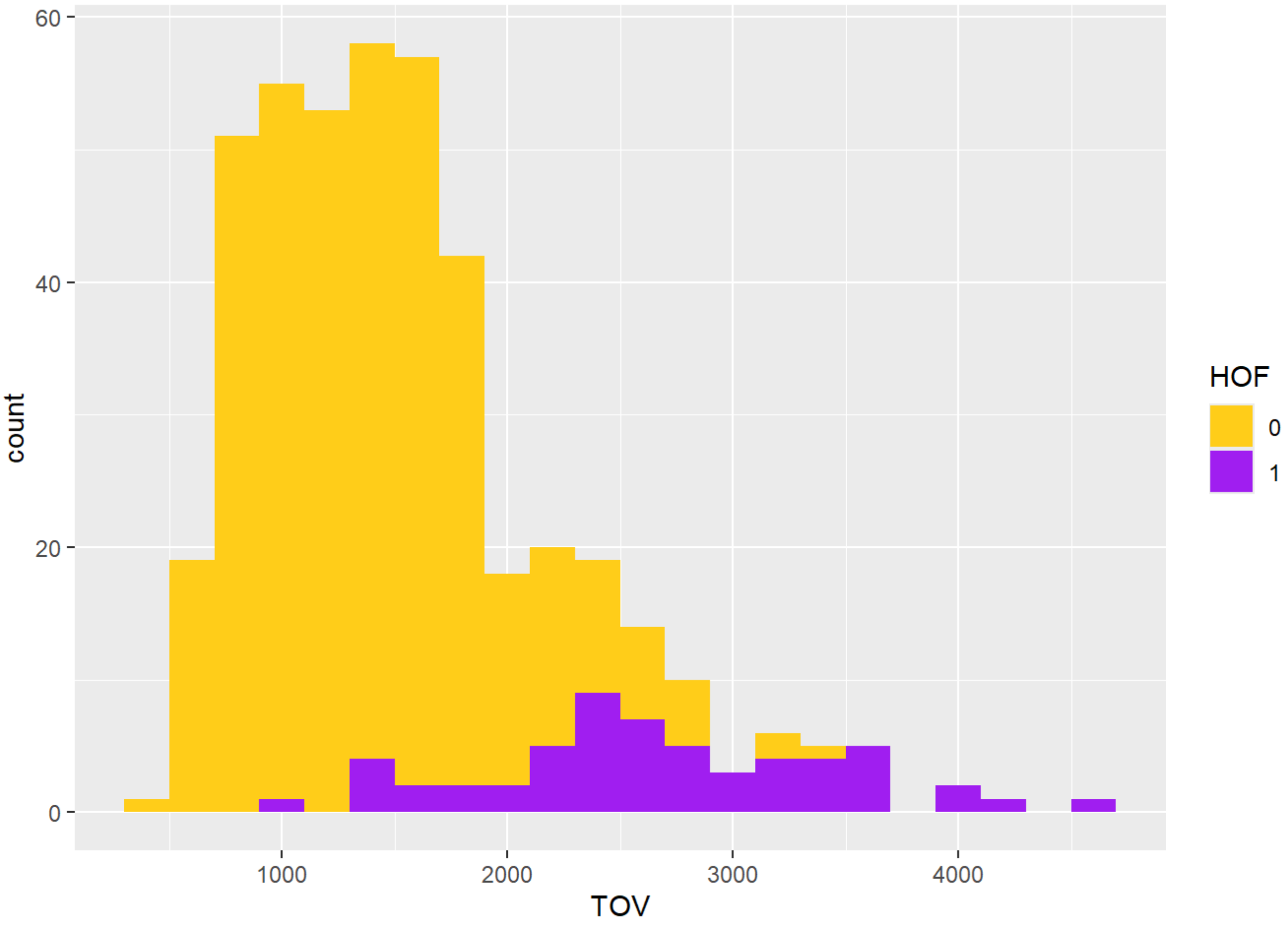
```
ggplot(NbaStats, aes(BLK)) +
  geom_histogram(binwidth = 200, aes(fill = HOF)) +
           scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



Having more blocks doesn’t correlate to Hall of Fame status until around the 2000 career blocks milestone. At this number, the more blocks one has, the higher their odds of making the Hall of Fame.

Hide

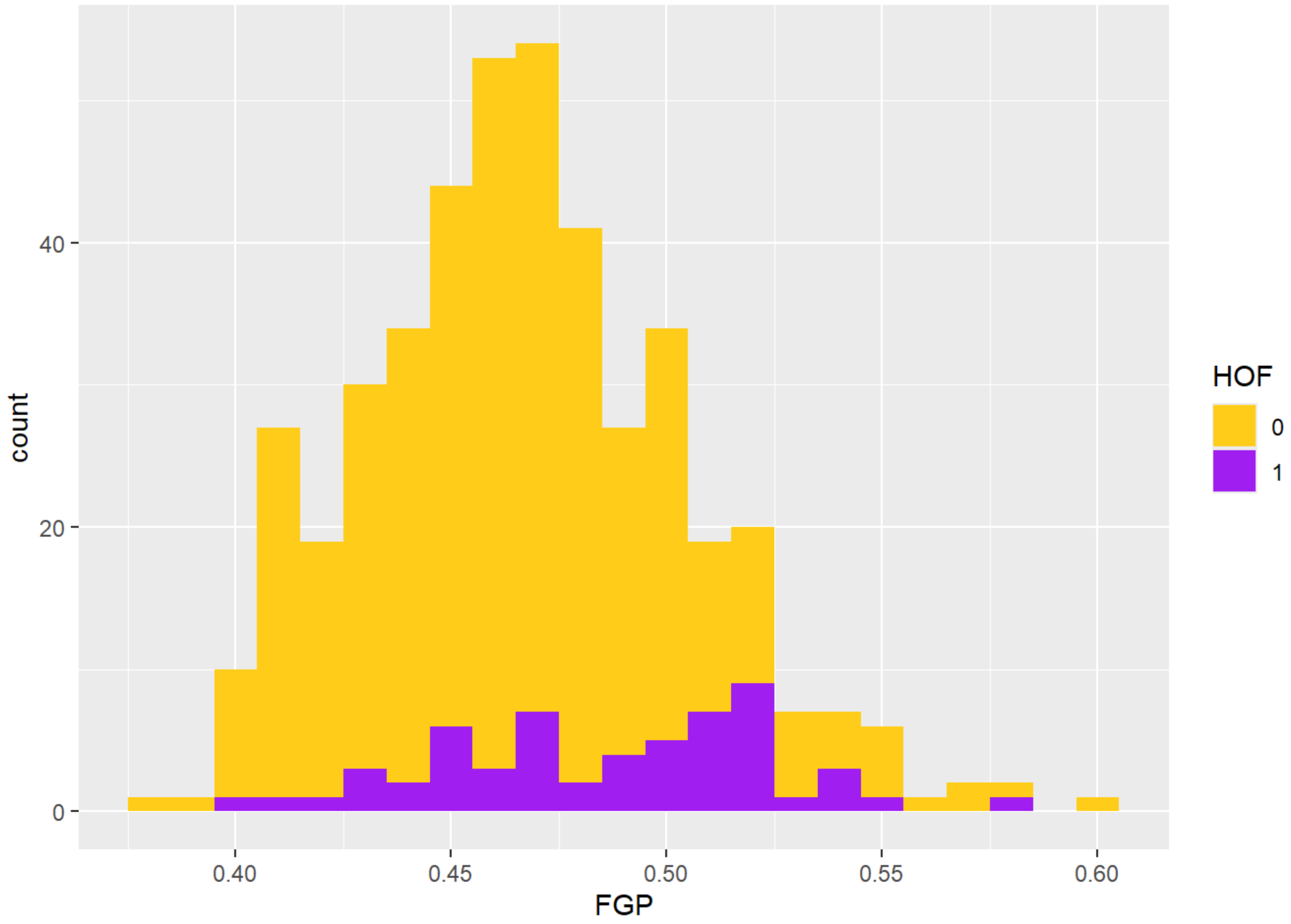
```
ggplot(NbaStats, aes(TOV)) +
  geom_histogram(binwidth = 200, aes(fill = HOF)) +
          scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



After the 2000 career turnovers milestone, the more turnovers a player has, the higher their likelihood of induction. All players with 4000 or more turnovers are in the the Hall of Fame. This may seem counterintuitive as turning the ball over hurts your team. However, a high amount of career turnovers indicates that the player handles the basketball more often throughout the course of the game, which is a right reserved only to the star players. Any player that turns the ball over often without having enough positive contributions to their team to justify this will not be in the NBA long enough to reach a high amount of career turnovers.

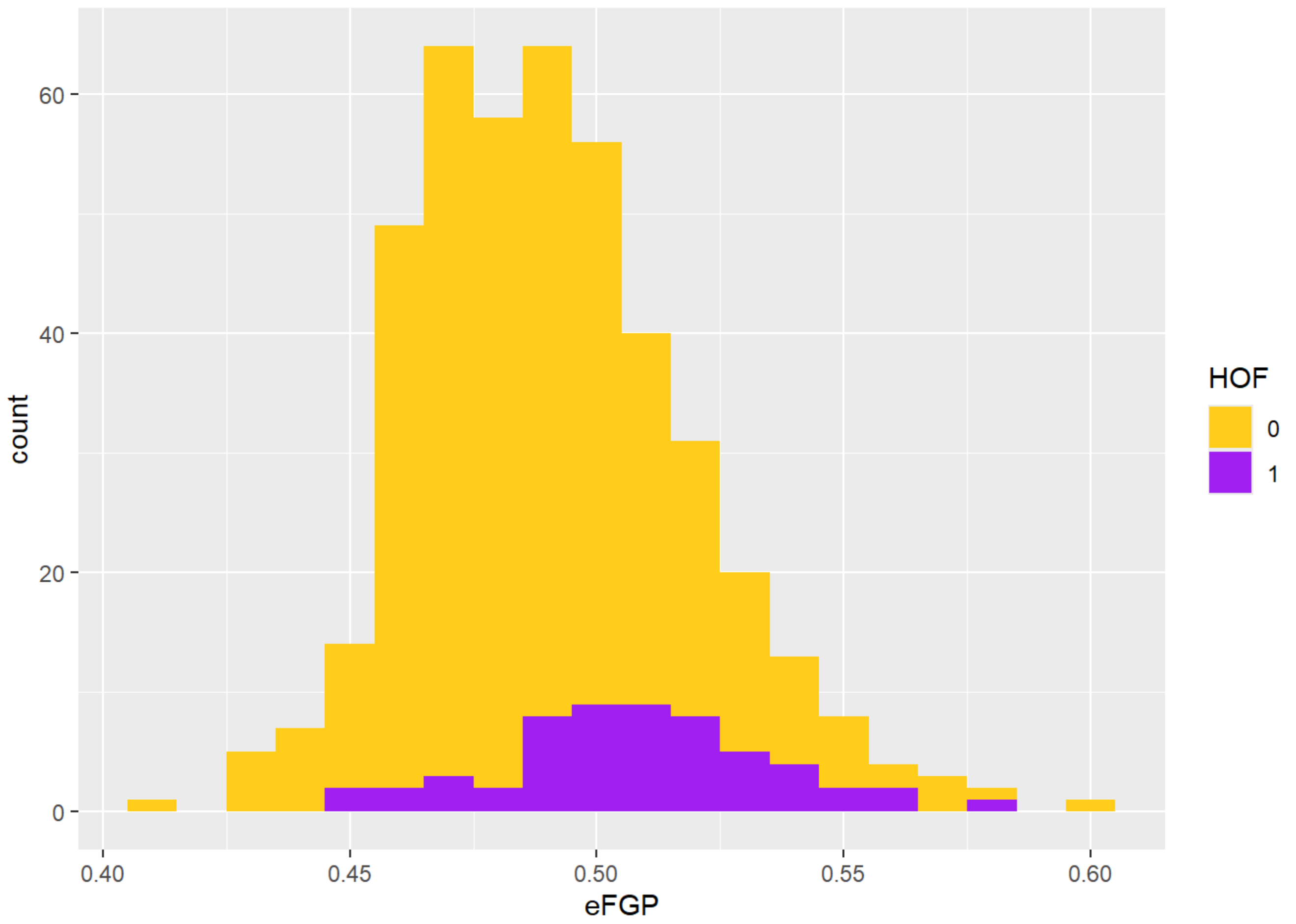
Hide

```
ggplot(NbaStats, aes(FGP)) +
  geom_histogram(binwidth = 0.01, aes(fill = HOF)) +
           scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



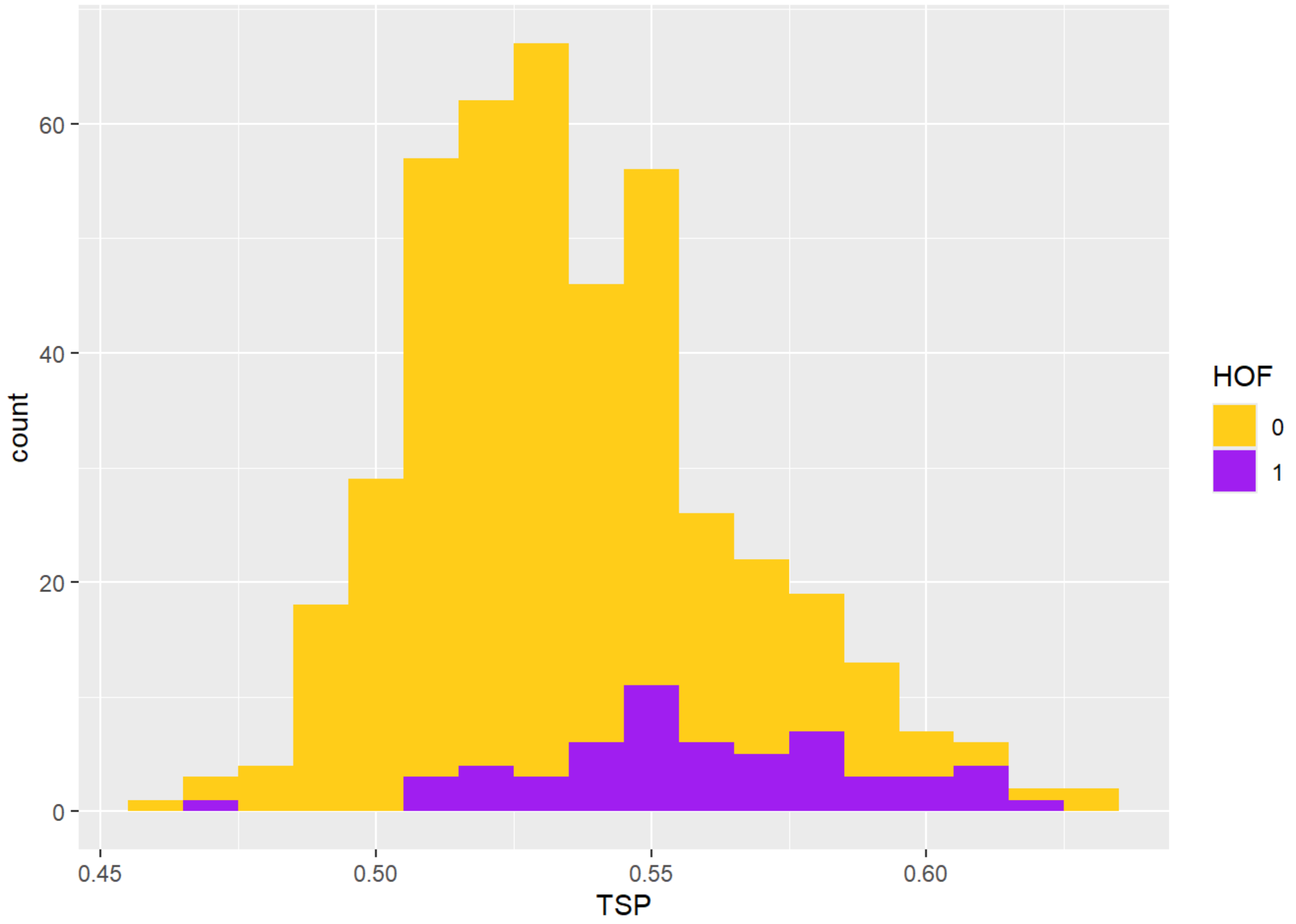
Hide

```
ggplot(NbaStats, aes(eFGP)) +
  geom_histogram(binwidth = 0.01, aes(fill = HOF)) +
          scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



Hide

```
ggplot(NbaStats, aes(TSP)) +
  geom_histogram(binwidth = 0.01, aes(fill = HOF)) +
          scale_fill_manual(values=c("#FFCE1B", "#A020F0"))
```



Contrary to my prior assumptions, it seems that the 3 player efficiency statistics (field goal percentage, effective field goal percentage, and true shooting percentage) do not have a large impact on Hall of Fame status. There are less Hall of Famers at the tail ends of the graph, though, which could indicate that a player that is too inefficient of a scorer has a lower chance of making the Hall of Fame. If a player is too efficient, this may be because they simply do not take many shots a game, which implies they are not a star player and will not make the Hall of Fame.

Setting Up Models

With a better understanding of the affect our predictors will have on a player making the Hall of Fame, we can begin building our models. We will randomly split our data into training and testing sets, create our recipe, and establish cross validation in our models.

Training/Testing Split

The first step to building our models and recipe will be to split the dataset into training and testing sets. The training set will be used to train our models, while the testing set will be reserved for the final steps when we will actually test the accuracy of our model. I decided to use a use a 70/30 split so that there were enough Hall of Famers in the testing set for reliable evaluation, and stratify on the response variable HOF. A random seed is set to ensure the training/testing split is the same every time.

Hide

```
set.seed(1200) #Setting seed so split is the same every time

NbaSplit <- initial_split(NbaStats, prop = 0.7, strata="HOF")
NbaTrain <- training(NbaSplit)
NbaTest <- testing(NbaSplit)
dim(NbaTrain)
```

```
## [1] 307  38
```

Hide

```
dim(NbaTest)
```

```
## [1] 133  38
```

Dimensions of the training set: 307 38 Dimensions of the testing set: 133 38

Building the Recipe

Of the 15 predictors, I will be using all of them except FGP (field goal percentage) and eFGP (effective field goal percentage) as TSP (true shooting percentage) encapsulates the same details that these variables do but in more detail. The predictors will be standardized.

Hide

```
hof_recipe <- recipe(HOF ~ MVP + DPOY + FSTM + FSTMD + AS + YRS + PTS + TRB + AST +
                           STL + BLK + TOV + TSP, data = NbaStats) %>%
  step_scale(all_predictors()) %>%
  step_center(all_predictors()) #Standardizing the Predictors
```

K-Fold Cross Validation

We will stratify our cross validation on the response variable, HOF, and use 10 folds.

Hide

```
NbaFolds <- vfold_cv(NbaTrain, v=10, strata=HOF)
```

The results will be saved to an RDA so they can be loaded anytime without any time commitment.

Hide

```
save(NbaFolds, NbaTrain, NbaTest, file = "C:/Users/Ethan Tran/PSTAT131/Final Project/RDA/n
ba_hof_model_setup.rda")
```

Prediction Model Building

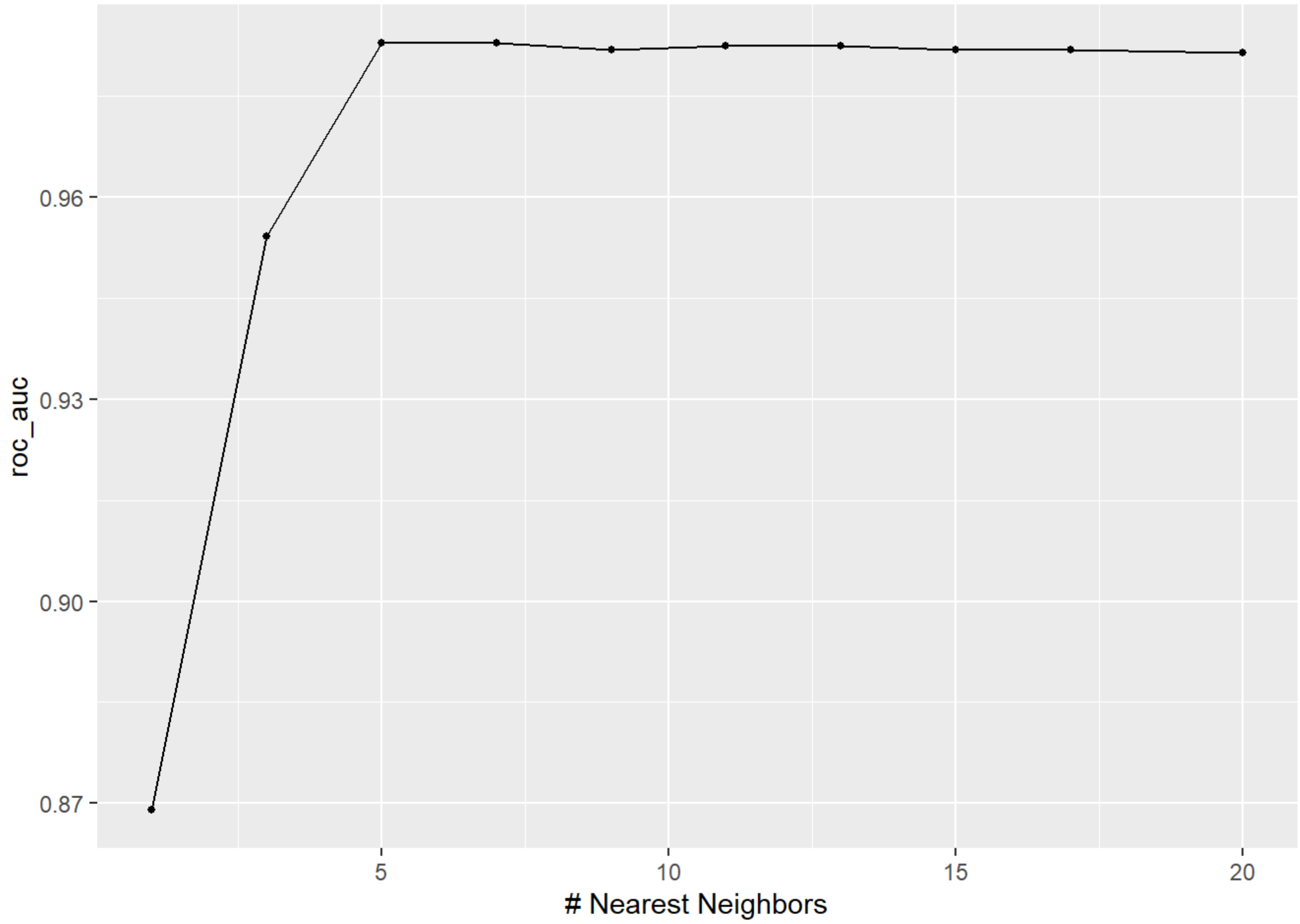
It is now time to build our models. Because these models require a bit of computing power, the code I used to create them can be found in separate R Markdown files. The models were saved and will be loaded below, from which we will analyze the results. I set the metric of performance to be roc_auc, as this is optimal in a binary classification problem with imbalanced data. The 6 models I used were logistic regression, k-nearest-neighbors, linear discriminant analysis, lasso regression, decision tree, and random forest.

Visualizing Results

The autoplot function in R is a great tool for visualizing the results of our tuned models. We will only be looking at the plots of the two best performing models, which were K-nearest-neighbors and random forest. These plots will detail the effects that changes in parameters will have on the roc_auc of our models.

Hide

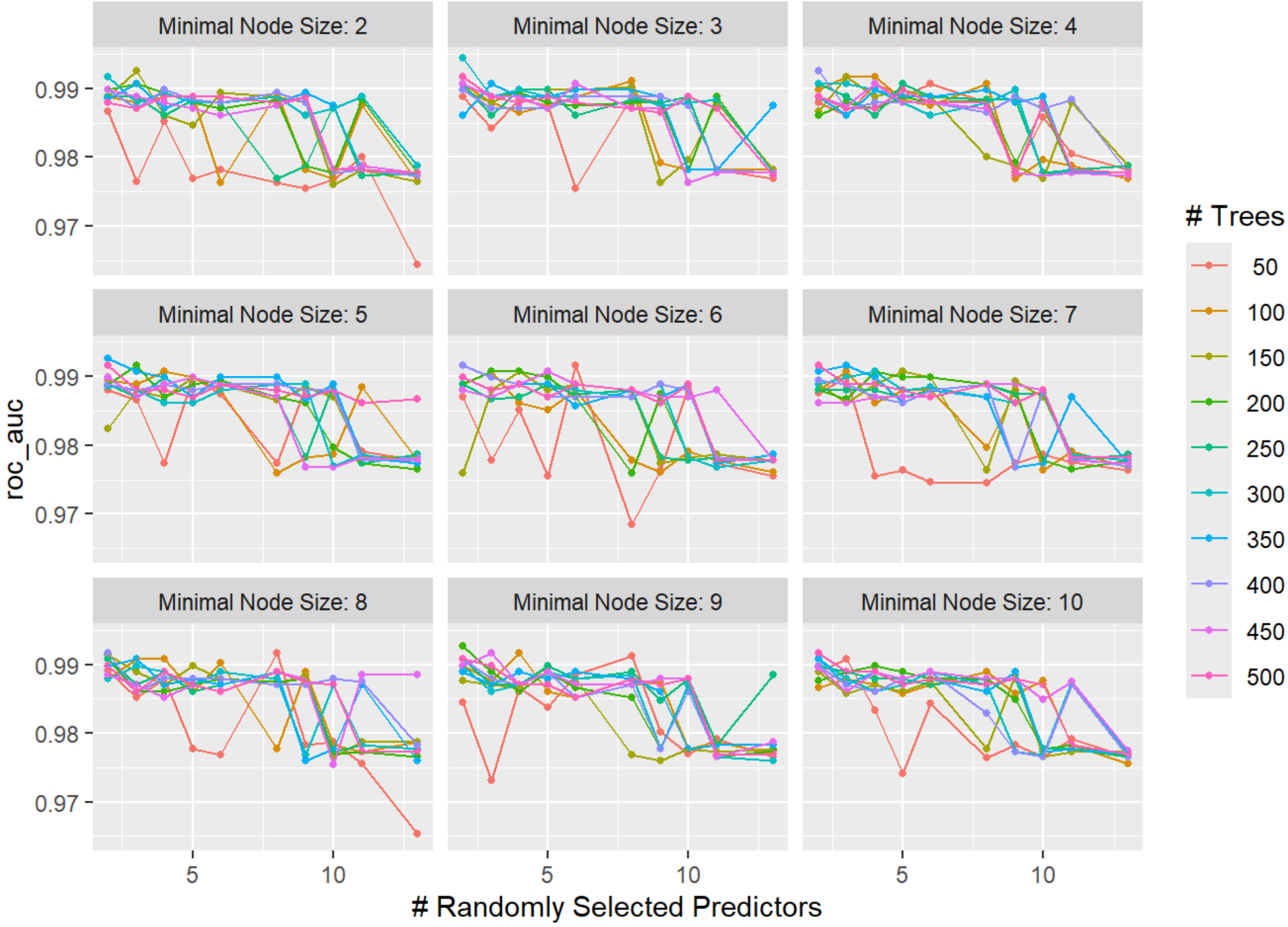
```
autoplot(nba_knn_tune_res)
```



Increasing the amount of nearest neighbors increased roc_auc of our model until the 5 nearest neighbors point. After this, the graph plateaus and increasing the amount of nearest neighbors doesn't significantly increase or decrease the roc_auc.

Hide

```
autoplot(nba_rf_tune_res)
```



For the random forest model, three different hyperparameters were tuned: mtry - The number of predictors that would be randomly sampled and given to the tree to make its decisions, trees - The number of trees to grow in the forest, and min_n - the minimum number of data values needed to create another split. Around the 8-10 predictors mark, increasing the number of predictors lowered the roc_auc value. The green and blue plots typically have higher roc_auc values, which tells us that having too little or too many trees lowered the value. The sweet spot was from around 200-350 trees. Node size didn’t have a visible affect on the roc_auc. The best model seems to be the top middle one, with about 250-300 trees, 2 randomly selected predictors, and a minimal node size of 3, achieving an roc_auc value of almost 1.

Hide

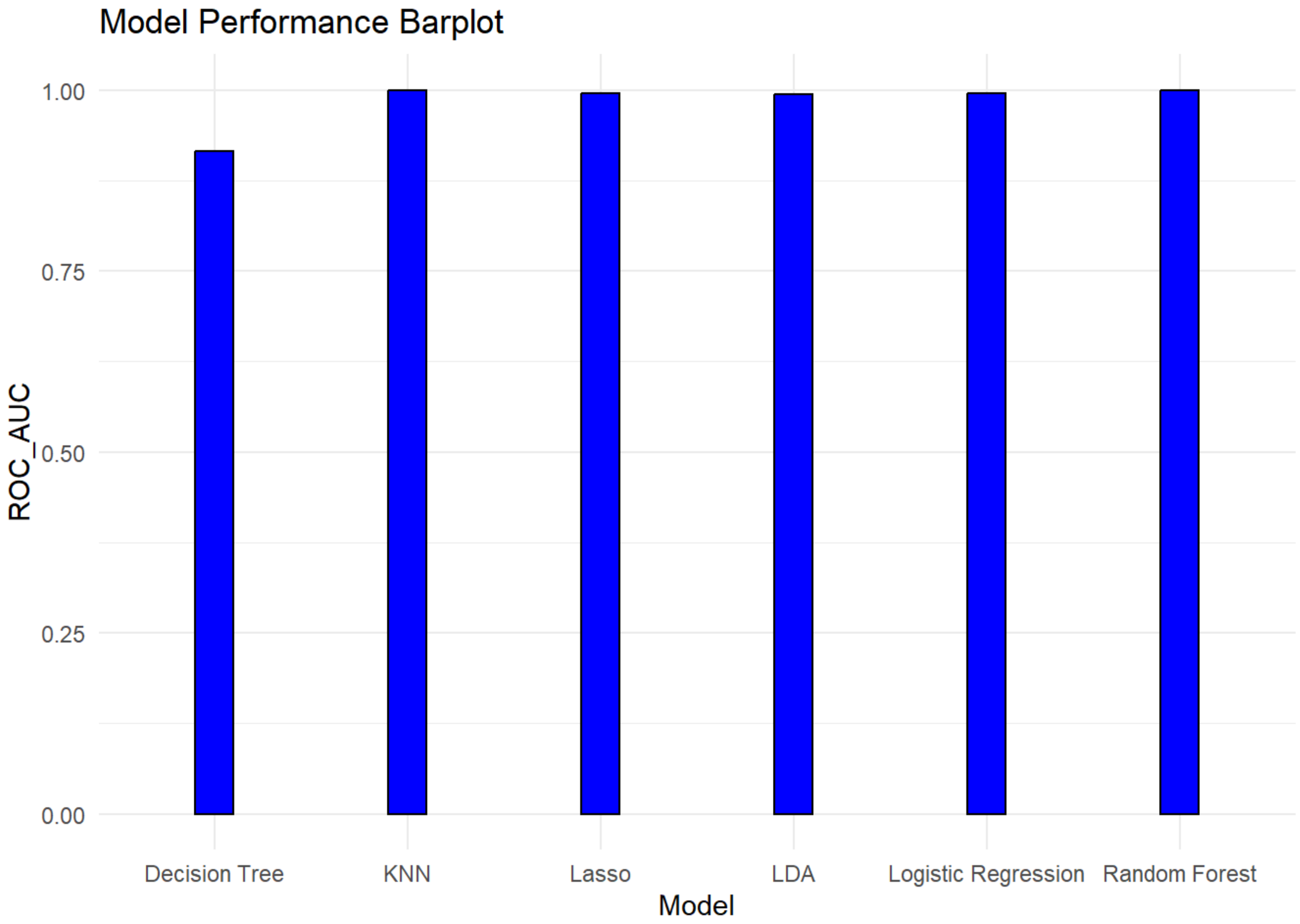
```
nba_lreg_auc <- augment(nba_lreg_fit, new_data = NbaTrain) %>%
  roc_auc(HOF, .pred_0)  #creating a tibble to display the roc_auc values of the models

nba_lda_auc <- augment(nba_lda_fit, new_data = NbaTrain) %>%
  roc_auc(HOF, .pred_0)

nba_knn_auc <- augment(nba_knn_final_fit, new_data = NbaTrain) %>%
  roc_auc(HOF, .pred_0)

nba_lasso_auc <- augment(nba_lasso_final_fit, new_data = NbaTrain) %>%
  roc_auc(HOF, .pred_0)

nba_decision_tree_auc <- augment(nba_dt_final_fit, new_data = NbaTrain) %>%
  roc_auc(HOF, .pred_0)

nba_random_forest_auc <- augment(nba_rf_final_fit, new_data = NbaTrain) %>%
  roc_auc(HOF, .pred_0)

nba_roc_aucs <- c(nba_lreg_auc$.estimate,
                           nba_lda_auc$.estimate,
                           nba_knn_auc$.estimate,
                           nba_lasso_auc$.estimate,
                           nba_decision_tree_auc$.estimate,
                           nba_random_forest_auc$.estimate)

nba_mod_names <- c("Logistic Regression",
            "LDA",
            "KNN",
            "Lasso",
            "Decision Tree",
            "Random Forest")

nba_results <- tibble(Model = nba_mod_names,
                      ROC_AUC = nba_roc_aucs)

nba_results <- nba_results %>%
  arrange(-nba_roc_aucs)

nba_results
```

```
## # A tibble: 6 × 2
##   Model               ROC_AUC
##   <chr>                 <dbl>
## 1 KNN                   1    
## 2 Random Forest         1    
## 3 Logistic Regression   0.996
## 4 Lasso                 0.995
## 5 LDA                   0.995
## 6 Decision Tree         0.916
```

The models all had high roc_auc scores, maybe a little too high. The 1.00 roc_auc values for the KNN and random forest model could potentially indicate overfitting, but we will have to see how the models perform on the testing set before jumping to conclusions.

Hide

```
nba_bar_plot <- ggplot(nba_results,
       aes(x = Model, y = ROC_AUC)) +
  geom_bar(stat = "identity", width=0.2, fill = "blue", color = "black") +
  labs(title = "Model Performance Barplot") +
  theme_minimal()
nba_bar_plot
```



Testing The Models

The next step will be to fit our best random forest model and our best k-nearest-neighbors model to the testing data and see how it does. An roc_auc of around 0.7 to 0.8 is considered good, and an roc_auc of 0.9 or above is excellent.

The Best Random Forest Model

Hide

```
show_best(nba_rf_tune_res, metric = "roc_auc") %>%
  select(-.estimator, .config) %>%
  dplyr::slice(1)
```

```
## # A tibble: 1 × 8
##    mtry trees min_n .metric  mean     n std_err .config
##   <int> <int> <int> <chr>   <dbl> <int>   <dbl> <chr>
## 1     2   300     3 roc_auc 0.994    10 0.00315 Preprocessor1_Model151
```

These are the parameters of the best random forest model.

Hide

```
nba_predict <- predict(nba_rf_final_fit,  # fitting our model to testing data
                              new_data = NbaTest,
                              type = "class")

nba_predict_with_actual <- nba_predict %>%
  bind_cols(NbaTest)

DT::datatable(nba_predict_with_actual, options = list(pageLength = 10, autoWidth = TRUE, s
crollX = TRUE)) #formatting the table to be interactive in the knitted file
```

Show 10 **entries** **Search:**

	.pred_class	Rk	Player	HOF	MVP	FSTM	FSTMD	DPOY	Y
1	1	1	Karl Malone	1	2	11	3	0	
2	1	2	Kobe Bryant	1	1	11	9	0	
3	1	7	Dominique Wilkins	1	0	1	0	0	
4	1	8	Tim Duncan	1	2	10	8	0	
5	1	10	Kevin Garnett	1	1	4	9	1	
6	1	23	Pau Gasol	1	0	0	0	0	
7	1	25	Mitch Richmond	1	0	0	0	0	
8	1	26	Joe Johnson	0	0	0	0	0	

9	1	32	Walter Davis	1	0	0	0	0
10	0	34	Terry Cummings	0	0	0	0	0

Showing 1 to 10 of 133 entries   Previous 1 2 3 4 5 … 14 Next

This is a table that shows the predictions the random forest model made for each player based off their stats, and the reality of whether or not they are in the Hall of Fame.

Hide

```
nba_augmented <- augment(nba_rf_final_fit, new_data = NbaTest) # used for ROC
DT::datatable(nba_augmented, options = list(pageLength = 10, autoWidth = TRUE, scrollX = T
RUE))
```

Show 10 entries   Search:

	.pred_class	.pred_0	.pred_1	Rk	Player	HOF
1	1	0.0116666666666667	0.9883333333333332	1	Karl Malone	1
2	1	0.0180555555555556	0.9819444444444446	2	Kobe Bryant	1
3	1	0.1330450513538749	0.866954948646125	7	Dominique Wilkins	1
4	1	0.0461111111111111	0.9538888888888888	8	Tim Duncan	1
5	1	0.0277777777777777	0.9722222222222221	10	Kevin Garnett	1
6	1	0.2768027357358344	0.7231972642641655	23	Pau Gasol	1
7	1	0.3337494870466723	0.6662505129533277	25	Mitch Richmond	1
8	1	0.2374587463059316	0.7625412536940683	26	Joe Johnson	0
9	1	0.343092079639265	0.656907920360735	32	Walter Davis	1

10	0	0.9124213872597492	0.08757861274025099	34	Terry Cummings	0

Showing 1 to 10 of 133 entries   Previous 1 2 3 4 5 … 14 Next

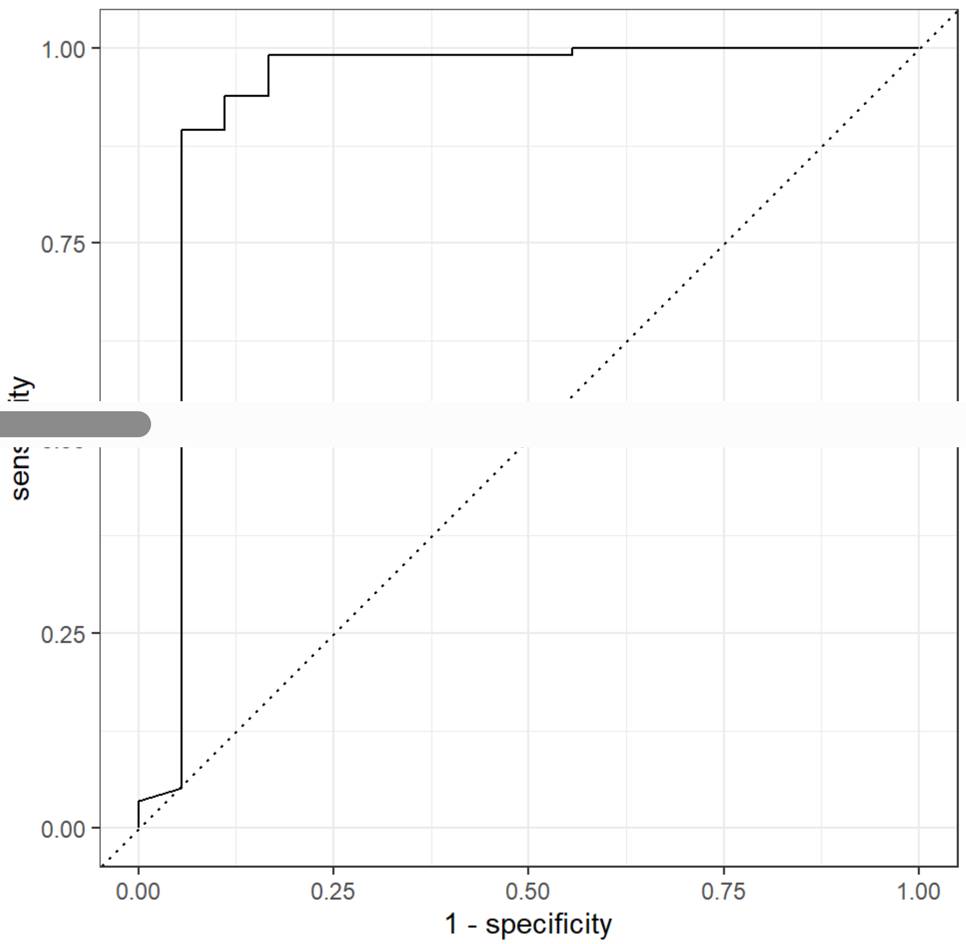
Hide

```
#formatting the table to be interactive in the knitted file
```

This table shows certainty the random forest model had for each prediction, and is helpful for analyzing how the roc_auc of the model is calculated.

Hide

```
nba_rf_roc_curve <- augment(nba_rf_final_fit, new_data = NbaTest) %>%
  roc_curve(HOF, .pred_0)  # computing the ROC curve for the random forest model

autoplot(nba_rf_roc_curve)
```



Before looking at the final roc_auc score, let's take a look at the ROC curve graph above. Ideally, we would like the curve to take a shape similar to that of an upside down L. This graph is quite close to what we are looking for, which indicates that the random forest model should have a pretty high roc_auc score.

Hide

```
nba_roc_auc <- augment(nba_rf_final_fit, new_data = NbaTest) %>%
  roc_auc(HOF, .pred_0)

nba_roc_auc
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.934
```

The random forest model had an roc_auc of about 0.9343, which is a great score.

The Best K-Nearest-Neighbors Model

Hide

```
show_best(nba_knn_tune_res, metric = "roc_auc") %>%
  select(-.estimator, .config) %>%
  dplyr::slice(1)
```

```
## # A tibble: 1 × 6
##   neighbors .metric  mean     n std_err .config
##       <int> <chr>   <dbl> <int>   <dbl> <chr>
## 1         5 roc_auc 0.983    10  0.0128 Preprocessor1_Model03
```

These are the parameters of the best k-nearest-neighbors model.

Hide

```
nba_predict2 <- predict(nba_knn_final_fit,  # fitting our model to testing data
                        new_data = NbaTest,
                        type = "class")

nba_predict_with_actual2 <- nba_predict2 %>%
  bind_cols(NbaTest)

DT::datatable(nba_predict_with_actual2, options = list(pageLength = 10, autoWidth = TRUE,
scrollX = TRUE)) #formatting the table to be interactive in the knitted file
```

Show 10 **entries** **Search:**

	.pred_class	Rk	Player	HOF	MVP	FSTM	FSTMD	DPOY	Y
1	1	1	Karl Malone	1	2	11	3	0	

2	1	2	Kobe Bryant	1	1	11	9	0
3	1	7	Dominique Wilkins	1	0	1	0	0
4	1	8	Tim Duncan	1	2	10	8	0
5	1	10	Kevin Garnett	1	1	4	9	1
6	1	23	Pau Gasol	1	0	0	0	0
7	0	25	Mitch Richmond	1	0	0	0	0
8	1	26	Joe Johnson	0	0	0	0	0
9	1	32	Walter Davis	1	0	0	0	0
10	0	34	Terry Cummings	0	0	0	0	0

Showing 1 to 10 of 133 entries

Previous 1 2 3 4 5 … 14 Next

This is a table that shows the predictions the k-nearest-neighbors model made for each player based off their stats, and the reality of whether or not they are in the Hall of Fame.

Hide

```
nba_augmented <- augment(nba_knn_final_fit, new_data = NbaTest) # used for ROC
DT::datatable(nba_augmented, options = list(pageLength = 10, autoWidth = TRUE, scrollX = T
RUE))
```

Show 10 **entries** **Search:**

	.pred_class	.pred_0	.pred_1	Rk	Player	HOF
1	1	0	1	1	Karl Malone	1
2	1	0	1	2	Kobe Bryant	1

3	1	0	1	7	Dominique Wilkins	1
4	1	0	1	8	Tim Duncan	1
5	1	0	1	10	Kevin Garnett	1
6	1	0.2581449104364529	0.7418550895635472	23	Pau Gasol	1
7	0	0.8947612513436317	0.1052387486563684	25	Mitch Richmond	1
8	1	0.1052387486563684	0.8947612513436317	26	Joe Johnson	0
9	1	0.2336303295411815	0.7663696704588184	32	Walter Davis	1
10	0	1	0	34	Terry Cummings	0

Showing 1 to 10 of 133 entries

Previous 1 2 3 4 5 … 14 Next

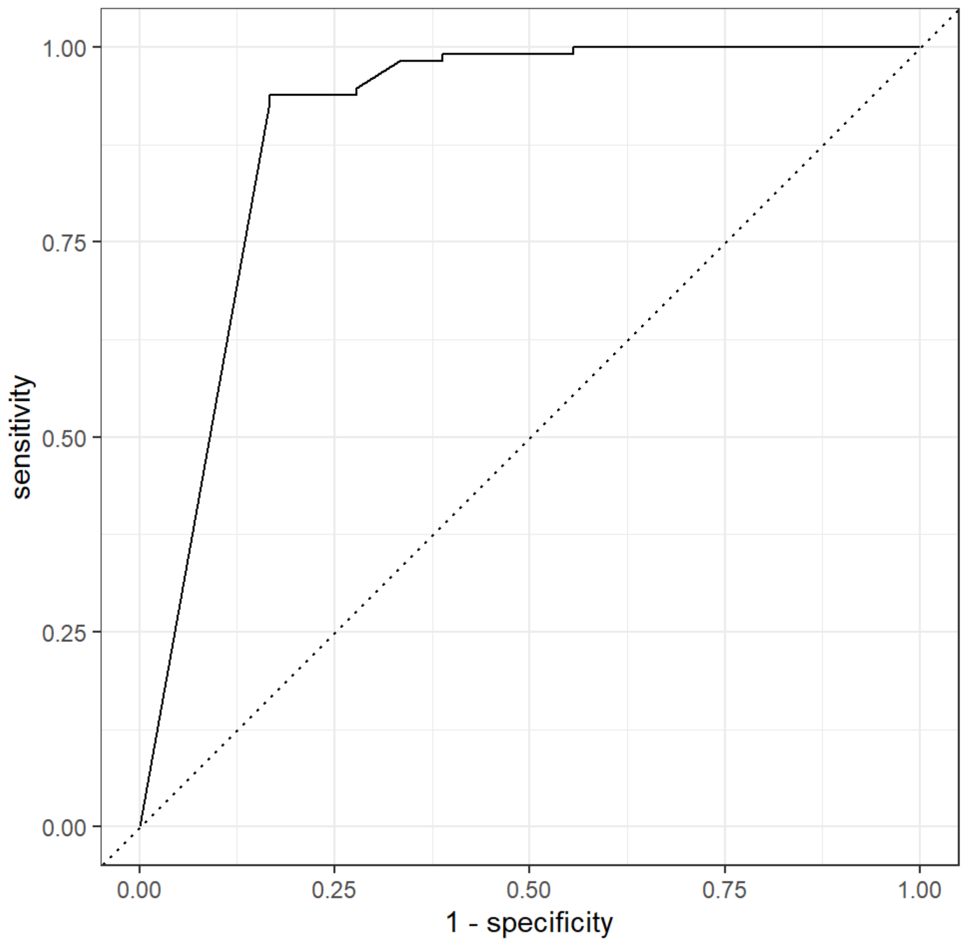
Hide

```
#formatting the table to be interactive in the knitted file
```

This table shows certainty the k-nearest-neighbors model had for each prediction, and is helpful for analyzing how the roc_auc of the model is calculated.

Hide

```
nba_knn_roc_curve <- augment(nba_knn_final_fit, new_data = NbaTest) %>%
  roc_curve(HOF, .pred_0)  # computing the ROC curve for the random forest model

autoplot(nba_knn_roc_curve)
```



The KNN model didn't have quite as ideal a shape as the random forest one, but it is still curving in the right direction, which means this model will also have a relatively good roc_auc score.

Hide

```
nba_roc_auc <- augment(nba_knn_final_fit, new_data = NbaTest) %>%
  roc_auc(HOF, .pred_0)

nba_roc_auc
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.900
```

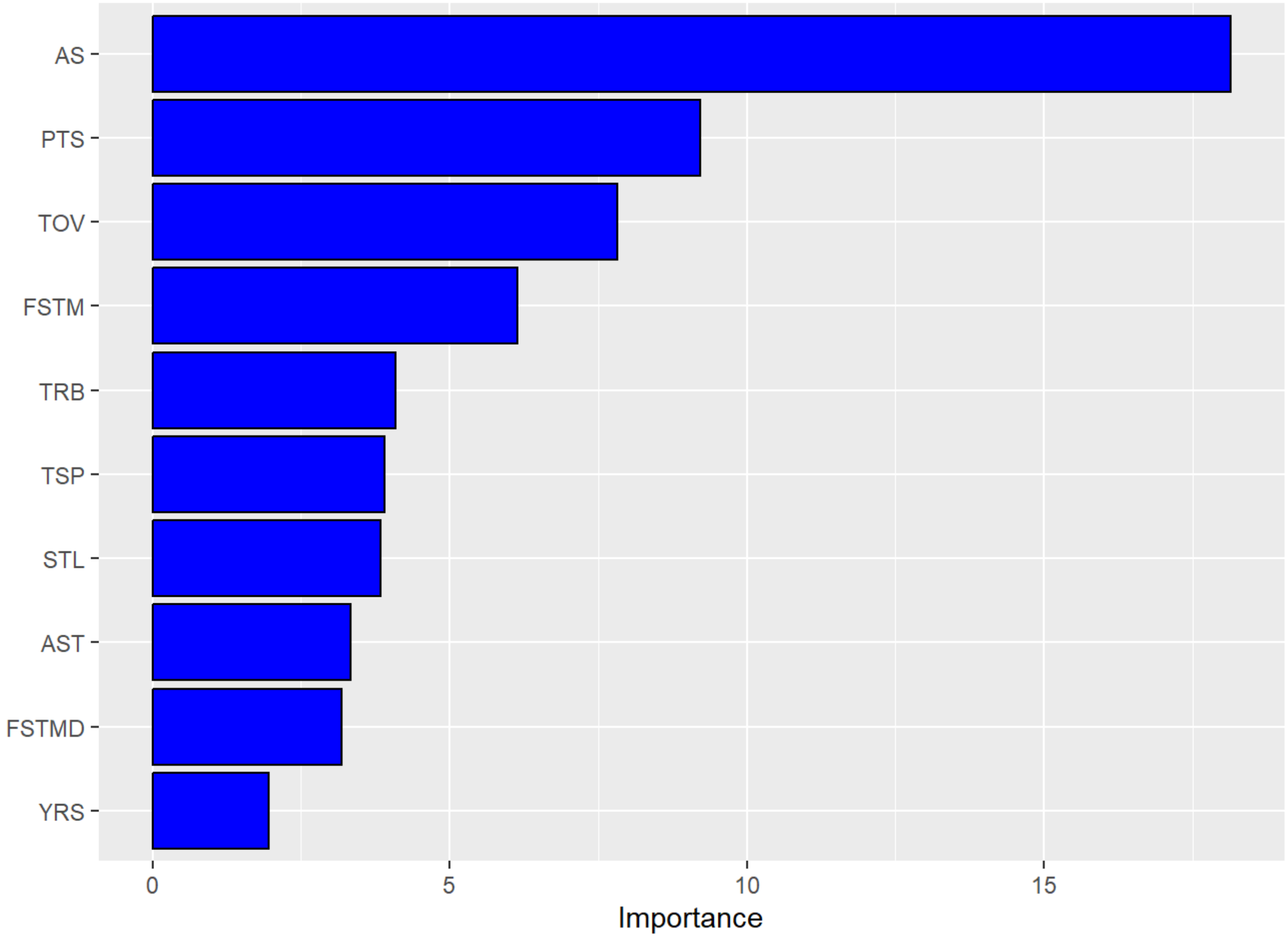
The KNN model had an roc_auc of about 0.900, so it also did well. Both the models may have suffered from a little overfitting, though.

Applying The Model

It's finally time to put our best model (random forest) to the test. Let's find out how well it will be able to predict a player's Hall of Fame status based on their individual stats. Let's quickly look at a variable importance chart first, though.

Hide

```
nba_rf_final_fit %>%
  extract_fit_engine() %>%
  vip(aesthetics = list(fill = "blue", color = "black"))
```



It seems that All Star appearances, Points, Turnovers, and First Team All NBA selections are the most important variables when it comes to predicting if a player will make the Hall of Fame or not. This aligns with our prior analysis of the distributions of these predictors earlier.

Test Cases

Lebron James is the greatest ever for many NBA fans, and an undisputed future first ballot Hall of Famer once he is eligible for induction. Let’s see if our model can properly predict a guaranteed Hall of Famer.

Hide

```
lbj_stats <- data.frame(PTS = 41003,
                        TRB = 11369,
                        AST = 11219,
                        AS = 20,
                        STL = 2291,
                        BLK = 1124,
                        TOV = 5305,
                        TSP = 0.589,
                        FSTM = 13,
                        FSTMD = 5,
                        MVP = 4,
                        DPOY = 0,
                        YRS = 21)
predict(nba_rf_final_fit, lbj_stats, type = "class")
```

```
## # A tibble: 1 × 1
##   .pred_class
##   <fct>
## 1 1
```

Nice! Now we have to test the model on a player who has no chance of making the Hall of Fame based on his current career stats and accolades. Sorry Dlo!

Hide

```
diangelo_russel_stats <- data.frame(PTS = 10405,
                        TRB = 2010,
                        AST = 3393,
                        AS = 1,
                        STL = 630,
                        BLK = 187,
                        TOV = 1577,
                        TSP = 0.547,
                        FSTM = 0,
                        FSTMD = 0,
                        MVP = 0,
                        DPOY = 0,
                        YRS = 9)
predict(nba_rf_final_fit, diangelo_russel_stats, type = "class")
```

```
## # A tibble: 1 × 1
##   .pred_class
##   <fct>
## 1 0
```

Correct again! Now to test the model on some players who I'm personally curious if they will make the Hall of Fame or not.

Hide

```
test_players <- read_csv("C:/Users/Ethan Tran/PSTAT131/Final Project/TestPlayers.csv")
nba_predict3 <- predict(nba_rf_final_fit,  # fitting our model to testing data
                              new_data = test_players,
                              type = "class")

nba_predict_with_actual3 <- nba_predict3 %>%
  bind_cols(test_players)

DT::datatable(nba_predict_with_actual3, options = list(pageLength = 10, autoWidth = TRUE,
scrollX = TRUE))
```

Show 10 **entries** **Search:**

	.pred_class	Player	MVP	DPOY	AS	FSTM	FSTMD	TSP
1	0	Draymond Green	0	1	4	0	4	0.542
2	1	Kyrie Irving	0	0	8	0	0	0.584
3	1	Kyle Lowry	0	0	6	0	0	0.569
4	0	Kevin Love	0	0	5	0	0	0.572
5	0	Klay Thompson	0	0	5	0	0	0.573
6	0	Rajon Rondo	0	0	4	0	2	0.501
7	0	Blake Griffin	0	0	6	0	0	0.559
8	0	LaMarcus Aldridge	0	0	7	0	0	0.541
9	0	Jimmy Butler	0	0	6	0	0	0.589
10	0	Amare Stoudemire	0	0	6	1	0	0.597

Showing 1 to 10 of 14 entries

Previous 1 2 Next

Hide

```
#formatting the table to be interactive in the knitted file
```

Conclusion

Throughout this project, we have analyzed our data in depth in order to build a model that can predict whether or not a player will make the Hall of Fame based on their career statistics and accolades. After testing and analyzing our six models, we determined that the random forest model was the best.

To potentially improve the project, I could have tried implementing a quadratic discriminant analysis model or a support vector machine. To improve upon the models I did build, I could have added playoff stats and accolades. Because some of the data was self compiled, this would have taken a long time to do. Another drawback is that playoff success in the NBA is directly correlated with how talented one's teammates are, which means adding playoff stats could skew the data as well. However, when looking at the 5 incorrect predictions my random forest model made on the testing set, we see 4 false negatives and one false positive. Joe Johnson, the false positive, is not too important as after doing some research, I learned that he will likely be inducted into the Hall of Fame in the future. When looking at the 4 false negatives, two of them, Vlade Divac and Toni Kukoc, were inducted partially based off their international and European League achievements. James Worthy and Michael Cooper, though, had extensive playoff resumes, as they were both multiple time NBA Finals winners, with James Worthy even earning a Finals MVP in 1988. I still think my decision to exclude basic playoffs stats and accolades was correct. I could have created my own predictors such as Finals MVPs with 500 total playoff points during the season, or Finals Wins with at least 750 minutes played, though, and I think this would have improved my model.

Overall, I am very happy with my Hall of Fame prediction model. I gained valuable knowledge and experience in the machine learning field, and I was able to put my passion for basketball to good use. Because I based my project on a topic that I am heavily invested in, sports analytics and the NBA in particular, the model building process was both enjoyable and insightful.