Final Project - Does Draft Round Predict if a Player Reaches the MLB?

Ben Steves

2020-12-11

NOTE\* All code pulling in data that is written in python is used from github user double\_dose\_larry.

<https://github.com/double-dose-larry/baseball_draft_data/blob/master/draft_data.ipynb>

pd.read\_json("https://lookup-service-prod.mlb.com/json/named.historical\_draft.bam?season=1965")

## historical\_draft  
## copyRight Copyright 2021 MLB Advanced Media, L.P. Use ...  
## queryResults {'created': '2021-12-27T03:09:45', 'totalSize'...

def get\_draft\_data(year):  
 print(f"getting year {year}")  
 json\_reply = requests.get(f"https://lookup-service-prod.mlb.com/json/named.historical\_draft.bam?season={year}").json()  
 df = pd.DataFrame(json\_reply["historical\_draft"]["queryResults"]["row"])  
 df["year"] = year  
 return df

draft\_df = pd.concat([ get\_draft\_data(yr) for yr in range(1965, 2020) ])

## getting year 1965  
## getting year 1966  
## getting year 1967  
## getting year 1968  
## getting year 1969  
## getting year 1970  
## getting year 1971  
## getting year 1972  
## getting year 1973  
## getting year 1974  
## getting year 1975  
## getting year 1976  
## getting year 1977  
## getting year 1978  
## getting year 1979  
## getting year 1980  
## getting year 1981  
## getting year 1982  
## getting year 1983  
## getting year 1984  
## getting year 1985  
## getting year 1986  
## getting year 1987  
## getting year 1988  
## getting year 1989  
## getting year 1990  
## getting year 1991  
## getting year 1992  
## getting year 1993  
## getting year 1994  
## getting year 1995  
## getting year 1996  
## getting year 1997  
## getting year 1998  
## getting year 1999  
## getting year 2000  
## getting year 2001  
## getting year 2002  
## getting year 2003  
## getting year 2004  
## getting year 2005  
## getting year 2006  
## getting year 2007  
## getting year 2008  
## getting year 2009  
## getting year 2010  
## getting year 2011  
## getting year 2012  
## getting year 2013  
## getting year 2014  
## getting year 2015  
## getting year 2016  
## getting year 2017  
## getting year 2018  
## getting year 2019

draft\_df.to\_csv("draft\_1965\_2019.csv", index=False)

NOTE\* All code from here on out is written by me

draft <- read.csv('draft\_1965\_2019.csv')  
draft <- draft %>%  
 #mutate(round = as.integer(round)) %>%  
 filter(draft\_type == "JR", year < 2018, round <= 20) %>%  
 replace\_na(list(so = 0)) %>%  
 mutate(year = as.factor(year)) %>%  
 mutate(mlb\_g = as.factor(mlb\_g)) %>%  
 separate(col = name\_first\_last, c("nameFirst", "nameLast"), sep = " ", extra = "merge", fill = "left")

## Overview

### Real-world question

Major League Baseball is arguably the world’s best league for baseball. Amateur baseball players across the world are hoping that one day they will be good enough to reach the MLB. The best of these amateur players (both college and high schoolers) will get drafted in the yearly June Draft, but this isn’t a guarantee that they will make the majors as an every day player. Generally speaking, players who are drafted in earlier rounds of the draft are considered better players than those drafted later on. However, a player’s draft position only tells us how good or how projectable that said player is as an amateur. Each player still has to grind through the minor leagues, which can be pretty draining for young players. This struggle for amateur players, along with skill level, can sometimes be completely independent of where someone was drafted. There’s usually going to be players drafted in the first round that never quite put their tools together. There are also players drafted in the 16th round who end up being very good.

The main question given this is: can we predict whether a player makes the majors based on what round they were drafted in? Could this also vary by year, since some years might have better draft classes than others?

### Data Source

The data comes from the MLB.com lookup service API and contains data on all drafted players from 1965-2019. The data contains a lot of draft profile data for players like round, pick, hometown, team and, position, weight. There are a few basic stats provided for those drafted who were/are in the MLB, like strikeouts and Earned Run Average (ERA) for pitchers, and home runs and batting average for hitters. One other notable variable is mlb\_g which is a variable indicating whether a drafted player made it to the major leagues, “Y” meaning yes and “N” meaning no. This is what will be the response variable in the modeling.

## Approach

### Problem Description

The main approach here is, based on the dataset, can we predict the variable mlb\_g based on variables round and year, where - mlb\_g is a boolean for whether a player made it to the majors - round is the round a player was drafted in - year is the year a player was drafted in

### Data

To figure this out I am going to carry out a classification task to try and see how the model does at predicting mlb\_g. From there I will test the model’s accuracy, sensitivity, and specificity.

#### Provenance

The data comes right from Major League Baseball’s API, and the data is owned by MLB. The data is available for use so long as it isn’t being used for commercial purposes. The terms of use are located here: <http://gdx.mlb.com/components/copyright.txt>

#### Structure

There are 18681 observations in the dataset and 38 variables. Each row represents a drafted player.

draft %>%  
 filter(nameFirst == "Bryan", nameLast == "Reynolds")

## hr nameFirst nameLast weight mlb\_g height\_inches era round\_sort round  
## 1 47 Bryan Reynolds 210 Y 3 -.-- 2 2  
## draft\_type rbi bats description sv height\_feet pick year  
## 1 JR 177 S June Amateur Draft NA 6 18 2016  
## name\_last\_first avg overall milb\_g team\_full league\_full  
## 1 Reynolds, Bryan 0.29 59 Y San Francisco Giants National League  
## primary\_position team\_abbrev birth\_date so throws league\_id l w  
## 1 OF SF 1995-01-27T00:00:00 0 R 104 NA NA  
## school trans\_date sb signed\_sw round\_desc player\_id team\_id  
## 1 Vanderbilt 2016-06-22T00:00:00 9 Y Round 2 668804 137

Important features of interest with types:

Ordinal: round, pick

Nominal: nameFirst, nameLast, mlb\_g, primary\_position

Numerical: year (although treated as a factor, so could be nominal), era, so

#### Appropriateness for task

I think for my approach this data is really solid. The mlb\_g parameter, especially, was perfect for what I was trying to predict, since I knew I wanted to do something related to draft data and predict how good a player might be based on draft round. The “Y” or “N” benchmark in my opinion is a good indicator of whether a player panned out in professional baseball.

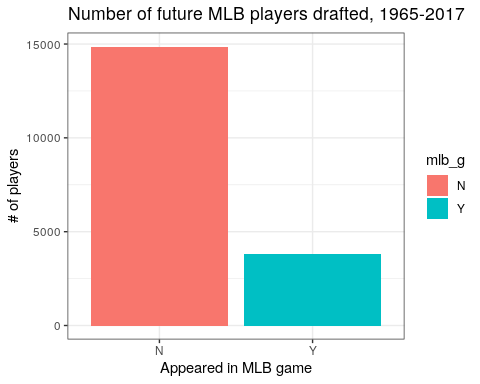
The only thing I may have hoped this data also had were hometown data or birth state data, because that is a pretty interesting predictor as well, since a lot of good players comes from warmer-weather states. I tried doing joins of other datasets to include this but most only had this kind of data for MLB players only, and not all drafted players, consisting of both major league and minor league players.

For the sake of data wrangling, I will most likely have to do a few things. For one, I only want to look at June Draft draftees, as the dataset has some other data on other drafts like the Rule V Draft and the now defunct August Legion Draft. I will also only consider the first 20 rounds, since generally players drafted after round 20 either make the majors on super rare occasions, or choose not to sign to improve their draft stock for the next year. Doing this will make sure there aren’t many repeat names, and it also simultaneously should make for a more interesting model. Also, players from really recent are probably not in the major leagues yet; it takes a couple of years for them to reach the majors generally. Because of this I’m going to filter out draftees from 2018 and 2019.

## Exploratory Data Analysis

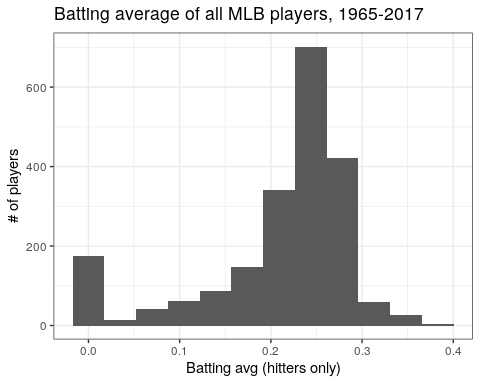
### Distributions:

draft %>%  
 group\_by(mlb\_g) %>%  
 summarize(n = n()) %>%  
 ggplot(aes(x = mlb\_g , y = n, fill = mlb\_g)) +  
 geom\_col() +  
 labs(x = "Appeared in MLB game", y = "# of players",  
 title3 = "Number of future MLB players drafted, 1965-2017")



There is a clear edge top the number of players who don’t reach the majors than those who do. This was expected since it’s generally difficult for players to reach the MLB after getting drafted.

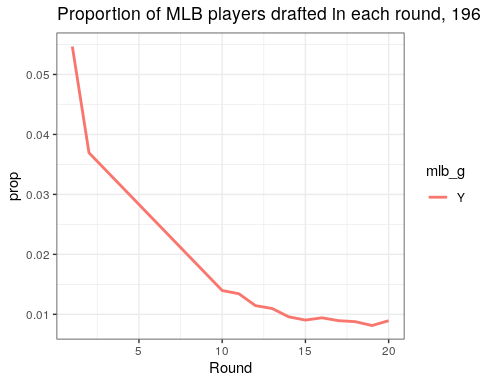
draft %>%  
 filter(avg < .4, primary\_position != "P") %>%  
 ggplot(aes(x = avg)) +  
 geom\_histogram(bins = 12) +  
 labs(x = "Batting avg (hitters only)", y = "# of players",  
 title = "Batting average of all MLB players, 1965-2017")



Overall there seems to be a larger number of of averages at about .250, which was expected. This is right about the league average. This only includes players that have made the MLB so this isn’t necesarily important in modeling, but still interesting.

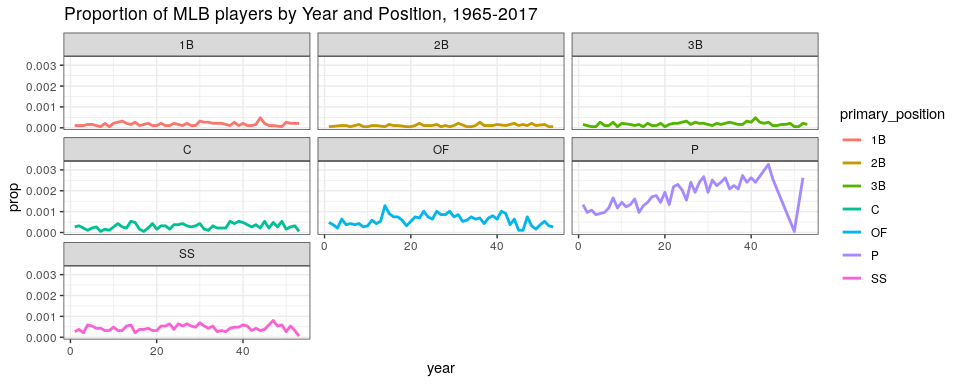
### Bivariate:

draft %>%  
 group\_by(round, mlb\_g) %>%  
 mutate(round = as.integer(round)) %>%  
 filter(mlb\_g == "Y") %>%  
 summarize(n = n()) %>%  
 mutate(prop = n / nrow(draft)) %>%  
 mutate(round = as.integer(round)) %>%  
 ggplot(aes(x = round, y = prop, color = mlb\_g)) +  
 geom\_line(size = 1) +  
 labs(title = "Proportion of MLB players drafted in each round, 1965-2017",  
 x = "Round", "Proportion of players drafted")



As the draft goes into later rounds, there is a worse chance that a player will not make the MLB.

draft %>%  
 group\_by(year, primary\_position, mlb\_g) %>%  
 filter(mlb\_g == "Y", primary\_position %in% c("3B", "P", "OF", "1B", "SS", "C", "2B")) %>%  
 mutate(year = as.numeric(year)) %>%  
 summarize(n = n()) %>%  
 mutate(prop = n / nrow(draft)) %>%  
 ggplot(aes(x = year, y = prop, color = primary\_position)) +  
 facet\_wrap(~primary\_position) +  
 geom\_line(size = 1) +  
 labs(title = "Proportion of MLB players by Year and Position, 1965-2017")

 Between positions, there isn’t really much of an increase in the number of MLB players by year, arguably besides pitcher. I figured there would be an uptick in number of pitchers who played in MLB in later years, however, which does seem to be the case in the graph as well. As the game has progressed, there has been more of an emphasis on crafting efficient bullpens. In the early days of baseball, a team would have a few starting pitchers and that was it. Now MLB teams roster about 13 pitchers, and bullpens are great to keep starting pitchers arms presereved. There is also a shorter window of time for relief pitchers to prove they have what it takes, which means that there can sometimes be a constant cycle of pitchers getting sent down to the minors while other pitchers making their MLB debuts are called up.

Note\* Year 0 is 1965, year 10 is 1975, etc. When I tried to plot years originally it didn’t look right since it was originally a factor/string, and when I converted it here it just numbered each string from 1-50 unfortunately.

## Modeling

### Setup

The model I am planning to fit is a tidymodels logistic regression model using the boolean variable “mlb\_g” as the response variable, and “round” and “year” as predictors. Round is chosen because I am curious at how well draft round can predict if a player makes it to the MLB. Generally we might assume that those drafted in rounds 1 and 2 have the best shot at reaching the majors, but this isn’t always 100% true. I am also curious about “year” because it is possible that certain years produced better players than others.

I plan on measuring the model in three ways:

* Accuracy: % of correct predictions
* Sensitivity: True positive rate
* Specificity: True negative rate

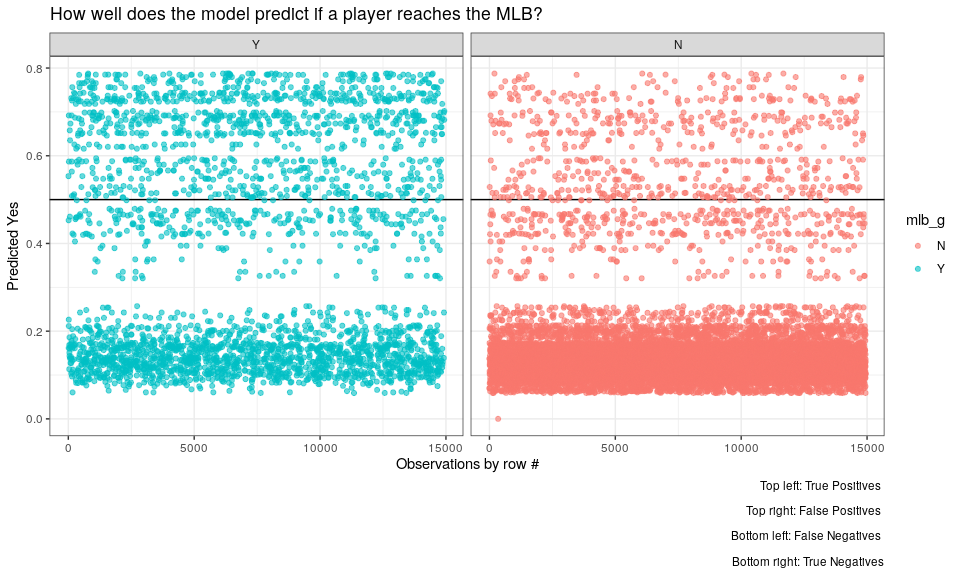
I will also carry out cross validation for the model to see its performance within the training set. After that I plan to measure the accuracy, sensitivity, and specificity for the test set.

set.seed(650)  
dsplit <- initial\_split(draft, prop = 4 / 5)  
dtrain <- training(dsplit)  
dtest <- testing(dsplit)

### Model 1

I chose a logistic regression for this model since our response variable (mlb\_g) is categorical. The only main hyperparameter is the engine “glm” which is just a basic regression engine. Predicted “Y” is higher if the player is more likely to have made the MLB.

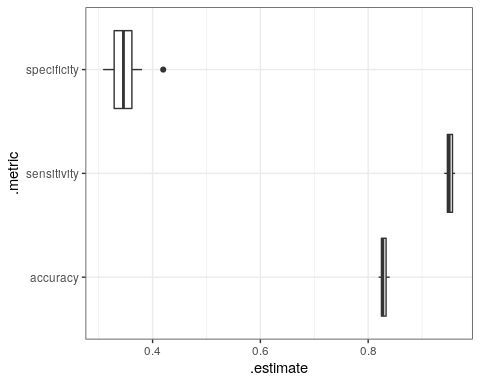
spec <- workflow() %>% add\_recipe(  
 recipe(mlb\_g ~ round + year, data = dtrain)) %>%   
 add\_model(logistic\_reg(mode = "classification") %>% set\_engine("glm"))  
dmodel <- spec %>% fit(dtrain)  
  
dmodel %>%   
 predict(dtrain, type = "prob") %>%  
 bind\_cols(dtrain) %>%  
 mutate(idx = row\_number()) %>%   
 ggplot(aes(x = idx, y = .pred\_Y, color = mlb\_g)) +  
 facet\_wrap(~fct\_relevel(mlb\_g, "Y", "N")) +  
 geom\_hline(yintercept = .5) +  
 geom\_point(alpha = 0.6) +  
 labs(title = "How well does the model predict if a player reaches the MLB?",  
 y = "Predicted Yes", x = "Observations by row #",  
 caption =   
 "Top left: True Positives \n  
 Top right: False Positives \n  
 Bottom left: False Negatives \n  
 Bottom right: True Negatives")



metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
dmodel %>%   
 predict(dtrain, type = "class") %>%   
 bind\_cols(dtrain) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)

## # A tibble: 3 × 3  
## .metric .estimator .estimate  
## <chr> <chr> <dbl>  
## 1 accuracy binary 0.829  
## 2 sensitivity binary 0.950  
## 3 specificity binary 0.354

resamples <- dtrain %>% vfold\_cv(v = 10, strata = mlb\_g)  
cv\_results <- spec %>%  
 fit\_resamples(resamples, metrics = metrics)  
cv\_results %>%  
 collect\_metrics(summarize = FALSE) %>%  
 ggplot(aes(x = .estimate, y = .metric)) + geom\_boxplot()



cv\_results %>%  
 collect\_metrics(summarize = TRUE)

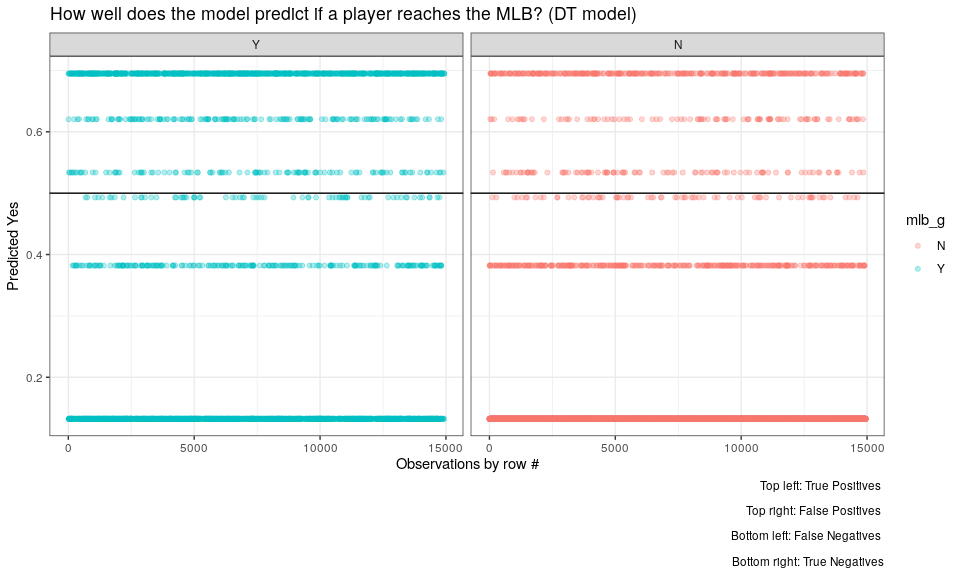
## # A tibble: 3 × 6  
## .metric .estimator mean n std\_err .config   
## <chr> <chr> <dbl> <int> <dbl> <chr>   
## 1 accuracy binary 0.828 10 0.00213 Preprocessor1\_Model1  
## 2 sensitivity binary 0.951 10 0.00236 Preprocessor1\_Model1  
## 3 specificity binary 0.349 10 0.0105 Preprocessor1\_Model1

The model is accurate rounded to about 83.2% in both the normal training set and the cross validation. The sensitivity of the model is good at about 95%, meaning that there are very few false negatives. There happen to be a lot of false positives though, which makes sense if we are only predicting by round and year, as this would show that there are many players in higher rounds that don’t end up making the majors.

### Model 2: Slight improvements (at least on the training set)

To try and improve accuracy, I fit a decision tree instead of a logistic regression. The model is the same except I use the hyperparameter cost\_complexity to make the model pretty specific, and set it to 0.0001. The engine is also switched to “rpart”.

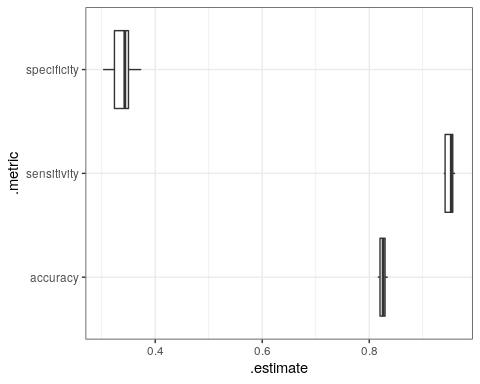
spec1 <- workflow() %>% add\_recipe(  
 recipe(mlb\_g ~ round + year, data = dtrain)) %>%  
 add\_model(decision\_tree(mode = "classification", cost\_complexity = 0.0001) %>% set\_engine("rpart"))  
model <- spec1 %>% fit(dtrain)  
  
model %>%   
 predict(dtrain, type = "prob") %>%  
 bind\_cols(dtrain) %>%  
 mutate(idx = row\_number()) %>%   
 ggplot(aes(x = idx, y = .pred\_Y, color = mlb\_g)) +  
 facet\_wrap(~fct\_relevel(mlb\_g, "Y", "N")) +  
 geom\_hline(yintercept = .5) +  
 geom\_point(alpha = 0.3) +  
 labs(title = "How well does the model predict if a player reaches the MLB? (DT model)",  
 y = "Predicted Yes", x = "Observations by row #",  
 caption =   
 "Top left: True Positives \n  
 Top right: False Positives \n  
 Bottom left: False Negatives \n  
 Bottom right: True Negatives")



metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
model %>%   
 predict(dtrain, type = "class") %>%   
 bind\_cols(dtrain) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)

## # A tibble: 3 × 3  
## .metric .estimator .estimate  
## <chr> <chr> <dbl>  
## 1 accuracy binary 0.832  
## 2 sensitivity binary 0.953  
## 3 specificity binary 0.360

resamples1 <- dtrain %>% vfold\_cv(v = 10, strata = mlb\_g)  
cv\_results1 <- spec1 %>%  
 fit\_resamples(resamples1, metrics = metrics)  
cv\_results1 %>%  
 collect\_metrics(summarize = FALSE) %>%  
 ggplot(aes(x = .estimate, y = .metric)) + geom\_boxplot()



cv\_results1 %>%  
 collect\_metrics(summarize = TRUE)

## # A tibble: 3 × 6  
## .metric .estimator mean n std\_err .config   
## <chr> <chr> <dbl> <int> <dbl> <chr>   
## 1 accuracy binary 0.825 10 0.00197 Preprocessor1\_Model1  
## 2 sensitivity binary 0.950 10 0.00260 Preprocessor1\_Model1  
## 3 specificity binary 0.338 10 0.00699 Preprocessor1\_Model1

With the decision tree, the training data accuracy is better by a little, 83.7% compared to 83.2%. The cross validation were worse by a little, at 83.0%. Overall the decision tree might improve the model a little, but only slightly, at least on the training data.

### Test - both models

I wasn’t 100% convinced that my decision tree was that much better, so rather than pick the best model, I just tested with both.

#### Decision tree:

metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
model %>%   
 predict(dtest, type = "class") %>%   
 bind\_cols(dtest) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)

## # A tibble: 3 × 3  
## .metric .estimator .estimate  
## <chr> <chr> <dbl>  
## 1 accuracy binary 0.819  
## 2 sensitivity binary 0.947  
## 3 specificity binary 0.322

#### Logistic reg

metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
dmodel %>%   
 predict(dtest, type = "class") %>%   
 bind\_cols(dtest) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)

## # A tibble: 3 × 3  
## .metric .estimator .estimate  
## <chr> <chr> <dbl>  
## 1 accuracy binary 0.822  
## 2 sensitivity binary 0.950  
## 3 specificity binary 0.326

In testing the model, the logistic regression proved to be better in accuracy by 0.5% Both the specificity and sensitivity were better on with the logistic regression as well.

## Summary

### Discussion of Findings

To debrief what I did, I first fit a logistic regression using response variable mlb\_g (“Y” if a player appeared in an MLB game), and predictors round (the round a player was selected in the draft) and year (year drafted). I created a graph to visualize the accuracy of the model. I then checked the accuracy, sensitivity, and specificity of this model to see how it performed. I did cross validation to analyze the results again in a slightly different method. I repeated this process but with a decision tree in an attempt to improve the model, which it did on the training data, but not on the test data. Finally I tested the data and found that the logistic regression performed the best. It finished with an accuracy of 82.2%, a sensitivity of 95.3%, and a specificity of 33.4%. The model being more sensitive that specific is good, since we want it to be accurate at predicting players that are/were MLB players. It’s much harder to predict the opposite, especially if said player was drafted in a high round.

I think the findings of this model make a lot of sense in retrospect. The false positives aren’t super surprising to me, since these were players that were predicted MLB players but never panned out. Many a year there are these super hyped-up amateur players that we think are going to be the next Miguel Cabrera…and then they bat 0.220 in the minors and five years later are out of baseball without ever getting their shot at the majors.

Nonetheless, this model helps us understand if draft round (and year) really have any effect in predicting whether or not a player makes the MLB. I’m not convinced that year is an great predictor (given the exploratory graph I made, which showed year not really having an effect on the number of draftees becoming MLB players), although it could be for pitchers given the reasons I stated in that section (see Exploratory Data Analysis). Overall I do think round can predict whether a player makes the MLB, meaning that overall players drafted in higher rounds are probably better and more MLB equipped than later round draft picks. This was a really interesting topic to try and use the predictive modeling process on.

### Limitations and Ethical Considerations

I think there are some identifiable biases to some degree, not exactly in how the data was compiled, but bias in how MLB teams treat their draft picks. Usually, players drafted in the first round are seen as an MLB team’s prize of the draft, and they are payed more than draftees in later rounds. If a player who was a former first round pick is not doing great in the minors, MLB General Managers might call them up anyway because they’re already paying that player a lot, and it would be almost a waste of money not to have them pitch in the majors, independent of their skill level. For example, if hypothetically a 1st round pick and a 14th round pick perform the exact same in the minors, with the exact same stats, the first round pick will probably be favored for a call-up because they cost their team more. Other than that, there isn’t any other obvious examples of biases in the draft and call-up process, or in the data.

### Future Directions

I am curious if there are any other possible predictors not present in this dataset that would be interesting to study, like home state or potentially an indicator if a player was drafted out of high school or college. One new approach I can think of would be predicting if high school or college players turn out better in the MLB. I could theoretically use a similar approach using mlb\_g as a predictor, but if I wanted to do a linear regression instead I could use another response variable, like ERA (Earned Run Average) for pitchers, or Batting average for hitters. These are two stats that measure the success of a player, but only work well if a player has played a lot of games, otherwise these stats might be majorly inflated or deflated. There was a dataset from SeanLahman, a prominent baseball writer, with info on where a player’s home state is, which could work, and is also an R package called “Lahman”. Unfortunately this only has hometown data on MLB players, so I may have to use this type of modeling on only MLB players rather than all drafted players, but it still might be pretty interesting.

## Appendix

knitr::opts\_chunk$set(echo = TRUE)  
library(tidyverse)  
library(reticulate)  
library(tidymodels)  
require(glmnet)  
library(ggridges)  
py\_run\_string("import matplotlib.pyplot as plt; plt.switch\_backend('agg')")  
theme\_set(theme\_bw())  
options(dplyr.summarise.inform = FALSE) # silence a warning message  
import numpy as np  
import pandas as pd  
import json  
import requests  
pd.set\_option("display.max\_columns",100)  
pd.read\_json("https://lookup-service-prod.mlb.com/json/named.historical\_draft.bam?season=1965")   
def get\_draft\_data(year):  
 print(f"getting year {year}")  
 json\_reply = requests.get(f"https://lookup-service-prod.mlb.com/json/named.historical\_draft.bam?season={year}").json()  
 df = pd.DataFrame(json\_reply["historical\_draft"]["queryResults"]["row"])  
 df["year"] = year  
 return df  
draft\_df = pd.concat([ get\_draft\_data(yr) for yr in range(1965, 2020) ])  
draft\_df.to\_csv("draft\_1965\_2019.csv", index=False)  
draft <- read.csv('draft\_1965\_2019.csv')  
draft <- draft %>%  
 #mutate(round = as.integer(round)) %>%  
 filter(draft\_type == "JR", year < 2018, round <= 20) %>%  
 replace\_na(list(so = 0)) %>%  
 mutate(year = as.factor(year)) %>%  
 mutate(mlb\_g = as.factor(mlb\_g)) %>%  
 separate(col = name\_first\_last, c("nameFirst", "nameLast"), sep = " ", extra = "merge", fill = "left")   
draft %>%  
 filter(nameFirst == "Bryan", nameLast == "Reynolds")  
draft %>%  
 group\_by(mlb\_g) %>%  
 summarize(n = n()) %>%  
 ggplot(aes(x = mlb\_g , y = n, fill = mlb\_g)) +  
 geom\_col() +  
 labs(x = "Appeared in MLB game", y = "# of players",  
 title3 = "Number of future MLB players drafted, 1965-2017")  
draft %>%  
 filter(avg < .4, primary\_position != "P") %>%  
 ggplot(aes(x = avg)) +  
 geom\_histogram(bins = 12) +  
 labs(x = "Batting avg (hitters only)", y = "# of players",  
 title = "Batting average of all MLB players, 1965-2017")  
draft %>%  
 group\_by(round, mlb\_g) %>%  
 mutate(round = as.integer(round)) %>%  
 filter(mlb\_g == "Y") %>%  
 summarize(n = n()) %>%  
 mutate(prop = n / nrow(draft)) %>%  
 mutate(round = as.integer(round)) %>%  
 ggplot(aes(x = round, y = prop, color = mlb\_g)) +  
 geom\_line(size = 1) +  
 labs(title = "Proportion of MLB players drafted in each round, 1965-2017",  
 x = "Round", "Proportion of players drafted")  
draft %>%  
 group\_by(year, primary\_position, mlb\_g) %>%  
 filter(mlb\_g == "Y", primary\_position %in% c("3B", "P", "OF", "1B", "SS", "C", "2B")) %>%  
 mutate(year = as.numeric(year)) %>%  
 summarize(n = n()) %>%  
 mutate(prop = n / nrow(draft)) %>%  
 ggplot(aes(x = year, y = prop, color = primary\_position)) +  
 facet\_wrap(~primary\_position) +  
 geom\_line(size = 1) +  
 labs(title = "Proportion of MLB players by Year and Position, 1965-2017")  
set.seed(650)  
dsplit <- initial\_split(draft, prop = 4 / 5)  
dtrain <- training(dsplit)  
dtest <- testing(dsplit)  
spec <- workflow() %>% add\_recipe(  
 recipe(mlb\_g ~ round + year, data = dtrain)) %>%   
 add\_model(logistic\_reg(mode = "classification") %>% set\_engine("glm"))  
dmodel <- spec %>% fit(dtrain)  
  
dmodel %>%   
 predict(dtrain, type = "prob") %>%  
 bind\_cols(dtrain) %>%  
 mutate(idx = row\_number()) %>%   
 ggplot(aes(x = idx, y = .pred\_Y, color = mlb\_g)) +  
 facet\_wrap(~fct\_relevel(mlb\_g, "Y", "N")) +  
 geom\_hline(yintercept = .5) +  
 geom\_point(alpha = 0.6) +  
 labs(title = "How well does the model predict if a player reaches the MLB?",  
 y = "Predicted Yes", x = "Observations by row #",  
 caption =   
 "Top left: True Positives \n  
 Top right: False Positives \n  
 Bottom left: False Negatives \n  
 Bottom right: True Negatives")  
metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
dmodel %>%   
 predict(dtrain, type = "class") %>%   
 bind\_cols(dtrain) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)  
resamples <- dtrain %>% vfold\_cv(v = 10, strata = mlb\_g)  
cv\_results <- spec %>%  
 fit\_resamples(resamples, metrics = metrics)  
cv\_results %>%  
 collect\_metrics(summarize = FALSE) %>%  
 ggplot(aes(x = .estimate, y = .metric)) + geom\_boxplot()  
cv\_results %>%  
 collect\_metrics(summarize = TRUE)  
spec1 <- workflow() %>% add\_recipe(  
 recipe(mlb\_g ~ round + year, data = dtrain)) %>%  
 add\_model(decision\_tree(mode = "classification", cost\_complexity = 0.0001) %>% set\_engine("rpart"))  
model <- spec1 %>% fit(dtrain)  
  
model %>%   
 predict(dtrain, type = "prob") %>%  
 bind\_cols(dtrain) %>%  
 mutate(idx = row\_number()) %>%   
 ggplot(aes(x = idx, y = .pred\_Y, color = mlb\_g)) +  
 facet\_wrap(~fct\_relevel(mlb\_g, "Y", "N")) +  
 geom\_hline(yintercept = .5) +  
 geom\_point(alpha = 0.3) +  
 labs(title = "How well does the model predict if a player reaches the MLB? (DT model)",  
 y = "Predicted Yes", x = "Observations by row #",  
 caption =   
 "Top left: True Positives \n  
 Top right: False Positives \n  
 Bottom left: False Negatives \n  
 Bottom right: True Negatives")  
metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
model %>%   
 predict(dtrain, type = "class") %>%   
 bind\_cols(dtrain) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)  
resamples1 <- dtrain %>% vfold\_cv(v = 10, strata = mlb\_g)  
cv\_results1 <- spec1 %>%  
 fit\_resamples(resamples1, metrics = metrics)  
cv\_results1 %>%  
 collect\_metrics(summarize = FALSE) %>%  
 ggplot(aes(x = .estimate, y = .metric)) + geom\_boxplot()  
cv\_results1 %>%  
 collect\_metrics(summarize = TRUE)   
metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
model %>%   
 predict(dtest, type = "class") %>%   
 bind\_cols(dtest) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)  
metrics <- yardstick::metric\_set(accuracy, sensitivity, specificity)  
dmodel %>%   
 predict(dtest, type = "class") %>%   
 bind\_cols(dtest) %>%   
 metrics(truth = mlb\_g, estimate = .pred\_class)