NBA Free Agency Contract Prediction 2021

Sumitro Datta

July 27, 2021

Introduction

Day 1 of NBA free agency is by far the biggest day of the offseason for the league. While last year's class itself wasn't as star studded as 2019, it still provided plenty of drama with the COVID-19 pandemic still hanging around like an ominous cloud. The Milwaukee Bucks' trade for **Bogdan Bogdanović** of the Sacramento Kings was leaked before the official opening of the moratorium and subsequently backtracked, allowing the Atlanta Hawks to swoop in and sign Bogdanović to a 4-year, `$72M` contract. **Anthony Davis** shut down speculation that he was going to take a shorter-term contract to maximize his earning potential in later years by agreeing to a 5-year, `$190M` maximum contract extension with the Los Angeles Lakers. **Jerami Grant** desired a shot at being a Number 1 option, signing a 3-year, `$60M` contract with the rebuilding Detroit Pistons after rejecting a similar deal to re-sign with the promising Denver Nuggets, who had reached the Western Conference Finals that year. And **Gordon Hayward** closed the fascinating Boston chapter of his NBA career, opting out of `$34.2M` and signing with the Charlotte Hornets to the tune of 4 years and `$120M`.

This year, the class is headlined by two players with player options, meaning they may not even hit the market if they pick their options up. At the start of the season, it was almost a guarantee that 35-year-old point guard **Chris Paul**, newly acquired in a trade by the Phoenix Suns, would accept his `$44M` player option. It was also widely assumed that small forward **Kawhi Leonard** of the Los Angeles Clippers would be opting out of his `$36M` option. Fast forward to now: Paul has helped lead the young Suns to a NBA Finals appearance, while Leonard suffered a partially torn ACL in the second round of the playoffs, sidelining him for possibly the entire 2022 season after surgery. Reports have trickled out that Paul might opt out and seek a 3-year & `$100M` deal, while it's a distinct possibility that Leonard opts in to continue his rehab under the Clippers.

In terms of restricted free agents, the top players are **Lonzo Ball, John Collins** & **Jarrett Allen**. Collins is a bouncy power forward who has spent the past three years catching lobs from Trae Young in Atlanta. He reportedly turned down a `$90M` extension, believing himself to be max-contract worthy, but he might be a casualty of the Hawks' future cap crunch. Ball might not have lived up to his astronomical hype as the No. 2 overall pick in 2016, but he's a solid point guard with plus defense & playmaking acumen. Allen was shipped out to the Cleveland Cavaliers as the key young piece in the trade that brought James Harden to the Brooklyn Nets, thus being freed from the shackles of a timeshare with fellow centre Deandre Jordan.

On the unrestricted free agent side, we've got **DeMar DeRozan, Mike Conley** & **Kyle Lowry** topping the lists. After spending the majority of his career as a shooting guard, DeRozan has reinvented himself as a point forward in San Antonio under the tutelage of legendary coach Gregg Popovich, averaging almost 7 assists a game. Conley struggled in his first season with the Utah Jazz, battling injuries and attempting to adapt on the fly to coach Quin Snyder's system. He was able to put it all together this season, playing an integral part in helping the Jazz finish with the league's best record and even making his first All-Star team. Speculation ran rampant that Lowry was going to be traded this year, possibly ending an extremely successful 8-year tenure with the Toronto Raptors that included an NBA championship in 2019. He ultimately stayed put, but with the Raptors trending towards a possible retool/rebuild and Lowry wanting to add to his ring count, media discussion quickly turned to offseason destinations. This shifted into overdrive when the Philadelphia 76ers & Miami Heat (who were both rumored to be heavily pursuing the Raptors point guard during the season) flamed out early in the playoffs.

What I wanted to do was predict what contracts this year's free agent class might get based off previous offseasons. Stars generally get star-type money, but in tiers below, contracts of comparable players usually come up in discussing contract value.

Methods/Analysis

Loading Packages

Let's start by loading required packages.

```r
if(!require(tidyverse))
  install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(tidymodels))
  install.packages("tidymodels", repos = "http://cran.us.r-project.org")
#for cleaning variable names
if(!require(janitor))
  install.packages("janitor", repos = "http://cran.us.r-project.org")
#for multinomial regression
if(!require(glmnet))
  install.packages("glmnet", repos = "http://cran.us.r-project.org")
#ranger is random forest algorithm wrapper
if(!require(ranger))
  install.packages("ranger", repos = "http://cran.us.r-project.org")
#kernlab is svm kernel wrapper
if(!require(kernlab))
  install.packages("kernlab", repos = "http://cran.us.r-project.org")
#for variable importance
if(!require(vip))
  install.packages("vip", repos = "http://cran.us.r-project.org")
#zoo allows rolling operations
if(!require(zoo))
  install.packages("zoo", repos = "http://cran.us.r-project.org")
#matrix stats
if(!require(matrixStats))
  install.packages("matrixStats", repos = "http://cran.us.r-project.org")
#rpart.plot shows the decision tree of an rpart result
if(!require(rpart.plot))
  install.packages("rpart.plot", repos = "http://cran.us.r-project.org")
#kableExtra allows more customization of tables
if(!require(kableExtra))
  install.packages("kableExtra")
if(!require(RColorBrewer))
  install.packages("RColorBrewer", repos = "http://cran.us.r-project.org")
#for dark background plots
if(!require(ggdark))
  install.packages("ggdark", repos = "http://cran.us.r-project.org")
#easy formatting of numbers as currency & percent
if(!require(formattable))
  install.packages("ggdark", repos = "http://cran.us.r-project.org")
```

Importing the Data

There are five files I'll import.

```r
#specify columns because otherwise birth year is read as logical
cols_for_stats=cols(
  .default = col_double(),
  player = col_character(),
  hof = col_logical(),
  pos = col_character(),
  lg = col_character(),
  tm = col_character()
)

advanced<-read_csv("Data/Advanced.csv",col_types = cols_for_stats) %>%
  select(seas_id:mp,ows:ws,vorp)
totals<-read_csv("Data/Player Totals.csv",col_types = cols_for_stats)
#max games per season for players on multiple teams
max_games_tots=totals %>% filter(tm=="TOT") %>% group_by(season,lg,tm) %>%
  summarize(max_games_tot=max(g,na.rm = TRUE)) %>% ungroup()
#max games per season for players on single team
max_games=totals %>% filter(tm !="TOT") %>% group_by(season,lg) %>%
  summarize(max_games_non_tot=max(g,na.rm = TRUE)) %>% ungroup()
#coalesce above two into one column in totals df
totals_enhanced=left_join(totals,max_games_tots) %>% left_join(.,max_games) %>%
  mutate(max_games_playable=coalesce(max_games_tot,max_games_non_tot)) %>%
  select(-c(max_games_non_tot,max_games_tot))
advanced_and_totals<-left_join(totals_enhanced,advanced) %>%
  #if player played for multiple teams in season, only take total row
  mutate(tm=ifelse(tm=="TOT","1TOT",tm)) %>%
  group_by(player_id,season) %>% arrange(tm) %>% slice(1) %>%
  mutate(tm=ifelse(tm=="1TOT","TOT",tm)) %>%
  arrange(season,player) %>%
  mutate(g_percent=g/max_games_playable*100,gs_percent=gs/g*100,.before=g) %>%
  select(-c(gs,max_games_playable)) %>%
  #filter for only last ten years for faster pre-processing
  filter(season > 2009) %>% ungroup()
```

For the statistical data, I've scraped total and advanced stats from Basketball-Reference and stored them in .csv files. The advanced stats I kept were cumulative (offensive win shares, defensive win shares and value over replacement player). This was actually part of a larger project to scrape complete statistics for teams, players and awards (the Kaggle dataset resides here). To my knowledge, my dataset is unique in that it includes BAA stats and ABA stats, which is not really of use here.

For players who played on multiple teams in one season, I kept their total stats and discarded the team-specific season portions. In the previous iteration of this project, we scaled games played and games started to a normal distribution due to fluctuations in games played between seasons caused by the COVID-19 pandemic. There was an initial desire to use totals to bake in availability/body fragility, but the shortened seasons would cause the model to declare all players to be fragile and underestimate their contract. We will convert the games started to a percentage of games played and we will change the games played to a percentage of maximum playable games. This maximum will differ for players who played for multiple teams in one season.

```r
past_free_agents<-read_csv("Data/2016-2020 Free Agents.csv")
```

I've updated the free agents training set from last year to include the 2020 free agents. As a quick refresh:

- removed retired players and players who signed from overseas (wouldn't have any contract year data) from the dataset
- set contract years to zero and salary to zero for players who:
  - went overseas (not many, considering the pandemic)
  - had explicitly non guaranteed first years in their contracts (training camp deals, two ways, ten days, exhibit 10s)
  - had blanks in their contract terms cell
- included option years and partially guaranteed years in my calculation of contract years
  - looked at it as both sides (player and team) intending to see out the contract
- gathered year 1 salary for remaining players using Spotrac, Basketball-Reference or Basketball-Insiders:
  - Capology was previously my main source, but it wasn't updated when I started compiling the 2020 free agents
  - Spotrac became the new main source as it cost nothing to look at the current year salary
  - Basketball-Reference has a transaction timeline, which is ideal for players who were waived/played for multiple teams
  - when all other sources were exhausted, turned to Basketball-Insiders
- general housekeeping like adding suffixes (Jr., II, III) to certain players to match up with statistical data

```r
#subtract one from year to match up with offseason in which contract was signed
salary_cap_hist<-read_csv("Data/Salary Cap History.csv") %>% mutate(season=season-1)
#create variable of first year salary as percentage of cap
#easier to compare across years
past_free_agents<-past_free_agents %>% select(-c(terms,Source)) %>%
  left_join(.,salary_cap_hist) %>%
  mutate(first_year_percent_of_cap=yr_1_salary/cap) %>%
  select(-c(yr_1_salary,cap))
```

The next file I used was salary cap history, also from Basketball-Reference. To somewhat normalize comparisons across years, I converted the first year salary to a percentage of the salary cap.

```r
current_fa<-read_csv("Data/Free Agents 2021.csv")
#separate out options to compare what players options get if declined
current_fa_options<-current_fa %>% filter(str_detect(type,"PO|CO")) %>%
  select(-c(experience,contract_yrs)) %>%
  rename(option_type=type,option_amt=first_year_percent_of_cap) %>%
  mutate(option_amt=currency(option_amt,digits=0))
#make player options all declined (UFA's)
#make club options ufa or rfa depending on exp
current_fa<-current_fa %>%
  mutate(type=case_when((type=="PO"|(type=="CO" & experience >= 4))~"UFA",
                        (type=="CO" & experience < 4)~"RFA",
                        TRUE~type)) %>%
  group_by(player) %>% select(-experience) %>% slice(1) %>% ungroup() %>%
  mutate(first_year_percent_of_cap=NA)
```

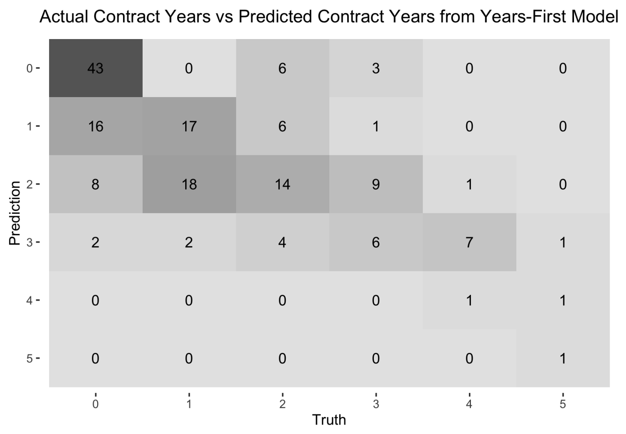
The last file loaded is our evaluation set: the 2021 free agent class, retrieved from Spotrac. Since the deadlines for some option decisions are all the way until August 1 and I wanted to run my models before then, I decided to scrape Spotrac on **June 30** (subject to change based on how fast models can be run). I had to edit this dataset to match the Basketball-Reference names (mainly adding diacritics to European names). In addition, I filtered out players with options. Players who decline player options and players who have their team options declined with more than 3 years of experience become unrestricted free agents. Players with less than or equal to 3 years of experience and a declined team option become restricted free agents. I'll use this fact to see which players & teams might decline their option. 2022 salary amounts of players with options were added manually.

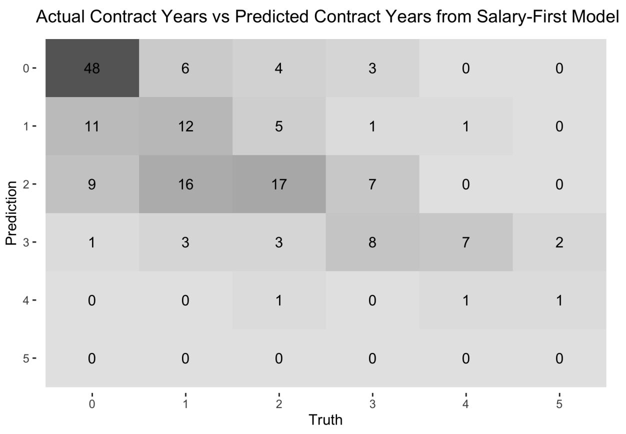
In the GitHub repository where this project is located, a file called `free agents.r` has more details on exactly how I scraped the train set, evaluation set and the salary cap history.

Retrospective on Last Year's Results

Before getting into pre-processing, we'll take a look at last year's results and see how the algorithms performed. We remove players who ended up not hitting the market due to them picking up their player option. Any player who was predicted but didn't end up in the free agent history must not have received a contract, so their NA's were replaced with zeroes.

The years accuracy of the years-first model was 49.1%, while the years accuracy of the salary-first model was 51.5%. Here's some confusion matrices on how each model handled the prediction of contract years.





The incorrect predictions were skewed toward predicting more years than received as evidenced by the sum of the lower triangle being greater than the upper triangle (models forecasting one year contracts for players who didn't receive a contract, two years for players who got one year, etc).

Here are the worst misses for both models.

player	yrs_y1s2	contract_yrs
Jordan McLaughlin	3	0
Thon Maker	3	0
Josh Gray	0	3
Drew Eubanks	0	3
Zylan Cheatham	0	3

player	yrs_s1y2	contract_yrs
Jordan McLaughlin	3	0
John Konchar	1	4
Josh Gray	0	3
Drew Eubanks	0	3
Zylan Cheatham	0	3

Gray, Eubanks & Cheatham were all deemed out-of-the-league material by both models, but the trio ended up securing 3 year contracts. Gray & Cheatham actually signed their contracts so they could be used as salary filler in the 4-team trade that sent Jrue Holiday to Milwaukee that sent the pair to the Oklahoma City Thunder. Gray and Cheatham were eventually waived a week after the trade. Eubanks was a bit player in the San Antonio Spurs rotation, averaging 14.4 minutes (12th on the team) & 5.8 points (10th).

McLaughlin and Maker were the opposite: predicted to land 3-year contracts and not receiving any full offers. McLaughlin re-signed on a two-way contract with the Minnesota Timberwolves. Maker signed an Exhibit 10 contract with the Cleveland Cavaliers and ended up making the opening night roster, but he was waived after 8 games. Konchar was surprisingly locked up to a 4-year, \$ 9M contract by the Memphis Grizzlies when the salary-first model only had him getting a single year.

Let's shift our focus to the salary predictions. First, the residual mean squared error of the years-first model was 0.030047, while the salary-first model had an RMSE of 0.0284566.

As we did with the years models, let's look at the most extreme salary misses.

player	y1s2_cap_percent	first_year_percent_of_cap
Hassan Whiteside	19.42%	2.13%
Montrezl Harrell	19.20%	8.48%
Gordon Hayward	17.89%	26.11%
Serge Ibaka	15.36%	8.48%
Jordan McLaughlin	6.63%	0.00%
Thon Maker	6.31%	0.00%
Goran Dragić	10.31%	16.49%
Shabazz Napier	6.02%	0.00%
Marc Gasol	8.27%	2.35%
Carmelo Anthony	8.27%	2.35%

player	s1y2_cap_percent	first_year_percent_of_cap
Hassan Whiteside	17.23%	2.13%
Montrezl Harrell	17.82%	8.48%
Gordon Hayward	18.12%	26.11%
Marc Gasol	9.69%	2.35%
Carmelo Anthony	8.95%	2.35%
Glenn Robinson III	8.06%	1.86%
Jerami Grant	11.39%	17.45%
Shabazz Napier	5.99%	0.00%
Furkan Korkmaz	5.90%	0.00%
Serge Ibaka	14.17%	8.48%

Harrell, Gasol & Ibaka seemingly took less money in exchange for a greater shot at a championship ring. Ibaka went to the Los Angeles Clippers, while Gasol joined the Los Angeles Lakers. Harrell actually didn't even change cities, swapping Clipper blue for Laker purple across the hall. Anthony returned to Portland, repaying the faith the Trail Blazers showed in him when they extended a contract offer to him during the 2020 season. Whiteside didn't receive any competitive offers with his reputation as an "empty-stats" player, and returned to his old stomping grounds in Sacramento with a near-minimum contract in hand. Upon further research, Korkmaz actually wasn't a free agent, having signed a 2-year contract in 2019. Dragić received a "legacy" contract from the Miami Heat, a 2-year balloon deal to thank him for his years of service with a team option on the 2nd year to maintain flexibility. Robinson signing with the Sacramento Kings on a 1-year, \$ 2M contract was surprising, as his potential as a rangy 3 & D wing should have enticed championship contenders.

On a more positive note, here's some players on which the models were very close on.

player	y1s2_cap_percent	first_year_percent_of_cap
JaKarr Sampson	1.75%	1.73%
Denzel Valentine	4.22%	4.25%
Jeff Green	2.40%	2.35%
Noah Vonleh	1.80%	1.86%
Avery Bradley	5.26%	5.16%
Fred VanVleet	19.61%	19.47%
Udonis Haslem	2.49%	2.35%
Harry Giles	1.69%	1.54%
Anthony Davis	30.15%	30.00%
Raul Neto	1.52%	1.73%

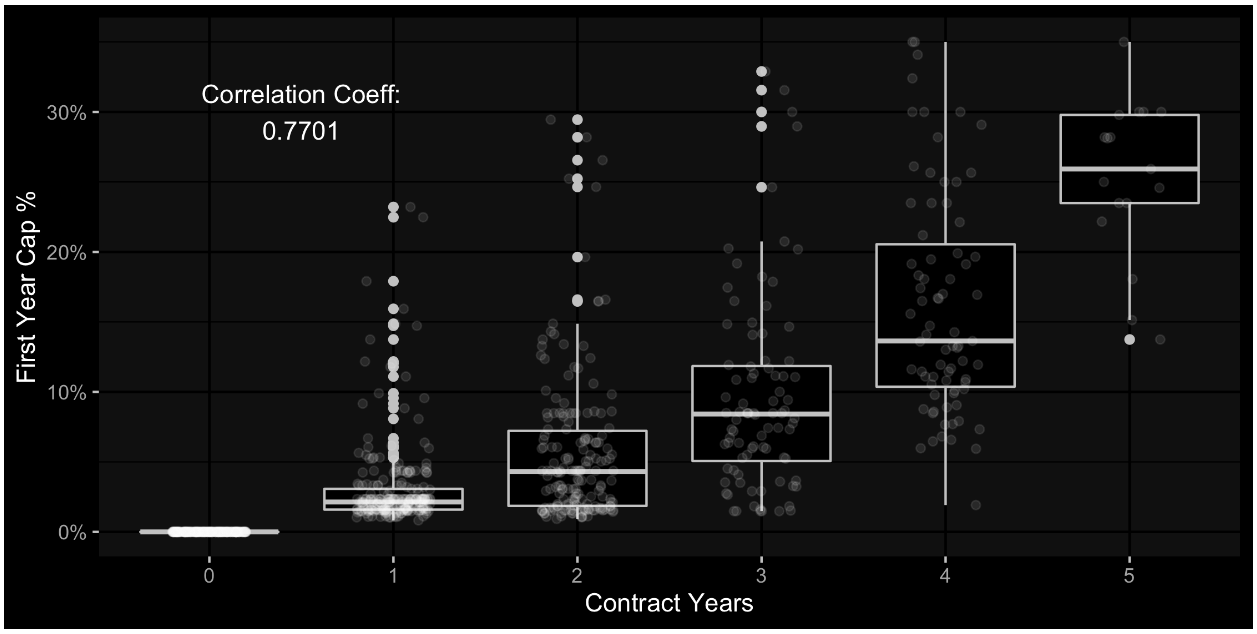
player	s1y2_cap_percent	first_year_percent_of_cap
Jordan Clarkson	10.62%	10.54%
Jakob Poeltl	7.62%	7.42%
Nicolas Batum	2.60%	2.35%
Patrick Patterson	2.56%	2.82%
Paul Millsap	8.89%	9.16%
Solomon Hill	1.72%	1.99%
J.J. Barea	2.05%	2.35%
James Ennis	3.32%	3.02%
Brandon Ingram	24.66%	25.00%
Chris Boucher	5.62%	5.96%

12 of the 20 closest predictions were for contracts that took up less than 5% of the salary cap. Notes on the remaining 8:

- Davis & Ingram re-signed on maximum contract extensions
- VanVleet epitomized the Toronto Raptors' challenging 2020 season in their makeshift home of Tampa Bay, exhibiting both peaks (a career- & franchise-high 54 points against the Orlando Magic in February) as well as valleys (shooting a league-worst 38.9% from the field for the entire season)
- Clarkson was a scoring machine off the bench for the Utah Jazz, eventually winning the Sixth Man of the Year award
- Boucher broke out, receiving votes for both the Sixth Man of the Year & the Most Improved Player awards and earning the nickname "Block Quebecois" for his ferocious shot-swatting prowess
- Bradley signed a 2-year deal with the Miami Heat with the expectation of being a defensive pest off the bench, but his season was beset by injuries and he was eventually traded to the Houston Rockets as part of a package to land Victor Oladipo
- Millsap provided a steady veteran presence to the Denver Nuggets, ceding minutes to young up-and-coming forward Michael Porter Jr. as well as trade deadline acquisition Aaron Gordon. This reduced role resulted in Millsap's scoring average falling below 10 points for the first time in thirteen seasons.
- Poeltl began the season on the bench for the San Antonio Spurs before being thrust into the starting lineup after the Spurs agreed to a buyout with LaMarcus Aldridge. He averaged career highs across the board, including minutes per game (26.7), rebounds per game (7.9) and points per game (8.6).

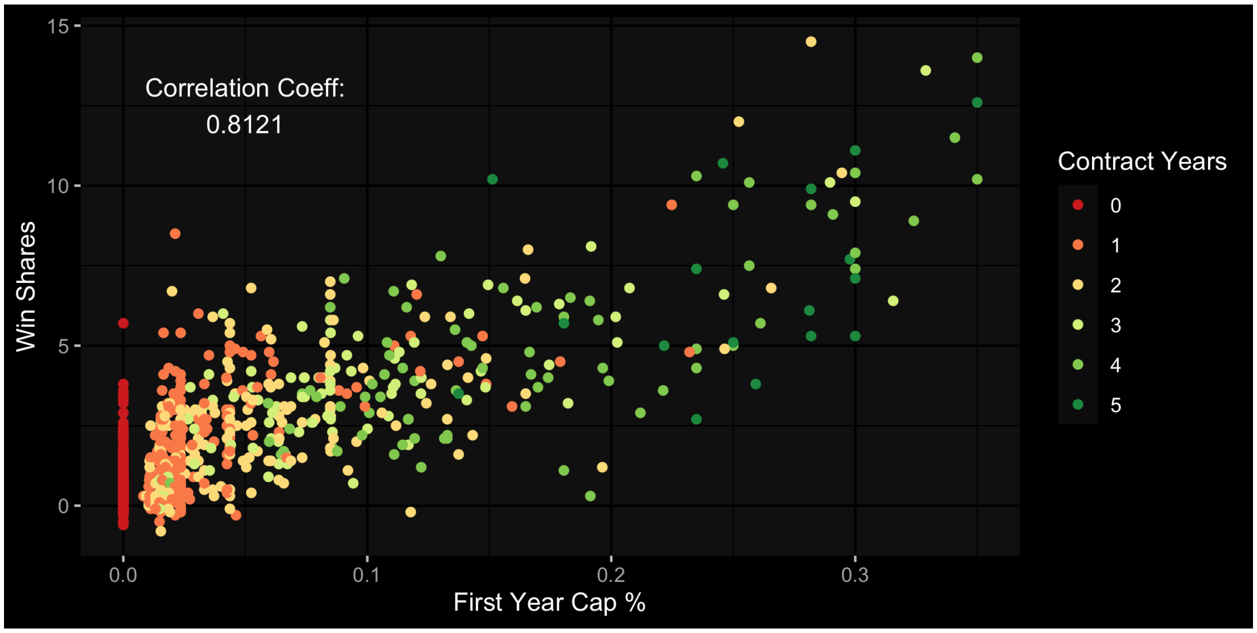
Data Exploration and Visualizations

Before we train models, let's see how some of the predictors and some of the targets interact. First, let's see how our targets correlate with each other. I've created a box and whisker plot, as well as added the points themselves in a transparent layer with some random variation to differentiate between points.



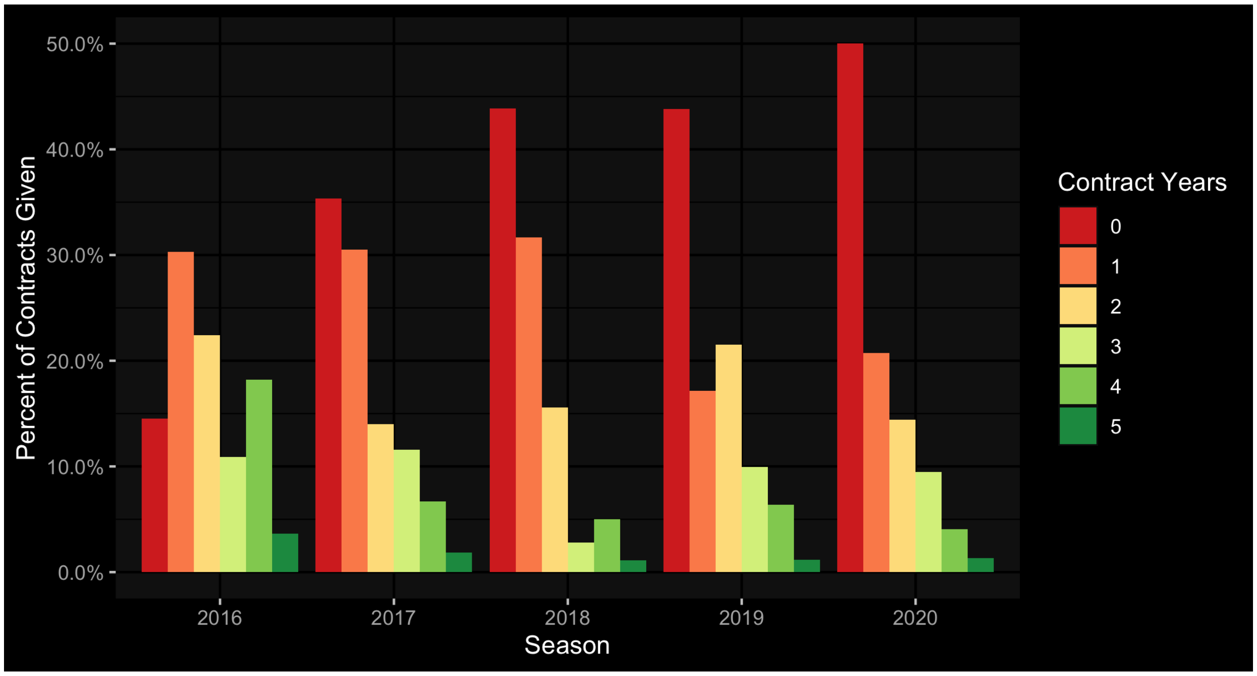
The correlation coefficient is 0.77, which shows that the two targets are strongly and positively correlated. The median value of the first year cap % (middle line in each box) increases with an increase in contract length. The highest increase in first year cap % is between a 4-year and 5-year contract.

Next, let's see if an all-encompassing advanced statistic has a relationship with first year cap percentage. Win Shares represent how much a player has contributed to his team's wins by comparing his output to a marginal player. Higher win shares generally indicate a better player.



With a correlation coefficient of 0.8121, win shares are highly correlated with first year cap percentage. This shouldn't be too groundbreaking: better players get paid more.

Finally, let's see how many contracts of each length were given in each offseason.



It comes as no surprise that as contract length increases, the percent of contracts of that length given out decreases. Although in 2016, the amount of players who didn't receive contracts was lower than the amount who received 1, 2, or 4 year contracts. 2016 was the year of the cap spike, when the salary cap jumped from $ 70 million to $ 94 million. Similar to a lot of people with new-found money, teams spent somewhat recklessly.

Pre-Processing

```
create_data<-function(x){
  a<-advanced_and_totals %>% group_by(player_id) %>%
    #three year sum
    mutate(across(-c(1:10,fg_percent,x3p_percent,
                     x2p_percent:e_fg_percent,ft_percent),
                  list(three_yrs=~rollapplyr(.,3,sum,partial=TRUE)),
                  .names="{col}_last_3_yrs")) %>%
    inner_join(.,x) %>%
    mutate(ws_per_48_last_3_yrs=ws_last_3_yrs/mp_last_3_yrs*48) %>%
    mutate(fg_percent_last_3_yrs=
             ifelse(fga_last_3_yrs==0,0,fg_last_3_yrs/fga_last_3_yrs),
           x3p_percent_last_3_yrs=
             ifelse(x3pa_last_3_yrs==0,0,x3p_last_3_yrs/x3pa_last_3_yrs),
           x2p_percent_last_3_yrs=
             ifelse(x2pa_last_3_yrs==0,0,x2p_last_3_yrs/x2pa_last_3_yrs),
           e_fg_percent_last_3_yrs=
             ifelse(fga_last_3_yrs==0,0,
                  (fg_last_3_yrs+0.5*x3p_last_3_yrs)/fga_last_3_yrs),
         ft_percent_last_3_yrs=
           ifelse(fta_last_3_yrs==0,0,ft_last_3_yrs/fta_last_3_yrs)) %>%
    #remove categories that aren't predictive vars or linear combo of others
    select(-c(hof,lg,pos,birth_year,tm,
              trb,trb_last_3_yrs,
              fg,fga,fg_last_3_yrs,fga_last_3_yrs,
              pts,pts_last_3_yrs)) %>%
    #convert contract year and last 3 year stats to per game (except games)
    mutate(across(c(mp,x3p:x3pa,x2p:x2pa,ft:fta,orb:pf),list(per_game=~./g)),
           .after="gs_percent") %>%
    select(-c(g,mp,x3p:x3pa,x2p:x2pa,ft:fta,orb:pf,ws)) %>%
    mutate(across(mp_last_3_yrs:pf_last_3_yrs,list(per_game=~./g_last_3_yrs)),
           .after="gs_percent_last_3_yrs") %>%
    select(-c(g_last_3_yrs,mp_last_3_yrs:pf_last_3_yrs,ws_last_3_yrs)) %>%
    ungroup() %>% mutate(contract_yrs=factor(contract_yrs,levels = 0:5)) %>%
    replace_na(list(fg_percent=0,x3p_percent=0,x2p_percent=0,
                    e_fg_percent=0,ft_percent=0))
  return(a)
}
```

I used regular season stats, although I do understand that some players get paid on the strength of playoff performance. I started off with contract year stats, because there's anecdotal evidence that players exert more effort in their contract year (*cough cough Hassan Whiteside*). All stats except for the advanced stats (OWS, DWS and VORP) were converted to per game. Percentages were left alone.

In addition to using contract year stats, I summed the past two years and the contract year.

Why I settled on 3 years:

- Players do get paid on past performance, so just using contract year stats was out of the question
- 2 years opens up the possibility of a fluke year
  - Kawhi would have his nine game season bring down his averages significantly from his Raptors season: adding another year somewhat lessens this effect
- On the other hand, it's quite unlikely that teams factor in stats from more than 4 years ago, a lot would have changed
  - the Trail Blazers didn't pay Carmelo to recapture his form of his 2013 year where he led the league in scoring (I would hope)
- Another reason I settled on 3 years is that I can keep the same model for restricted free agents
  - my thought is that the rookie year is a bonus: great if you did well, but doesn't matter in the grand scheme of things if you did poorly
  - rookie extension is more based on how you improved over the course of that initial contract
  - For example, if Luka Dončić had a worse rookie year but had the same level of play that he has achieved in his second and third year (as well as next year), I highly doubt that Dallas would offer him a significantly less amount of money due to that substandard rookie year

I performed the same processing on the three-year totals, using the three-year game total as the denominator for converting to per game. I had to calculate the three-year percentages, and also re-engineered the win shares per 48 minutes metric.

I removed categories that were linear combinations of one another. For example, total rebounds can be found by simply adding up offensive and defensive rebounds, and points are just the result of 2*(number of

2 point field goals made) & 3*(number of 3 point field goals made). I kept age and experience as predictor variables, but removed position because I felt it would ultimately reflect in the stats themselves.

I changed contract years from a numeric column to a factor/category column. This changes its prediction from a regression problem to a classification problem. A 2.5-year contract makes no sense, so it is in our best interest to discretize the years and store them as factors rather than round a regression result.

The final step was to replace missing values in the shooting percentages with zeroes. These NA's were originally due to lack of attempts.

```
train_set<-create_data(past_free_agents)
eval_set<-create_data(current_fa)
rm(advanced_and_totals,past_free_agents)
```

Since the models depend on contract year stats, these players were removed from the evaluation set:

player	type
Deonte Burton	UFA
Troy Daniels	UFA
Zach Collins	RFA

Training Models

There is no need for a subset of the training set to be withheld as a test set before running the models on the evaluation set, because there is built-in cross validation. In the previous iteration of this project, I utilized leave-one-out cross validation since the dataset is relatively small (<1000 observations). How this works is that the model is run excluding one observation. Then, the model attempts to predict the result of that excluded observation. This is repeated for every observation. However, on this run, I've decided to use k-fold cross validation. I'm hoping to get similar or even better results while cutting down significantly on training time.

As we saw in data visualization, the two target variables (contract years and first year salary as a percentage of the salary cap) are fairly well correlated, as they have a Pearson correlation coefficient of 0.77. The way I chose to handle this is:

- predict one target first without the other as a predictor
- choose the best model (be that a single model or an ensemble of multiple models)
- use the first target's predictions as an input to predict the second target

One potential problem is compounding errors. If there's an incorrect year prediction, it might lead to an incorrect salary prediction.

The Models

I used a total of six models.

- linear regression model as a baseline for salary, and multinomial regression as a baseline for years
  - the separation is due to the classification/regression split
- k-nearest neighbors model: take the distance between the statistics of two players (the absolute value of the difference) and then take the average of the outcome variable of the k nearest neighbours
  - the intuition being that similar players get similar contracts

- decision tree model (`rpart`): maybe as a player passes certain statistical thresholds, their contract increases
  - only using for predicting the contract years; since there are so many different salary percentages, a solitary decision tree would either be useless or far too complicated
- random forest model (`ranger`): reduces instability by averaging multiple trees
  - costs interpretability as there is no tree diagram that is representative of the decisions made
- support vector machine model: attempt to separate classes with a hyperplane
  - support vectors are the points closest to the hyperplane, named as such because the hyperplane would change if those points were removed
  - I believe this image from Wikipedia succinctly explains an SVM

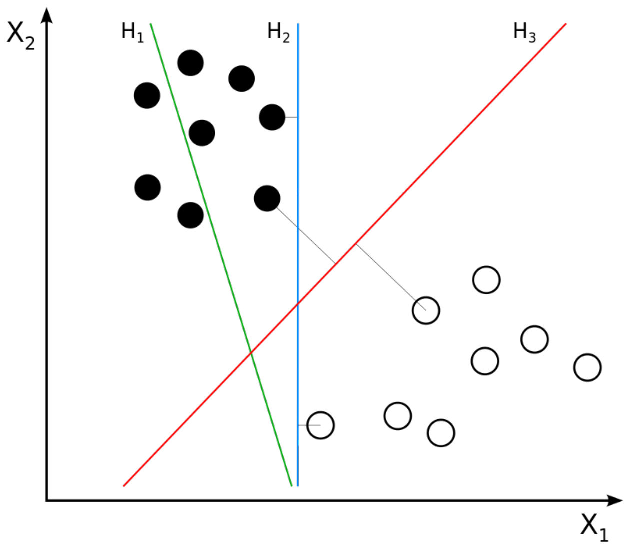


Figure 1: H1 does not separate the classes. H2 does, but only with a small margin. H3 separates them with the maximal margin. By User:ZackWeinberg, based on PNG version by User:Cyc - This file was derived from: Svm separating hyperplanes.png, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=22877598

Predicting Years First, then Salary

We'll start by predicting years first and then salary with years as an input.

With caret, everything could be thrown into the train function: cross-validation, tuning, data, etc. Tidymodels is more explicit in its steps. The standard 10-fold cross validation is how the data is going to be resampled. In addition, we'll stratify selection, so there is sufficient data to predict all contract year lengths. A hypothetical (albeit unlikely) scenario that could occur without stratification is a fold could include all instances of restricted free agents (which are more rare) in the test portion, and none in the train portion. The chosen algorithm would be confused as to how to predict something for which it has no training.

Recipes is tidymodels' pre-processing package. The first step to change the roles of some variables. The season_id, season, player_id & player variables are of no use as predictors, but are useful to identify observations, so they are given the "id" role. Since we are predicting years first, the first_year_percent_of_cap variable will be removed. Finally, we will convert the free agent type column to a numeric column.

```r
cv=vfold_cv(train_set,v=10,strata=type,repeats=5)
y1_recipe=recipe(contract_yrs~.,data=train_set) %>%
  update_role(seas_id:player,new_role="id") %>%
  step_rm(first_year_percent_of_cap) %>% step_dummy(type)
```

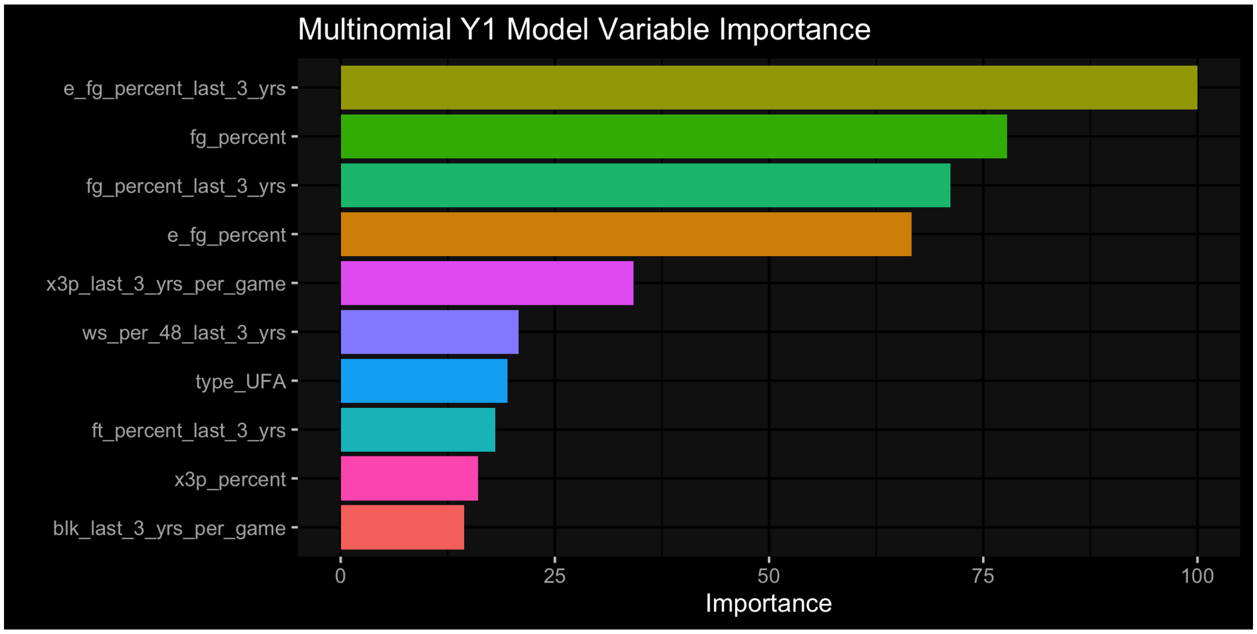
Initially, accuracy was chosen as the metric to determine the best submodel by cross-validation. However, with the inherent imbalance of the outcome classes, the F1 score is a better metric. As an extreme example, if there were only two classes with a 90:10 split, a classifier could achieve 90% accuracy by simply predicting the more populous class for every case. On the other hand, the F1 score attempts to minimize both false positives & false negatives. However, the default averaging for F1 is macro-averaging, which gives equal weights to all classes and is exactly the problem we were trying to distance ourselves from by not choosing accuracy. We need to change the averaging function to be macro-weighted. In order to include it within yardstick's metric_set, we have to create a new function wrapping the metric, set the options within the wrapper & formalize it as a new metric.

```r
macro_weight_f1 <- function(data,truth,estimate,beta = 1,estimator="macro_weighted",
                            na_rm = TRUE,
                            event_level = yardstick_event_level(),...){
  f_meas(data=data,
         truth = !! rlang::enquo(truth),
         estimate = !! rlang::enquo(estimate),
         estimator = "macro_weighted",
         beta=beta,
         na_rm=na_rm,
         event_level=event_level,
         ...
         )
}
macro_weight_f1 <- new_class_metric(macro_weight_f1,"maximize")
```

First up is the multinomial regression model.

```r
wf=workflow() %>% add_recipe(y1_recipe) %>%
  add_model(multinom_reg(penalty=0) %>%
              set_engine("glmnet") %>%
              set_mode("classification"))
tune_multinom=wf %>% tune_grid(resamples=cv,
                               metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_multinom %>% select_best())
fit_y1_multinom=fit(wf,train_set)
```

Let's get the most important variables of the multinomial model.



Effective field goal percentage & field goal percentage in both their current-year & last three-years forms comprise the top 4 most important variables.

Next up is the k-nearest neighbors model.

```
wf=workflow() %>% add_recipe(y1_recipe) %>%
  add_model(nearest_neighbor(neighbors = tune()) %>%
  set_engine("kknn") %>% set_mode("classification"))
tune_knn=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(neighbors=5:50),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_knn %>% select_best())
fit_y1_knn=fit(wf,train_set)
tune_knn %>% select_best() %>% knitr::kable()
```

neighbors	.config
50	Preprocessor1_Model46

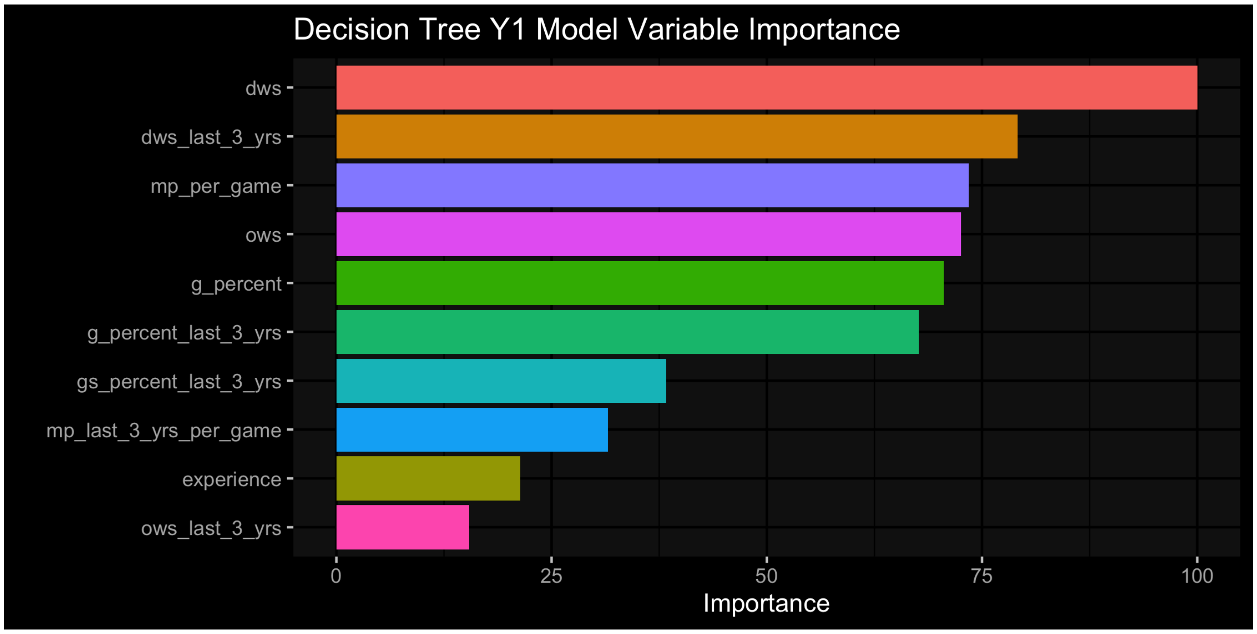
Looks like the best knn model is right at the upper limit of the tune grid with using 50 closest comparable players.

The next model is the decision tree model. We can tune the cost complexity parameter (cp) in this model. CP is the minimum for how much the residual sum of squares must improve for another partition to be added. A CP that is too high will have too few branches, while a CP that is too low will be difficult to follow since there are many branches. We lean towards the lower end of the spectrum. We will also change the min_n, which is the minimum number of data points in a node that are required for the node to be split further. The default is 20, but the 5-year portion of the dataset is small, so we will decrease min_n to 1. Since there exists an element of randomness in choosing samples to model a decision tree, we need to set a seed to keep the work reproducible.

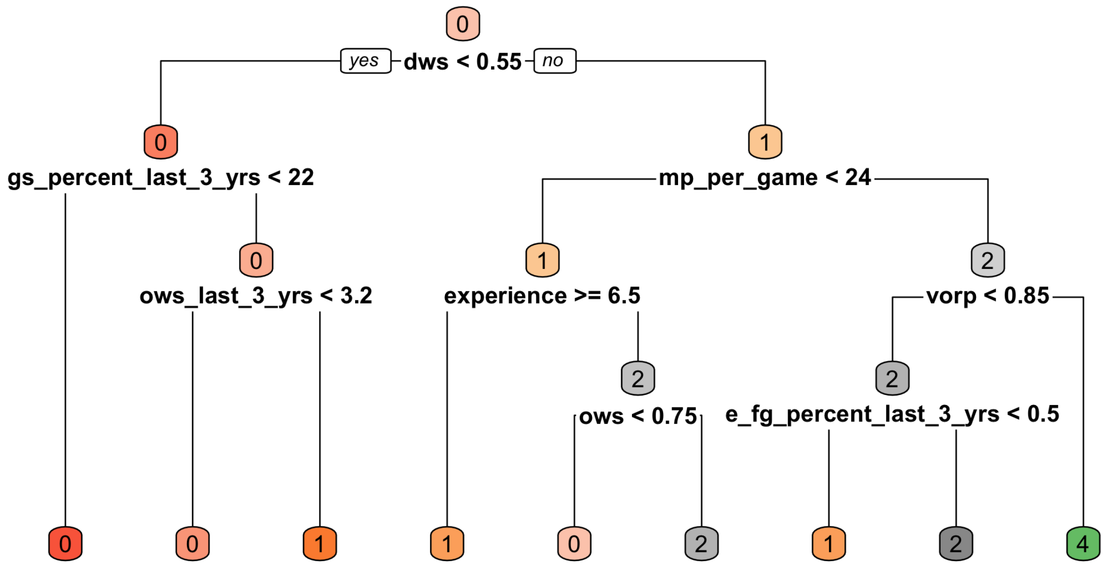
```
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(y1_recipe) %>%
  add_model(decision_tree(cost_complexity = tune(),min_n=tune()) %>%
  set_engine("rpart") %>% set_mode("classification"))
tune_tree=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(cost_complexity=0.1^seq(2,5),
                                         min_n=seq(1:10)),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_tree %>% select_best())
fit_y1_tree=fit(wf,train_set)
tune_tree %>% select_best() %>% knitr::kable()
```

cost_complexity	min_n	.config
0.01	1	Preprocessor1_Model01

Let's get the most important variables of the decision_tree model.



The individual contract-year components of the holistic advanced stat of win shares rank within the top-4 most important variables. Durability and availabilty are also highly desirable characteristics, as evidenced by the high values of contract-year minutes per game, contract-year games played percentage and last-3-years games played percentage.



The decision tree does not predict any 3-year or 5-year contracts, which is . . . not ideal to say the least.

The decision tree maximizes its prediction when a player does all of the following:

- has defensive win shares above 0.55 in the contract year
- plays more than 24 minutes per game in the contract year
- has a value over replacement player above 0.85

Full disclosure: when I included values > 0.1 in tuning the cost complexity, I got a decision tree with one decision (more of a decision stump to be honest):

- if the contract-year defensive win shares was above 0.55, predict one contract year
- else, predict zero contract years

While the F1 score was higher is this case, the model is practically useless. This is a refreshing example to hammer home the point that one should never blindly follow a solitary metric.

The next model is a random forest. Since `ranger` is a *random* forest algorithm, we need to set a seed to keep the work reproducible.

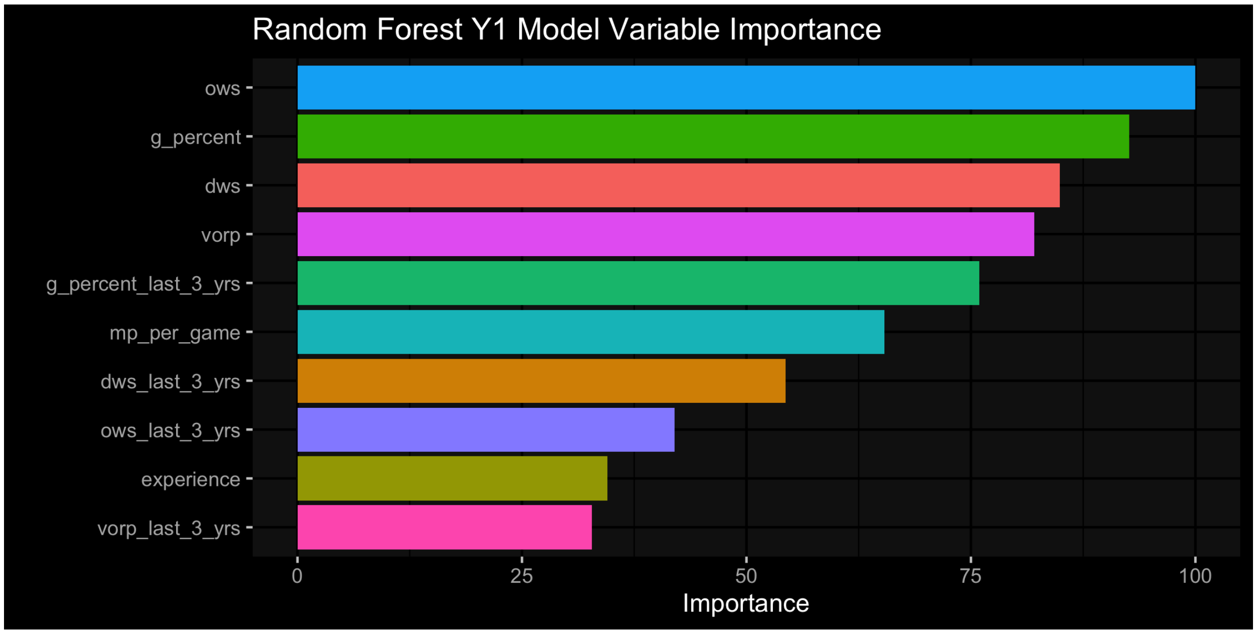
Random forest algorithms require an explicit call for variable importance, so we'll ask for permutation importance. A simplified explanation for permutation importance is shuffling a predictor's values and seeing how much the error increases. As a predictor's importance increases, it is difficult for the rest of the model to compute accurate predictions without it. There are 3 tuning parameters:

- trees is the number of decision trees to create
  - the default is 500, which we'll keep
- mtry is the number of variables to split at each decision node
  - the default is the rounded square root of the number of variables, which in this case would be `round(sqrt(55))`

    - we will try all integers between 5 & 10
- min_n is the same as the decision tree
    - the default is 1 for a classification problem like this, which we'll keep

```r
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(y1_recipe) %>%
  add_model(rand_forest(mtry = tune()) %>%
  set_engine("ranger",importance="permutation") %>% set_mode("classification"))
tune_forest=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(mtry=5:10),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_forest %>% select_best())
fit_y1_forest=fit(wf,train_set)
tune_forest %>% select_best() %>% knitr::kable()
```

mtry	.config
9	Preprocessor1_Model5



The contract year versions of the holistic advanced stats comprise 3 of the top 5 most important variables in the random forest model. The other two variables in the top 5 are percentage of games played in the contract year and percentage of games played in the past 3 years.

Finally, we train the support vector machine on the model. There are two tuning parameters:

- rbf_sigma: determines how well to fit the training data (higher=fit closer to training)
    - with a high sigma, every workaday big with middling stats might be predicted to get close to Mozgov money
- cost: tradeoff between smoothness of boundaries and correct classification
    - with a high cost, leads to too wiggly of a boundary, and might not generalize to test sets
    - tests using C=0.25, C=0.5 and C=1

```
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(y1_recipe) %>%
  add_model(svm_rbf(rbf_sigma=tune(),cost=tune()) %>%
  set_engine("kernlab") %>% set_mode("classification"))
tune_svm=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(rbf_sigma=0.1^seq(2:5),
                                         cost=c(0.25,0.5,1)),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_svm %>% select_best())
fit_y1_svm=fit(wf,train_set)
```

```
## maximum number of iterations reached 3.252306e-05 3.24728e-05maximum number of iterations reached 0.
```

```
tune_svm %>% select_best() %>% knitr::kable()
```

cost	rbf_sigma	.config
0.25	1e-04	Preprocessor1_Model04

Unfortunately, there is no concept of variable importance for an SVM model.

Let's look at some performance metrics, and see which models we want to take along as inputs for predicting salary.

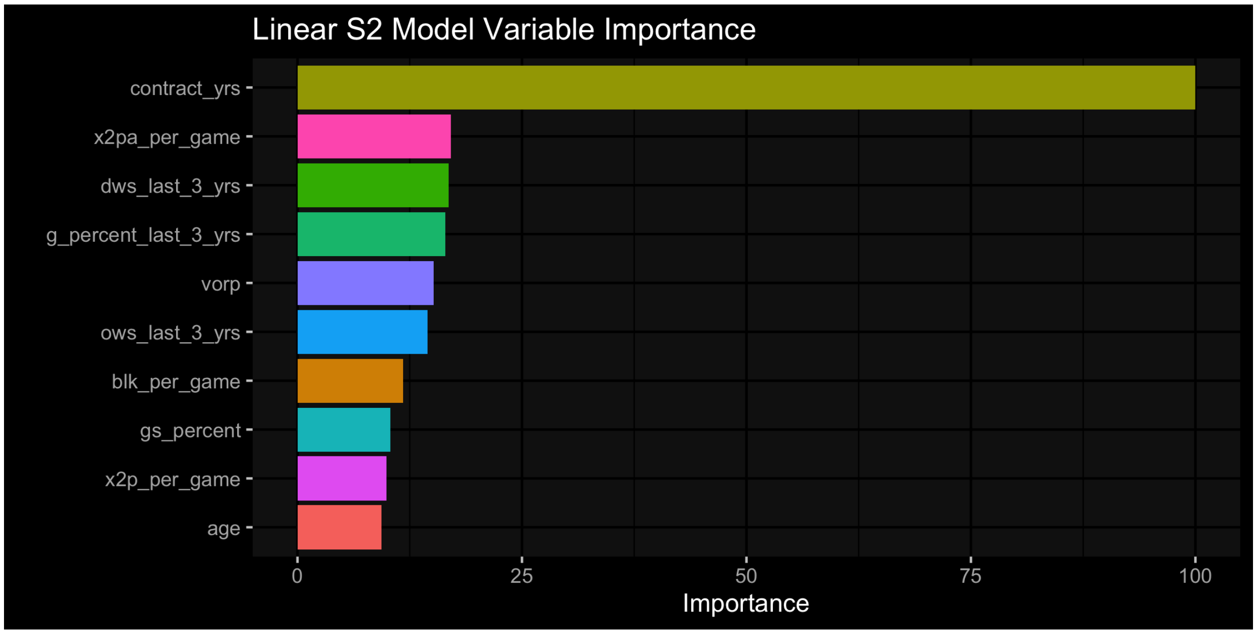
Method	Correct Predict %	Off by >1 Yr	Max Year Predicts	F1 Score
Multinomial	62.73%	12.42%	16	0.4826198
KNN	63.95%	13.54%	3	0.4828747
Decision Tree	54.89%	17.41%	0	0.4640077
Random Forest	98.07%	1.32%	17	0.4979823
SVM	38.90%	36.05%	0	0.5582287

The SVM suffers from the same problem that I talked about when fitting the decision tree. It has the highest F1 score, but the lowest accuracy. In fact, there are almost as many predictions more than 1 year off as there are correct predictions. The random forest stands far above the rest with 98% accuracy! It also has the highest F1-score barring the SVM. In last year's project, the random forests had the best performance as well, but had difficulty distinguishing 5-year contracts. Shifting to a classification problem seems to have solved that. The multinomial model had significantly more max-year predictions than KNN, whereas KNN pipped the multinomial model in F1-score and had ~1% better accuracy. While anticlimactic, I think the best course of action is to simply use the random forest model to make contract-year predictions. We'll convert the prediction column to use it as a numeric predictor.

```
train_set_after_y1=train_set %>% select(-contract_yrs) %>%
  bind_cols(contract_yrs=
              as.numeric(
                as.character(
                  predict(fit_y1_forest,new_data=train_set) %>% pull()
                  )
                )
            )
cv=vfold_cv(train_set_after_y1,v=10,strata=type,repeats=5)
s2_recipe=recipe(first_year_percent_of_cap~.,data=train_set_after_y1) %>%
  update_role(seas_id:player,new_role="id") %>% step_dummy(type)
```

Now I can run through the models for salary using the predicted years as an input. The models I won't reuse are the multinomial model and the decision tree. The multinomial model is for classification only, while we are attempting to predict a continuous outcome in the first year salary as a percentage of the salary cap. We'll sub in a linear regression model as our new baseline. Since there's so many different salary percentages, a decision tree model would be useless or far too complicated.

```
wf=workflow() %>% add_recipe(s2_recipe) %>%
  add_model(linear_reg() %>%
              set_engine("lm") %>%
              set_mode("regression"))
tune_lin=wf %>% tune_grid(resamples=cv,metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_lin %>% select_best(metric="rmse"))
fit_s2_lin=fit(wf,train_set_after_y1)
prettify_vip(fit_s2_lin,"Linear S2 Model Variable Importance")
```



Contract years dwarf all other variables in terms of importance.

```
wf=workflow() %>% add_recipe(s2_recipe) %>%
  add_model(nearest_neighbor(neighbors=tune()) %>%
              set_engine("kknn") %>%
              set_mode("regression"))
tune_knn=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(neighbors=5:50),
                        metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_knn %>% select_best(metric="rmse"))
fit_s2_knn=fit(wf,train_set_after_y1)
tune_knn %>% select_best(metric="rmse") %>% knitr::kable()
```

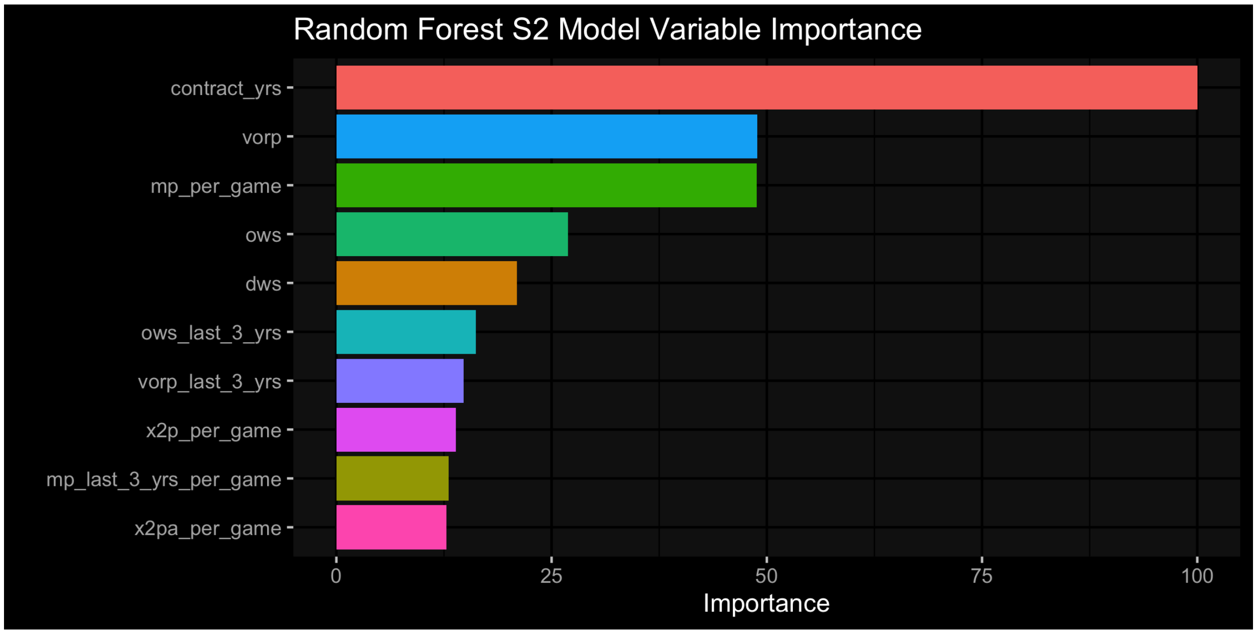
neighbors	.config
21	Preprocessor1_Model17

The best knn salary model is around the middle of the tune grid, using 22 comparables.

```r
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(s2_recipe) %>%
  add_model(rand_forest(mtry = tune()) %>%
  set_engine("ranger",importance="permutation") %>% set_mode("regression"))
tune_forest=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(mtry=5:10),
                        metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_forest %>% select_best(metric="rmse"))
fit_s2_forest=fit(wf,train_set_after_y1)
tune_forest %>% select_best(metric="rmse") %>% knitr::kable()
```

mtry	.config
10	Preprocessor1_Model6

```r
prettify_vip(fit_s2_forest,"Random Forest S2 Model Variable Importance")
```



Like the linear model, contract years are by far the most important variable in the random forest model, although not to the same extent. The 2nd-thru-5th most important variables are all from the contract-year subset of statistics, 3 of these being advanced stats.

```r
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(s2_recipe) %>%
  add_model(svm_rbf(rbf_sigma=tune(),cost=tune()) %>%
  set_engine("kernlab") %>% set_mode("regression"))
tune_svm=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(rbf_sigma=0.1^seq(2:5),
                                         cost=c(0.25,0.5,1)),
                        metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_svm %>% select_best(metric="rmse"))
fit_s2_svm=fit(wf,train_set_after_y1)
tune_svm %>% select_best(metric="rmse") %>% knitr::kable()
```

cost	rbf_sigma	.config
1	0.01	Preprocessor1_Model10

Here are the performance metrics for the salary portion of the Y1S2 model.

Method	Off By >5%	Within 2%	MAE	RMSE
Linear	6.42%	71.59%	0.0186438	0.0270035
KNN	6.82%	76.99%	0.0186072	0.0305135
Random Forest	0.31%	91.75%	0.0158618	0.0266997
SVM	4.89%	83.30%	0.0157178	0.0267516

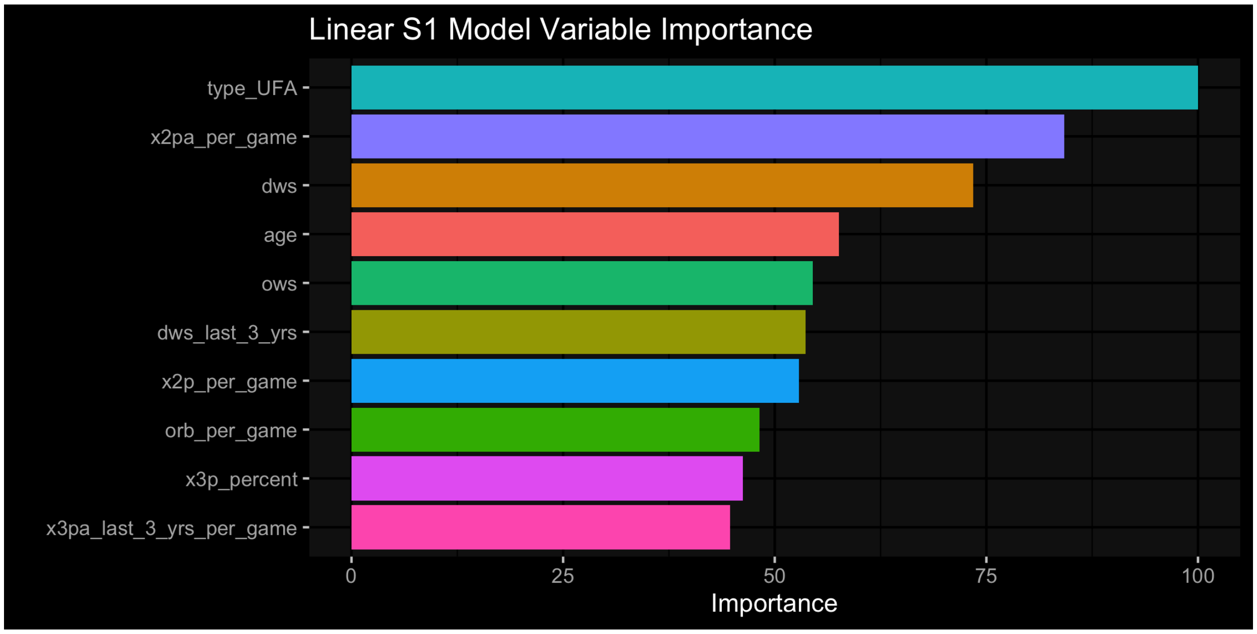
Mean absolute error is the measure of the average difference between forecasts, while the residual mean squared error penalizes large errors. The random forest has the best RMSE, and also tops the leaderboard of maximizing predictions within 2% of actual value, and minimizing predictions that are more than 5% away. The linear model also has a very close RMSE to the random forest & SVM, but it suffers on the dataset-specific metrics. The KNN model has the worst RMSE, but still performs better on predictions than the linear model. With the SVM & random forest being so close in traditional metrics as well as the SVM being no slouch on dataset-specific metrics, we will take the mean of the SVM & random forest as our salary prediction in the Y1S2 model.

Predicting Salary First, then Years

Now I'll switch up the order, predicting salary first and then years.

```
cv=vfold_cv(train_set,v=10,strata=type,repeats=5)
s1_recipe=recipe(first_year_percent_of_cap~.,data=train_set) %>%
  update_role(seas_id:player,new_role="id") %>%
  step_rm(contract_yrs) %>% step_dummy(type)
```

```
wf=workflow() %>% add_recipe(s1_recipe) %>%
  add_model(linear_reg() %>%
              set_engine("lm") %>%
              set_mode("regression"))
tune_lin=wf %>% tune_grid(resamples=cv,metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_lin %>% select_best(metric="rmse"))
fit_s1_lin=fit(wf,train_set)
prettify_vip(fit_s1_lin,"Linear S1 Model Variable Importance")
```



The most important factor here is the type of free agent a player is. Next on the importance graph is amount of 2-point shots attempted per game in the contract year. 2PA/G is a measure of scoring opportunity: leaders in this category are usually those also near the top in points per game, as their talent warrants them taking that increased volume of shots.

```
wf=workflow() %>% add_recipe(s1_recipe) %>%
  add_model(nearest_neighbor(neighbors=tune()) %>%
              set_engine("kknn") %>%
              set_mode("regression"))
tune_knn=wf %>% tune_grid(resamples=cv,
                          grid=expand.grid(neighbors=5:50),
                          metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_knn %>% select_best(metric="rmse"))
fit_s1_knn=fit(wf,train_set)
tune_knn %>% select_best(metric="rmse") %>% knitr::kable()
```

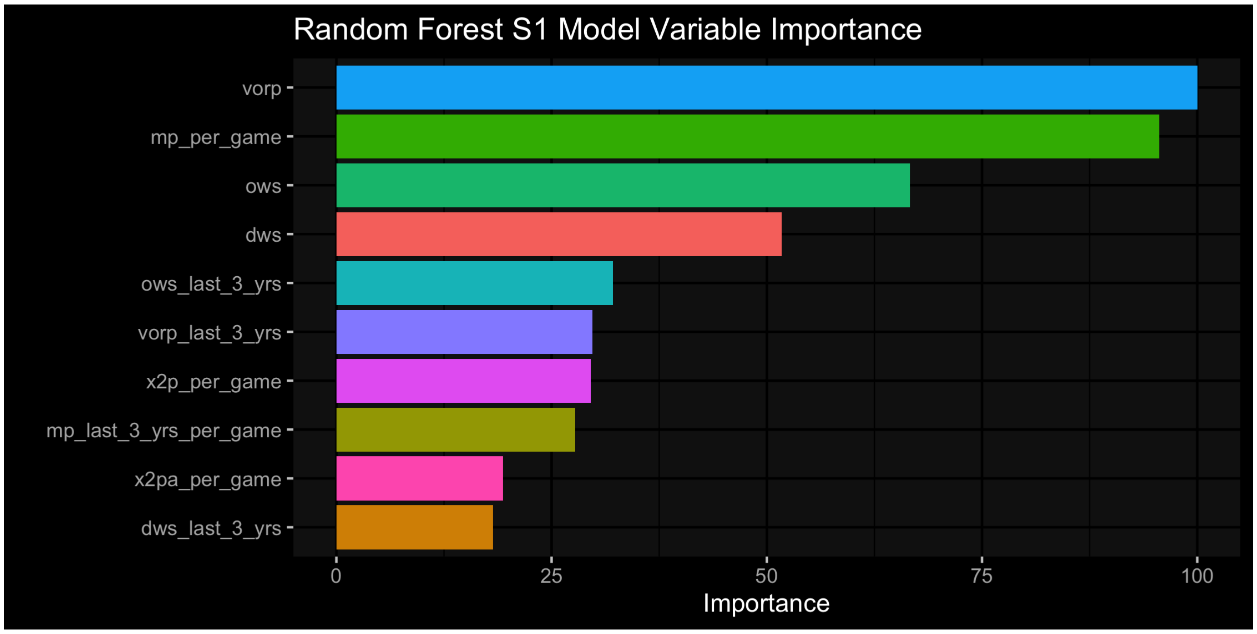
neighbors	.config
25	Preprocessor1_Model21

The best KNN model is smack-dab in the middle of the tune grid, taking the 25 closest players in similarity to construct a player's salary prediction.

```
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(s1_recipe) %>%
  add_model(rand_forest(mtry = tune()) %>%
  set_engine("ranger",importance="permutation") %>% set_mode("regression"))
tune_forest=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(mtry=5:10),
                        metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_forest %>% select_best(metric="rmse"))
fit_s1_forest=fit(wf,train_set)
tune_forest %>% select_best(metric="rmse") %>% knitr::kable()
```

mtry	.config
10	Preprocessor1_Model6

```
prettify_vip(fit_s1_forest,"Random Forest S1 Model Variable Importance")
```



The salary-first forest is more widely clustered at the top than the salary-second forest, having 5 variables >50% importance vs 1 respectively. Excluding contract years in the salary-second forest, the variables of importance follow the exact same order in both forests.

```
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(s1_recipe) %>%
  add_model(svm_rbf(rbf_sigma=tune(),cost=tune()) %>%
  set_engine("kernlab") %>% set_mode("regression"))
tune_svm=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(rbf_sigma=0.1^seq(2:5),
                                         cost=c(0.25,0.5,1)),
                        metrics=metric_set(rmse,mae))
wf=wf %>% finalize_workflow(tune_svm %>% select_best(metric="rmse"))
fit_s1_svm=fit(wf,train_set)
tune_svm %>% select_best(metric="rmse") %>% knitr::kable()
```

cost	rbf_sigma	.config
1	0.01	Preprocessor1_Model10

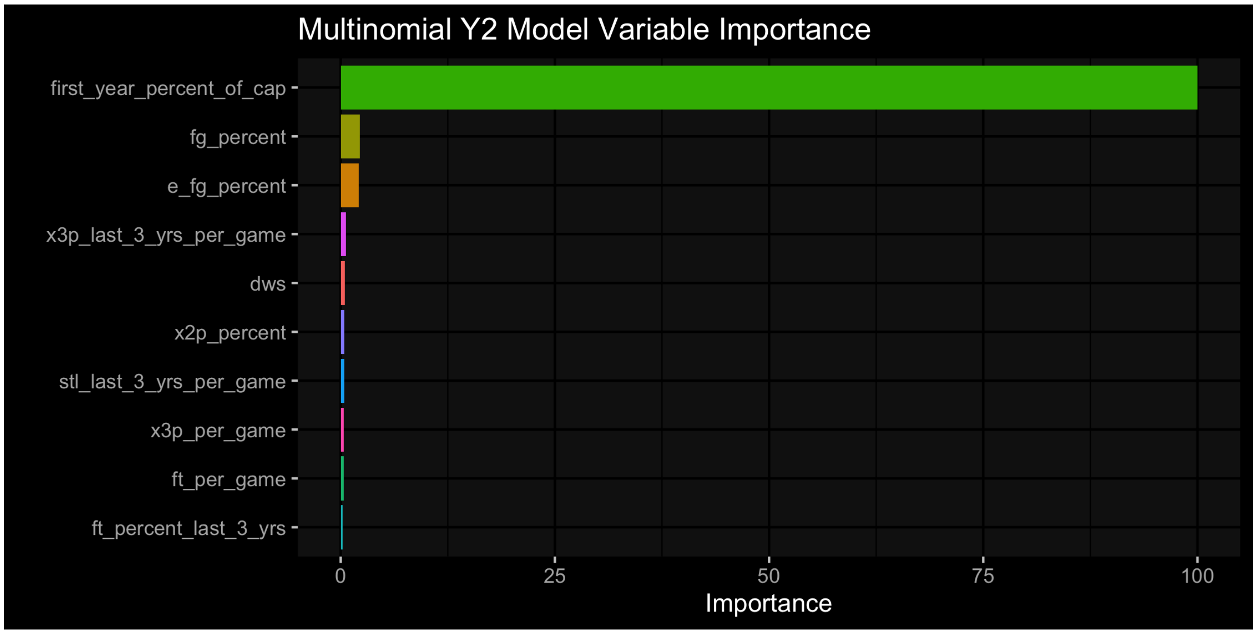
Below are the performance metrics for the salary-first models.

Method	Off By >5%	Within 2%	MAE	RMSE
Linear	8.86%	61.61%	0.0227934	0.0326051
KNN	8.35%	72.61%	0.0212010	0.0333765
Random Forest	0.92%	89.51%	0.0209039	0.0318733
SVM	6.82%	76.58%	0.0201047	0.0311694

Once again, the random forest finishes second to the SVM model in traditional metrics but beats it out in our dataset-specific metrics. All 4 models performed worse when predicting salary-first vs salary-second. With the SVM dipping below 80%, we will use the random forest as our salary prediction.

```
train_set_after_s1=train_set %>% select(-first_year_percent_of_cap) %>%
  bind_cols(first_year_percent_of_cap=predict(fit_s1_forest,new_data=train_set) %>%
            pull())
cv=vfold_cv(train_set_after_s1,v=10,strata=type,repeats=5)
y2_recipe=recipe(contract_yrs~.,data=train_set_after_s1) %>%
  update_role(seas_id:player,new_role="id") %>% step_dummy(type)
```

```
wf=workflow() %>% add_recipe(y2_recipe) %>%
  add_model(multinom_reg(penalty=0) %>%
            set_engine("glmnet") %>%
            set_mode("classification"))
tune_multinom=wf %>% tune_grid(resamples=cv,metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_multinom %>% select_best())
fit_y2_multinom=fit(wf,train_set_after_s1)
prettify_vip(fit_y2_multinom,"Multinomial Y2 Model Variable Importance")
```



The multinomial years-second model essentially discards all other variables in favor of the predicted first-year-percent-of-cap variable.

```r
wf=workflow() %>% add_recipe(y2_recipe) %>%
  add_model(nearest_neighbor(neighbors = tune()) %>%
  set_engine("kknn") %>% set_mode("classification"))
tune_knn=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(neighbors=5:50),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_knn %>% select_best())
fit_y2_knn=fit(wf,train_set_after_s1)
tune_knn %>% select_best() %>% knitr::kable()
```

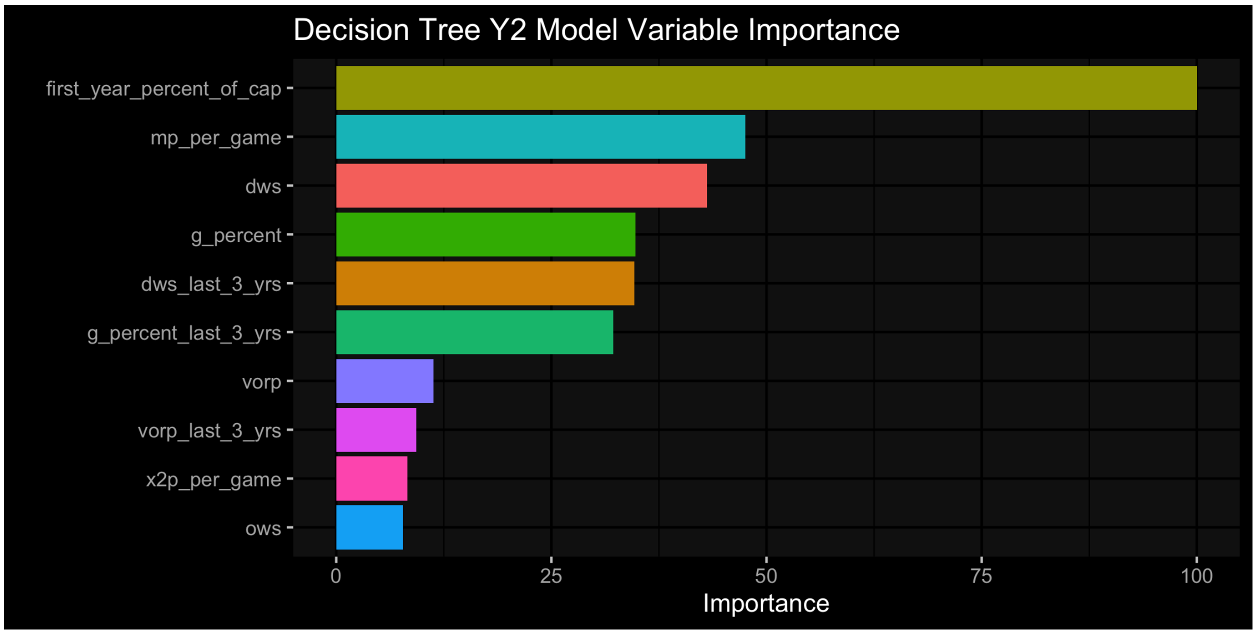
neighbors	.config
48	Preprocessor1_Model44

All 50 neighbors are needed in order for the KNN years-second model to classify a player's contract years.

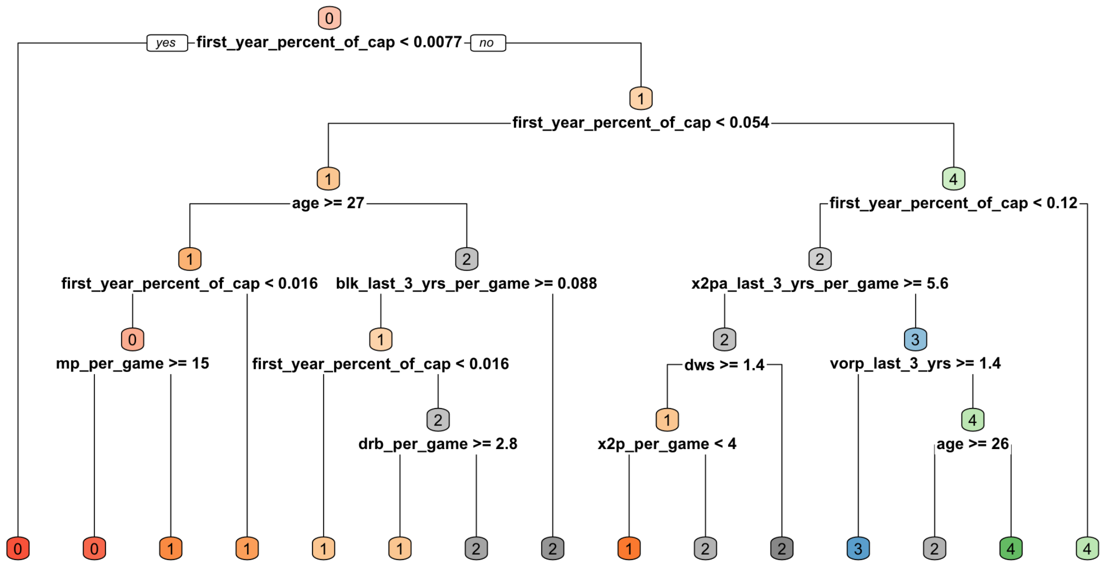
```r
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(y2_recipe) %>%
  add_model(decision_tree(cost_complexity = tune(),min_n=tune()) %>%
  set_engine("rpart") %>% set_mode("classification"))
tune_tree=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(cost_complexity=0.1^seq(2,5),
                                         min_n=seq(1:10)),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_tree %>% select_best())
fit_y2_tree=fit(wf,train_set_after_s1)
tune_tree %>% select_best() %>% knitr::kable()
```

cost_complexity	min_n	.config
0.01	1	Preprocessor1_Model01

```
prettify_vip(fit_y2_tree,"Decision Tree Y2 Model Variable Importance")
```



First-year-percent-of-cap shoots up to 1st in importance in the years-second decision tree, much like we saw in the years-second multinomial model. Contract-year minutes per game & defensive win shares are pretty much tied for second in terms of variable importance.

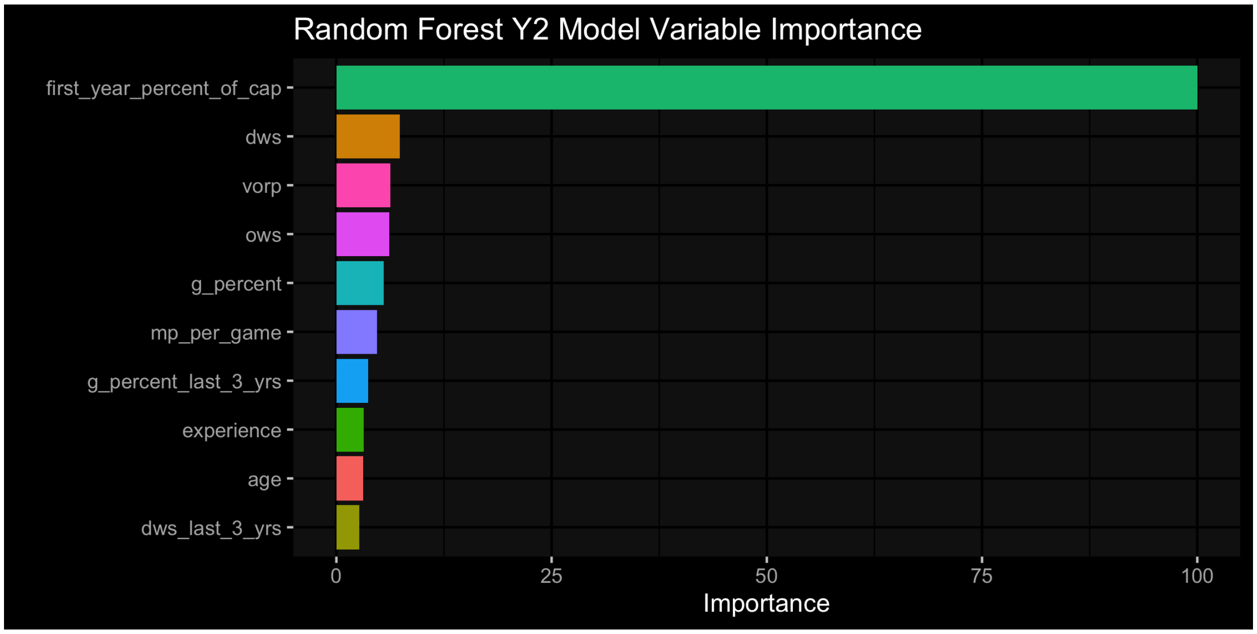

0
yes
first_year_percent_of_cap < 0.0077
no
1
first_year_percent_of_cap < 0.054
1
age >= 27
4
first_year_percent_of_cap < 0.12
1
first_year_percent_of_cap < 0.016
2
blk_last_3_yrs_per_game >= 0.088
2
x2pa_last_3_yrs_per_game >= 5.6
0
mp_per_game >= 15
1
first_year_percent_of_cap < 0.016
2
dws >= 1.4
3
vorp_last_3_yrs >= 1.4
2
drb_per_game >= 2.8
1
x2p_per_game < 4
4
age >= 26
0
0
1
1
1
1
2
2
1
2
2
3
2
4
4

The years-second decision tree predicts some three year contracts, but still can't seem to figure out the division between 4- and 5-year contracts. It maximizes its prediction at 4 contract years if the first-year-percent-of-cap is greater than 12%.

```r
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(y2_recipe) %>%
  add_model(rand_forest(mtry = tune()) %>%
  set_engine("ranger",importance="permutation") %>% set_mode("classification"))
tune_forest=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(mtry=5:10),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_forest %>% select_best())
fit_y2_forest=fit(wf,train_set_after_s1)
tune_forest %>% select_best() %>% knitr::kable()
```

mtry	.config
10	Preprocessor1_Model6

```r
prettify_vip(fit_y2_forest,"Random Forest Y2 Model Variable Importance")
```



The variable importance graph looks like the multinomial model with a slightly less severe division between the first-year-percent-of-cap variable importance and the rest of the variable importances.

```r
set.seed(100,sample.kind = "Rounding")
wf=workflow() %>% add_recipe(y2_recipe) %>%
  add_model(svm_rbf(rbf_sigma=tune(),cost=tune()) %>%
  set_engine("kernlab") %>% set_mode("classification"))
tune_svm=wf %>% tune_grid(resamples=cv,
                        grid=expand.grid(rbf_sigma=0.1^seq(2:5),
                                         cost=c(0.25,0.5,1)),
                        metrics=metric_set(macro_weight_f1))
wf=wf %>% finalize_workflow(tune_svm %>% select_best())
fit_y2_svm=fit(wf,train_set_after_s1)
```

```
## maximum number of iterations reached 0.00211794 0.002072318
```

```
tune_svm %>% select_best() %>% knitr::kable()
```

cost	rbf_sigma	.config
0.25	1e-04	Preprocessor1_Model04

Last but not least, here are the years performance metrics of the S1Y2 model.

Method	Correct Predict %	Off by >1 Yr	Max Year Predicts	F1 Score
Multinomial	80.75%	5.60%	17	0.6619937
KNN	64.26%	12.93%	3	0.4911126
Decision Tree	74.03%	6.72%	0	0.6541921
Random Forest	99.90%	0.10%	17	0.6495498
SVM	38.90%	36.05%	0	0.5578424

The random forest drops to third in F1 score, but has no predictions that are off by more than one year. We will use the random forest as the sole model to generate Y2 predictions.

Results

Now it's time to test on the 2021 data.

Years First, Salary Second

As a reminder, we're using a random forest as our years prediction, and using the mean of a random forest and SVM to predict salary.

```
eval_y1_predict=as.numeric(as.character(
  predict(fit_y1_forest,new_data=eval_set) %>% pull()))
eval_s2_predict_forest=predict(fit_s2_forest,
                               new_data=eval_set %>% select(-contract_yrs) %>%
                                 bind_cols(contract_yrs=eval_y1_predict))
eval_s2_predict_svm=predict(fit_s2_svm,
                               new_data=eval_set %>% select(-contract_yrs) %>%
                                 bind_cols(contract_yrs=eval_y1_predict))
eval_s2_predict=tibble(forest=eval_s2_predict_forest %>% pull(),
                       svm=eval_s2_predict_svm %>% pull()) %>% rowwise() %>%
  mutate(m=mean(c_across(forest:svm))) %>% ungroup() %>% pull(m)
```

Salary First, Years Second

For the S1Y2 model, we are using random forests for each component.

```
eval_s1_predict=predict(fit_s1_forest,new_data=eval_set) %>% pull()
eval_y2_predict=as.numeric(as.character(
  predict(fit_y2_forest,new_data=eval_set %>% select(-first_year_percent_of_cap) %>%
            bind_cols(first_year_percent_of_cap=eval_s1_predict)) %>%
  pull()))
```

Player Option Decisions and Selected Free Agents

Before I look at which players might be accepting or declining their option and then some selected high-profile free agents, let's look at the distribution of contract years.

yrs_Y1S2	n
0	95
1	52
2	34
3	7
4	12

yrs_S1Y2	n
0	55
1	78
2	46
3	8
4	13

Last year, both the Y1S2 & the S1Y2 had only 4-5 players being out of the league. This didn't make sense, as the annual rookie draft brings in at least 30 new players. The problem was that there were players who were predicted to receive 1 year contracts but receiving salaries that were less than the minimum. My solution was to convert any salary predictions less than the minimum to zero, and any years predictions to zero where the salary prediction is zero.

For the 2021-2022 season, the salary cap is $112,414,000. The minimum contract (from RealGM) is $1,489,065. Therefore the minimum first year percent of cap in 2021 is 1.32%.

```r
first_yr_min=1489065/112414000
final_results<-final_results %>%
  mutate(yrs_Y1S2=ifelse(yr1_cap_percent_Y1S2<=first_yr_min,0,yrs_Y1S2)) %>%
  mutate(yr1_cap_percent_Y1S2=ifelse(yrs_Y1S2==0,0,yr1_cap_percent_Y1S2)) %>%
  mutate(yrs_S1Y2=ifelse(yr1_cap_percent_S1Y2<=first_yr_min,0,yrs_S1Y2)) %>%
  mutate(yr1_cap_percent_S1Y2=ifelse(yrs_S1Y2==0,0,yr1_cap_percent_S1Y2))
```

yrs_Y1S2	n
0	95
1	52
2	34
3	7
4	12

yrs_S1Y2	n
0	76
1	64
2	39
3	8
4	13

The Y1S2 predictions remain unchanged, while some 1-year and a handful of 2-year contracts in the S1Y2 predictions got converted to 0-year due to the salary being below the minimum threshold.

Another task is to make the information more digestible. Contracts are usually reported in terms of total value, and players usually get five percent raises per year, so I'll add a column for each method for total contract value assuming the projected cap value of $112.414 million.

```r
final_results<-final_results %>%
  mutate(total_Y1S2=
           round(yr1_cap_percent_Y1S2*112414000*((1.05)^(yrs_Y1S2)-1)/0.05,digits=-4),
         total_S1Y2=
             round(yr1_cap_percent_S1Y2*112414000*((1.05)^(yrs_S1Y2)-1)/0.05,digits=-4))
```

Player Option Decisions

player	Y1S2 Cap %	yrs_Y1S2	total_Y1S2	S1Y2 Cap %	yrs_S1Y2	total_S1Y2	Option Type	2021 Option
Chris Paul	26.71%	4	$129,410,000	26.50%	4	$128,400,000	PO	$44,211,146
Kawhi Leonard	31.90%	4	$154,560,000	29.53%	4	$143,080,000	PO	$36,016,200
Goran Dragić	3.52%	1	$3,960,000	4.96%	1	$5,580,000	CO	$19,440,000
Andre Iguodala	2.86%	1	$3,220,000	3.39%	1	$3,810,000	CO	$15,000,000
Justise Winslow	1.92%	1	$2,160,000	1.80%	1	$2,020,000	CO	$13,000,000
Josh Richardson	5.83%	1	$6,550,000	7.79%	2	$17,950,000	PO	$11,615,328
Montrezl Harrell	10.16%	2	$23,410,000	11.11%	2	$25,600,000	PO	$9,720,900
Serge Ibaka	6.93%	2	$15,970,000	6.77%	2	$15,600,000	PO	$9,720,900
Avery Bradley	0.00%	0	$0	1.84%	1	$2,070,000	CO	$5,916,750
Kris Dunn	0.00%	0	$0	0.00%	0	$0	PO	$5,005,350
Willie Cauley-Stein	2.50%	1	$2,810,000	3.38%	1	$3,800,000	CO	$4,100,000
Bobby Portis	7.24%	2	$16,680,000	8.37%	2	$19,290,000	PO	$3,804,150
Ryan Arcidiacono	0.00%	0	$0	1.81%	1	$2,030,000	CO	$3,000,000
Edmond Sumner	2.88%	2	$6,640,000	2.32%	2	$5,350,000	CO	$2,320,000
Mitchell Robinson	7.89%	2	$18,180,000	9.76%	3	$34,590,000	CO	$1,802,057
DaQuan Jeffries	0.00%	0	$0	0.00%	0	$0	CO	$1,701,593
Didi Louzada	0.00%	0	$0	0.00%	0	$0	CO	$1,517,981

Looking past the aforementioned Leonard & Paul, it's not a very inspired class of free agents with options. It's very clear that Dragić and Iguodala will have their club options declined. Coincidentally, both play for the Heat, who will look to retool after a disappointing first-round loss against the eventual 2021 champion Milwaukee Bucks, just a year after advancing to the NBA Finals. Winslow is also probably getting his option declined: he was actually part of the package that Miami sent to the Memphis Grizzlies to acquire Iguodala. Kris Dunn is very likely to pick up his player option after missing a majority of the year with a sprained MCL in his right knee, and the Knicks are probably elated to accept the club option on Mitchell Robinson.

Selected Free Agents

Now let's see what the models have predicted for this year's other notable free agents. I'll present them five at a time and comment on some projections. We'll look at restricted free agents first.

player	age	Y1S2 Cap %	yrs_Y1S2	total_Y1S2	S1Y2 Cap %	yrs_S1Y2	total_S1Y2
John Collins	23	17.42%	3	$61,730,000	16.69%	3	$59,150,000
Lonzo Ball	23	15.47%	4	$74,950,000	12.66%	4	$61,340,000
Jarrett Allen	22	13.93%	3	$49,370,000	12.88%	3	$45,640,000
Devonte' Graham	25	12.25%	4	$59,350,000	10.76%	4	$52,130,000
Kendrick Nunn	25	12.82%	4	$62,120,000	9.65%	4	$46,760,000

The biggest between-model discrepancies for RFAs belong to Ball & Nunn, who lose `$13M` & `$16M` respectively when switching to the S1Y2 model. Graham finds himself stuck behind 2021 No. 2 overall pick LaMelo Ball & fringe All-Star Terry Rozier in the Charlotte Hornets guard rotation. A team could pry him from the Hornets with a big enough offer sheet, as Charlotte would probably like to use that cap space to address their centre position.

player	age	Y1S2 Cap %	yrs_Y1S2	total_Y1S2	S1Y2 Cap %	yrs_S1Y2	total_S1Y2
Duncan Robinson	26	11.36%	4	$55,040,000	10.67%	4	$51,700,000
Lauri Markkanen	23	11.84%	4	$57,370,000	8.21%	4	$39,780,000
Josh Hart	25	6.61%	2	$15,230,000	6.89%	2	$15,880,000
Gary Trent Jr.	22	6.73%	2	$15,510,000	6.57%	2	$15,140,000
Bruce Brown	24	5.80%	2	$13,370,000	6.32%	2	$14,560,000

Robinson is often compared to the Nets' Joe Harris as both are sweet-shooting wings opening up the floor for superstar teammates. Harris provides more defensive chops & a midrange game, while Robinson is renowned for his off-ball movement. Last year, I projected Harris for 3-years and `$55-57M`, and he ended up getting 4-years and `$72M`.

I was genuinely surprised by the projections for Markkanen & Trent Jr. In fact, I would say they should be flipped. The fact is, these models are exceedingly retrospective, with the only prospective variables being age & experience. They see Markkanen as an efficient scorer from both 2-point & 3-point range, while also being a serviceable rebounder. On the other hand, going by stats alone, Trent was inefficient with minimal contributions in rebounds and assists as well as a negative VORP. Never mind that Markkanen lost his starting lineup spot in the second half of the season, or that Trent turned down a contract extension before the season for 4 years & `$60M`.

On to the unrestricted free agents.

player	age	Y1S2 Cap %	yrs_Y1S2	total_Y1S2	S1Y2 Cap %	yrs_S1Y2	total_S1Y2
DeMar DeRozan	31	22.80%	2	$52,540,000	25.23%	2	$58,140,000
Mike Conley	33	20.73%	4	$100,440,000	18.01%	4	$87,260,000
Kyle Lowry	34	15.60%	2	$35,950,000	17.71%	2	$40,810,000
Norman Powell	27	17.07%	4	$82,710,000	15.11%	4	$73,210,000
Evan Fournier	28	15.35%	4	$74,370,000	12.88%	4	$62,410,000

Conley is the only non-option free agent to break the `$100M` barrier, and that too by the slimmest of margins. DeRozan is a wildcard, as he has made an estimated `$175M` in career earnings and might take a pay cut in order to pursue a championship. Perhaps he does this to team up with best friend & former Raptors running mate Lowry, who was looking for a 2-year, `$50M` contract extension from teams interested in his services at the trade deadline. Fournier & Powell are in similar situations: acquired at the trade deadline by teams looking to bolster their playoff aspirations, they declined their player options to seek a larger payday. Powell offers more defense while Fournier is a better playmaker, but they are both scorers at heart, and as Bill Russell so eloquently stated: This game has always been and always will be about getting buckets.

player	age	Y1S2 Cap %	yrs_Y1S2	total_Y1S2	S1Y2 Cap %	yrs_S1Y2	total_S1Y2
Kelly Olynyk	29	14.79%	4	$71,660,000	12.54%	4	$60,760,000
Dennis Schröder	27	11.34%	2	$26,130,000	13.58%	1	$15,270,000
Tim Hardaway Jr.	28	9.58%	1	$10,770,000	12.92%	4	$62,600,000
Danny Green	33	10.83%	3	$38,380,000	11.21%	2	$25,830,000
Reggie Bullock	29	10.92%	4	$52,910,000	9.54%	4	$46,220,000

Olynyk was sent to Houston as part of the trade that brought Victor Oladipo to Miami, putting up solid numbers for a tanking Rockets squad. Schröder has been rumored to be aiming for a `$120M-$140M` contract, but projections don't look to be as rosy. THJ is the first example of wildly different predictions between the two models, with a staggering discrepancy of `$52M`!

player	age	Y1S2 Cap %	yrs_Y1S2	total_Y1S2	S1Y2 Cap %	yrs_S1Y2	total_S1Y2
Kelly Oubre Jr.	25	9.41%	2	$21,690,000	10.71%	2	$24,680,000
Nicolas Batum	32	9.73%	3	$34,480,000	10.24%	3	$36,290,000
Richaun Holmes	27	9.44%	2	$21,750,000	10.44%	2	$24,060,000
T.J. McConnell	28	9.15%	3	$32,430,000	8.92%	3	$31,610,000
Derrick Rose	32	8.69%	2	$20,030,000	9.25%	2	$21,320,000

Oubre had a nightmare start to his tenure with the Golden State Warriors, only making 7 of his first 50 3-point shots for a putrid 14% 3-point percentage. He never really recovered, with his numbers down across the board from his 2020 season with the Suns. Batum rebuilt his value after being waived & stretched by the Hornets, becoming a valuable glue guy for the Clippers. McConnell giveth & McConnell taketh, as he was 10th in the league for assists and topped the leaderboard in steals.

Conclusion

Using a combination of contract year stats and last-three-years stats, we set out to predict NBA free agent contracts for the 2021 offseason. Since we had correlated targets in contract length and first year percent of salary cap, we predicted one target first and used its predictions as an input for the second target. The contract years were predicted as a classification problem, while the salary was predicted as a regression model We used six models: linear (for salary) & multinomial (for contract length) as baselines, k-nearest-neighbors, decision tree (only for predicting contract length), a random forest and a support vector machine. Chris Paul, Kawhi Leonard & Mike Conley are all projected to get above $ 90 million in total contract value.

The models suffer from being unable to quantify intangibles like playing reputation, team fit, within-season role changes, or willingness to take a reduced salary to be on a championship contender. The models also can't determine which team will sign which player. That highly depends on a sequence of events: if Team A signs this player, they don't have enough money to resign Player B, who then goes to Team C for less money, etc.

Since salary and contract length were correlated, maybe I should have predicted them as a tuple instead of sequentially. Unfortunately, caret didn't have that capability of multi-target regression, but tidymodels does. Perhaps I should have implemented a time factor or weighted recent years more heavily, as team decision makers may have gotten smarter. I didn't remove highly correlated predictors by design, as I wanted the models themselves to perform feature selection and determine what the most important variables were.

Future work could include trying more models, like boosting (in which models are added sequentially, with later models in the sequence attempting to correct the errors of earlier models). Another example might be adding a draft pedigree variable, as teams might be more willing to:

A. take a chance on a player with mediocre surface-level stats, as the team could believe the player was misused and the inherent talent is still there
B. pay up for a player with good stats, as the high pick of the player could signal to the player's higher potential

A final option might be predicting a third target: whether a contract will end in an option year. Star players are more likely to demand the last year of their contract as a player option in order to take ownership of their future.