How do different factor affect the NBA players' salary

CREST GOLD

Chun Yin, Wong Jamin Kwan Ho, Mok Jack Cheuk Hei, Ng Adrian
King Yiu, Chow Kinson

2022-08-18

Contents

*Webpage version: https://rpubs.com/Jamin/nba_player_salary_202223

1 Executive Summary

This report aims to find the correlation of the player statistics in the year before and the salary of that player in the year after.

We found player statistics for 2021-2022 season and their respective salary in 2022-2023 season. We have collect the player statistics and salary of 384 NBA players that played in 2021-2022 and have salary in the 2022-23 season. The player statistics is collected from Basketball Reference and NBA official website. The detailed links are listed below in the Source of files section.

We started with merging the the data from different source and removing repeated variables.

In the exploratory data analysis, we start with univariate analysis of the response variable salary and then explore the correlation of individual variables with salary.

In the modelling section, we use the RMSE score on the train set to determine the final model. We have 10 fold cross validation on each model (the 10 fold all use RMSE to hyperparameter tuning). The root mean squared error (RMSE) is a measure of prediction error while R-squared is a measure of the proportion of variation explained by the model.
We have included the following modelling methods:

- Linear Regression (Train RMSE: 0.4996, R-squared: 0.7285)
- Lasso Regression (Train RMSE: 0.5246, R-squared: 0.7076)
- Elastic Net Regression (Train RMSE: 0.5134, R-squared: 0.7178)
- Stepwise Regression (Train RMSE: 0.5072, R-squared: 0.7535)
- Decision Tree (Train RMSE: 0.5096, R-squared: 0.6438)
- Random Forest (Train RMSE: 0.2264, R-squared: 0.7097)
- Neural Network (Train RMSE: 0.6170, R-squared: 0.6624)

As the Random Forest Model has the lowest RMSE, we chose it as the final model for our prediction. It has 0.4835 of estimate out-of-bag RMSE.

In the interpretation part, we will mainly use linear regression and principle component analysis to analyse. We will not use the random forest model as it is hard to give interpretation on a random forest model.

In the analysis, imperial units are used as it is the common unit systems in the NBA. All the salary is record in USD.

2 Understanding of the problem

We want to use player statistics to find out what kind of players with which kind of attribute will result in higher salary in modern NBA.
There are 59 explanatory predictive variable after removing repetitive variable in the raw dataset with salary as the responsive variable.

2.1 Source of files:

player_PGstats_2021.csv – NBA players statistics per game in 2021-2022 season
source: https://www.basketball-reference.com/leagues/NBA_2022_per_game.html
player_Adstats_2021.csv – NBA players advance statistics in 2021-2022 season source: https://www.basketball-reference.com/leagues/NBA_2022_advanced.html
salary2022.csv – NBA players contract in 2022-2023 season onward
source: https://www.basketball-reference.com/contracts/players.html
bio.csv – NBA player bio (height and weight)
source: https://www.nba.com/stats/players/bio/?Season=2021-22&SeasonType=Regular%20Season&sort=PLAYER_NAME&dir=1

3 Loading and Exploring Data

3.1 Loading libraries required

Load libraries required for the analysis.

```
library(knitr)
library(plyr)
library(dplyr)
library(tidyr)
library(caret)
library(ggplot2)
library(corrplot)
library(stringr)
library(scales)
library(randomForest)
library(glmnet)
library(rpart)
library(lubridate)
library(plotly)
library(forcats)
library(psych)
library(ggExtra)
library(reshape2)
library(tree)
library(MASS)
library(Metrics)
library(rattle)
library(neuralnet)
library(sigmoid)
opts_chunk$set(echo = TRUE, cache = TRUE)
opts_chunk$set(tidy.opts = list(width.cutoff = 80), tidy = TRUE, fig.height = 6, fig.width = 9)
```

3.2 Loading data files

Read in the data that were downloaded from the sources.

```
pgstats <- read.csv("files/2022/player_PGstats_2021.csv")
adstats <- read.csv("files/2022/player_Adstats_2021.csv")
salary <- read.csv("files/2022/salary2022.csv")
bio <- read.csv("files/2022/bio.csv")
```

3.3 File description

3.3.1 player_PGstats_2021.csv

NBA players statistics per game in 2021-2022 season
source: https://www.basketball-reference.com/leagues/NBA_2022_per_game.html
The file contains NBA player per-game statistics in 2021-22 season, including points, assists, rebounds, blocks, and steal et al..

```
dim(pgstats)
```

```
## [1] 812  31
```

```
str(pgstats)
```

```
## 'data.frame':    812 obs. of  31 variables:
##  $ Rk       : int  1 2 3 4 5 6 6 6 7 8 ...
##  $ Player   : chr  "Precious Achiuwa" "Steven Adams" "Bam Adebayo" "Santi Aldama" ...
##  $ Pos      : chr  "C" "C" "C" "PF" ...
##  $ Age      : int  22 28 24 21 36 23 23 23 26 23 ...
##  $ Tm       : chr  "TOR" "MEM" "MIA" "MEM" ...
##  $ G        : int  73 76 56 32 47 65 50 15 66 56 ...
##  $ GS       : int  28 75 56 0 12 21 19 2 61 56 ...
##  $ MP       : num  23.6 26.3 32.6 11.3 22.3 22.6 26.3 9.9 27.3 32.3 ...
##  $ FG       : num  3.6 2.8 7.3 1.7 5.4 3.9 4.7 1.1 3.9 6.6 ...
##  $ FGA      : num  8.3 5.1 13 4.1 9.7 10.5 12.6 3.2 8.6 9.7 ...
##  $ FG.      : num  0.439 0.547 0.557 0.402 0.55 0.372 0.375 0.333 0.448 0.677 ...
##  $ X3P      : num  0.8 0 0 0.2 0.3 1.6 1.9 0.7 2.4 0 ...
##  $ X3PA     : num  2.1 0 0.1 1.5 1 5.2 6.1 2.2 5.9 0.2 ...
##  $ X3P.     : num  0.359 0 0 0.125 0.304 0.311 0.311 0.303 0.409 0.1 ...
##  $ X2P      : num  2.9 2.8 7.3 1.5 5.1 2.3 2.8 0.4 1.5 6.6 ...
##  $ X2PA     : num  6.1 5 12.9 2.6 8.8 5.3 6.5 1 2.7 9.6 ...
##  $ X2P.     : num  0.468 0.548 0.562 0.56 0.578 0.433 0.434 0.4 0.533 0.688 ...
##  $ eFG.     : num  0.486 0.547 0.557 0.424 0.566 0.449 0.45 0.438 0.588 0.678 ...
##  $ FT       : num  1.1 1.4 4.6 0.6 1.9 1.2 1.4 0.7 1 2.9 ...
##  $ FTA      : num  1.8 2.6 6.1 1 2.2 1.7 1.9 0.8 1.1 4.2 ...
##  $ FT.      : num  0.595 0.543 0.753 0.625 0.873 0.743 0.722 0.917 0.865 0.708 ...
##  $ ORB      : num  2 4.6 2.4 1 1.6 0.6 0.7 0.1 0.5 3.4 ...
##  $ DRB      : num  4.5 5.4 7.6 1.7 3.9 2.3 2.6 1.5 2.9 7.3 ...
##  $ TRB      : num  6.5 10 10.1 2.7 5.5 2.9 3.3 1.5 3.4 10.8 ...
##  $ AST      : num  1.1 3.4 3.4 0.7 0.9 2.4 2.8 1.1 1.5 1.6 ...
##  $ STL      : num  0.5 0.9 1.4 0.2 0.3 0.7 0.8 0.3 0.7 0.8 ...
```

```
##  $ BLK      : num  0.6 0.8 0.8 0.3 1 0.4 0.4 0.3 0.3 1.3 ...
##  $ TOV      : num  1.2 1.5 2.6 0.5 0.9 1.4 1.7 0.5 0.7 1.7 ...
##  $ PF       : num  2.1 2 3.1 1.1 1.7 1.6 1.8 1 1.5 1.7 ...
##  $ PTS      : num  9.1 6.9 19.1 4.1 12.9 10.6 12.8 3.5 11.1 16.1 ...
##  $ player_id: chr  "achiupr01" "adamsst01" "adebaba01" "aldamsa01" ...
```

3.3.2 player_Adstats_2021.csv

player_Adstats_2021.csv – NBA players advance statistics in 2021-2022 season source: https://www.basketball-reference.com/leagues/NBA_2022_advanced.html
This file contains NBA player advance statistics in 2021-22 season, including PER (player efficiency rating), win shares, box score, and VORP (value over replacement player) et al..

```
dim(adstats)
```

```
## [1] 812  30
```

```
str(adstats)
```

```
## 'data.frame':    812 obs. of  30 variables:
##  $ Rk       : int  1 2 3 4 5 6 6 6 7 8 ...
##  $ Player   : chr  "Precious Achiuwa" "Steven Adams" "Bam Adebayo" "Santi Aldama" ...
##  $ Pos      : chr  "C" "C" "C" "PF" ...
##  $ Age      : int  22 28 24 21 36 23 23 23 26 23 ...
##  $ Tm       : chr  "TOR" "MEM" "MIA" "MEM" ...
##  $ G        : int  73 76 56 32 47 65 50 15 66 56 ...
##  $ MP       : int  1725 1999 1825 360 1050 1466 1317 149 1805 1809 ...
##  $ PER      : num  12.7 17.6 21.8 10.2 19.6 10.5 10.5 10.2 12.7 23 ...
##  $ TS.      : num  0.503 0.56 0.608 0.452 0.604 0.475 0.474 0.497 0.609 0.698 ...
##  $ X3PAr    : num  0.259 0.003 0.008 0.364 0.1 0.497 0.483 0.688 0.684 0.018 ...
##  $ FTr      : num  0.217 0.518 0.466 0.242 0.223 0.16 0.153 0.25 0.13 0.428 ...
##  $ ORB.     : num  8.7 17.9 8.7 9.4 7.8 2.7 3 0.8 1.9 12 ...
##  $ DRB.     : num  21.7 22 26.1 16.1 18.7 11.5 11 15.6 10.9 24.5 ...
##  $ TRB.     : num  14.9 19.9 17.5 12.6 13.4 7.1 6.9 8.5 6.5 18.4 ...
##  $ AST.     : num  6.9 16.1 17.5 7.7 6.3 16.1 16.1 15.5 7.6 8.2 ...
##  $ STL.     : num  1.1 1.6 2.2 0.8 0.6 1.5 1.5 1.7 1.2 1.2 ...
##  $ BLK.     : num  2.3 2.7 2.6 2.5 4 1.5 1.4 2.4 1 3.7 ...
##  $ TOV.     : num  11.3 19.6 14.4 9.9 8 11.3 11.2 13.1 6.7 12.7 ...
##  $ USG.     : num  18.5 12 25 18.4 22.4 24.1 24.8 17.9 15.2 18.1 ...
##  $ X        : logi  NA NA NA NA NA NA ...
##  $ OWS      : num  0.4 3.8 3.6 -0.1 2.1 -1.1 -1.1 0 2.8 5.4 ...
##  $ DWS      : num  2.1 3 3.5 0.4 1 1.1 0.9 0.2 1.4 3 ...
##  $ WS       : num  2.5 6.8 7.2 0.3 3.1 0.1 -0.1 0.2 4.2 8.5 ...
##  $ WS.48    : num  0.07 0.163 0.188 0.044 0.141 0.003 -0.005 0.07 0.11 0.225 ...
##  $ X.1      : logi  NA NA NA NA NA NA ...
##  $ OBPM     : num  -2 1 1.7 -4.2 1.3 -1.8 -1.7 -2.9 0.6 2.7 ...
##  $ DBPM     : num  -0.6 1 2.1 -1.5 -0.6 -1.1 -1.3 1.2 -0.2 1.2 ...
##  $ BPM      : num  -2.6 2 3.8 -5.7 0.7 -2.9 -3 -1.7 0.4 3.9 ...
##  $ VORP     : num  -0.2 2 2.7 -0.3 0.7 -0.3 -0.3 0 1.1 2.7 ...
##  $ player_id: chr  "achiupr01" "adamsst01" "adebaba01" "aldamsa01" ...
```

3.3.3 salary2022.csv

salary2022.csv – NBA players contract in 2022-2023 season onward
source: https://www.basketball-reference.com/contracts/players.html
This file contains NBA players' salary, types of contract and the guaranteed amount of money from the contract.

```
dim(salary)
```

```
## [1] 448  12
```

```
str(salary)
```

```
## 'data.frame':    448 obs. of  12 variables:
##  $ Rk          : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Player      : chr  "Stephen Curry" "Russell Westbrook" "LeBron James" "Kevin Durant" ...
##  $ Tm          : chr  "GSW" "LAL" "LAL" "BRK" ...
##  $ X2022.23    : chr  "$48070014" "$47063478" "$44474988" "$44119845" ...
##  $ X2023.24    : chr  "$51915615" "" "" "$46407433" ...
##  $ X2024.25    : chr  "$55761216" "" "" "$49856021" ...
##  $ X2025.26    : chr  "$59606817" "" "" "$53282609" ...
##  $ X2026.27    : chr  "" "" "" "" ...
##  $ X2027.28    : chr  "" "" "" "" ...
##  $ Signed.Using: chr  "Bird" "Bird Rights" "Bird" "Bird" ...
##  $ Guaranteed  : chr  "$215353662" "$47063478" "$44474988" "$193665908" ...
##  $ player_id   : chr  "curryst01" "westbru01" "jamesle01" "duranke01" ...
```

3.3.4 bio

bio.csv – NBA player bio (height and weight)
source: https://www.nba.com/stats/players/bio/?Season=2021-22&SeasonType=Regular%20Season&sort=PLAYER_NAME&dir=1
This file contains the height and weight of NBA players in season 2021-22.

```
dim(bio)
```

```
## [1] 605   4
```

```
str(bio)
```

```
## 'data.frame':    605 obs. of  4 variables:
##  $ Player: chr  "Aaron Gordon" "Aaron Henry" "Aaron Holiday" "Aaron Nesmith" ...
##  $ Team  : chr  "DEN" "PHI" "PHX" "BOS" ...
##  $ Weight: int  235 210 185 215 190 225 200 241 174 240 ...
##  $ Height: chr  "6-8" "6-5" "6-0" "6-5" ...
```

4 Preprocessing Data

4.1 Merge data tables

We merge the tables by their primary key (pgstats.player_id) and foreign key (salary.player_id) by inner join (only take the entries which exist in both table). We treat the players that received salary but have not played any game as outliers.

Merge player per-game statistics and advance statistics.

```
merged <- merge(pgstats, adstats, by = c("player_id", "Tm"))
```

Since there is players that has changed team in the middle of the season, We use the total stats (indicated by Tm = "TOT").

```
repeated <- names(which(table(merged$player_id) > 1))

for (i in 1:length(repeated)) {
    id <- repeated[i]

    use <- merged[which(merged$player_id == id & merged$Tm == "TOT"), ]
    temp_tm <- merged$Tm[max(which(merged$player_id == id))]
    merged <- merged[merged$player_id != id, ]
    merged <- rbind(merged, use)
    merged$Tm[merged$player_id == id] <- temp_tm
}
```

Remove and rename some variable to increase readability.

```
merged <- merged %>%
    dplyr::select(!c(Rk.y, Player.y, Pos.y, Age.y, MP.y, G.y, X, X.1)) %>%
    rename(Rk = Rk.x, Player = Player.x, Position = Pos.x, Age = Age.x, Game_played = G.x,
        FGpct = FG., X3Ppct = X3P., X2Ppct = X2P., eFGpct = eFG., FTpct = FT., TSpct = TS.,
        ORBpct = ORB., DRBpct = DRB., TRBpct = TRB., ASTpct = AST., STLpct = STL.,
        BLKpct = BLK., TOVpct = TOV., USGpct = USG., WSper48 = WS.48, Game_started = GS,
        MP = MP.x)
```

The salary dataset contains multiple entries for players who had changed their team in the middle of the season. The salary for each repeated entries are the same.

```
salary <- salary %>%
    group_by(Player, player_id, X2022.23) %>%
    summarise(Tm = Tm[1], Signed.Using = Signed.Using[1], Guaranteed = sum(as.numeric(grep(pattern = "[(
        Guaranteed))), Rk = Rk[1])
```

Merge the remain datasets.

```
merged <- merge(merged, salary, by = c("player_id"))
bio <- bio %>%
    dplyr::select(!Team) %>%
    rename(Player.x = Player)
merged <- merge(merged, bio, by = c("Player.x"), all.x = TRUE)
```

4.2 Save table

```
write.csv(merged, "dataset/all2022.csv")
```

4.2.1 Read in the saved table

```
all <- as.data.frame(read.csv("dataset/all2022.csv"))
```

```
dim(all)
```

```
## [1] 384  60
```

There are 384 entries.

```
str(all)
```

```
## 'data.frame':    384 obs. of  60 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Player.x    : chr  "Aaron Gordon" "Aaron Holiday" "Aaron Nesmith" "Aaron Wiggins" ...
##  $ player_id   : chr  "gordoaa01" "holidaa01" "nesmiaa01" "wiggiaa01" ...
##  $ Tm.x        : chr  "DEN" "WAS" "BOS" "OKC" ...
##  $ Rk.x        : int  198 244 406 581 250 85 448 97 328 497 ...
##  $ Position    : chr  "PF" "PG" "SF" "SG" ...
##  $ Age         : int  26 25 22 23 35 30 20 27 28 19 ...
##  $ Game_played : int  75 63 52 50 69 81 61 41 39 72 ...
##  $ Game_started: int  75 15 3 35 69 44 12 18 10 13 ...
##  $ MP          : num  31.7 16.2 11 24.2 29.1 28.6 20.2 28 15.9 20.7 ...
##  $ FG          : num  5.8 2.4 1.4 3.1 3.9 3.5 3 2.5 2.4 3.5 ...
##  $ FGA         : num  11.1 5.4 3.5 6.7 8.2 9 7.5 6.2 4.5 7.3 ...
##  $ FGpct       : num  0.52 0.447 0.396 0.463 0.467 0.391 0.408 0.398 0.534 0.474 ...
##  $ X3P         : num  1.2 0.6 0.6 0.8 1.3 1.9 0.9 1 0.2 0.4 ...
##  $ X3PA        : num  3.5 1.6 2.2 2.8 3.8 4.8 3.2 3.1 0.5 1.6 ...
##  $ X3Ppct      : num  0.335 0.379 0.27 0.304 0.336 0.404 0.289 0.333 0.286 0.248 ...
##  $ X2P         : num  4.6 1.8 0.8 2.3 2.6 1.6 2.1 1.5 2.3 3.1 ...
##  $ X2PA        : num  7.7 3.7 1.3 4 4.4 4.2 4.2 3.2 4 5.7 ...
##  $ X2Ppct      : num  0.605 0.477 0.612 0.573 0.582 0.378 0.498 0.462 0.568 0.539 ...
##  $ eFGpct      : num  0.573 0.504 0.481 0.525 0.546 0.499 0.47 0.48 0.551 0.502 ...
##  $ FT          : num  2.3 0.9 0.4 1.2 1.2 2.7 0.6 1.4 1.1 2.3 ...
##  $ FTA         : num  3.1 1.1 0.5 1.7 1.4 3.3 0.8 1.8 1.6 3.2 ...
##  $ FTpct       : num  0.743 0.868 0.808 0.729 0.842 0.822 0.7 0.795 0.651 0.711 ...
##  $ ORB         : num  1.7 0.4 0.3 1 1.6 0.6 1.2 0.8 1.3 1.9 ...
##  $ DRB         : num  4.2 1.6 1.4 2.5 6.1 4.3 4 2.8 2.8 3.5 ...
##  $ TRB         : num  5.9 1.9 1.7 3.6 7.7 4.9 5.2 3.6 4.1 5.5 ...
##  $ AST         : num  2.5 2.4 0.4 1.4 3.4 3 2.1 4 1.2 2.6 ...
##  $ STL         : num  0.6 0.7 0.4 0.6 0.7 1 0.6 1.7 0.3 0.8 ...
##  $ BLK         : num  0.6 0.1 0.1 0.2 1.3 0.3 0.6 0.4 0.6 0.9 ...
##  $ TOV         : num  1.8 1.1 0.6 1.1 0.9 1.1 1.5 1.4 1.1 2 ...
##  $ PF          : num  2 1.5 1.3 1.9 1.9 2.7 1.4 2.6 2.6 3 ...
##  $ PTS         : num  15 6.3 3.8 8.3 10.2 11.7 7.6 7.4 6 9.6 ...
```

```
##  $ PER          : num  15.3 12.6 7.3 10.3 16.7 13.7 12 11.7 13.1 16 ...
##  $ TSpct        : num  0.602 0.544 0.507 0.556 0.574 0.559 0.485 0.528 0.577 0.552 ...
##  $ X3PAr        : num  0.312 0.305 0.632 0.409 0.466 0.534 0.432 0.492 0.119 0.223 ...
##  $ FTr          : num  0.276 0.201 0.143 0.252 0.167 0.363 0.11 0.285 0.358 0.442 ...
##  $ ORBpct       : num  6.1 2.6 2.9 4.3 6 2.1 6 3.3 9 10.1 ...
##  $ DRBpct       : num  14.3 10.3 13.6 11 22.2 16.6 20.8 11.2 19.3 18.9 ...
##  $ TRBpct       : num  10.3 6.5 8.4 7.6 14.3 9.2 13.3 7.3 14.1 14.5 ...
##  $ ASTpct       : num  11.6 20.7 5.4 8.5 16.4 15.7 16.5 18.5 10.6 19.1 ...
##  $ STLpct       : num  0.9 2 1.7 1.2 1.2 1.8 1.5 3 1 1.9 ...
##  $ BLKpct       : num  1.7 0.7 0.8 0.8 4.2 1.1 2.9 1.1 3.5 4.1 ...
##  $ TOVpct       : num  12.5 15.4 13.8 12.6 9.6 9.7 16.4 16.5 17.4 18.8 ...
##  $ USGpct       : num  19.7 18.7 17.2 15.3 14.8 17.7 20 13.3 17.1 22 ...
##  $ OWS          : num  3.2 0.5 -0.4 0.5 3.7 3.2 -1.1 0.7 0.4 0.8 ...
##  $ DWS          : num  2 0.9 0.9 0.8 3.8 2.9 1.5 1.2 0.5 1.3 ...
##  $ WS           : num  5.2 1.5 0.4 1.2 7.6 6.1 0.4 2 0.9 2.1 ...
##  $ WSper48      : num  0.105 0.068 0.038 0.048 0.181 0.126 0.014 0.082 0.07 0.068 ...
##  $ OBPM         : num  0.5 -1.9 -4.9 -3.4 1.4 -0.4 -2.7 -2.2 -3.2 -1.6 ...
##  $ DBPM         : num  -1.1 0.3 0.7 -0.9 2.9 1.2 0.1 2.3 0.3 0.6 ...
##  $ BPM          : num  -0.6 -1.7 -4.3 -4.3 4.3 0.8 -2.6 0.2 -2.9 -1 ...
##  $ VORP         : num  0.9 0.1 -0.3 -0.7 3.2 1.7 -0.2 0.6 -0.1 0.4 ...
##  $ Player.y     : chr  "Aaron Gordon" "Aaron Holiday" "Aaron Nesmith" "Aaron Wiggins" ...
##  $ X2022.23     : chr  "$19690909" "$1968175" "$3804360" "$1563518" ...
##  $ Tm.y         : chr  "DEN" "ATL" "IND" "OKC" ...
##  $ Signed.Using: chr  "Bird" "Minimum Salary" "1st Round Pick" "" ...
##  $ Guaranteed   : int  1 1 1 0 1 1 1 1 1 1 ...
##  $ Rk.y         : int  63 362 266 430 48 139 281 154 261 278 ...
##  $ Weight       : int  235 185 215 190 240 214 NA 186 250 NA ...
##  $ Height       : chr  "6-8" "6-0" "6-5" "6-4" ...
```

4.3 Remove repeated variables

4.3.1 Player Name

```
sum(all$Player.x != all$Player.y)
```

```
## [1] 1
```

Since there is no difference in the player name, I will remove player.y and renaming player.x to name

```
all <- all %>%
    dplyr::select(!Player.y) %>%
    rename(name = Player.x)
```

4.3.2 Rank

It is rank in their original respective table (alphabetical order of player name in player statistics tables and salary in 2022-23 season for salary table).
Since, it doesn't carry any extra information, I will remove both of the variables.

```
all <- all %>%
    dplyr::select(!c(Rk.x, Rk.y))
```

4.3.3 Team

Tm.x is the team the player in in 2021-22 season while Tm.y is the team of 2022-23 season. I will change Tm.x to team_2021 and Tm.y to team_2022.

```
all <- all %>%
    rename(team_2021 = Tm.x, team_2022 = Tm.y)
```

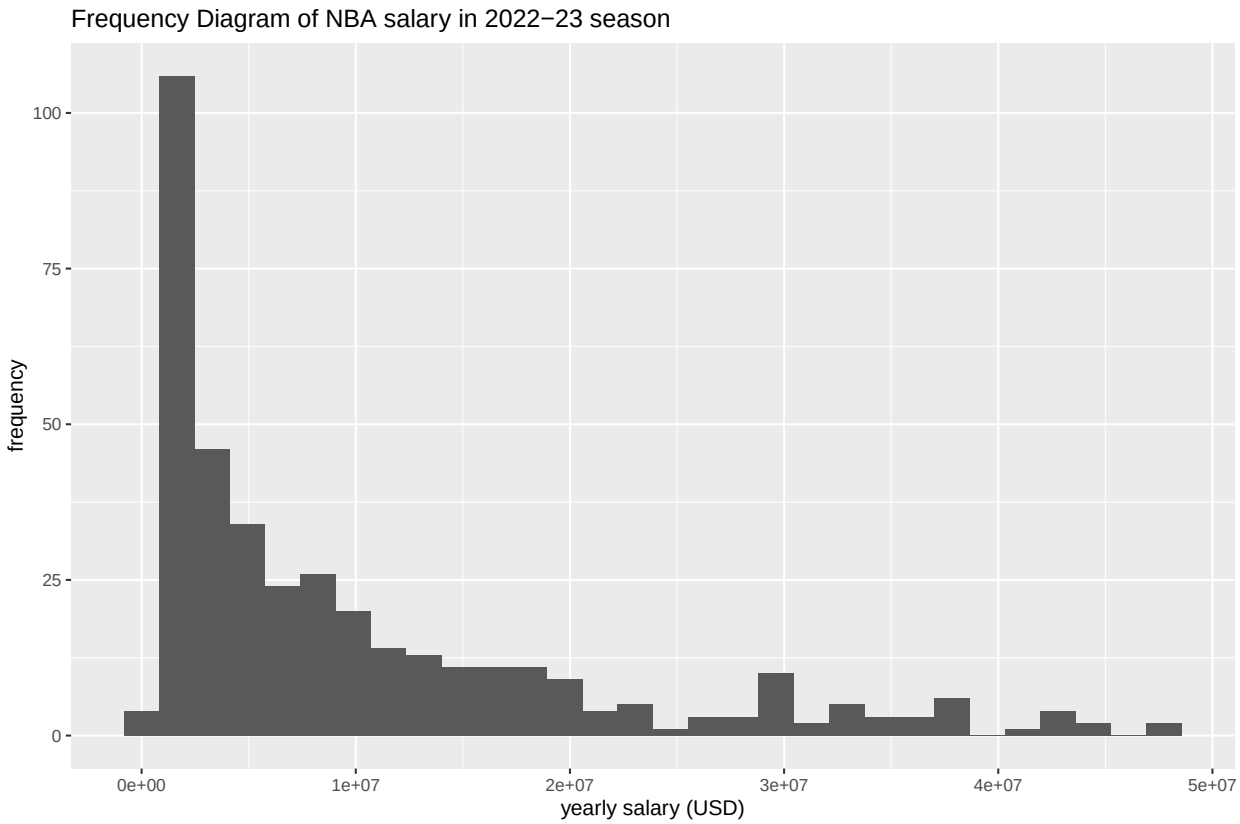
5 Exploratory analysis

5.1 The response variable: salary

The aim of this project is to predict the salary next year. I will remove the salary of 2023-24 season onward and change the X2022.23 to numeric variables.

```
all <- all %>%
    rename(salary = X2022.23) %>%
    mutate(salary = as.numeric(str_extract(salary, "[0-9]+")))
```

```
ggplot(data = all %>%
    drop_na(salary), aes(x = salary)) + geom_histogram(bins = 30) + labs(title = "Frequency Diagram of
    x = "yearly salary (USD)", y = "frequency")
```



The salary is highly right skewed. This is expected as the top NBA players are paid more in order for the team to keep their top players.
There is high frequency concentrated on 0 to 2 million range. This might be because of the existence of minimum salary in NBA, which is 1 million to 3 million per year depending on their experience (Adams, 2022). I will keep that in mind.

```
summary(all$salary)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##    333333   2196960   5837760 10131574 13955840 48070014         1
```

The salary of salary of NBA players in 2022-23 season (who played in 2021-22 season) ranges from 3.3 million USD to 48.1 million USD. The median of the salary is 5.8 million USD and the mean is 10 million USD.

5.2 Important Numeric Variables

I will first use the correlation with salary to get a feel on the numeric variables on the response variables

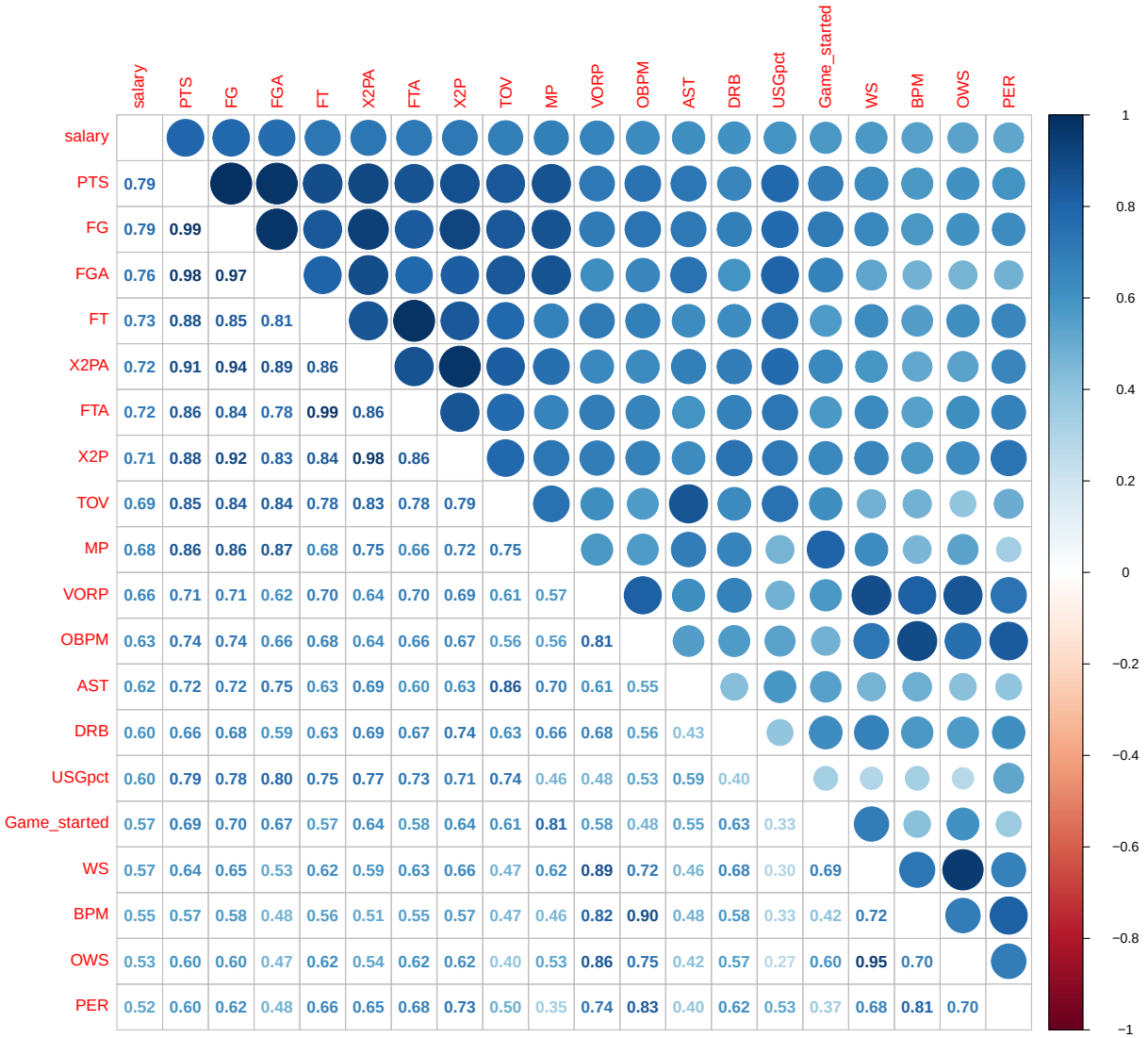
5.2.1 Correlation with salary 2022-23

```
numVar <- which(sapply(all, is.numeric))
numVarNames <- names(numVar)
length(numVarNames)
```

```
## [1] 50
```

There are 50 numeric variables

```
all_numVar <- all[, numVar]
all_numVar <- dplyr::select(all_numVar, !X)

cor_Mat <- cor(all_numVar, use = "pairwise.complete.obs")

cor_names <- names(sort(cor_Mat[, "salary"], decreasing = TRUE))[1:20]

cor_Mat <- cor_Mat[cor_names, cor_names]

corrplot.mixed(cor_Mat, tl.pos = "lt")
```



We chose three variables that show high correlation with the salary: Points per game, Value Over replacement player, and assists per game.
The correlation shows a high multicollinearity among predictive variables.
For example:

- FG, FGA, FT, X2PA, FTA, X2P, TOV, and MP all have correlation higher 0.8 with PTS.
- OWS has 0.95 correlation with WS
- BPM has 0.9 correlation with OBPM

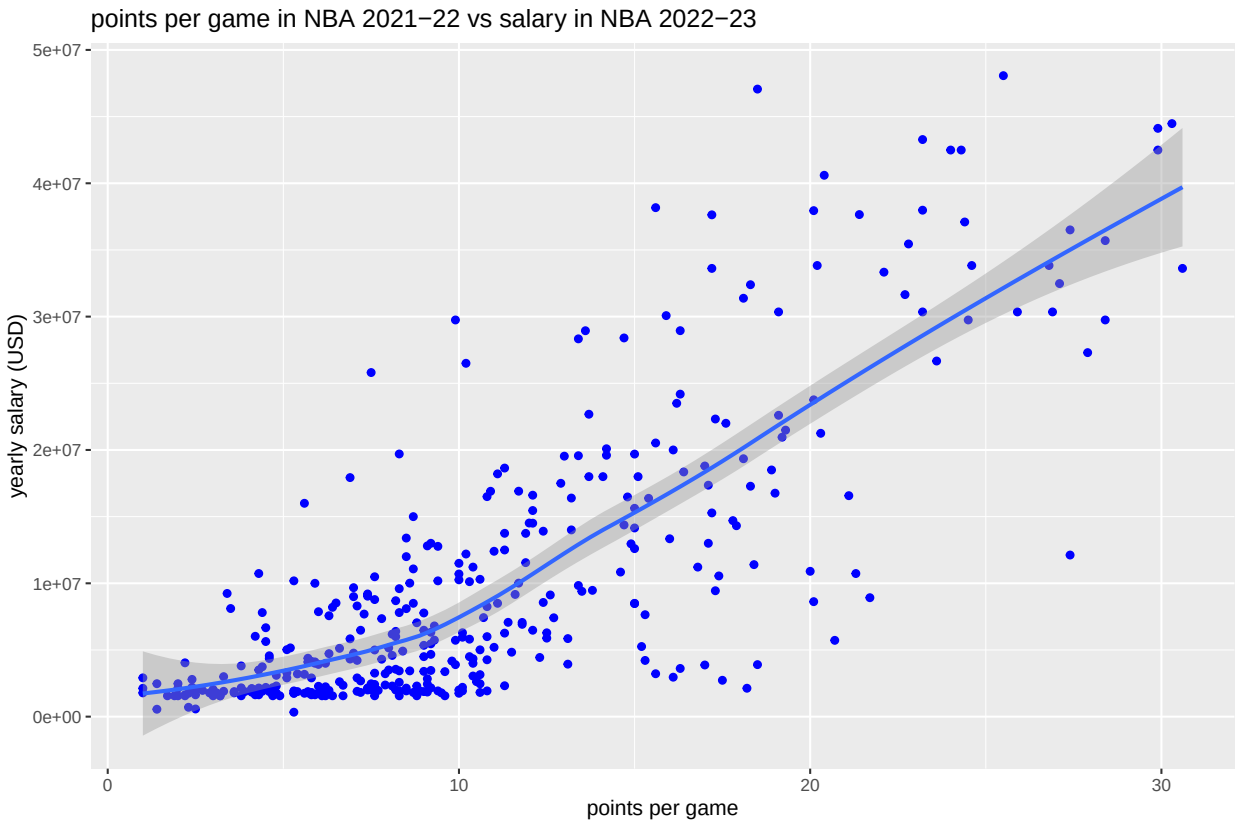
5.2.2 Points

PTS:

`PTS - Points per game`

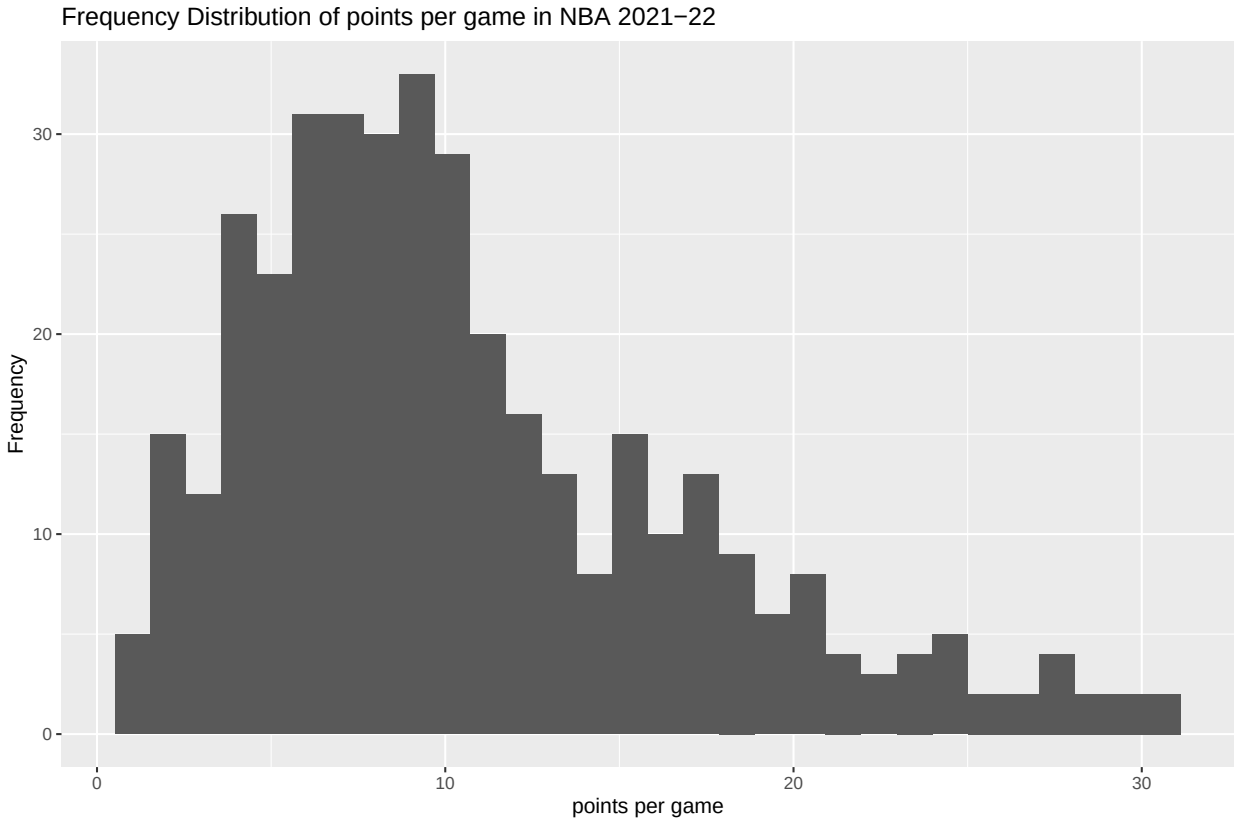
It has the highest correlation with salary among the numeric variables (0.7944907). It is the average point per game played.

```
ggplot(all %>%
    drop_na(PTS, salary), aes(x = PTS, y = salary)) + geom_point(col = "blue") +
    geom_smooth(formula = y ~ x, method = "loess") + labs(title = "points per game in NBA 2021-22 vs sa
    x = "points per game", y = "yearly salary (USD)")
```



There is a clear linear correlation between salary and points per game. The correlation is smaller when the points per game is below about 9 but increase after it goes above 9 points.

```
ggplot(all %>%
    drop_na(PTS, salary), aes(x = PTS)) + geom_histogram(bins = 30) + labs(title = "Frequency Distributi
    x = "points per game", y = "Frequency")
```



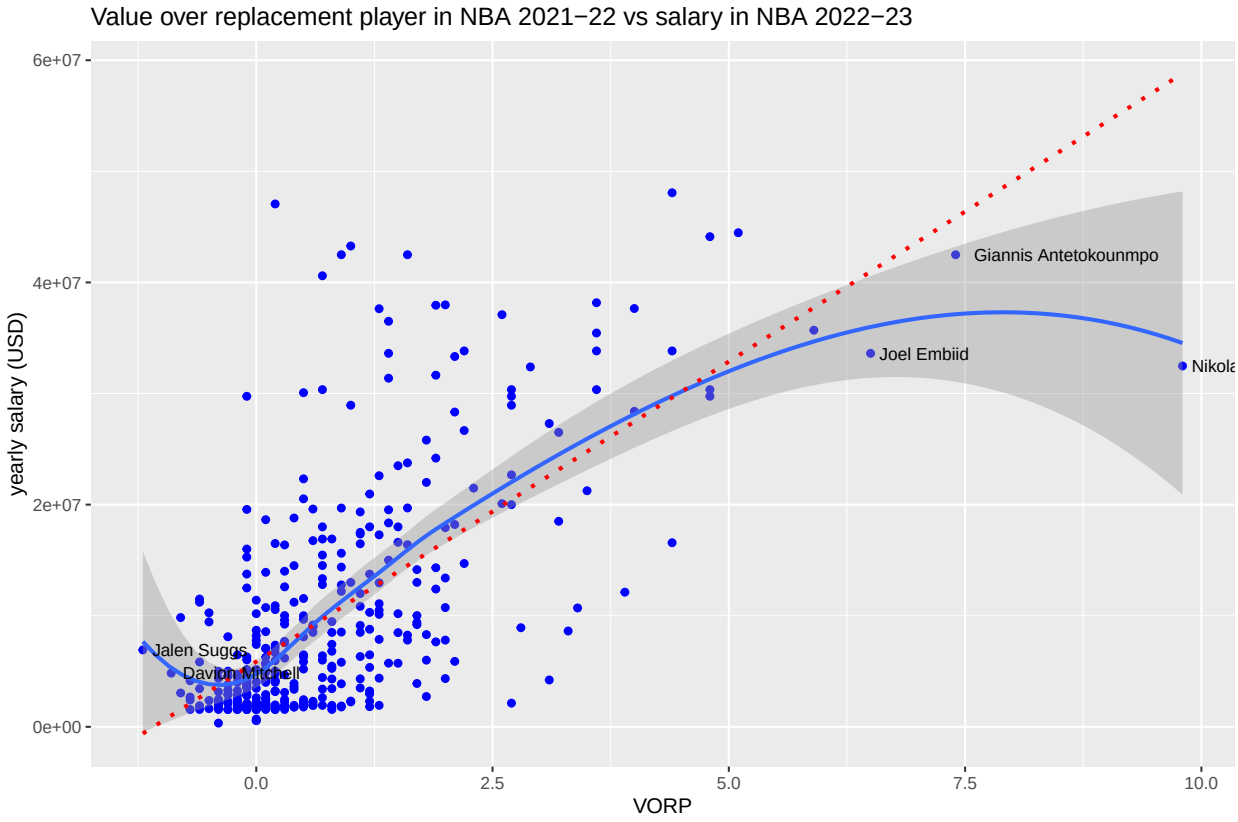
5.2.3 Value Over Replacement player

Value Over Replacement Player:

```
VORP - Value Over Replacement Player (available since the 1973-74 season in the NBA); a box score estima
```

Although FG, FGA, FT etc are more highly correlated, they are also highly correlated to points per game (> 0.75). I will look at the next one that is not highly correlated to points per game. It has a correlation of (0.6645534) with salary. It is a estimate of the value provide player over a below average player.

```
ggplot(all %>%
    drop_na(VORP, salary), aes(x = VORP, y = salary, label = ifelse(VORP <= -0.9 |
    VORP > 6, name, ""))) + geom_point(col = "blue") + geom_smooth(formula = y ~
    x, method = "loess") + geom_smooth(formula = y ~ x, method = "glm", linetype = "dotted",
    col = "red", se = FALSE) + labs(title = "Value over replacement player in NBA 2021-22 vs salary in
    x = "VORP", y = "yearly salary (USD)") + geom_text(size = 3, hjust = -0.1)
```



It shows clear linear correlation except some in both extreme of the VORP. The non-linear part are due to the outliers while the general trend still shows positive correlation.

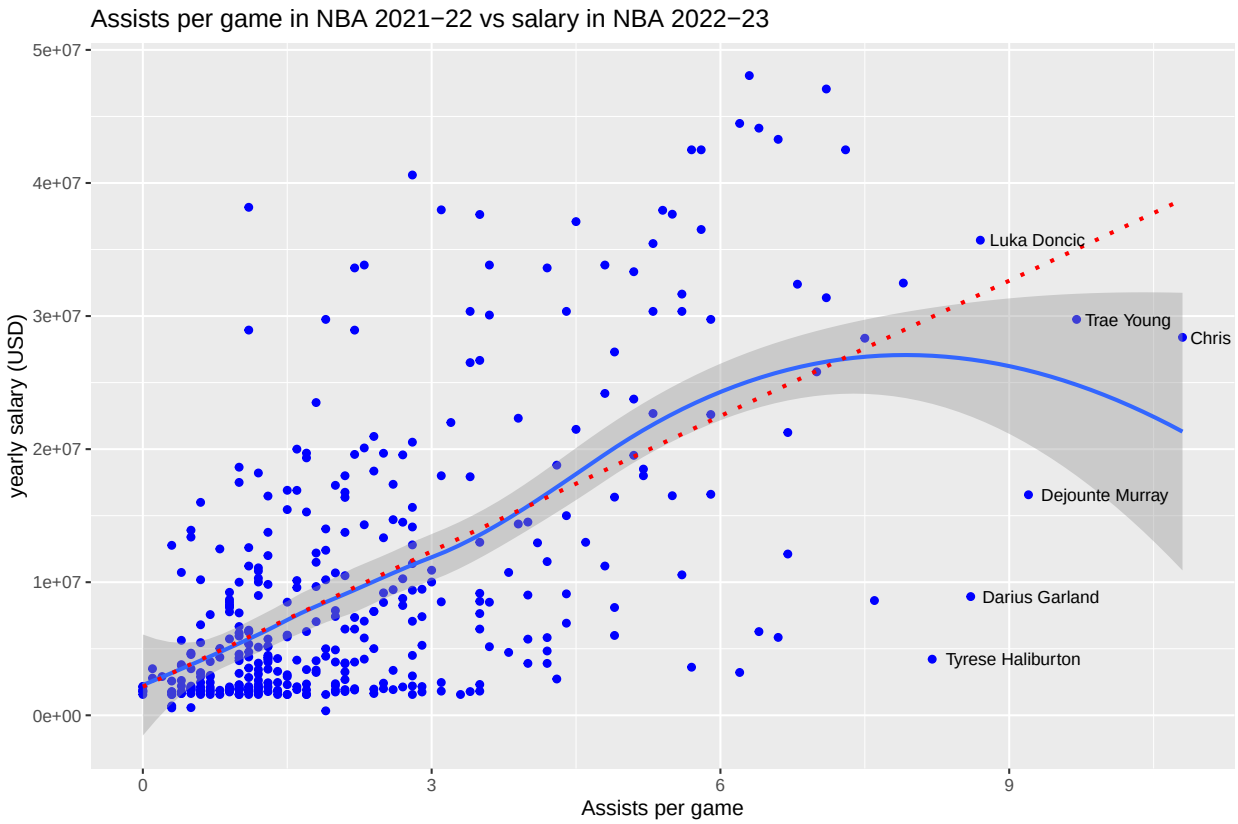
5.2.4 Assists

Assists:

`AST - Assists per game`

It has a high correlation with salary while not a having such a high correlation with points per game. It has a correlation of (0.6154281) with salary. It is the passes that lead to a field goal for the team.

```
ggplot(all %>%
    drop_na(AST, salary), aes(x = AST, y = salary, label = ifelse(AST > 8, name,
    ""))) + geom_point(col = "blue") + geom_smooth(formula = y ~ x, method = "loess") +
    geom_smooth(formula = y ~ x, method = "glm", linetype = "dotted", col = "red",
        se = FALSE) + labs(title = "Assists per game in NBA 2021-22 vs salary in NBA 2022-23",
    x = "Assists per game", y = "yearly salary (USD)") + geom_text(size = 3, hjust = -0.1)
```



It show positive correlation until it goes above 6 assists per game where it shows negative correlation. This maybe explained by that the players with high assist are usually not the first attacking choice of the team which means they might not be the top player of the team and thus lower salary.

6 Imputing missing data and factorising character variables.

6.1 Impute missing data

```
Nacol <- names(which(colSums(is.na(all) | all == "") > 0))
sort(colSums(sapply(all[Nacol], function(x) is.na(x) | x == "")), decreasing = TRUE)
```

```
## Signed.Using       Weight       Height       X3Ppct     Position        FTpct
##           49           17           17           14            8            5
##       salary
##            1
```

6.1.1 Salary

I will impute the salary since it is the most important variable of the dataset (response variable).

```
kable(all[is.na(all$salary), c("X", "name", "salary")])
```

	X	name	salary
146	146	Ish Wainright	NA

The salary of Ish Wainright is 125000 USD spotrac (n.d.) (source: spotrac, Accessed: 05/08/2022).

```
all$salary[all$name == "Ish Wainright"] <- 125000
```

6.1.2 Signed.Using

Signed.Using:

`The type of contract use to sign`

Changing the value with capitalisation difference to the same to make it the same factor.

```
unique(all$Signed.Using)
```

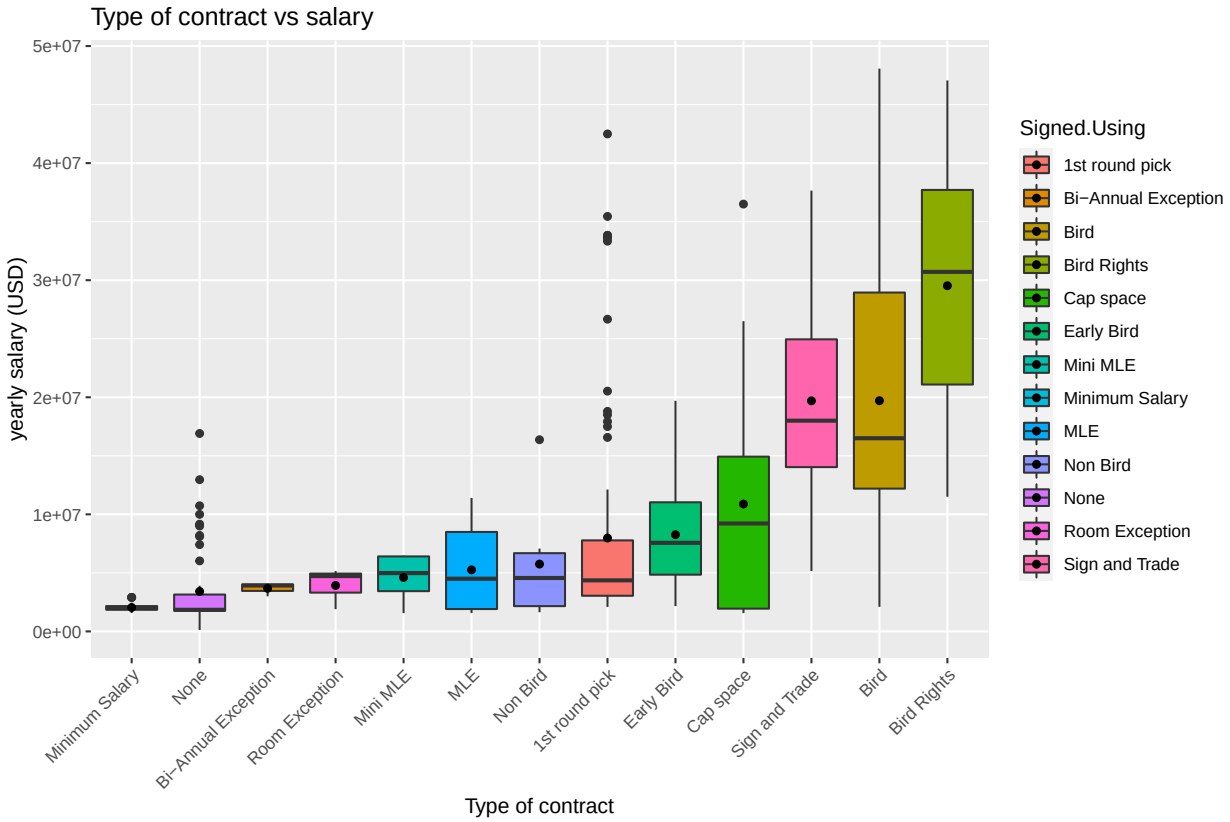
```
##  [1] "Bird"                "Minimum Salary"      "1st Round Pick"
##  [4] ""                    "Cap Space"           "MLE"
##  [7] "Bi-Annual Exception" "Mini MLE"            "Early Bird"
## [10] "1st Round pick"      "Sign and Trade"      "Bird Rights"
## [13] "Room Exception"      "1st round pick"      "Non Bird"
## [16] "Cap space"
```

```
all$Signed.Using[grep("^1st [Rr]ound [Pp]ick", all$Signed.Using)] <- "1st round pick"
all$Signed.Using[grep("Cap [Ss]pace", all$Signed.Using)] <- "Cap space"
```

THe NAs mean nothing special about the signing of the contract. The NAs are replaced by "None".

```
all$Signed.Using[is.na(all$Signed.Using) | all$Signed.Using == ""] <- "None"
```

```
ggplot(all, aes(x = fct_reorder(as.factor(Signed.Using), salary, .fun = "mean"),
    y = salary, fill = Signed.Using)) + geom_boxplot() + geom_point(stat = "summary",
    fun = "mean") + labs(title = "Type of contract vs salary", x = "Type of contract",
    y = "yearly salary (USD)") + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



There are no clear ordinal element in the Signed.Using variable so it will be kept as a character variable.

6.1.3 Height and Weight

Some of the players height and weight are missing. We manually searched up each player and input them. The data are from Basketball Reference.

```
kable(all[which(is.na(all$Weight) | is.na(all$Height)), c("name", "Weight", "Height")])
```

	name	Weight	Height
7	Aleksej Pokusevski	NA	NA
10	Alperen Şengün	NA	NA
22	Boban Marjanović	NA	NA
24	Bogdan Bogdanović	NA	NA
25	Bojan Bogdanović	NA	NA
29	Brandon Boston Jr.	NA	NA
75	Dāvis Bertāns	NA	NA
185	Jonas Valančiūnas	NA	NA
204	Jusuf Nurkić	NA	NA
245	Luka Dončić	NA	NA
253	Marcus Morris	NA	NA
285	Nikola Jokić	NA	NA
286	Nikola Vučević	NA	NA
315	Robert Williams	NA	NA
346	Théo Maledon	NA	NA

	name	Weight	Height
373	Vlatko Čančar	NA	NA
378	Willy Hernangómez	NA	NA

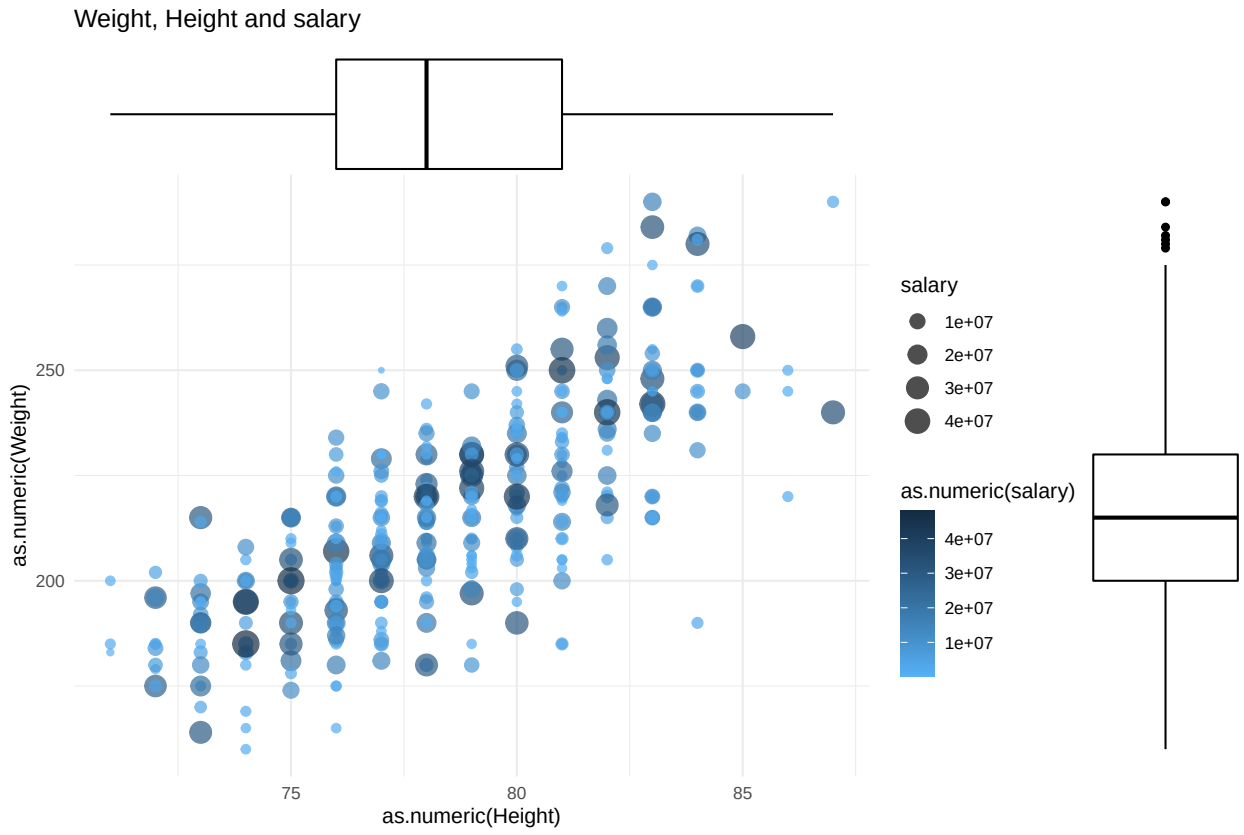
I will manually search up their height and weight.

```
all$Weight[which(is.na(all$Weight))] <- c(190, 235, 290, 220, 226, 185, 225, 265,
    290, 230, 218, 284, 260, 237, 175, 236, 250)

all$Height[which(is.na(all$Height))] <- c("7-0", "6-9", "7-3", "6-6", "6-7", "6-7",
    "6-10", "6-11", "6-11", "6-7", "6-8", "6-11", "6-10", "6-8", "6-4", "6-8", "6-11")
```

Change height unit from feet to inches.

```
heightinch <- as.numeric(sapply(all$Height, function(x) substring(x, 1, 1))) * 12 +
    as.numeric(sapply(all$Height, function(x) substring(x, 3, nchar(x))))

all$Height <- heightinch
```

```
g1 <- ggplot(all, aes(x = as.numeric(Height), y = as.numeric(Weight), col = as.numeric(salary),
    size = salary)) + geom_point(alpha = 0.7) + theme_minimal() + labs(title = "Weight, Height and sala
    scale_color_continuous(high = "#132B43", low = "#56B1F7")

ggMarginal(g1, type = "boxplot")
```



There clear positive correlation between height and weight but there is no visible correlation with salary.

6.1.4 X3Ppct

X3Ppct:

`3 point field goal percentage`

Some players had not made any 3 points attempt in the season which become NA.

```
kable(all[which(is.na(all$X3Ppct)), c("name", "X3P", "X3PA", "X3Ppct")])
```

	name	X3P	X3PA	X3Ppct
21	Bismack Biyombo	0	0	NA
82	DeAndre Jordan	0	0	NA
147	Ivica Zubac	0	0	NA
150	Jaden Springer	0	0	NA
176	Jericho Sims	0	0	NA
269	Mitchell Robinson	0	0	NA
273	Moses Brown	0	0	NA
280	Nic Claxton	0	0	NA
281	Nick Richards	0	0	NA
291	Onyeka Okongwu	0	0	NA
351	Tony Bradley	0	0	NA
364	Tyrell Terry	0	0	NA
368	Udoka Azubuike	0	0	NA
370	Vernon Carey Jr.	0	0	NA

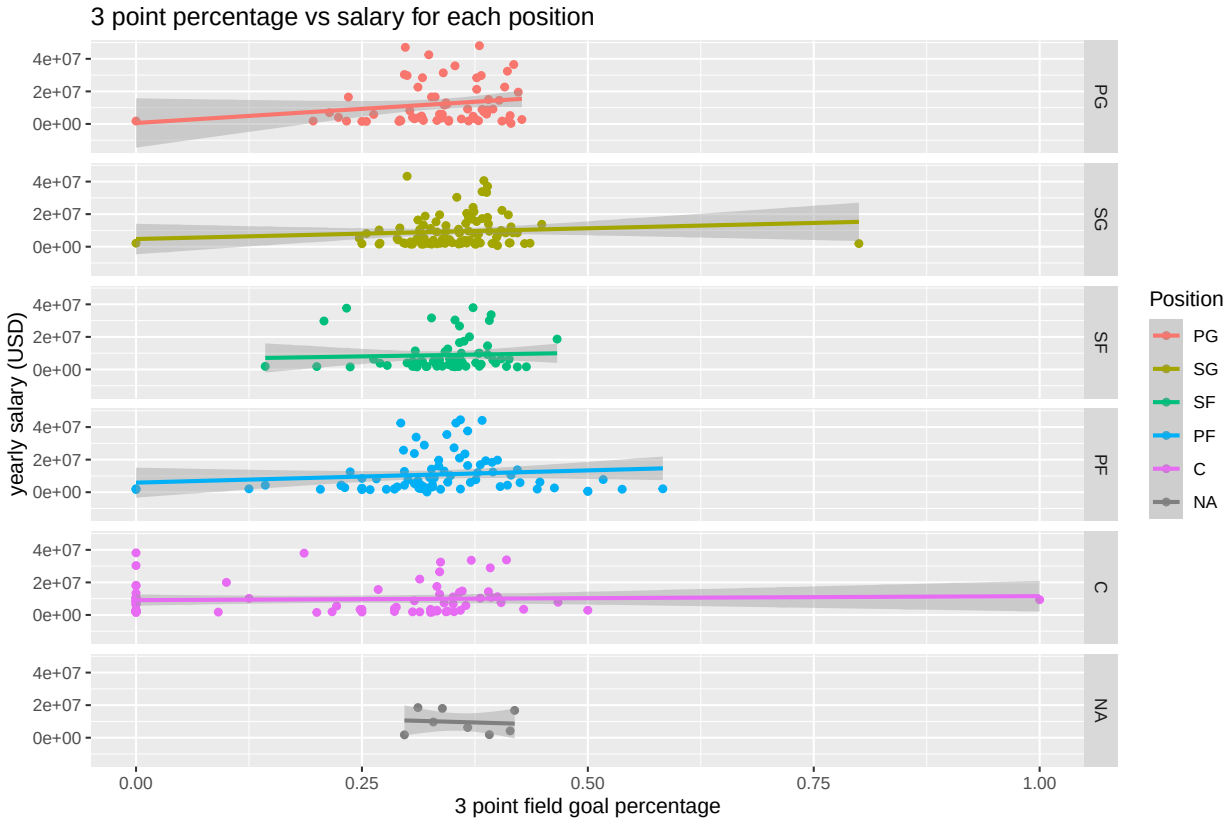
I will impute by setting to 0 if there is no 3 point attempt.

```
all$X3Ppct[which(is.na(all$X3Ppct))] <- sapply(which(is.na(all$X3Ppct)), function(x) ifelse(all$X3PA[x]
    0, 0, all$X3P[x]/all$X3PA[x]))
```

```
ggplot(all, aes(x = X3Ppct, y = salary)) + geom_point() + geom_smooth(col = "red",
    formula = y ~ x, method = "glm") + labs(title = "3 point percentage vs salary",
    x = "3 point field goal percentage", y = "yearly salary (USD)")
```

It shows a slight but not significant correlation between salary and 3 point percentage. Most players' three point percentage lies between 20% and 43%.

```
ggplot(all, aes(x = X3Ppct, y = salary, col = Position)) + geom_point() + geom_smooth(formula = y ~
    x, method = "glm") + facet_grid(Position ~ .) + labs(title = "3 point percentage vs salary for each
    x = "3 point field goal percentage", y = "yearly salary (USD)")
```



There are little to none correlation if separate to each position. This might be the result of the modern NBA requires everyone to have certain degree of three point ability regardless of their position.

6.1.5 Position

The position of the following is

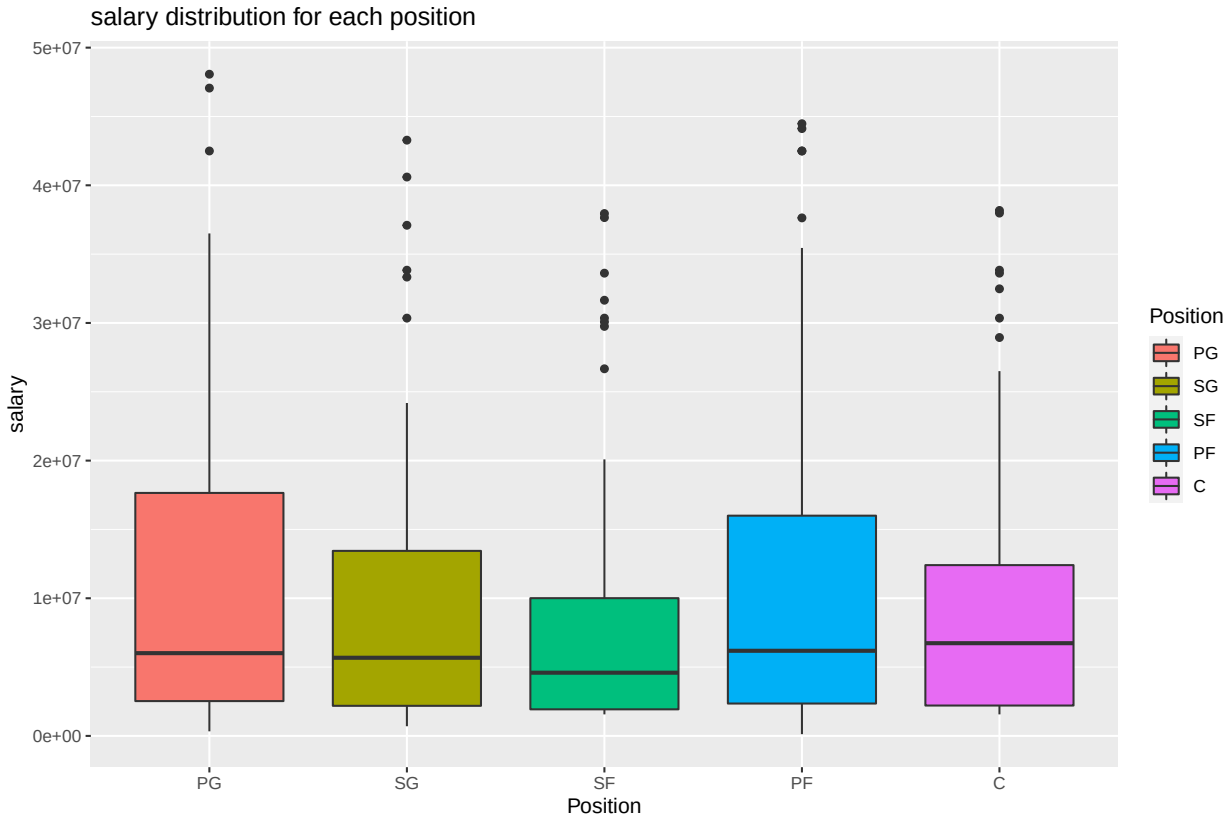
```
kable(all[is.na(all$Position), c("name", "Position")])
```

	name	Position
18	Armoni Brooks	NA
97	Didi Louzada	NA
99	Domantas Sabonis	NA
202	Justin Holiday	NA
237	Larry Nance Jr.	NA
287	Norman Powell	NA
331	Spencer Dinwiddie	NA
365	Tyrese Haliburton	NA

I will impute manually by search up their primary position

```
all$Position[is.na(all$Position)] <- c("SG", "SF", "PF", "SG", "PF", "SG", "PG",
    "PG")
```

```
ggplot(all, aes(x = Position, y = salary, fill = Position)) + geom_boxplot() + labs(title = "salary dis
```



Although there are a number corresponding to each position in basketball, the above show no ordinal correlation with the salary. Hence, I will keep it as a factor.

6.1.6 FTpct

FTpct:

`Free throw percentage`

Some players had not made a free throw attempt throughout the season which is record as NA.

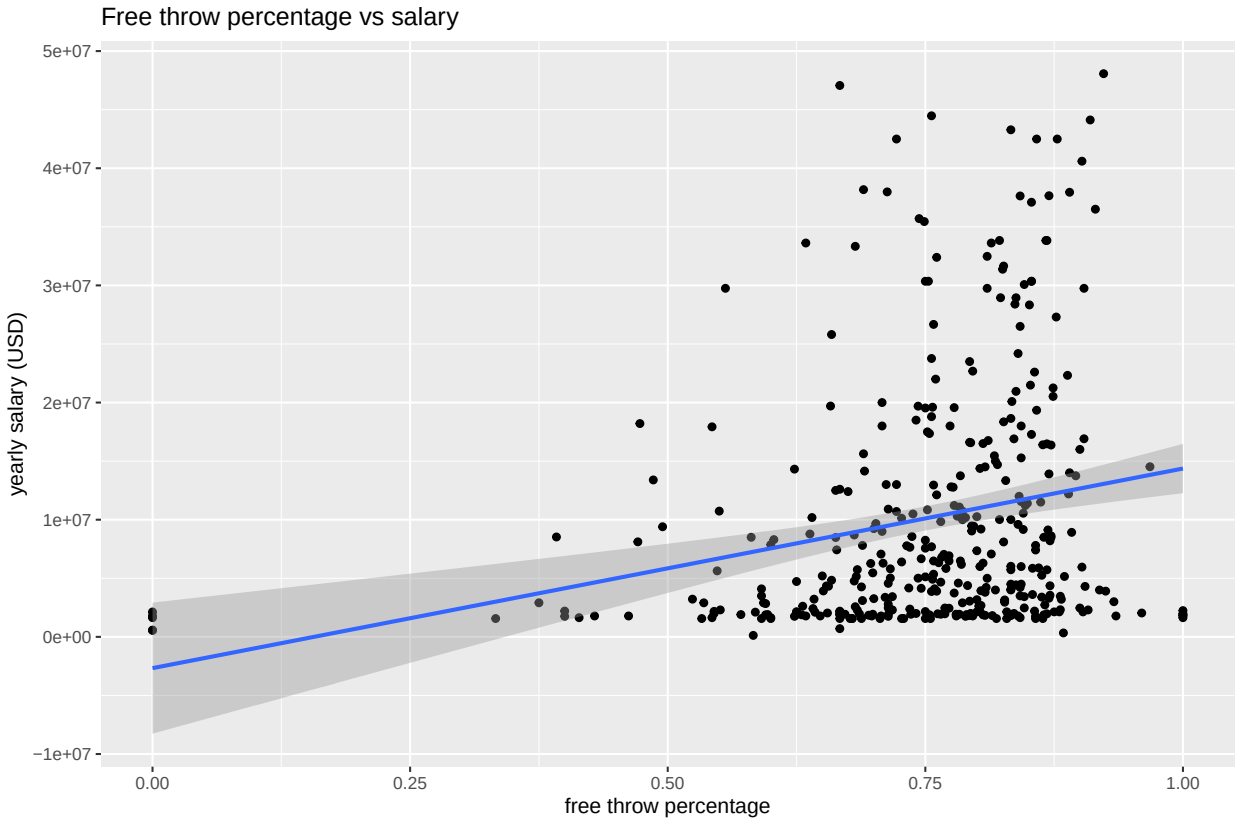
```
kable(all[which(is.na(all$FTpct)), c("FT", "FTA", "FTpct")])
```

	FT	FTA	FTpct
150	0	0	NA
205	0	0	NA
251	0	0	NA
325	0	0	NA
364	0	0	NA

I will impute by setting to 0 if there is no free throw attempt.

```
all$FTpct[which(is.na(all$FTpct))] <- sapply(which(is.na(all$FTpct)), function(x) ifelse(all$FTA[x] ==
    0, 0, all$FT[x]/all$FTA[x]))
```

```
ggplot(all, aes(x = FTpct, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = "glm") + labs(title = "Free throw percentage vs salary", x = "free throw percentage",
    y = "yearly salary (USD)")
```



6.1.7 Guaranteed

Guaranteed:

`The amount of a player's remaining salary that is guarenteed.`

Since it is a direct indication of the salary, I will remove this variable.

```
all <- dplyr::select(all, !Guaranteed)
```

6.2 Factorizing Character Variables

Find all character variables:

```
chrVar <- names(which(sapply(all, is.character)))
chrVar
```

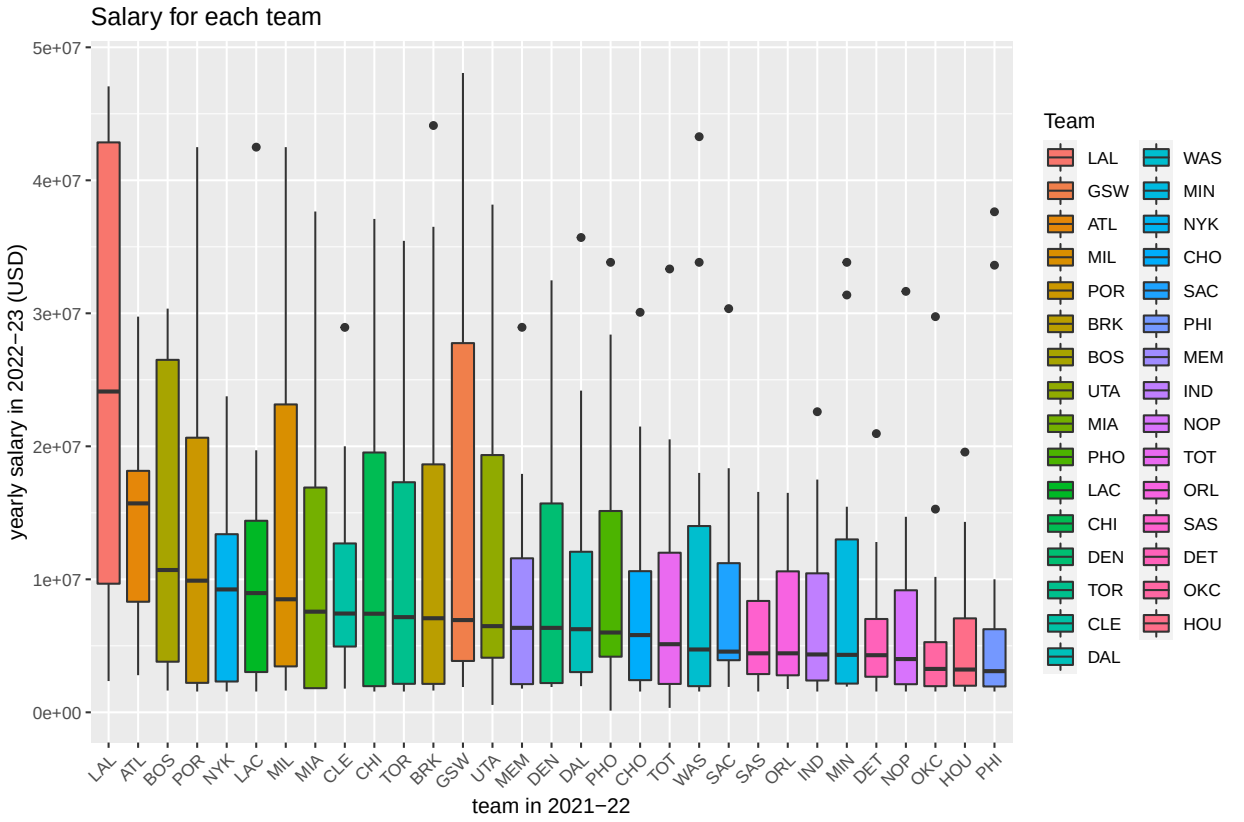
```
## [1] "name"         "player_id"    "team_2021"    "team_2022"    "Signed.Using"
```

6.2.1 Player Id and playe name

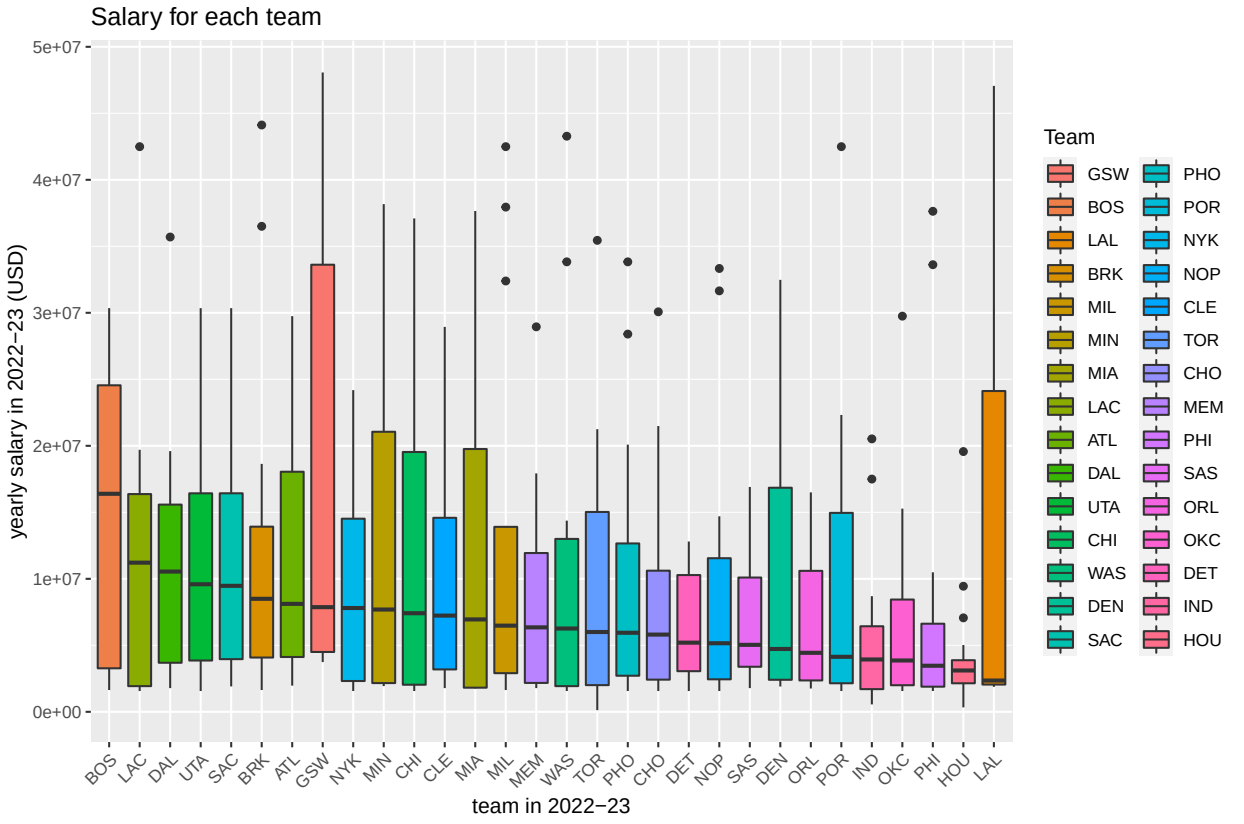
I will keep tempfor now to keep track of each entries but will remove them before fitting the model.

6.2.2 Team

```
ggplot(all, aes(x = fct_reorder(as.factor(team_2021), salary, median, .desc = TRUE),
    y = salary, fill = reorder(as.factor(team_2021), salary, .fun = "mean", decreasing = TRUE))) +
    geom_boxplot() + labs(title = "Salary for each team", x = "team in 2021-22",
    y = "yearly salary in 2022-23 (USD)") + theme(axis.text.x = element_text(angle = 45,
    hjust = 1)) + guides(fill = guide_legend(title = "Team"))
```




```
ggplot(all, aes(x = fct_reorder(as.factor(team_2022), salary, median, .desc = TRUE),
    y = salary, fill = reorder(as.factor(team_2022), salary, .fun = "mean", decreasing = TRUE))) +
    geom_boxplot() + labs(title = "Salary for each team", x = "team in 2022-23",
    y = "yearly salary in 2022-23 (USD)") + theme(axis.text.x = element_text(angle = 45,
    hjust = 1)) + guides(fill = guide_legend(title = "Team"))
```




```
all$team_2021 <- as.factor(all$team_2021)
all$team_2022 <- as.factor(all$team_2022)
```

6.2.3 Signed.Using

I will also remove this variable as this might be a direct indication to the salary of the player.

```
all <- dplyr::select(all, !Signed.Using)
```

7 Visualization

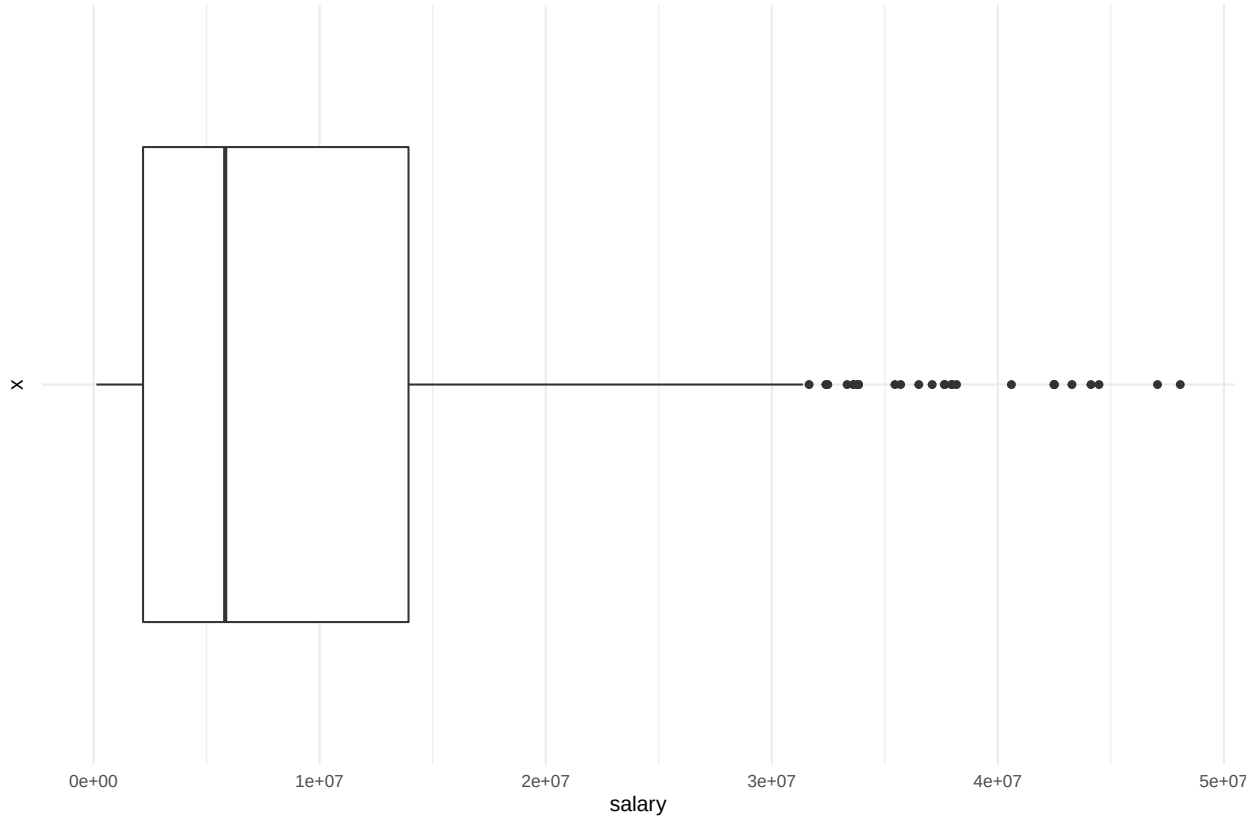
7.1 The response variable: salary

Although I have already done some visualization, I will visualize it again.

```
summary(all$salary)
```

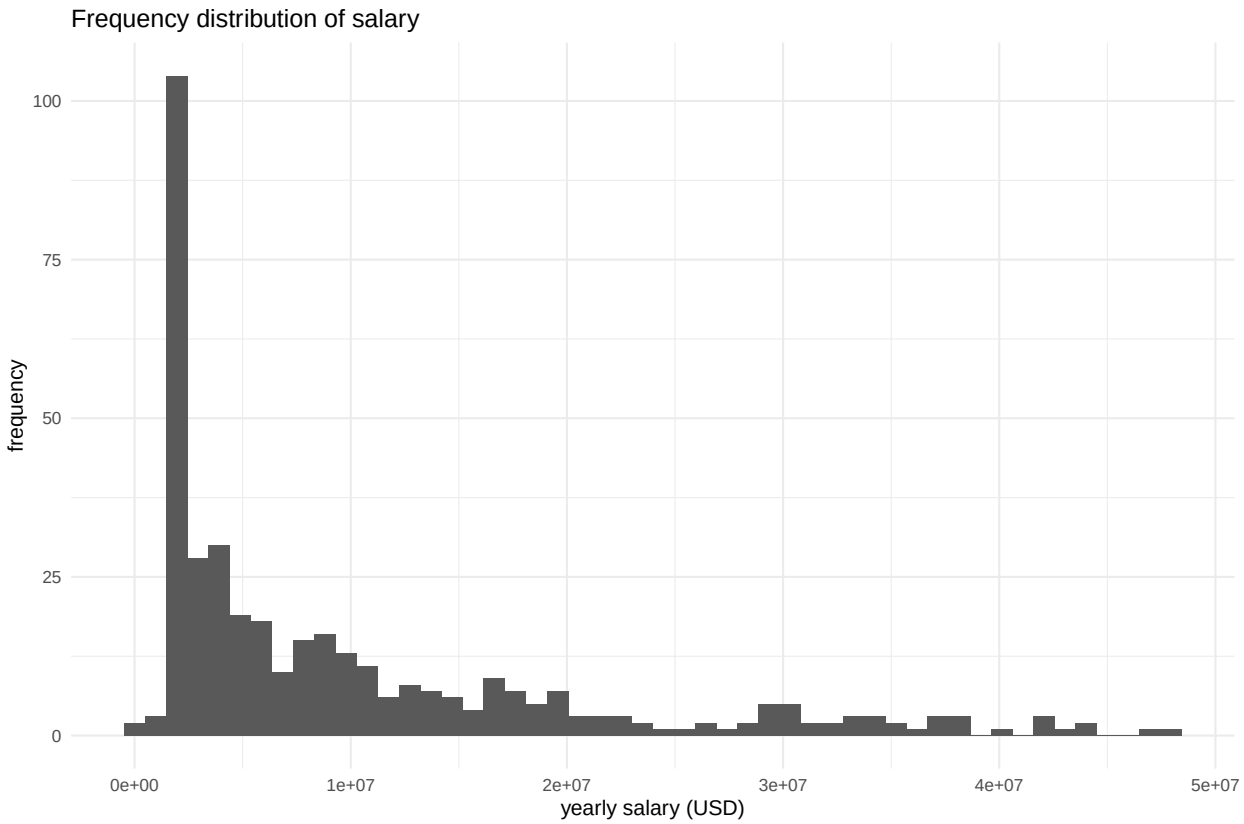
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##   125000  2189160  5823098 10105516 13931408 48070014
```

```
ggplot(data = all, aes(x = "", y = salary)) + geom_boxplot() + labs(y = "salary") +
    coord_flip() + theme_minimal()
```



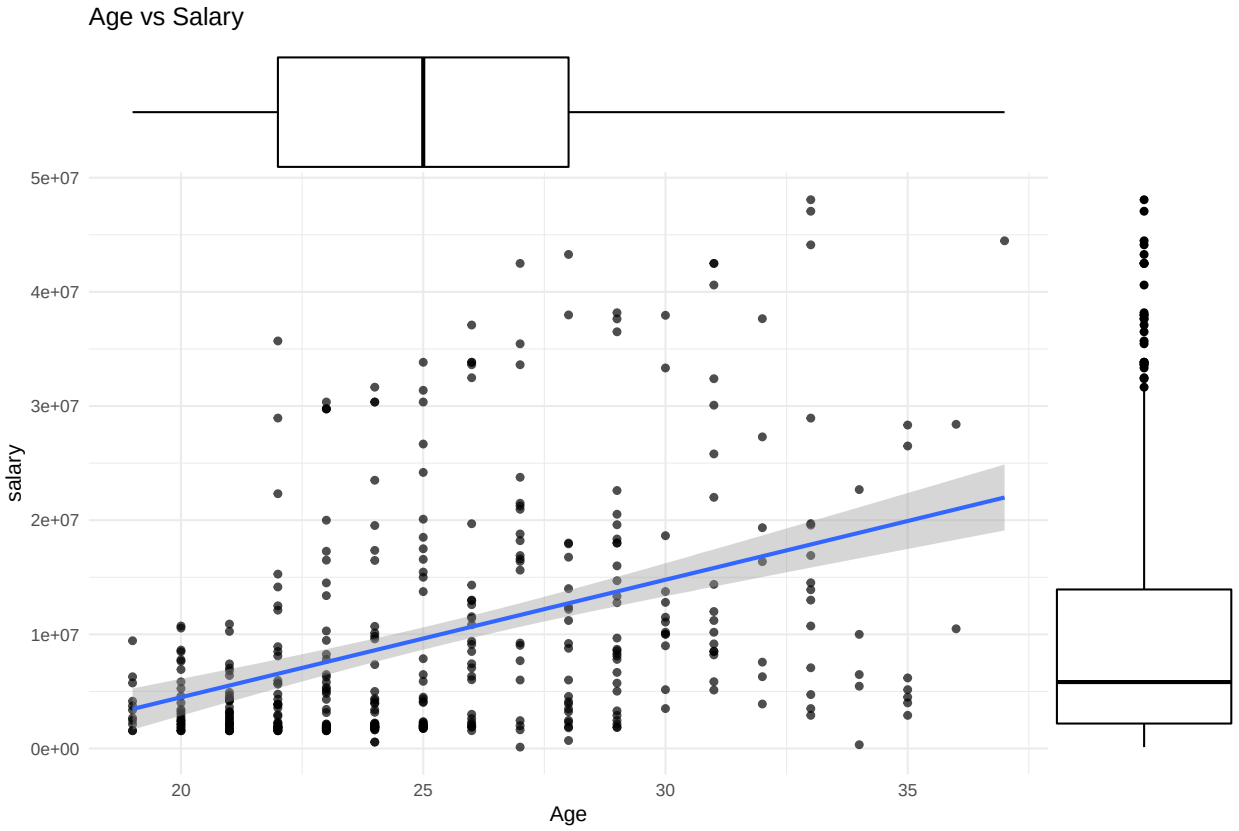
The salary (in USD) ranges from 0.1 million to 48.1 million with mean of 10.1 million and median of 5.8 million.

```
ggplot(data = all, aes(x = salary)) + geom_histogram(bins = 50) + labs(title = "Frequency distribution o
    x = "yearly salary (USD)", y = "frequency") + theme_minimal()
```



As from above, the data is highly right skewed and has a large spike in about 2 million.

```
g1 <- ggplot(all, aes(x = Age, y = salary)) + geom_point(alpha = 0.7) + theme_minimal() +
    geom_smooth(formula = y ~ x, method = "glm") + labs(title = "Age vs Salary")
ggMarginal(g1, type = "boxplot")
```



There are a clear positive correlation of age and salary. This can be due to various reason including the existence of rookie contract, experience players general earns more and players with longer experience have a higher minimum salary.

7.2 Grouping predictors

Group variables to different categories:

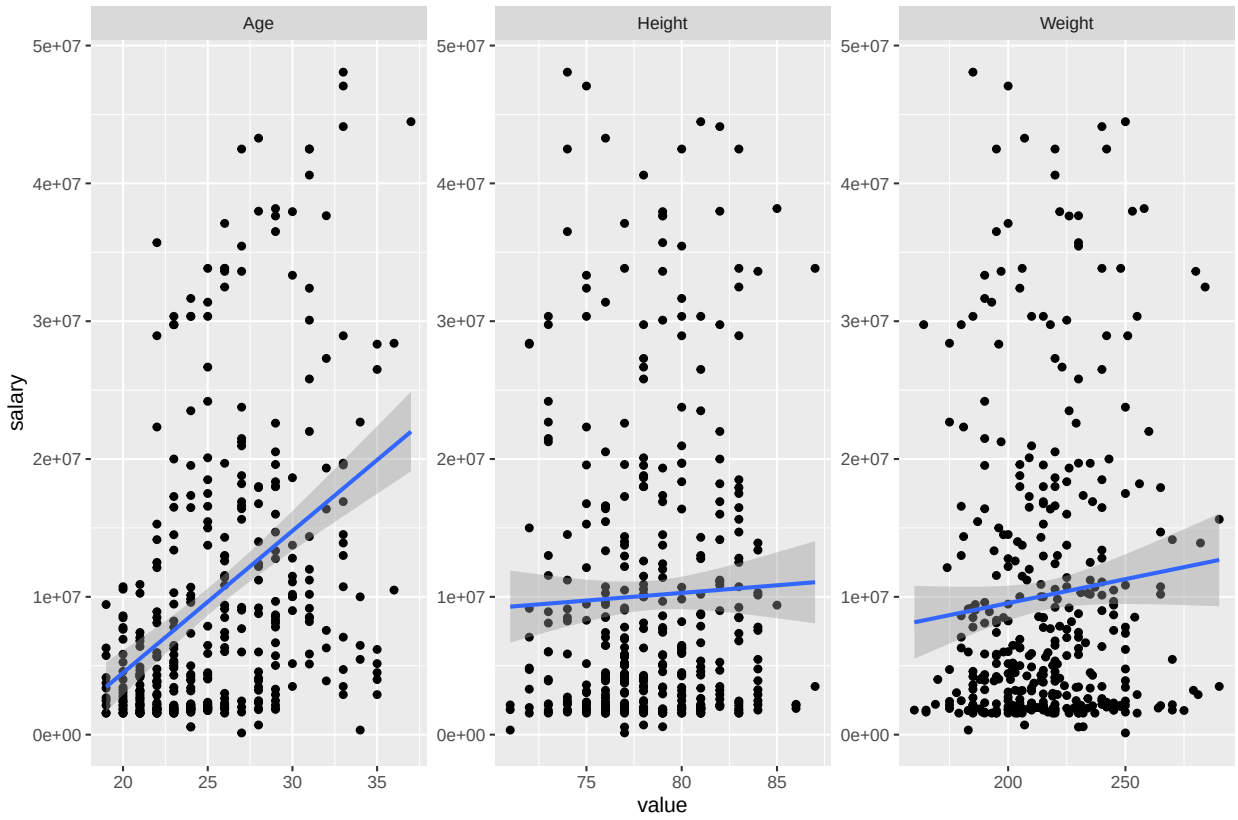
- Player bios: Basic information of the players
- Attendance: Measures of players' actually played
- shooting: General shoot attributes
- 2 pointers: 2 point shooting attributes
- 3 pointers: 3 point shooting attributes
- Free throw: Free throw shooting attributes
- Rebounding: Rebounding attributes
- Playmaking: Team play attributes
- Defence: defence related attributes
- Advance: Advance statistics that measure overall performance

```
pbio <- c("Age", "Height", "Weight")

attendence <- c("Game_played", "Game_started", "MP", "USGpct")

shooting <- c("FG", "FGA", "FGpct", "eFGpct", "PTS", "TSpct")

X2_point <- c("X2P", "X2PA", "X2Ppct")
```

```
X3_point <- c("X3P", "X3PA", "X3Ppct", "X3PAr")

Free_throw <- c("FT", "FTA", "FTpct", "FTr")

rebounding <- c("ORB", "DRB", "TRB", "ORBpct", "DRBpct", "TRBpct")

playmaking <- c("AST", "TOV", "ASTpct", "TOVpct")

defence <- c("STL", "BLK", "PF", "STLpct", "BLKpct")

overall_adstats <- c("PER", "OWS", "DWS", "WS", "WSper48", "OBPM", "DBPM", "BPM",
    "VORP")
```

7.2.1 Player Bio

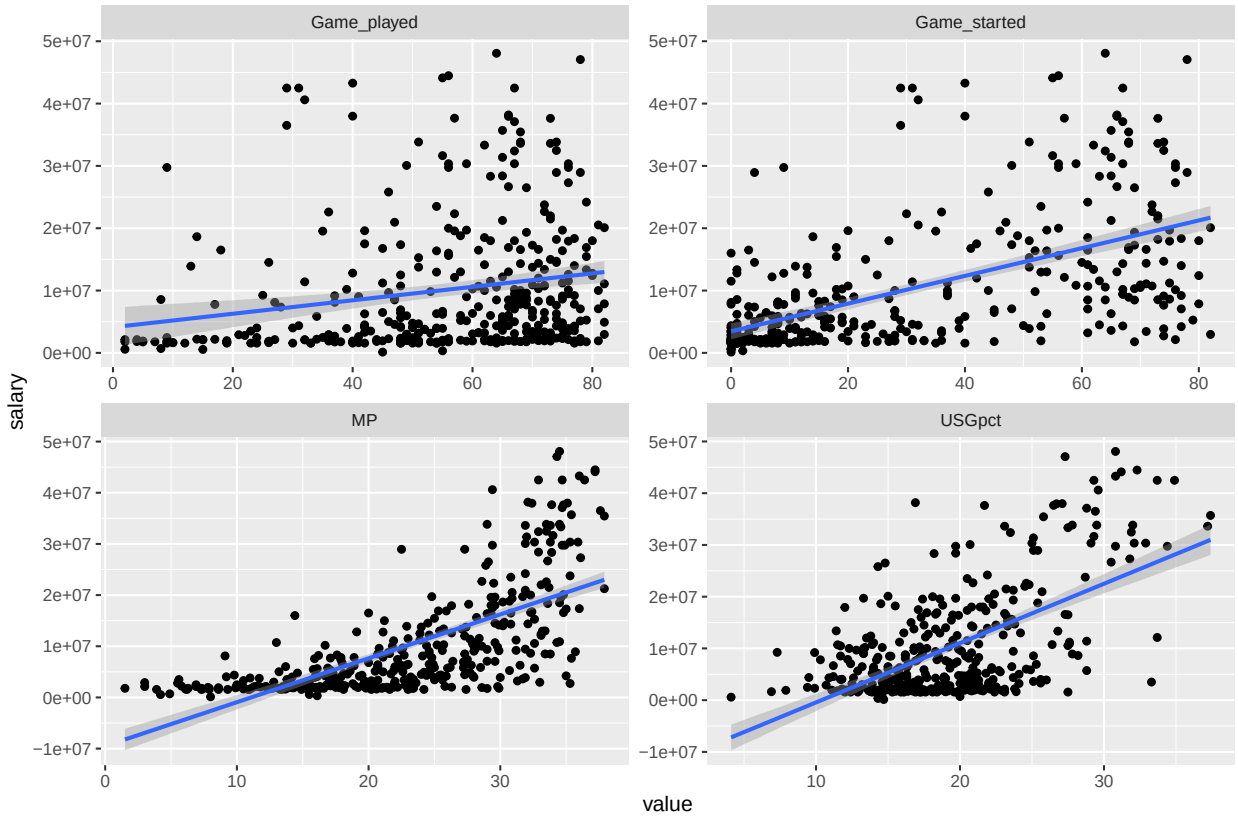
```
all_bio <- melt(all, id.vars = "salary", measure.vars = pbio)

ggplot(all_bio, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



The age show clear correlation while height and weight do not show much. It is reasonable as different position requires different height and weight and different player with different height has their own play style that suits their body. There is no clear correlation of their body and their performance and thus salary.

7.2.2 Attendence

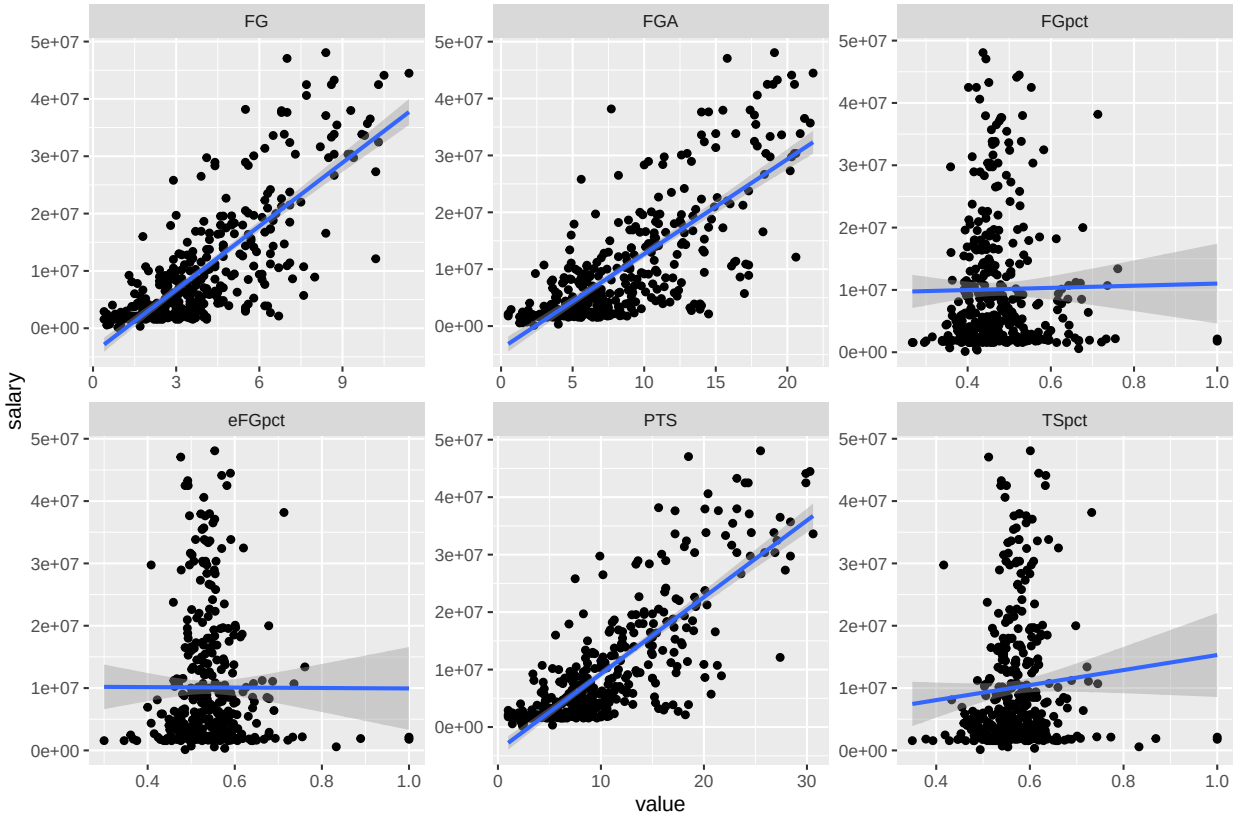
```
all_attendence <- melt(all, id.vars = "salary", measure.vars = attendence)

ggplot(all_attendence, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



All of the attributes show positive correlation. This make sense as player with stronger performance get played more and get paid more. Both attendance and salary are related to player performance.

7.2.3 Overall shooting

```
all_ovSh <- melt(all, id.vars = "salary", measure.vars = shooting)

ggplot(all_ovSh, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



Field goal percentage is directly correlated to FG and FGA and it doesn't have large correlation to salary. Effective field goal percentage is removed as it is similar to true shooting percentage and it has less correlation.

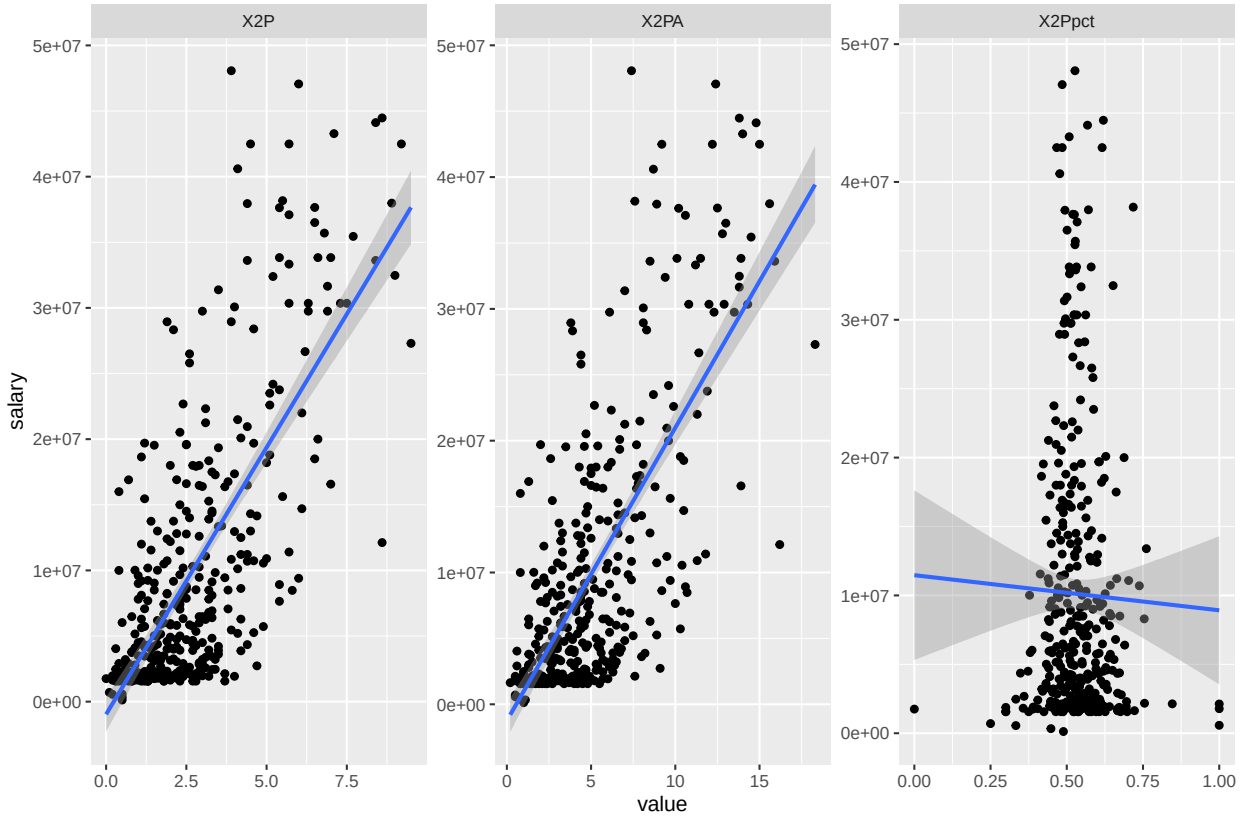
```
cor(all$FG, (all$FGA * all$FGpct))
```

```
## [1] 0.9999042
```

```
all <- all %>%
    dplyr::select(!eFGpct)
```

7.2.4 2 Pointers

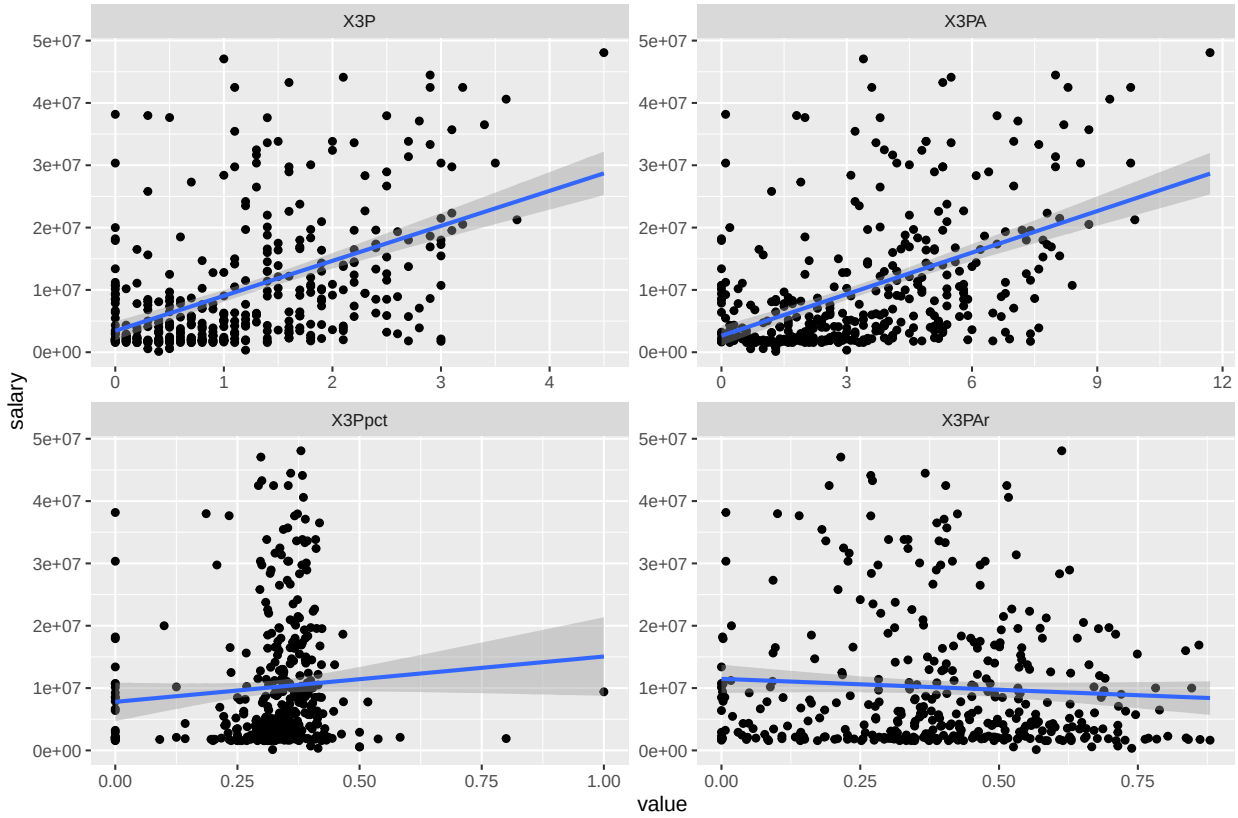
```
all_X2 <- melt(all, id.vars = "salary", measure.vars = X2_point)

ggplot(all_X2, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



It is out of expectation that 2 point field goal percentage has a negative correlation with salary. This can be due to players who shoot more tends to decrease in shooting percentage from fatigue while player with few attempt is easier to maintain high shooting percentage. At the same time players who shoot more are usually the top player of the team and thus have higher salary.

7.2.5 3 Pointers

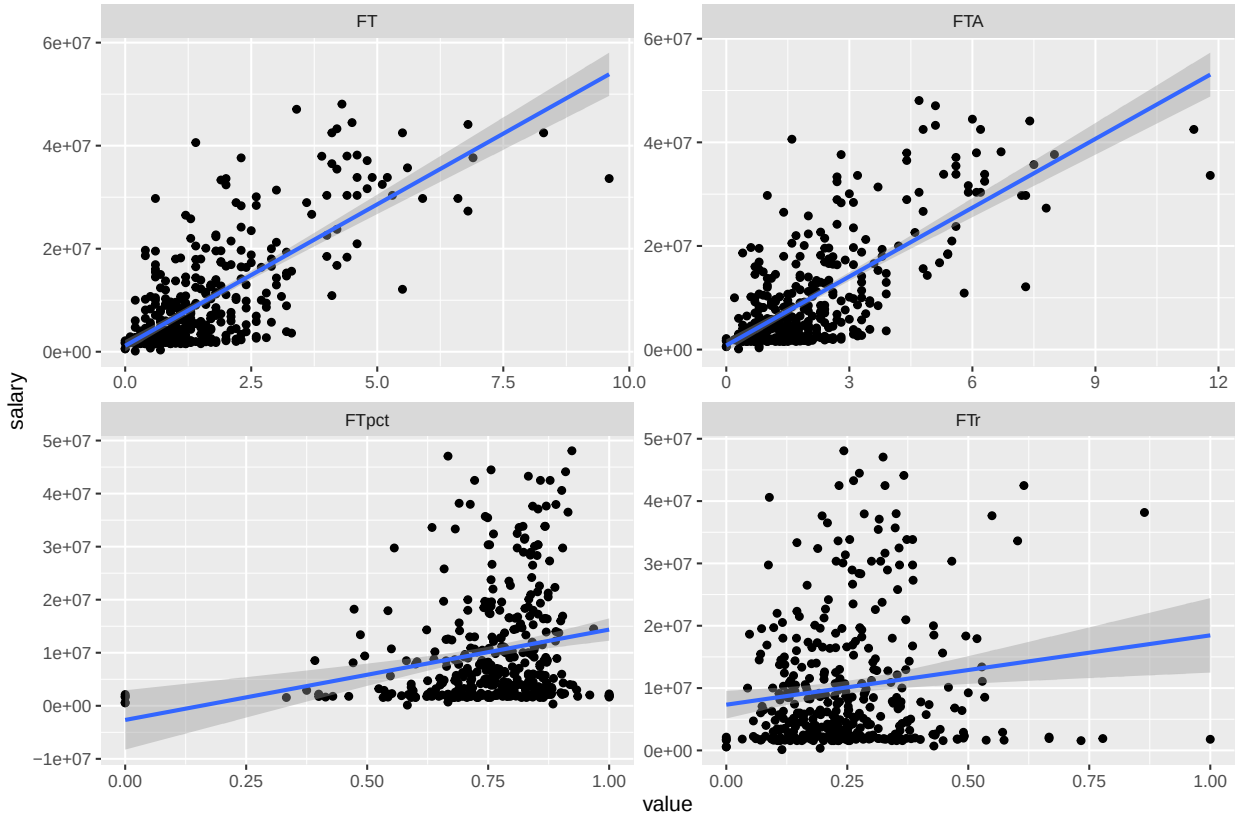
```
all_X3 <- melt(all, id.vars = "salary", measure.vars = X3_point)

ggplot(all_X3, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



All except 3 point attempt rate shows a positive correlation which make sense, as really high 3 point attempt rate shows that the players have small attempt rate and relies on little ways of scoring which are often a role player and thus paid less.

7.2.6 Free throw

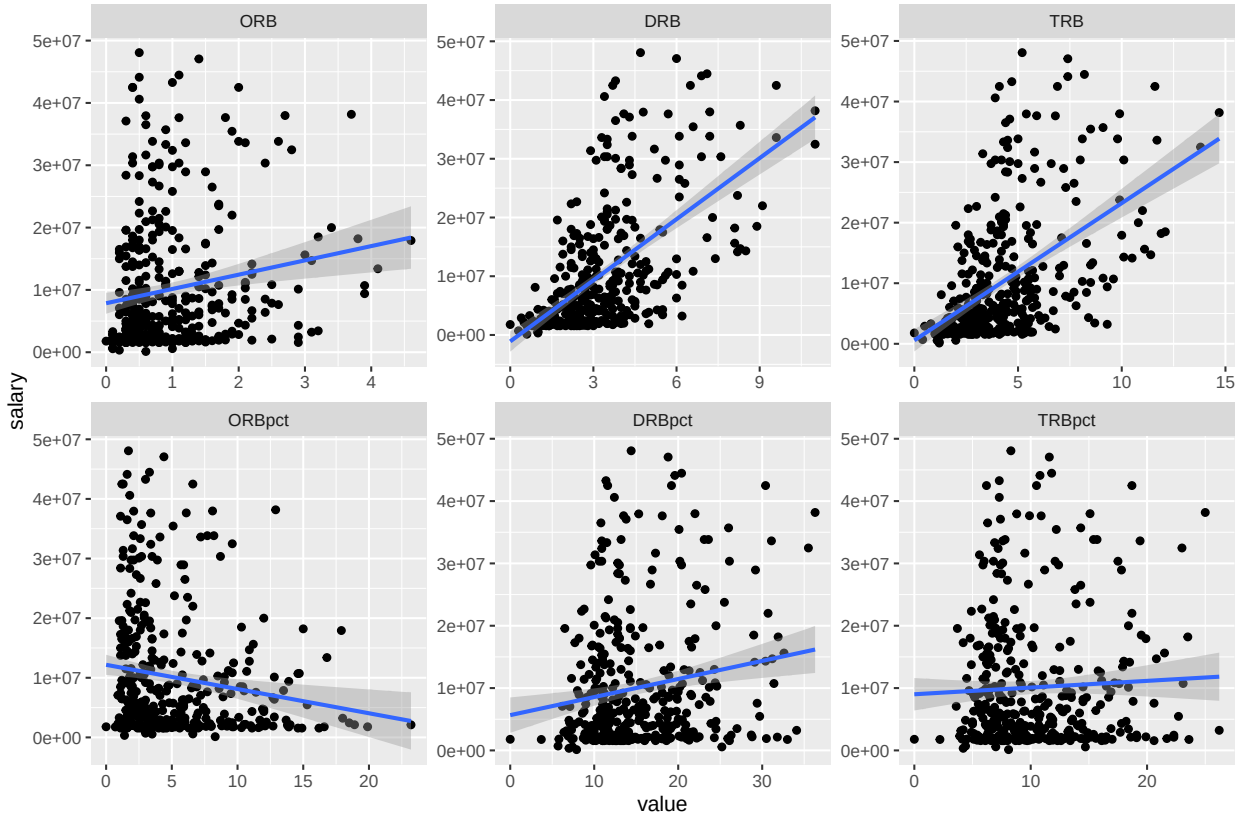
```
all_ft <- melt(all, id.vars = "salary", measure.vars = Free_throw)

ggplot(all_ft, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



All attributes have a positive correlation with salary.

7.2.7 Rebounding

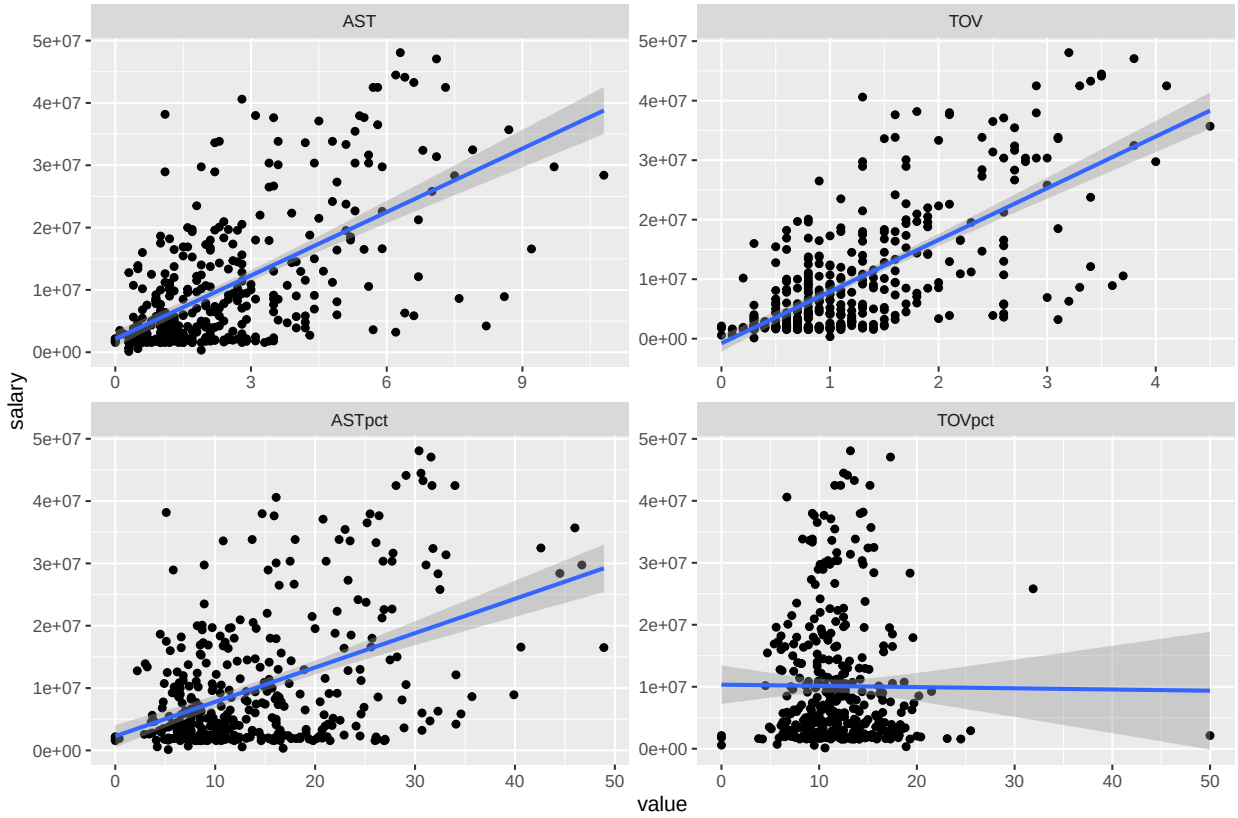
```
all_rb <- melt(all, id.vars = "salary", measure.vars = rebounding)

ggplot(all_rb, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



It is out of expectation that the offensive rebound percentage show a negative correlation. This might be explained by only role player will attempt to grab offensive rebound to conserve main player stamina.

7.2.8 Playmaking

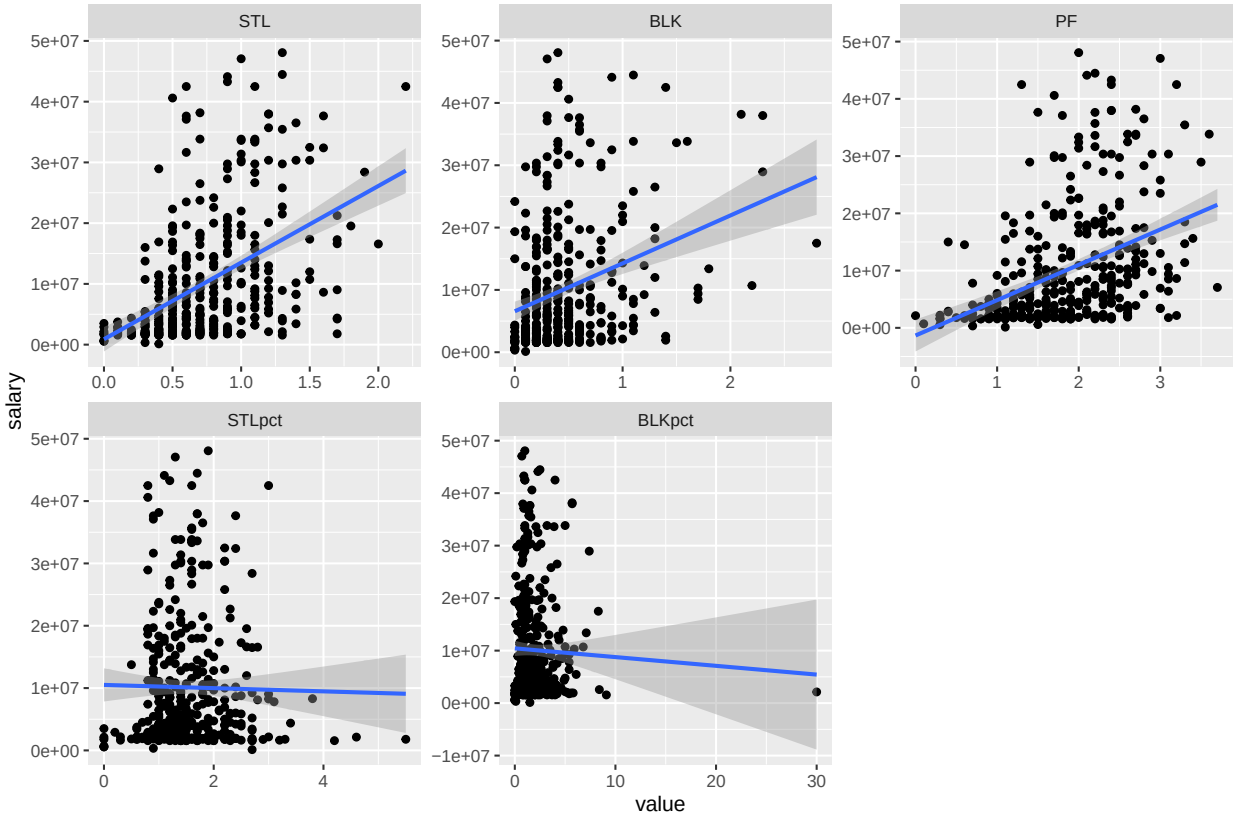
```
all_pm <- melt(all, id.vars = "salary", measure.vars = playmaking)

ggplot(all_pm, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



The plots show a strong correlation of turnover and salary. This is because the top players have more usage of the ball and thus easier to turnover the ball.

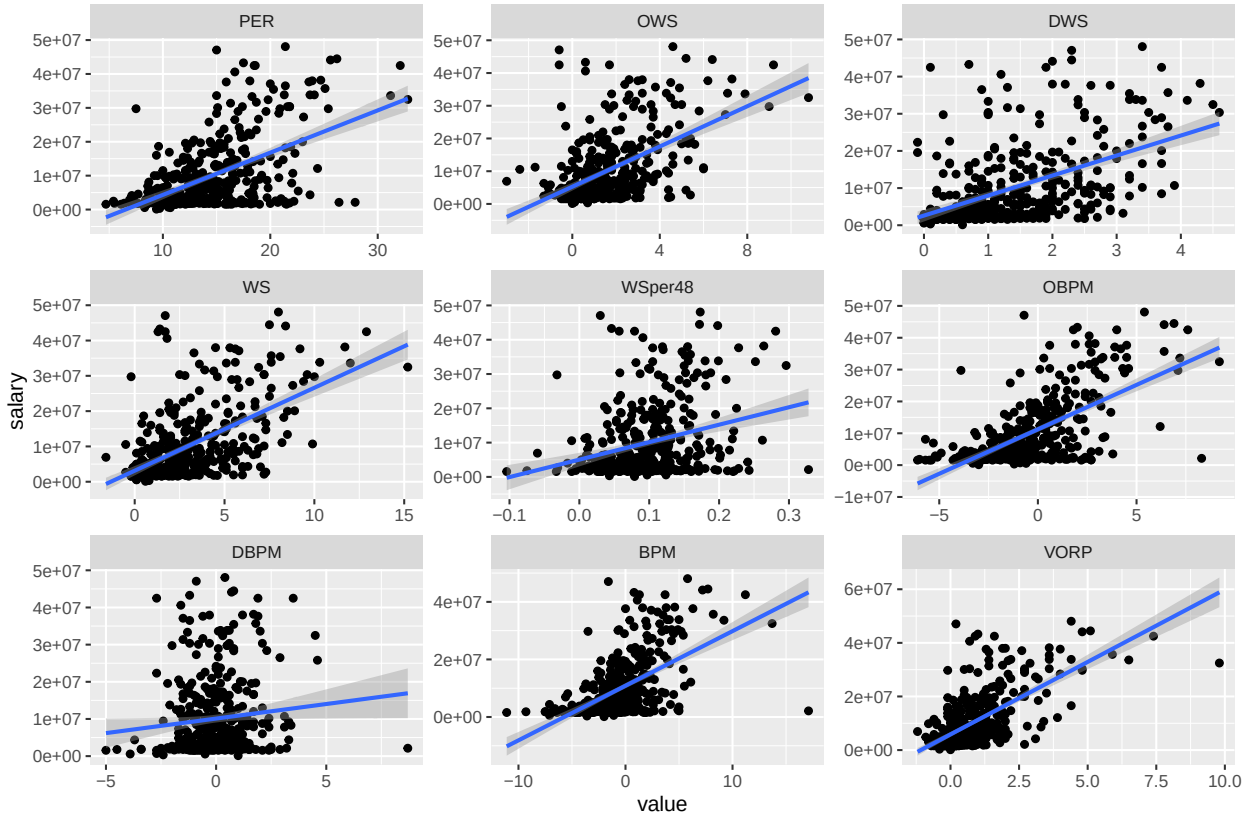
7.2.9 Defending

```
all_df <- melt(all, id.vars = "salary", measure.vars = defence)

ggplot(all_df, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



7.2.10 Overall performance

```
all_ovAd <- melt(all, id.vars = "salary", measure.vars = overall_adstats)

ggplot(all_ovAd, aes(x = value, y = salary)) + geom_point() + geom_smooth(formula = y ~
    x, method = glm) + facet_wrap(vars(variable), scales = "free")
```



8 Feature Engineering

To reduce complexity of the model, I will combine and delete some variables. A high complexity model will result in overfitting which will lead to a lower accuracy in predicting unseen data.
Before this, I will save a copy of original data for possible future operation.

```
write.csv(all[, !names(all) %in% c("X")], "dataset/cleaned_data.csv")
```

8.1 Character variables

8.1.1 Player Id and name

I will remove player id and name since it unique for each player and thus cannot use in regression, but I will save the player id and name in another variables to take reference from.

```
players <- all[, c("player_id", "name")]
all <- dplyr::select(all, !c(player_id, name))
```

8.1.2 Teams

It show little correlation and this research is mainly about the players' individual statistics. Hence, it is removed.

```
cor(as.numeric(fct_reorder(as.factor(all$team_2021), all$salary, median)), all$salary)
```

```
## [1] 0.2288759
```

```
cor(as.numeric(fct_reorder(as.factor(all$team_2022), all$salary, median)), all$salary)
```

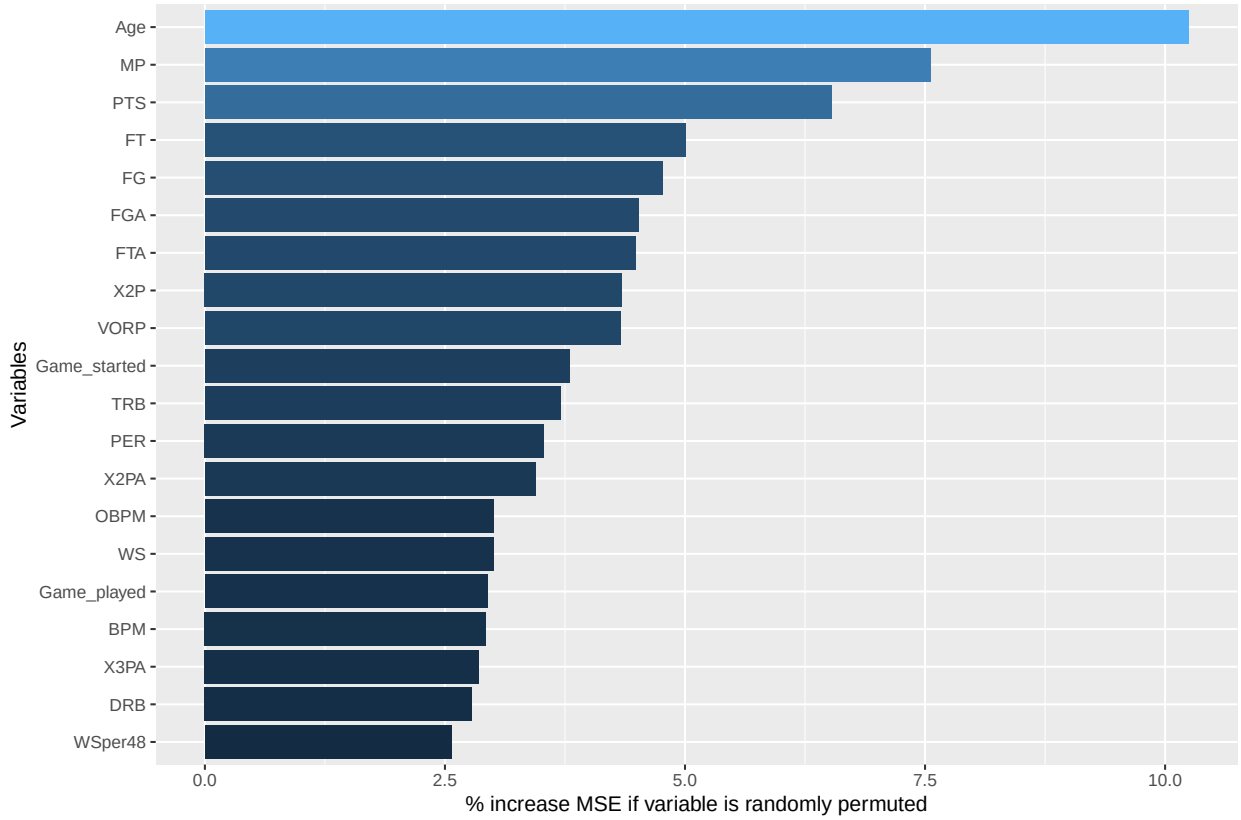
```
## [1] 0.1704062
```

There is no clear correlation between salary and team as each team will have varying salary for their star players and bench players. I will remove this variable.

```
all <- dplyr::select(all, !c(team_2021, team_2022))
```

8.2 Importance of each variable

```
quick_rf <- randomForest(salary ~ ., data = all, ntree = 100, importance = TRUE)

imp_rf <- randomForest::importance(quick_rf)
imp_df <- data.frame(Variables = row.names(imp_rf), MSE = imp_rf[, 1])
imp_df <- imp_df[order(imp_df$MSE, decreasing = TRUE), ]

ggplot(imp_df[1:20, ], aes(x = reorder(Variables, MSE), y = MSE, fill = MSE)) + geom_bar(stat = "identi
    labs(x = "Variables", y = "% increase MSE if variable is randomly permuted") +
    coord_flip() + theme(legend.position = "none")
```



8.3 Net Possession Gained

There are steal, block and offensive rebounds per game record but their correlation is not very strong. I will combine them into possession gain to make a stronger variable as these action will all result in a possession gain for the team.

```
cor(all$STL, all$salary)
```

```
## [1] 0.460643
```

```
cor(all$BLK, all$salary)
```

```
## [1] 0.2902457
```

```
cor(all$ORB, all$salary)
```

```
## [1] 0.1674434
```

Removing steal, block and offensive rebounds per game and add possession gain.

```
all <- all %>%
    mutate(possGain = STL + BLK + ORB) %>%
    dplyr::select(!c(STL, BLK, ORB))
```

8.4 Possession Lost

I will combine turn-overs and personal fouls to become possession. These variables have a low importance in the random forest. These actions will result in an possession lost.

```
imp_df$MSE[imp_df$Variables %in% c("TOV", "PF")]
```

```
## [1] 2.234827 2.210181
```

```
all <- all %>%
    mutate(possLost = TOV + PF) %>%
    dplyr::select(!c(TOV, PF))
```

8.5 field goal missed + remove field goal percentage and field goal attempt

I will remove anything about two pointer as it is just portion of field goal that is not three pointers.

```
all <- all %>%
    dplyr::select(!c(X2P, X2PA, X2Ppct))
```

I will remove free throw, field goal and 3 point percentage and attempts. I will replace them by free throw, field goal and 3 three point missed.

```
all <- all %>%
    mutate(FGM = FGA - FG, X3M = X3PA - X3P, FTM = FTA - FT) %>%
    dplyr::select(!c(FGA, FGpct, X3PA, X3Ppct, FTA, FTpct))
```

8.6 Removing Game played and add starter

I will remove game started and replace it will starter. It will be define whether that player has start for more than 50% of their game played. I will also remove game played as it is directly related to minutes played while minutes played has a stronger correlation and importance. I will also change minutes play per game to minutes played in total

```
all <- all %>%
    mutate(starter = ifelse(Game_started/Game_played >= 0.5, 1, 0), MP = MP * Game_played) %>%
    dplyr::select(!c(Game_started, Game_played))
all$starter <- as.factor(all$starter)
```

8.7 Win share

I will remove win share and win share per 48 as it is just sum of defensive and offensive win share.

```
all <- dplyr::select(all, !c(WS, WSper48))
```

8.8 Total rebound

I will remove total rebound as it is the sum of offensive and defensive rebound while defensive rebound has stronger correlation and importance.

```
all <- dplyr::select(all, !TRB)
```

8.9 Box score

I will remove Box Plus or minus as it is the sum of offensive and defensive box score.

```
cor(all[, c("OBPM", "DBPM", "BPM", "salary")])[, "salary"]
```

```
##      OBPM      DBPM       BPM    salary
## 0.6356575 0.1026086 0.5481368 1.0000000
```

```
imp_df[imp_df$Variables %in% c("OBPM", "DBPM", "BPM"), ]
```

```
##      Variables        MSE
## OBPM      OBPM 3.00731196
## BPM        BPM 2.92681805
## DBPM      DBPM 0.07547817
```

```
cor(all$OBPM + all$DBPM, all$BPM)
```

```
## [1] 0.9998765
```

```
all$BPM <- NULL
```

8.10 Free throw rate and three point rate

I will remove them as their are directly related to field goal and field goal missed.

```
all$FTr <- NULL
all$X3PAr <- NULL
```

9 Preparing data for modelling

As I am not sure about the effect of the variables on the model, I will not remove any extra variable but look at the results first

```
rm(list = ls()[!ls() %in% c("all", "players")])
```

9.1 Preprocessing predictor variables

```
numVar <- names(which(sapply(all, is.numeric)))
salary <- all$salary
player_name <- all$name
numVar <- numVar[!numVar %in% c("salary", "X", "name")]
all_num <- all[, numVar]
all_fac <- all[, !names(all) %in% c(numVar, "salary", "name", "X")]
```

There are 30 numeric predictors and 2 factor predictor.

9.1.1 Removing skewness of variables

I will use min max normalization for variable with negative value to turn all data to positive.
Variables that are highly right skewed (skewness > 0.8) are natural logged to reduce skewness.
Variables that are highly left skewed (skewness < -0.8) are squared to reduce skewness.

```
log_names <- c()
minMax <- c()
sq_names <- c()
for (i in 1:ncol(all_num)) {
    if (any(all_num[, i] <= 0)) {
        process <- preProcess(as.data.frame(all_num[, i]), method = "range")
        all_num[, i] <- predict(process, as.data.frame(all_num[, i]))
        minMax <- c(minMax, i)
    }
```

```
    if (skew(all_num[, i]) > 0.8) {
        all_num[, i] <- log(all_num[, i] + 1)
        log_names <- c(log_names, i)
    } else if (skew(all_num[, i]) < -0.8) {
        all_num[, i] <- all_num[, i]^2
        sq_names <- c(sq_names, i)
    }
}
log_names <- names(all_num)[log_names]
minMax_names <- names(all_num)[minMax]
sq_names <- names(all_num)[sq_names]
```

These column have been log + 1 due to its skewness.
(The + 1 is to prevent logging 0 which result in NA)

9.1.2 Normalizing Data

The remaining data is normalized by feature scaling and mean normalization.

```
all_num[!names(all_num) %in% log_names] <- as.data.frame(scale(all_num[!names(all_num) %in%
    log_names]))
```

9.1.3 One hot encoding for categorical variables

I will convert all the remaining variables to numeric (it is required by many machine learning algorithm).

```
all_fac <- as.data.frame(model.matrix(~. - 1, as.data.frame(all_fac)))
```

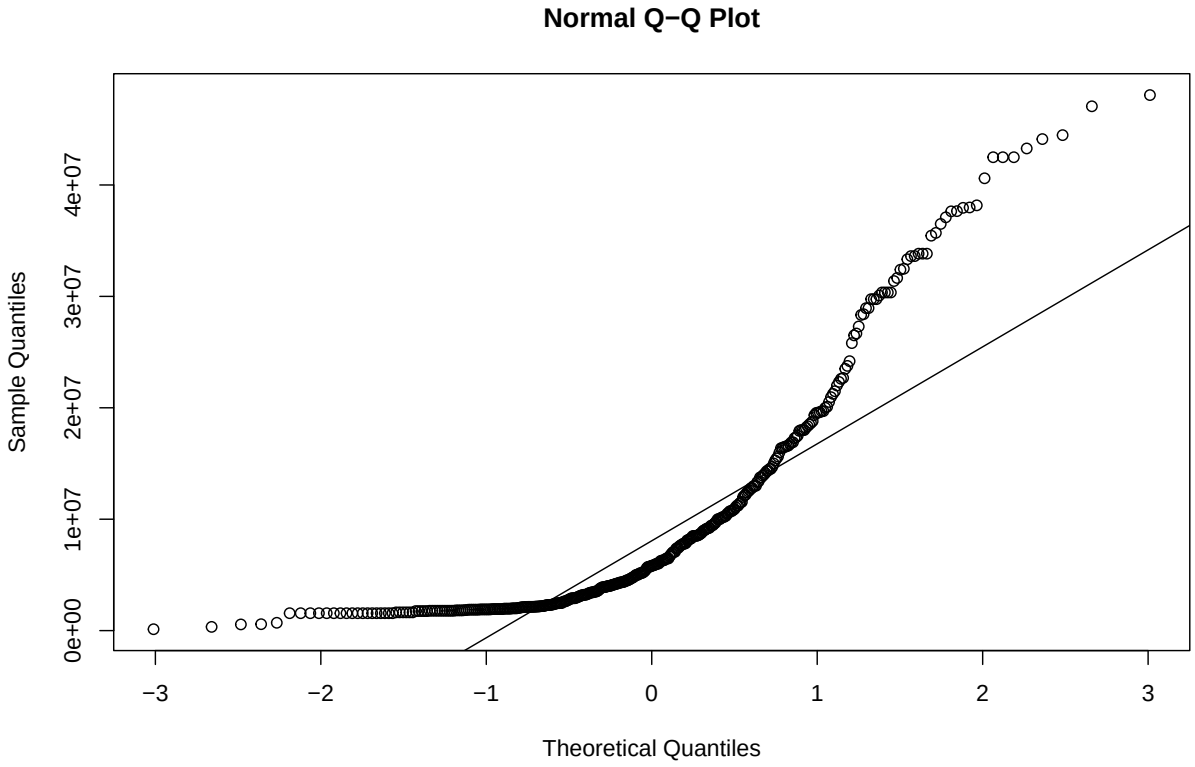
9.2 Dealing with the skewness of response variable

```
skew(salary)
```

```
## [1] 1.564389
```

The skewness of salary is too high which will be harder to fit a model

```
qqnorm(salary)
qqline(salary)
```



This is a QQ plot where the x axis are the theoretical quantiles while y axis are the sample quantile. The diagonal line is where sample theoretical quantiles perfectly aligned. The placementment of the point indicate the skewness of the sample.

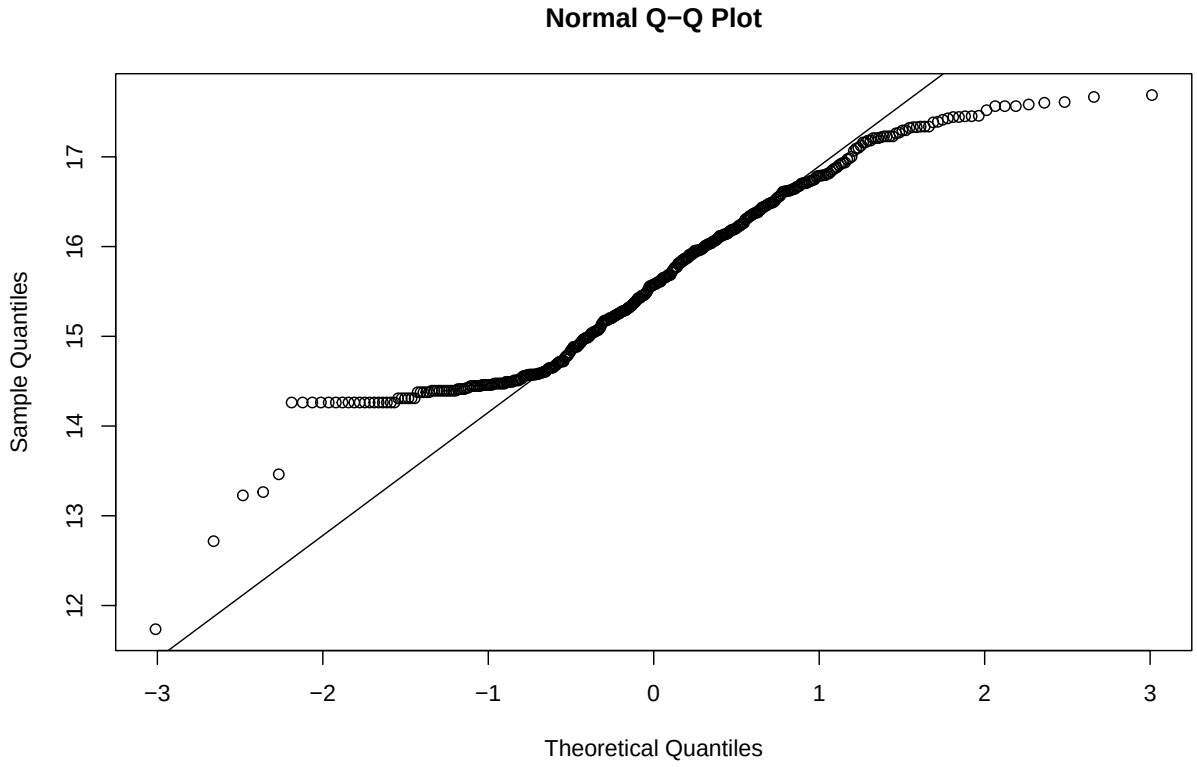
Salary is too right skewed and not normally distributed.

```
salary <- log(salary)
skew(salary)
```

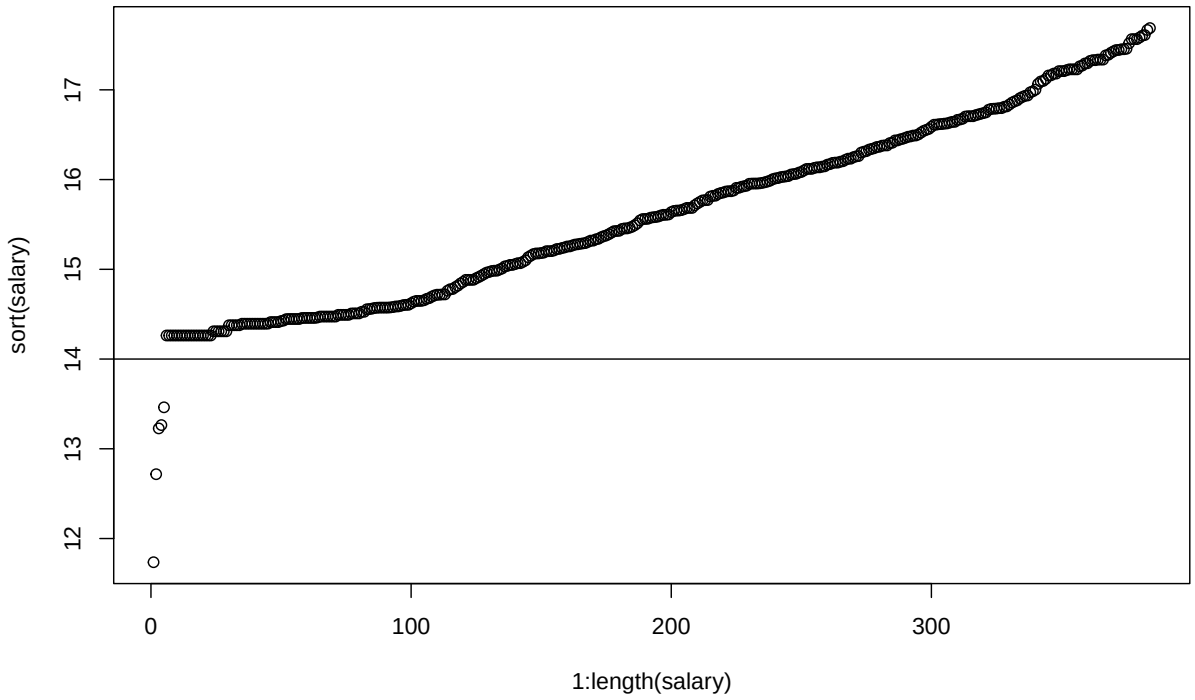
```
## [1] 0.04653786
```

After logging, the points lie more towards the line (less skewed).

```
qqnorm(salary)
qqline(salary)
```




```
plot(x = 1:length(salary), y = sort(salary))
abline(h = 14)
```



9.2.1 Combining data

I treat log(salary) < 14 as outlier and remove them from the data.

```
all_noNorm <- cbind(X = 1:nrow(all), salary = salary, all[, numVar], all_fac)
alldata <- cbind(X = 1:nrow(all), salary = salary, all_num, all_fac)
row.names(alldata) <- players$name
name <- players$name[alldata$salary > 14]
ind <- which(alldata$salary > 14)
alldata <- alldata[alldata$salary > 14, ]
all_noNorm <- all_noNorm[alldata$salary > 14, ]
```

9.3 Spliting training and testing set

I will use train-test split and 10 fold cross validation in the training set,

```
inTrain <- sample(1:2, size = nrow(alldata), prob = c(0.8, 0.2), replace = TRUE)
train <- alldata[inTrain == 1, ]
test <- alldata[inTrain == 2, ]
train_noNorm <- all_noNorm[inTrain == 1, ]
test_noNorm <- all_noNorm[inTrain == 2, ]
```

```
write.csv(alldata[, -c(1, 2)], "dataset/normalized_data.csv")
```

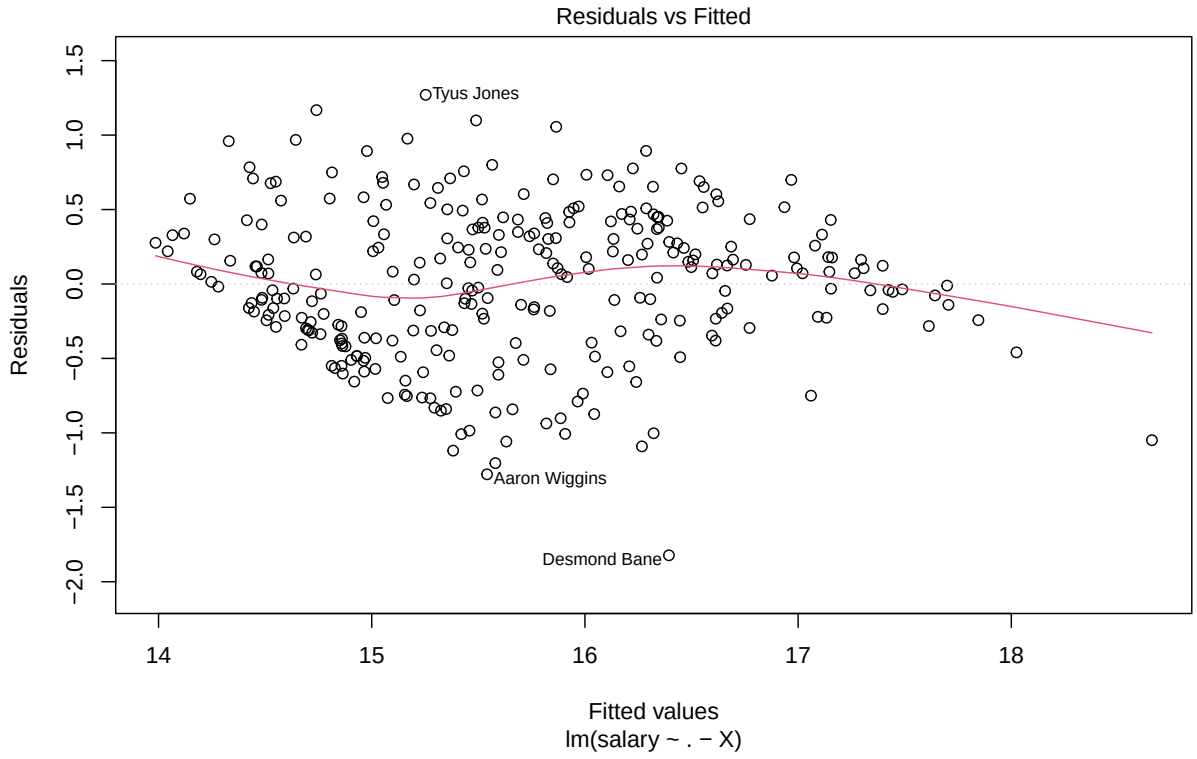
10 Modelling

In the modelling I will use the cross validation root mean squared error to determine which model to choose.

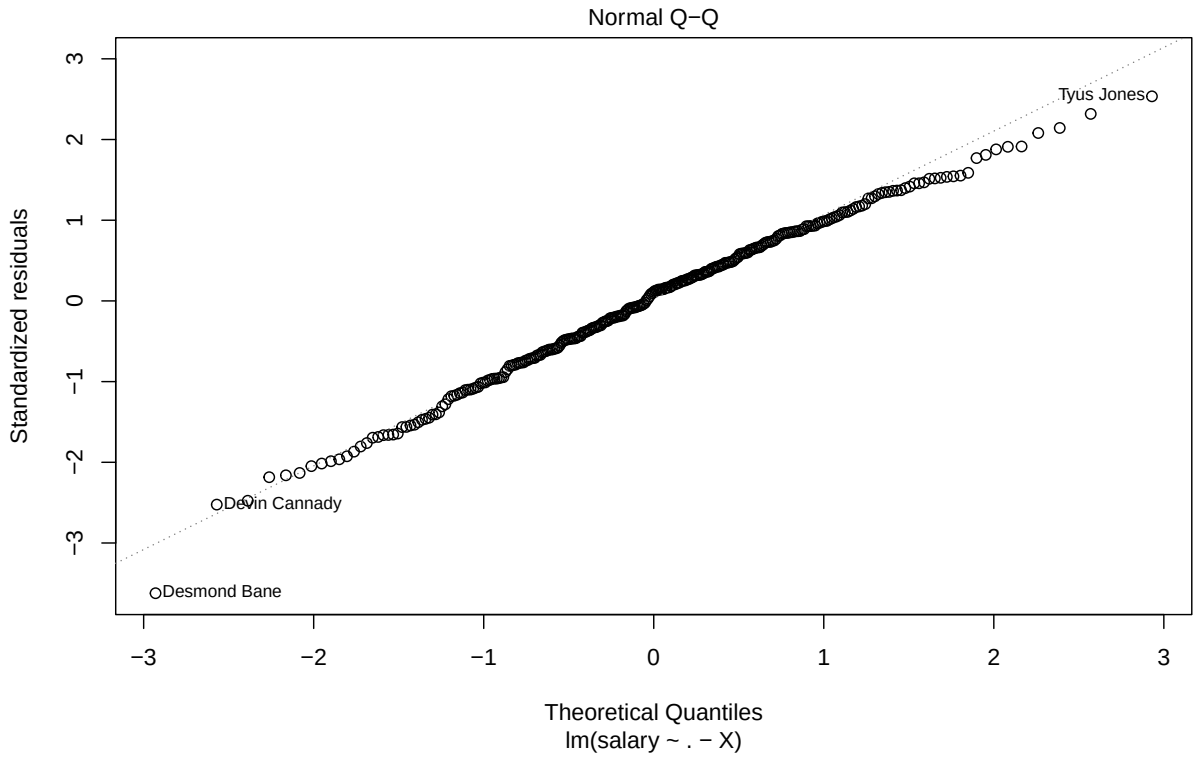
10.1 Linear regression

```
mod_lm <- lm(salary ~ . - X, data = train)
```

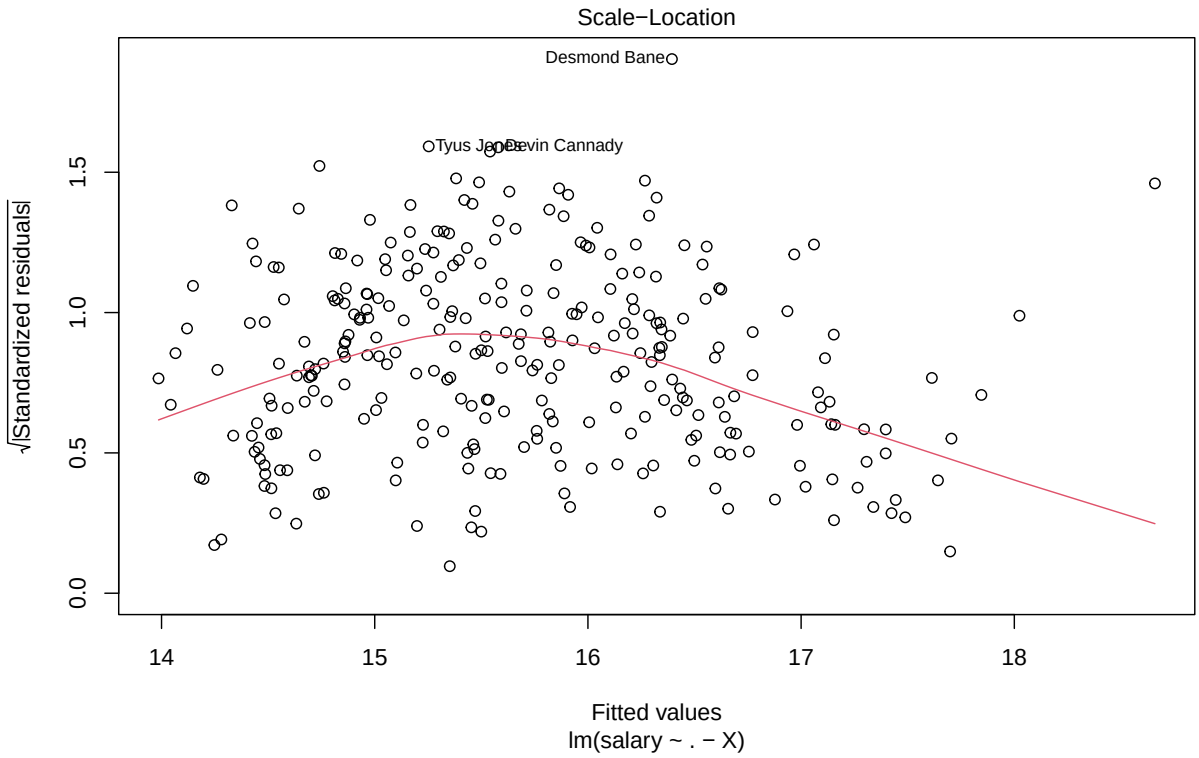
```
plot(mod_lm, which = 1)
```



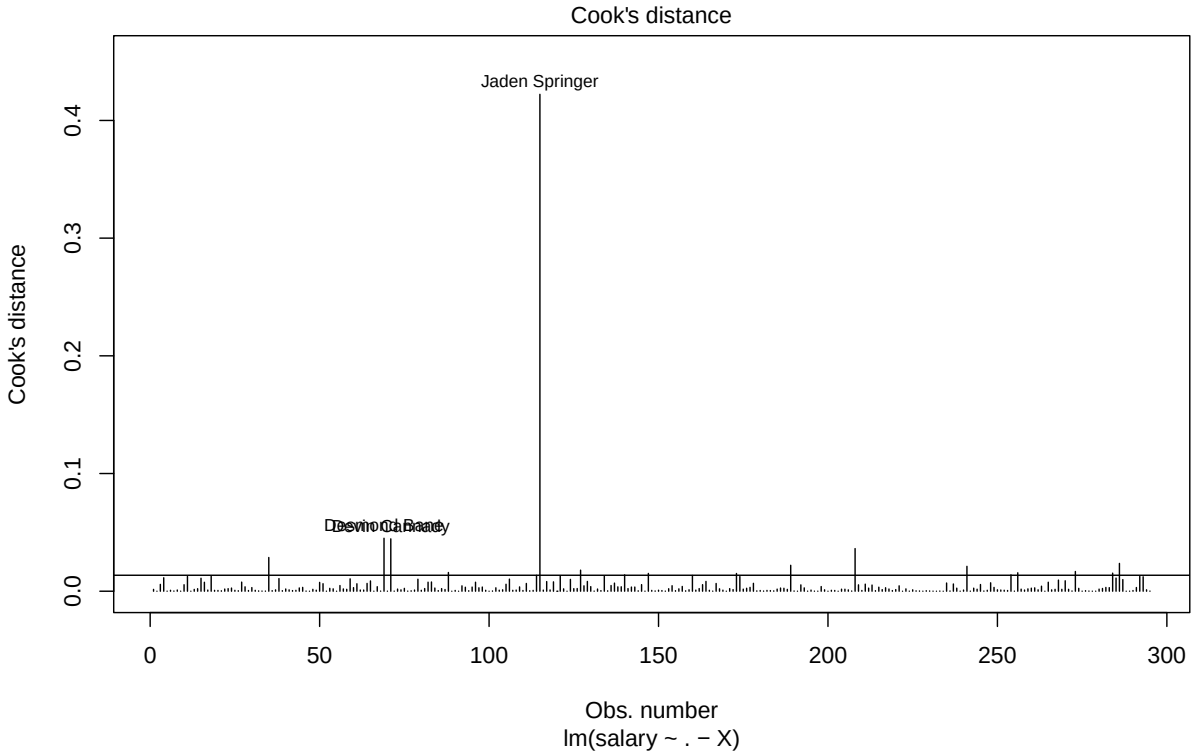

```
plot(mod_lm, which = 2)
```




```
plot(mod_lm, which = 3)
```




```
plot(mod_lm, which = 4)
abline(h = 4/nrow(train))
```



The cook's distance measures the residual and leverage of that point. It represent the degree of influence on the regression model. We will take a look at the player with high cook's distance.

```
sort(cooks.distance(mod_lm), decreasing = TRUE)[1:3]
```

```
## Jaden Springer   Desmond Bane  Devin Cannady
##     0.42218052     0.04500047     0.04449516
```

```
players[players$name %in% names(which(cooks.distance(mod_lm) > 4/nrow(train))), ]
```

```
##     player_id              name
## 43  edwarca01     Carsen Edwards
## 91   banede01       Desmond Bane
## 93  cannade01      Devin Cannady
## 114 kaminfr01     Frank Kaminsky
## 150 sprinja01     Jaden Springer
## 166 culveja01     Jarrett Culver
## 180 harrijo01         Joe Harris
## 189 poolejo01       Jordan Poole
## 220  loveke01         Kevin Love
## 240 jamesle01       LeBron James
## 264 portemi01 Michael Porter Jr.
## 308 tuckera01      Rayjon Tucker
## 322 westbru01  Russell Westbrook
## 324   beysa01         Saddiq Bey
## 344 taylote01       Terry Taylor
```

```
## 365 halibty01  Tyrese Haliburton
## 367 jonesty01         Tyus Jones
```

```
summary(mod_lm)
```

```
##
## Call:
## lm(formula = salary ~ . - X, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.82226 -0.31639  0.05619  0.35760  1.27033
##
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  16.483750   1.394815  11.818  < 2e-16 ***
## Age           0.307495   0.037006   8.309 5.45e-15 ***
## MP           -0.328533   0.100108  -3.282  0.00117 **
## FG            1.411302   1.044245   1.352  0.17771
## X3P          -0.162821   0.181761  -0.896  0.37119
## FT           -0.317065   1.277467  -0.248  0.80418
## DRB           2.461169   1.768175   1.392  0.16514
## AST           3.371172   1.418029   2.377  0.01816 *
## PTS          -1.402118   0.895750  -1.565  0.11873
## PER           0.373310   0.677644   0.551  0.58218
## TSpct        -2.156322   2.921740  -0.738  0.46117
## ORBpct        2.857177   3.851514   0.742  0.45886
## DRBpct        6.086728   6.086241   1.000  0.31821
## TRBpct      -11.152085   9.186810  -1.214  0.22588
## ASTpct       -3.406084   1.245786  -2.734  0.00669 **
## STLpct       -0.259637   0.752566  -0.345  0.73037
## BLKpct        0.398177   1.442842   0.276  0.78279
## TOVpct        0.989632   1.092366   0.906  0.36580
## USGpct        0.034019   0.155156   0.219  0.82662
## OWS           2.474102   1.415557   1.748  0.08168 .
## DWS           0.230342   0.102941   2.238  0.02610 *
## OBPM          0.137806   0.139308   0.989  0.32348
## DBPM         -0.044575   0.090066  -0.495  0.62108
## VORP         -1.675764   1.884377  -0.889  0.37467
## Weight       -0.040123   0.059448  -0.675  0.50032
## Height        0.134180   0.071508   1.876  0.06172 .
## possGain      0.411751   1.275648   0.323  0.74712
## possLost     -0.001725   0.107122  -0.016  0.98717
## FGM           0.181024   0.270827   0.668  0.50447
## X3M           0.333070   0.161365   2.064  0.04001 *
## FTM           1.307387   0.587963   2.224  0.02704 *
## PositionC    -0.220577   0.196897  -1.120  0.26364
## PositionPF   -0.135904   0.143901  -0.944  0.34583
## PositionPG   -0.207653   0.124694  -1.665  0.09706 .
## PositionSF   -0.214658   0.115568  -1.857  0.06439 .
## PositionSG          NA         NA      NA       NA
## starter1      0.206174   0.108258   1.904  0.05796 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.5332 on 259 degrees of freedom
## Multiple R-squared:  0.7608, Adjusted R-squared:  0.7285
## F-statistic: 23.54 on 35 and 259 DF,  p-value: < 2.2e-16
```

10.1.1 Glossary of the summary

- Residual
  - The difference between each predict value and actual value
  - It ranges from -1.8222 to 1.2703
- Coefficients
  - Estimate
    - The estimate value of the weight
  - Std. Error
    - The standard error of the weight (Similar to standard deviation)
  - t value
    - The number of standard error of the estimate weight from 0
  - p value
    - The probability of getting the estimate weight if the actual weight is 0
- Residual standard error
  - The standard deviation of residuals
- R-squared
  - The estimate proportion of the responsive variable the model has accounted for
- F-statistics
  - p value: the probability of getting these weight provided that the null hypothesis is all weight is zero.

The r squared of the model is 0.7608406

```
rmse(train$salary, mod_lm$fitted.values)
```

```
## [1] 0.4996236
```

```
rmse(test$salary, predict(object = mod_lm, newdata = test))
```

```
## [1] 0.5427067
```

The train root mean squared error is 1.2277041 while the testing (out of bag) root mean squared error is 0.4640182.

10.2 Lasso regression

```
train_control <- trainControl(method = "cv", number = 10)
param_grid <- expand.grid(alpha = 1, lambda = seq(0.001, 0.1, by = 5e-04))

mod_lasso <- train(salary ~ . - X, data = train, method = "glmnet", trControl = train_control,
    tuneGrid = param_grid)
```

Final model hyperparameter

```
mod_lasso$bestTune
```

```
##    alpha lambda
## 30     1 0.0155
```

lambda is the regularization penalt.

The coefficient of the final model:

```
coef(mod_lasso$finalModel, mod_lasso$bestTune$lambda)
```

```
## 37 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept) 14.627020415
## Age          0.281093896
## MP           .
## FG           0.216383524
## X3P          .
## FT           .
## DRB          0.683145734
## AST          .
## PTS          .
## PER          .
## TSpct        .
## ORBpct      -0.065845405
## DRBpct       .
## TRBpct       .
## ASTpct      -0.117123868
## STLpct      -0.200340328
## BLKpct       0.786544105
## TOVpct       .
## USGpct       .
## OWS          0.004649154
## DWS          0.003991165
## OBPM         .
## DBPM         .
## VORP         1.322332179
## Weight       .
## Height       0.037242954
## possGain     0.315031195
## possLost     .
## FGM          0.254924634
## X3M          0.108150531
## FTM          0.672180412
```

```
## PositionC   -0.017661106
## PositionPF   0.034136845
## PositionPG   .
## PositionSF   .
## PositionSG   0.090828654
## starter1     0.332252523
```

```
mod_lasso$results$Rsquared[mod_lasso$results$lambda == mod_lasso$bestTune$lambda]
```

```
## [1] 0.707579
```

The r squared of the best tuned model is 0.707579.

```
rmse(predict(mod_lasso), train$salary)
```

```
## [1] 0.5245659
```

```
rmse(predict(mod_lasso, test), test$salary)
```

```
## [1] 0.5291957
```

The train root mean squared error is 1.2025854 while the testing (out of bag) root mean squared error is 0.4811225.

10.3 Elastic Net Regression

```
mod_elaNet <- train(salary ~ . - X, data = train, method = "glmnet", tuneLength = 10,
    trControl = trainControl(method = "cv", number = 10))
```

```
mod_elaNet$results$Rsquared[which.min(mod_elaNet$results$RMSE)]
```

```
## [1] 0.7177684
```

The R squared is 0.7177684

```
rmse(predict(mod_elaNet), train$salary)
```

```
## [1] 0.5133823
```

```
rmse(predict(mod_elaNet, test), test$salary)
```

```
## [1] 0.5344562
```

The train root mean squared error is 1.2143706 while the testing (out of bag) root mean squared error is 0.4679066.

10.4 Step AIC

```
mod_stepAIC <- stepAIC(mod_lm, scope = list(upper = ~., lower = ~1), trace = FALSE,
    direction = "both")
```

```
RSS <- sum((mod_stepAIC$fitted.values - train$salary)^2)
TSS <- sum((train$salary - mean(train$salary))^2)
RSQ <- 1 - RSS/TSS
RSQ
```

```
## [1] 0.7535305
```

The R-squared is 0.7535305 and the AIC value is -364.5193241

```
mod_stepAIC$coefficients
```

```
## (Intercept)         Age          MP          FG         DRB         AST
##  15.6844715   0.2904833  -0.2329144   1.4273136   2.1189764   2.6561159
##         PTS      TRBpct      ASTpct      BLKpct         OWS         DWS
##  -1.2030373  -1.9317407  -2.3767727   1.0820445   1.4693787   0.1176143
##      Height         FGM         X3M         FTM    starter1  PositionSG
##   0.1077515   0.2414456   0.1694718   1.2876026   0.1944891   0.1708528
```

```
rmse(mod_stepAIC$fitted.values, train$salary)
```

```
## [1] 0.5072018
```

```
rmse(predict(mod_stepAIC, test), test$salary)
```

```
## [1] 0.5224379
```

The train root mean squared error is 1.2212577 while the testing (out of bag) root mean squared error is 0.4618056.

10.5 Decision Tree

```
mod_dt <- train(salary ~ ., data = train_noNorm[, -1], method = "rpart", trControl = trainControl("cv",
    number = 10), tuneLength = 20)
```

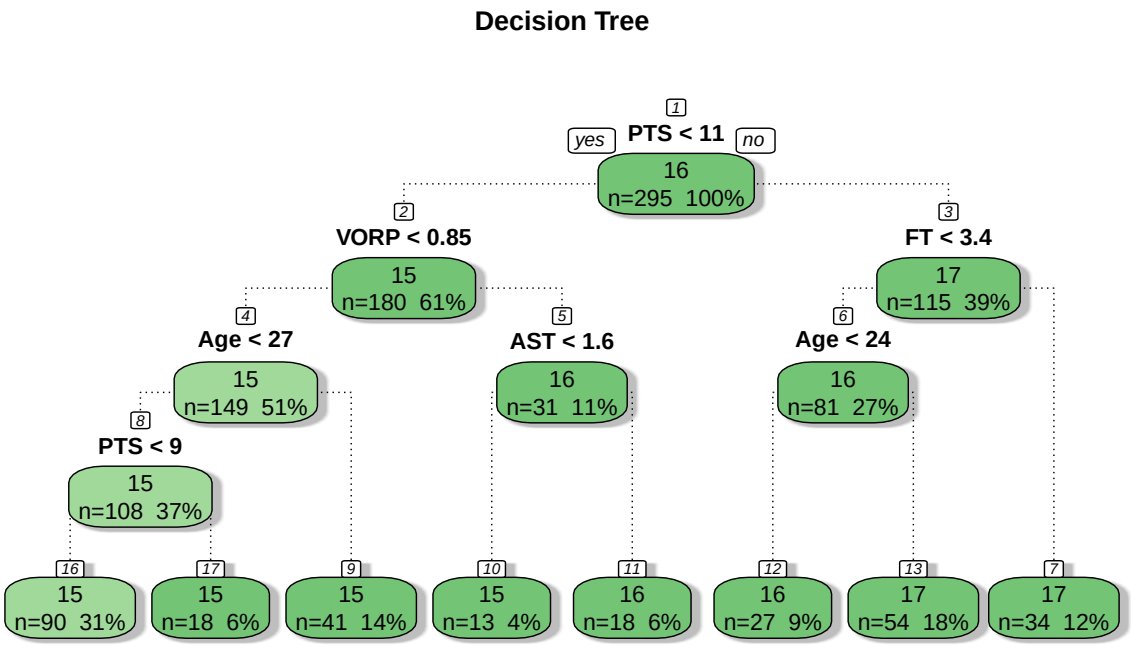
Final model hyperparameter

```
mod_dt$bestTune
```

```
##            cp
## 13 0.01447693
```

cp indicate the complexity of the tree.

```
fancyRpartPlot(mod_dt$finalModel, main = "Decision Tree", type = 1)
```




```
mod_dt$results$Rsquared[as.numeric(row.names(mod_dt$bestTune))]
```

```
## [1] 0.6438431
```

The R-squared of the model is 0.6438431

```
rmse(train_noNorm$salary, predict(mod_dt))
```

```
## [1] 0.509649
```

```
pred <- predict(mod_dt, test_noNorm)
rmse(test_noNorm$salary, pred)
```

```
## [1] 0.5939312
```

The train root mean squared error is 1.2205111 while the testing (out of bag) root mean squared error is 1.3594825.

10.6 Random Forest

```
mod_rft <- train(salary ~ ., train[, -c(1)], method = "rf", trControl = trainControl(method = "cv",
    number = 10))
```

Final model hyperparameter:

```
mod_rft$bestTune
```

```
##   mtry
## 3   36
```

mtry is the number of sample in each resampling.

```
rsq_rft <- mod_rft$results$Rsquared[as.numeric(row.names(mod_rft$bestTune))]
```

The R-squared is 0.7034341.

```
rmse(predict(mod_rft$finalModel, train), train$salary)
```

```
## [1] 0.2264029
```

```
rmse(predict(mod_rft$finalModel, test), test$salary)
```

```
## [1] 0.4720607
```

The train root mean squared error is 0.3016118 while the testing (out of bag) root mean squared error is 0.286766.

10.7 Neural Network

For the neural network, I have tested with different number of layers and node and this is the final mode hyperparameter with 100 node in first hidden layer, 75 nodes on second hidden layer and 50 node in third hidden layer. (The experimental record can be found in neural_network_train_record.md)

```
tunegrid_neural <- c(100, 75, 50)
tunegrid_neural <- as.data.frame(t(matrix(tunegrid_neural, nrow = 3)))
colnames(tunegrid_neural) <- c("layer1", "layer2", "layer3")
mod_neur <- train(salary ~ . - X, data = train, method = "neuralnet", tuneGrid = tunegrid_neural,
    trControl = trainControl(method = "cv", number = 10, verboseIter = TRUE), linear.output = TRUE)
mod_neur$results$RMSE
```

```
mod_neur$results$Rsquared
```

```
## [1] 0.6624476
```

The R-squared is 0.6624476.

```
mod_neur$results$RMSE[as.numeric(row.names(mod_neur$bestTune))]
```

```
## [1] 0.6170212
```

```
rmse(test$salary, predict(mod_neur$finalModel, test))
```

```
## [1] 0.6328699
```

The train root mean squared error is 0.6170212 while the testing (out of bag) root mean squared error is 0.3725195.

11 Final Model (random forest)

Random forest model will be used since it has the lowest training RMSE.

11.1 Hyperparameters

11.1.1 Mtry

```
mtry - Number of variables randomly sampled as candidates at each split
```

```
mod_rft$finalModel$mtry
```

```
## [1] 36
```

The number of sample per split is 36.

11.1.2 ntree

```
ntree - number of decision in the random forest
```

```
mod_rft$finalModel$ntree
```

```
## [1] 500
```

The number of decision in the random forest is 500.

11.2 Results

11.2.1 Root Mean Squared Error

Training error:

```
rmse(predict(mod_rft), train$salary)
```

```
## [1] 0.2264029
```

```
absErr <- mean(abs(exp(predict(mod_rft)) - exp(train$salary)))
absErr
```

```
## [1] 1587135
```

The root mean squared error is 1.242339.
The average difference in predicted variable and actual value in the training set is 1.6 million USD.

Testing error (out-bag-error): This is the expected error for unseen data.

```
rmse(predict(mod_rft, test), test$salary)
```

```
## [1] 0.4720607
```

```
absErr <- mean(abs(exp(predict(mod_rft, test)) - exp(test$salary)))
absErr
```

```
## [1] 3034959
```

The out-of-bag root mean squared error is 0.286766.
The average difference in predicted variable and actual value in the training set is 3 million USD. This mean for unseen data the expected error for the salary is plus or minus 3 million USD.

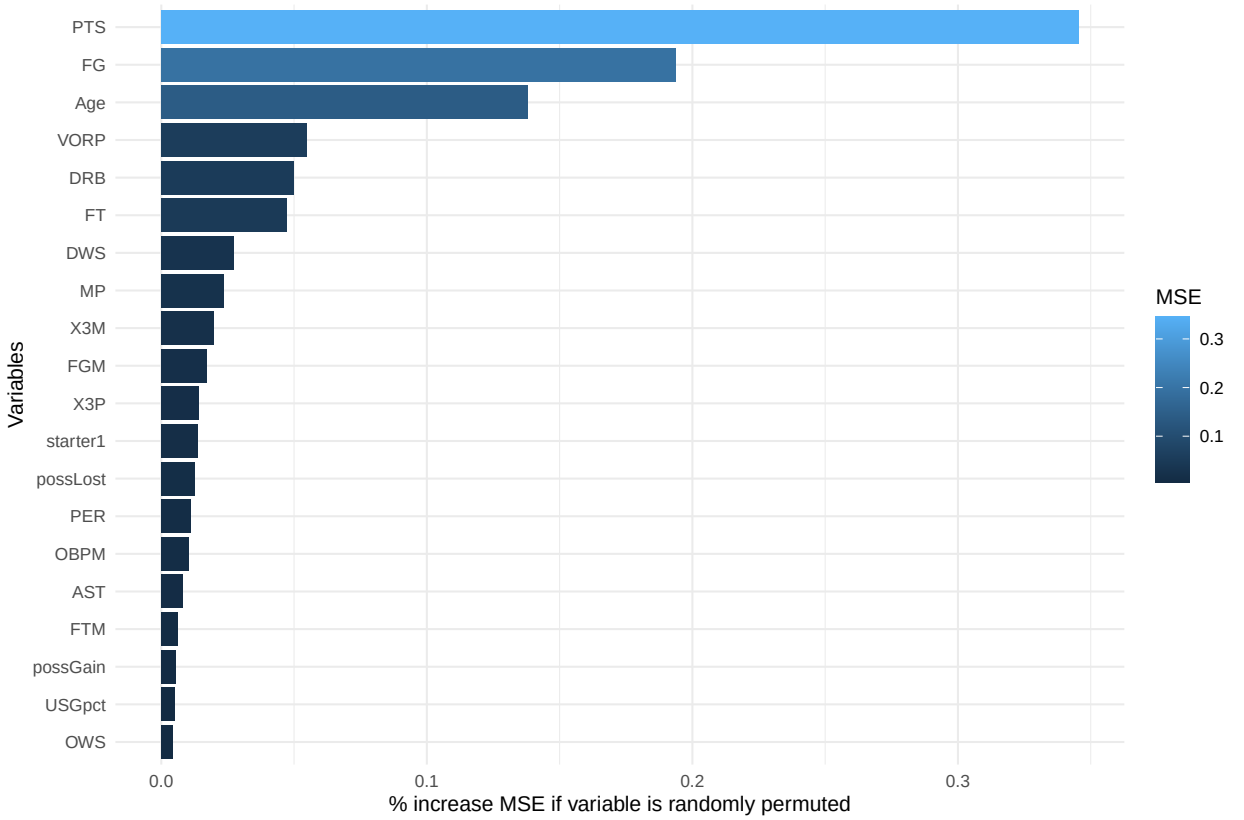
11.2.2 R-squared

```
RSQ <- mod_rft$results$Rsquared[as.numeric(row.names(mod_rft$bestTune))]
```

The R-squared of the final model is 0.7034341, which the model has accounted for 70.3% of variation of the salary.

11.3 Variable importance analysis

```
imp_rf <- mod_rft$finalModel$importance
imp_df <- data.frame(Variables = row.names(imp_rf), MSE = imp_rf[, 1])
imp_df <- imp_df[order(imp_df$MSE, decreasing = TRUE), ]

g1 <- ggplot(imp_df[1:20, ], aes(x = reorder(Variables, MSE), y = MSE, fill = MSE)) +
    geom_bar(stat = "identity") + labs(x = "Variables", y = "% increase MSE if variable is randomly per
    coord_flip() + theme(legend.position = "none") + theme_minimal()
g1
```



The length of the bar the percentage increase in MSE is randomly permuted. The higher the value the more important the variable is. From the graph, points per game, field goal and age are the top three most important in the model.

12 Interpretation

12.1 Pinciple Component Analysis

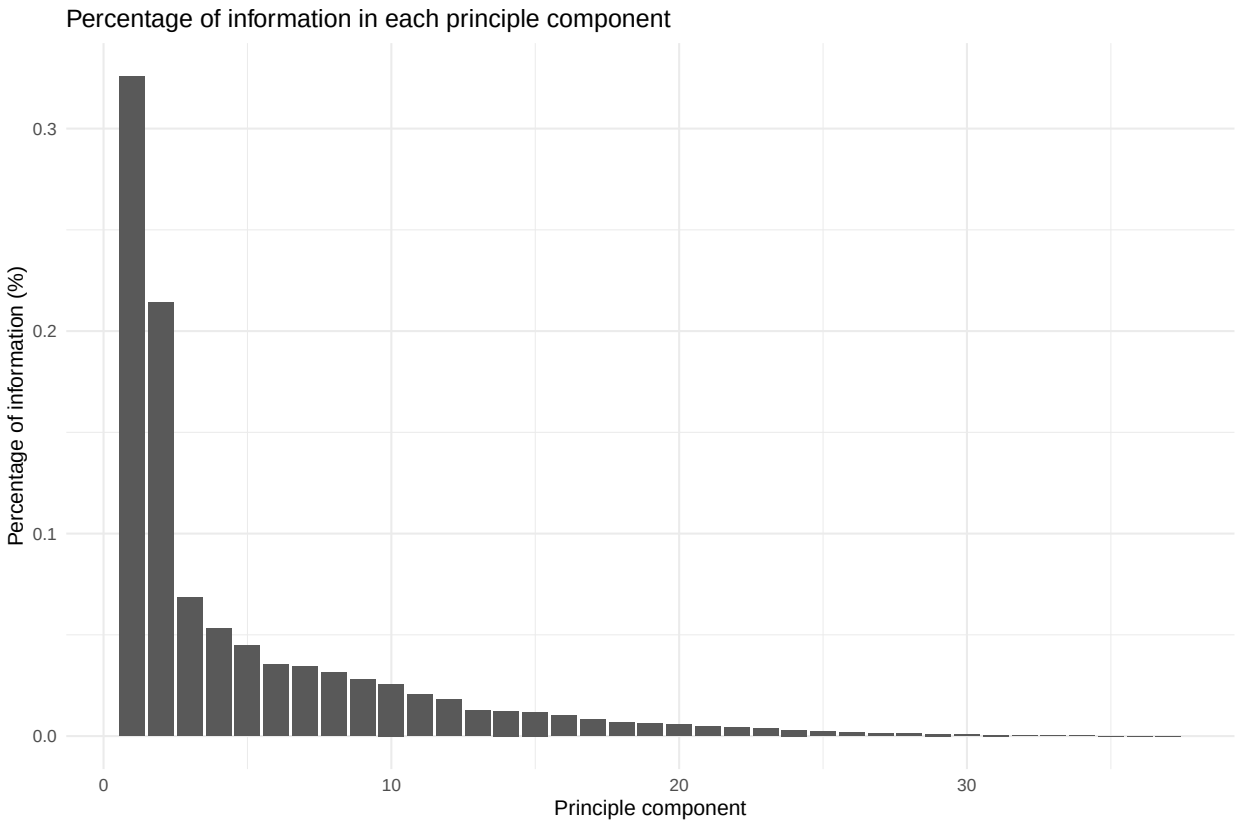
I will use principle component analysis to investigate the importance and correlation between predictive variables.

```
all_pca <- alldata
all_pca$X <- NULL
all_pca <- as.matrix(all_pca)
rownames(all_pca) <- row.names(alldata)

pca <- prcomp(all_pca, scale = TRUE)

pca.var <- pca$sdev^2
pca.var.per <- round(pca.var/sum(pca.var), 5)
```

12.1.1 Component analysis

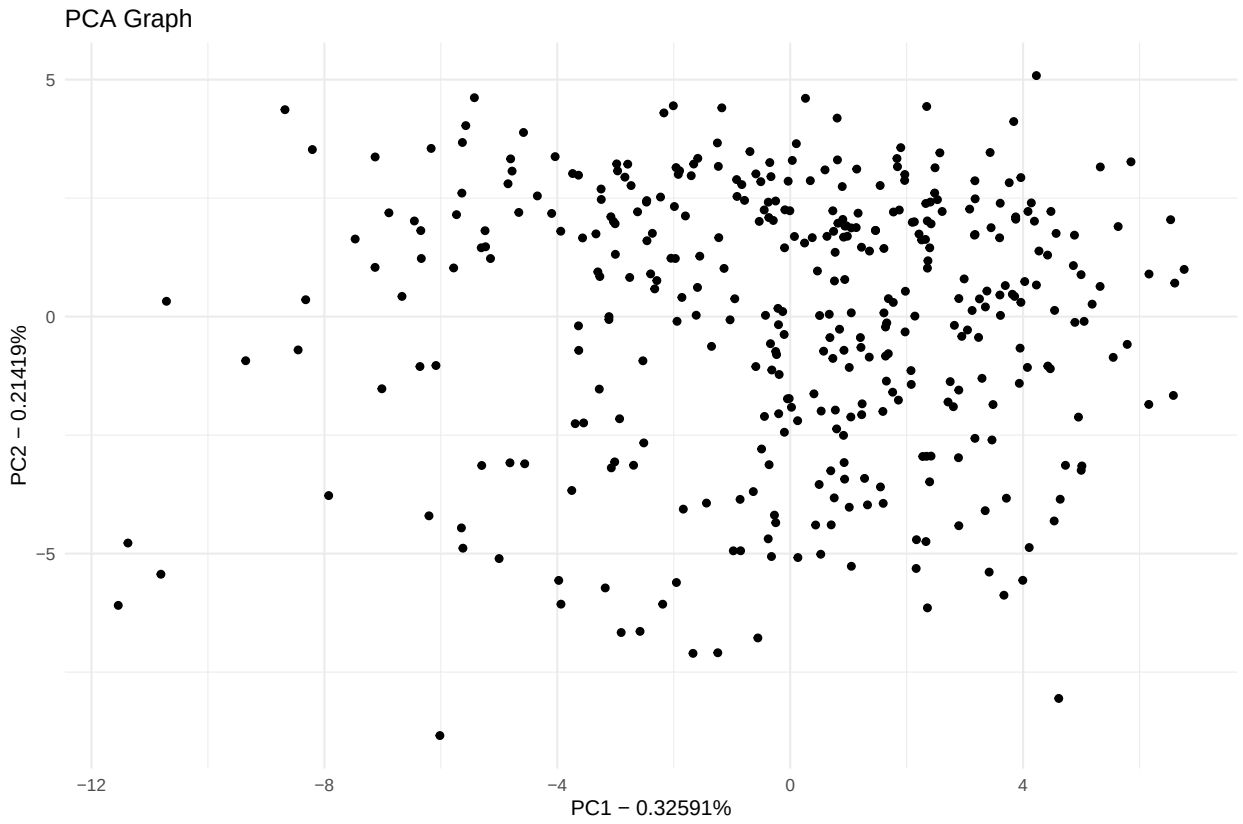
Percentage of information contained in the each principle component:

```
ggplot(data.frame(value = pca.var.per, index = 1:length(pca.var.per)), aes(x = index,
    y = pca.var.per)) + geom_bar(stat = "identity") + theme_minimal() + labs(title = "Percentage of inf
    x = "Principle component", y = "Percentage of information (%)")
```

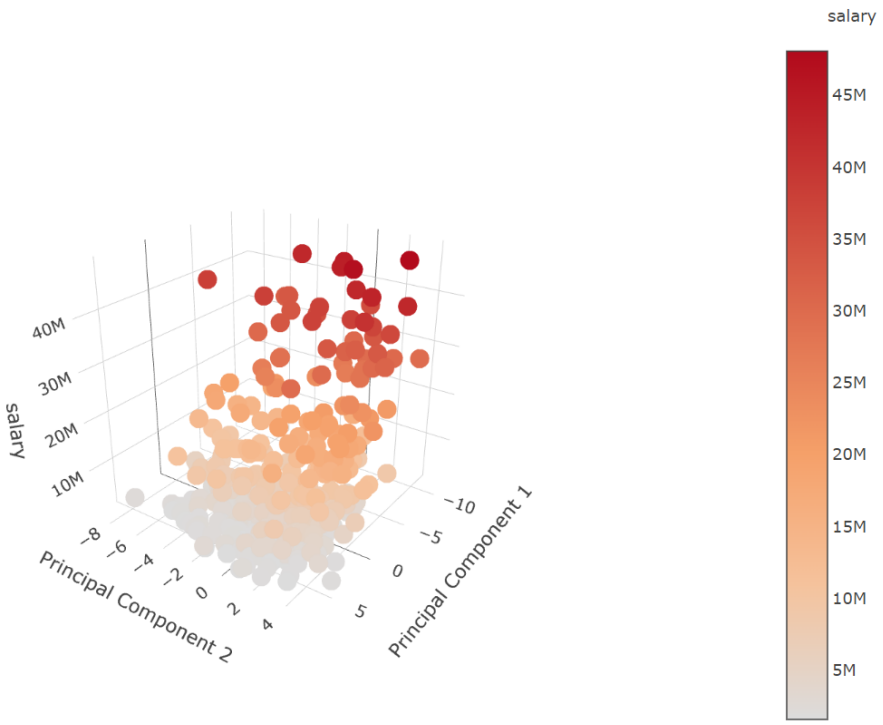


The plot shows that first 2 components has already represent more than 50% of the variation in the dataset. The following analysis will focus on the first 2 component.

```
pca.data <- data.frame(Sample = rownames(pca$x), PC1 = pca$x[, 1], PC2 = pca$x[,
    2], id = 1:nrow(pca$x), salary = all$salary[ind])

ggplot(pca.data, aes(x = PC1, y = PC2)) + geom_point() + xlab(paste("PC1 - ", pca.var.per[1],
    "%", sep = "")) + ylab(paste("PC2 - ", pca.var.per[2], "%", sep = "")) + theme_minimal() +
    ggtitle("PCA Graph")
```

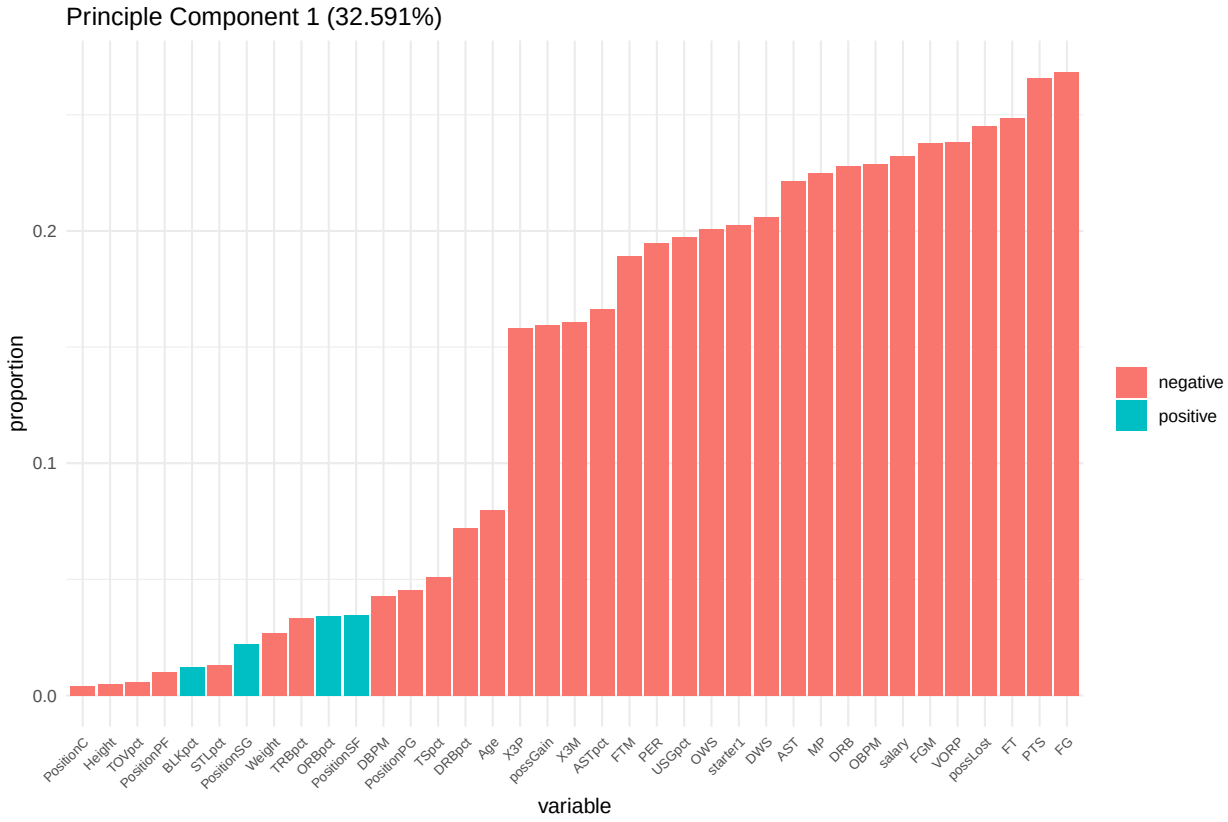


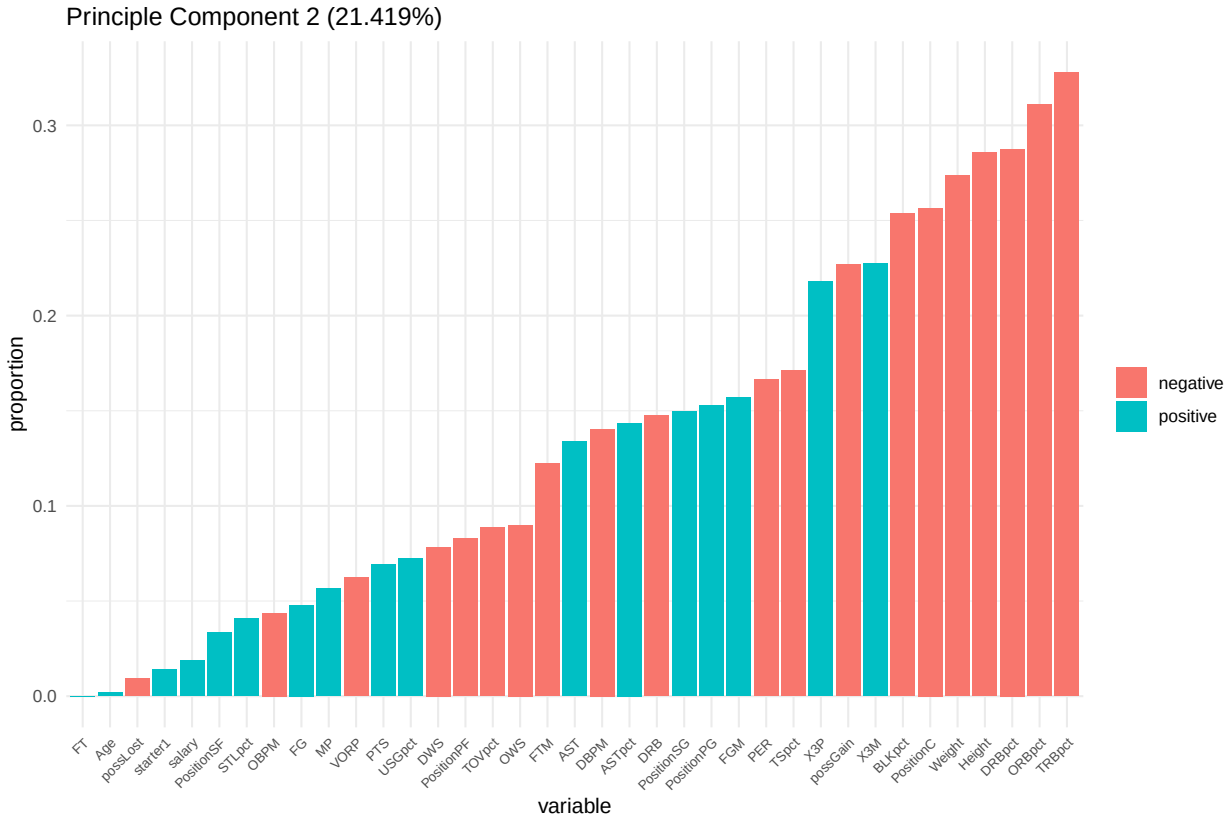

```
fig <- plot_ly(pca.data, x = ~PC1, y = ~PC2, z = ~salary, marker = list(color = ~salary,
    colorscale = c("#FFE1A1", "#683531"), showscale = TRUE)) %>%
    add_markers() %>%
    layout(scene = list(xaxis = list(title = "Principle Component 1"), yaxis = list(title = "Principle
        zaxis = list(title = "salary")), annotations = list(x = 1.13, y = 1.05, text = "salary",
        xref = "paper", yref = "paper", showarrow = FALSE))
fig
```

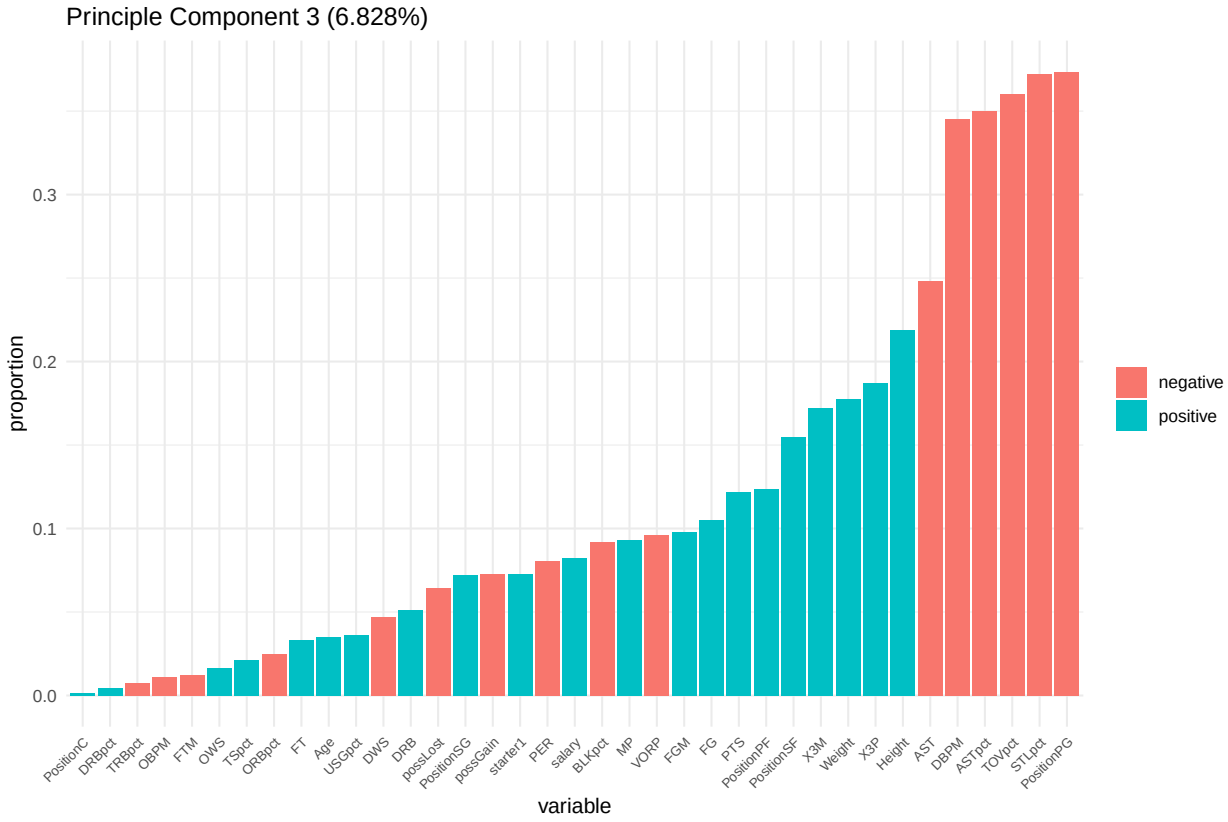


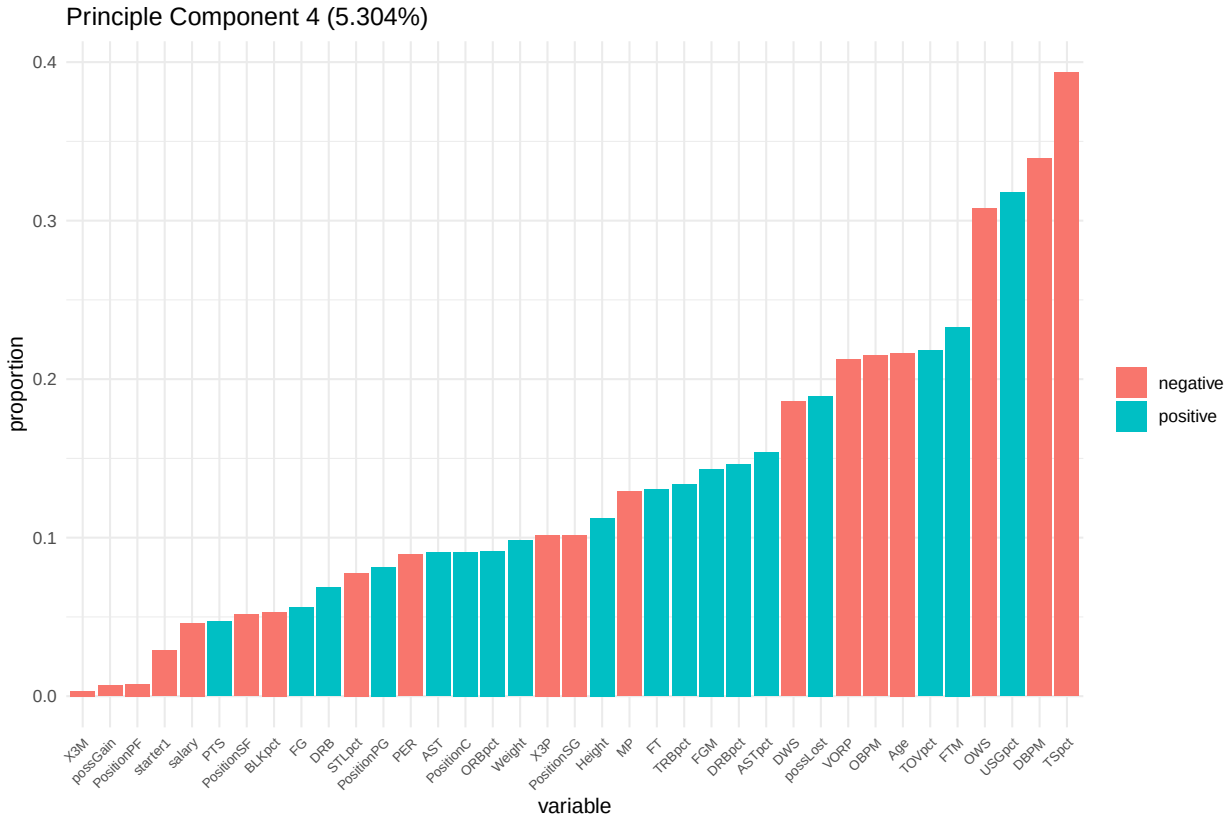
Both the first and second principle components show negative correlation with salary.
I will plot out the principle components that contains more than 5% of information of the original dataset.

```
count = 0
for (i in 1:length(pca.var.per)) {
    if (pca.var.per[i] < 0.05) {
        break
    } else {
        print(ggplot(data.frame(score = pca$rotation[, i], var = names(pca$rotation[,
            i])), aes(x = reorder(var, abs(score)), y = abs(score), fill = ifelse(pca$rotation[,
            i] > 0, "positive", "negative"))) + geom_bar(stat = "identity") + theme_minimal() +
            theme(axis.text.x = element_text(angle = 45, hjust = 1, size = 7)) +
            scale_color_manual(values = list(positive = "blue", negative = "red")) +
            labs(title = paste0("Principle Component ", as.character(i), " (", as.character(pca.var.per
                100), "%)"), x = "variable", y = "proportion") + guides(fill = guide_legend(title = ""))
    }
}
```


Principle Component 1 (32.591%)
proportion
0.0
0.1
0.2
variable
PositionC
Height
TOVpct
PositionPF
BLKpct
STLpct
PositionSG
Weight
TRBpct
ORBpct
PositionSF
DBPM
PositionPG
TSpct
DRBpct
Age
X3P
possGain
X3M
ASTpct
FTM
PER
USGpct
OWS
starter1
DWS
AST
MP
DRB
OBPM
salary
FGM
VORP
possLost
FT
PTS
FG
negative
positive


Principle Component 2 (21.419%)
proportion
0.0
0.1
0.2
0.3
variable
FT
Age
possLost
starter1
salary
PositionSF
STLpct
OBPM
FG
MP
VORP
PTS
USGpct
DWS
PositionPF
TOVpct
OWS
FTM
AST
DBPM
ASTpct
DRB
PositionSG
PositionPG
FGM
PER
TSpct
X3P
possGain
X3M
BLKpct
PositionC
Weight
Height
DRBpct
ORBpct
TRBpct
negative
positive


Principle Component 3 (6.828%)
proportion
0.0
0.1
0.2
0.3
PositionC DRBpct TRBpct OBPM FTM OWS TSpct ORBpct FT Age USGpct DWS DRB possLost PositionSG possGain starter1 PER salary BLKpct MP VORP FGM FG PTS PositionPF PositionSF X3M Weight X3P Height AST DBPM ASTpct TOVpct STLpct PositionPG
variable
negative
positive

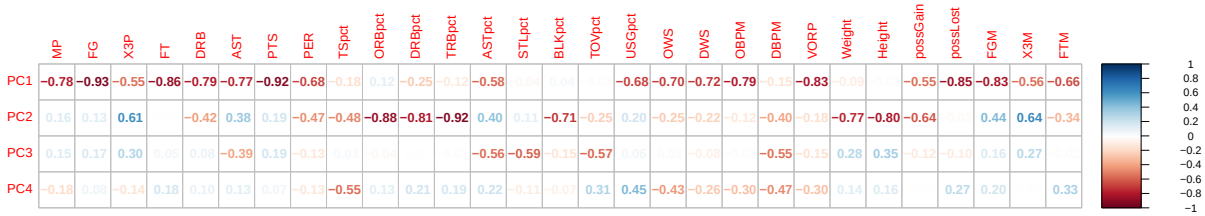

Principle Component 4 (5.304%)
proportion
0.0
0.1
0.2
0.3
0.4
X3M possGain PositionPF starter1 salary PTS PositionSF BLKpct FG DRB STLpct PositionPG PER AST PositionC ORBpct Weight X3P PositionSG Height MP FT TRBpct FGM DRBpct ASTpct DWS possLost VORP OBPM Age TOVpct FTM OWS USGpct DBPM TSpct
variable
negative
positive

12.1.2 Correlation between variables

```
cor_Mat <- cor(data.frame(alldata[, numVar], pca$x[, 1:4]))

cor_high <- names(which(rowSums(abs(cor_Mat[, c("PC1", "PC2", "PC3", "PC4")]) > 0.5) >
    0))
cor_high <- cor_high[!cor_high %in% c("PC1", "PC2", "PC3", "PC4")]
options(scipen = 100)
round(cor_Mat[cor_high, c("PC1", "PC2", "PC3", "PC4")], digits = 2)
```

```
##             PC1   PC2   PC3   PC4
## MP        -0.78  0.16  0.15 -0.18
## FG        -0.93  0.13  0.17  0.08
## X3P       -0.55  0.61  0.30 -0.14
## FT        -0.86  0.00  0.05  0.18
## DRB       -0.79 -0.42  0.08  0.10
## AST       -0.77  0.38 -0.39  0.13
## PTS       -0.92  0.19  0.19  0.07
## PER       -0.68 -0.47 -0.13 -0.13
## TSpct     -0.18 -0.48  0.03 -0.55
## ORBpct     0.12 -0.88 -0.04  0.13
## DRBpct    -0.25 -0.81  0.01  0.21
## TRBpct    -0.12 -0.92 -0.01  0.19
## ASTpct    -0.58  0.40 -0.56  0.22
## STLpct    -0.04  0.11 -0.59 -0.11
## BLKpct     0.04 -0.71 -0.15 -0.07
## TOVpct    -0.02 -0.25 -0.57  0.31
## USGpct    -0.68  0.20  0.06  0.45
## OWS       -0.70 -0.25  0.03 -0.43
## DWS       -0.72 -0.22 -0.08 -0.26
## OBPM      -0.79 -0.12 -0.02 -0.30
## DBPM      -0.15 -0.40 -0.55 -0.47
## VORP      -0.83 -0.18 -0.15 -0.30
## Weight    -0.09 -0.77  0.28  0.14
## Height    -0.02 -0.80  0.35  0.16
## possGain  -0.55 -0.64 -0.12 -0.01
## possLost  -0.85 -0.03 -0.10  0.27
## FGM       -0.83  0.44  0.16  0.20
## X3M       -0.56  0.64  0.27  0.00
## FTM       -0.66 -0.34 -0.02  0.33
```

```
corrplot(cor_Mat[c("PC1", "PC2", "PC3", "PC4"), cor_high], tl.pos = "lt", method = "number")
```



12.1.2.1 First principle component

It is highly correlated with minutes played, field goal, free throw, defensive rebound, assists, points, offensive box score, value over replacement player, possession lost (turnovers and personal fouls), and field goal missed. This means that the above variables vary together. If one increase, then the remaining ones tend to increase as well. I think these statistics are correlated through minutes played as better player usually have higher minutes and better statistics in these criterias.

12.1.2.2 Second principle component

It is highly correlated with offensive rebound percentage, defensive rebound percentage, total rebound percentage, weight, and height. This means these variables tend to vary together. These statistics are all related to rebounding. Weight and height are crucial in grabbing rebound as higher weight can lead to easier box out while higher in height can lead to easier catching of the ball which all leads to better rebounding.

12.2 Linear Regression

Some variable has been removed due to their multicollinearity with each other and their interpretability. We will define absolute value of correlation larger than 0.75 as too highly correlated for linear regression.

- PTS, FGM, FT, possLost are removed due to their high correlation with FG
- OBPM, OWS are removed due to their high correlation with VORP
- ASTpct is removed due to its high correlation with AST.
- X3M is removed due to its high correlation with X3P.
- Positions are removed due to them having similar weight in the linear model above

```
alldata_lm <- dplyr::select(alldata, !c(X, PTS, FGM, FT, possLost, OBPM, OWS, ASTpct,
    X3M, PositionPG, PositionSG, PositionSF, PositionPF, PositionC))

mod_lm_sh <- train(salary ~ ., alldata_lm, tuneLength = 20, trControl = trainControl(method = "cv",
    number = 10), method = "lm")
```

```
summary(mod_lm_sh)$r.squared
```

```
## [1] 0.7296182
```

```
rmse(predict(mod_lm_sh), alldata$salary)
```

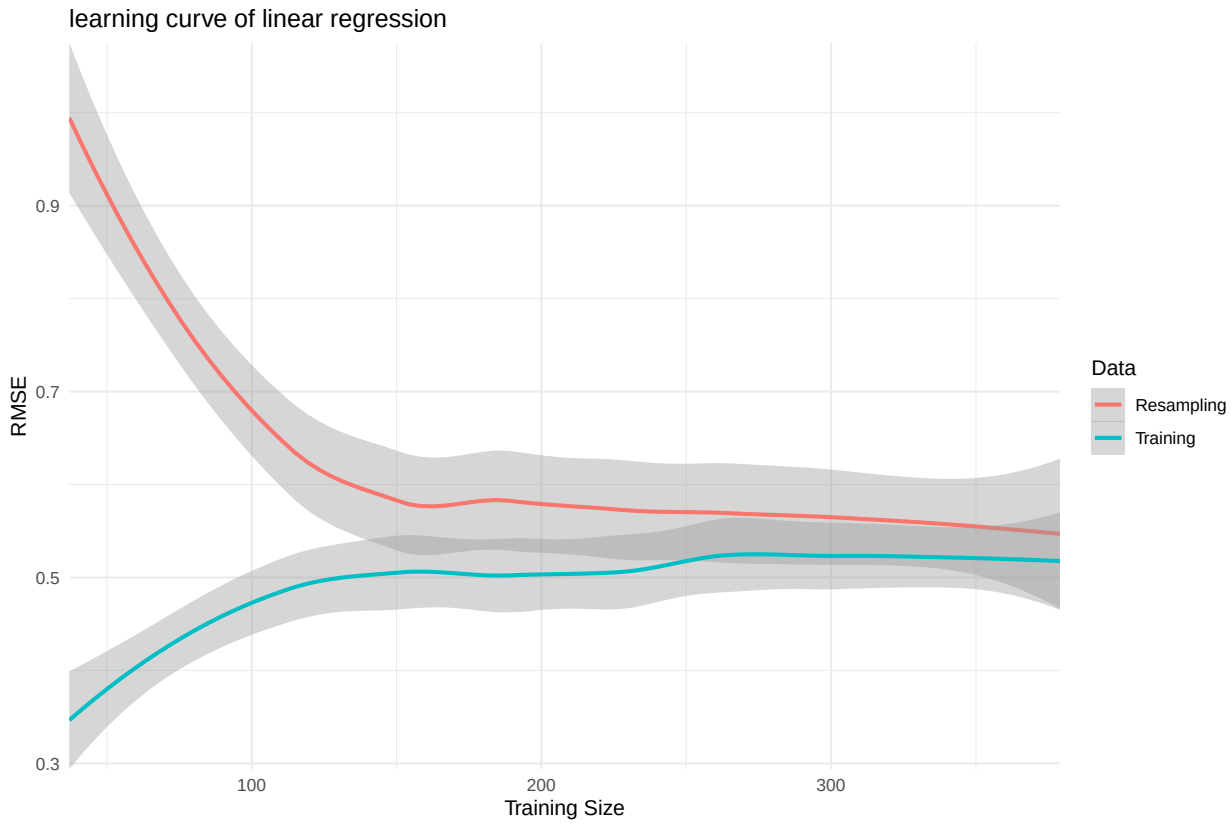
```
## [1] 0.5218133
```

The Rsquared of this model is 0.7296182 which is still high enough for interpretation and the residual (1.3822702) is also similar.

12.2.1 learning curve

```
train_control <- trainControl(method = "cv", number = 10)
mod_lm_lcurve <- learning_curve_dat(dat = alldata_lm, method = "lm", outcome = "salary",
    tuneLength = 20, trControl = train_control, metric = "RMSE", proportion = (1:10)/10)
```

```
ggplot(mod_lm_lcurve, aes(x = Training_Size, y = RMSE, color = Data)) + geom_smooth(method = loess,
    span = 0.8, formula = y ~ x) + scale_x_continuous(expand = c(0, 0)) + scale_y_continuous(expand = c
    0)) + labs(title = "learning curve of linear regression", x = "Training Size",
    y = "RMSE") + theme_minimal()
```



As the learning curve's resampling error and train error are both converging to the a low RMSE, this show that the model with these variable is neither overfitting or underfitting.

12.2.2 Coefficients

```
summary(mod_lm_sh)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.76806 -0.33955  0.01651  0.36756  1.27412
##
## Coefficients:
##              Estimate Std. Error t value             Pr(>|t|)
## (Intercept) 15.300815   0.559333  27.355 < 0.0000000000000002 ***
## Age          0.292665   0.032188   9.092 < 0.0000000000000002 ***
## MP          -0.143047   0.078545  -1.821              0.069414 .
```

```
## FG           0.642475   0.285371   2.251            0.024973 *
## X3P          0.117674   0.056309   2.090            0.037348 *
## DRB          2.452007   1.265765   1.937            0.053517 .
## AST          0.498903   0.652059   0.765            0.444709
## PER         -0.140897   0.450991  -0.312            0.754908
## TSpct       -1.365468   1.937087  -0.705            0.481330
## ORBpct       2.010968   3.163803   0.636            0.525435
## DRBpct       1.653886   4.968362   0.333            0.739419
## TRBpct      -5.077570   7.377477  -0.688            0.491743
## STLpct      -0.314362   0.601318  -0.523            0.601447
## BLKpct       2.201886   1.119657   1.967            0.050011 .
## TOVpct      -0.055161   0.819803  -0.067            0.946392
## USGpct       0.029995   0.087718   0.342            0.732596
## DWS          0.076833   0.079427   0.967            0.334027
## DBPM         0.009965   0.067802   0.147            0.883240
## VORP         1.501991   0.728270   2.062            0.039896 *
## Weight      -0.045318   0.049069  -0.924            0.356339
## Height       0.129969   0.052146   2.492            0.013144 *
## possGain    -0.697663   0.986468  -0.707            0.479885
## FTM          0.786640   0.472979   1.663            0.097163 .
## starter1     0.298772   0.085892   3.478            0.000567 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5392 on 355 degrees of freedom
## Multiple R-squared:  0.7296, Adjusted R-squared:  0.7121
## F-statistic: 41.65 on 23 and 355 DF,  p-value: < 0.00000000000000022
```

We will take a look at variable with significant weight (p-value < 0.1).

Age:

```
##                    Estimate                  Std. Error
## 0.29266520037294724287946 2 0.03218768576382054213969 0
##                     t value                    Pr(>|t|)
## 9.09245860421277818375074 3 0.00000000000000000699556 7
```

The age show very significant positive weight in the model. The weight is 0.293.
The possibility of age result in that weight for null hypothesis being 0 is less than $2 \cdot 10^{-14}\%$. The weight is more than 9 standard deviation away from 0.
The NBA minimum salary increase with the years that the players have played in the league from 953k for 0 experience and 2.721M for 10 or more years in the league.
In addition, player with higher age are often more experienced, while player who are more experience usually gives more confidence to the team. They have already proof with time their ability to play in NBA, which let the team willing to give them a higher salary.

Starter:
(starter1 = 1 means the player starts for more to 50% of the game, starter1 = 0 means the player starts for leess than 50% of the game)

```
##     Estimate   Std. Error      t value     Pr(>|t|)
## 0.2987724122 0.0858921270 3.4784609781 0.0005669854
```

The starter factor variable show very significant positive weight in the model. The weight is 0.299.
The possibility of the weight resulting in this weight is 0.057%, which is about 3.5 standard error away from 0.
Starter player are most often the best players in the team and played the most in a game. Hence, it is reasonable that the starter player get a higher salary.

Field Goal:

```
##   Estimate Std. Error    t value   Pr(>|t|)
## 0.64247509 0.28537135 2.25136505 0.02497346
```

The field goal has a weight of 0.642 and shows high significance.
The p-value is $2.5 \cdot 10^{-2}$ and 2.3 standard error from 0.
Players with more field goal means that they have more fire power and stronger ability to score. As the NBA encourages offense, player with strong offensive ability are usually paid more. A high means a higher field goal attempt rate (correlation of FG, FGA = 0.9417247), which represent the players' offensive importance in the team. Hence, these players with high field goal will usually paid more.

3 Point Field Goal:

```
##   Estimate Std. Error    t value   Pr(>|t|)
## 0.11767364 0.05630899 2.08978435 0.03734845
```

The three point field goal has a weight of 0.118 and shows high significance.
The p-value is 3.7% and 2.1 standard error away from 0.
In the modern NBA, spacing has become more and more important and valued by every team in the league. It can lead to more wide open chance, more defense loophole from the opponent team and thus easier points. Three point ability of a team is one of the crucial factor that determine the teams offensive spacing. Hence, players with strong 3 point ability are often valuable to the team. A high 3 point field goal indicate that the player has a high 3 point ability which will lead to a higher salary.

Value Over Replacement Player:

```
Value over Replacement Player (VORP) converts the BPM rate into an estimate of each player's overall co
```

(source: https://www.basketball-reference.com/about/bpm2.html)

```
##   Estimate Std. Error    t value   Pr(>|t|)
## 1.50199100 0.72827021 2.06240897 0.03989593
```

The value over replacement player has weight of 1.5 and show high significance.
The p-value is 3.4% and 2.1 standard error away from 0.
The value over replacement player (VORP) is a overall estimate of the players' contribution to a team comparing with a rotational player. The higher the VORP, the better the performance of the player and thus usually have a higher salary.

Height:

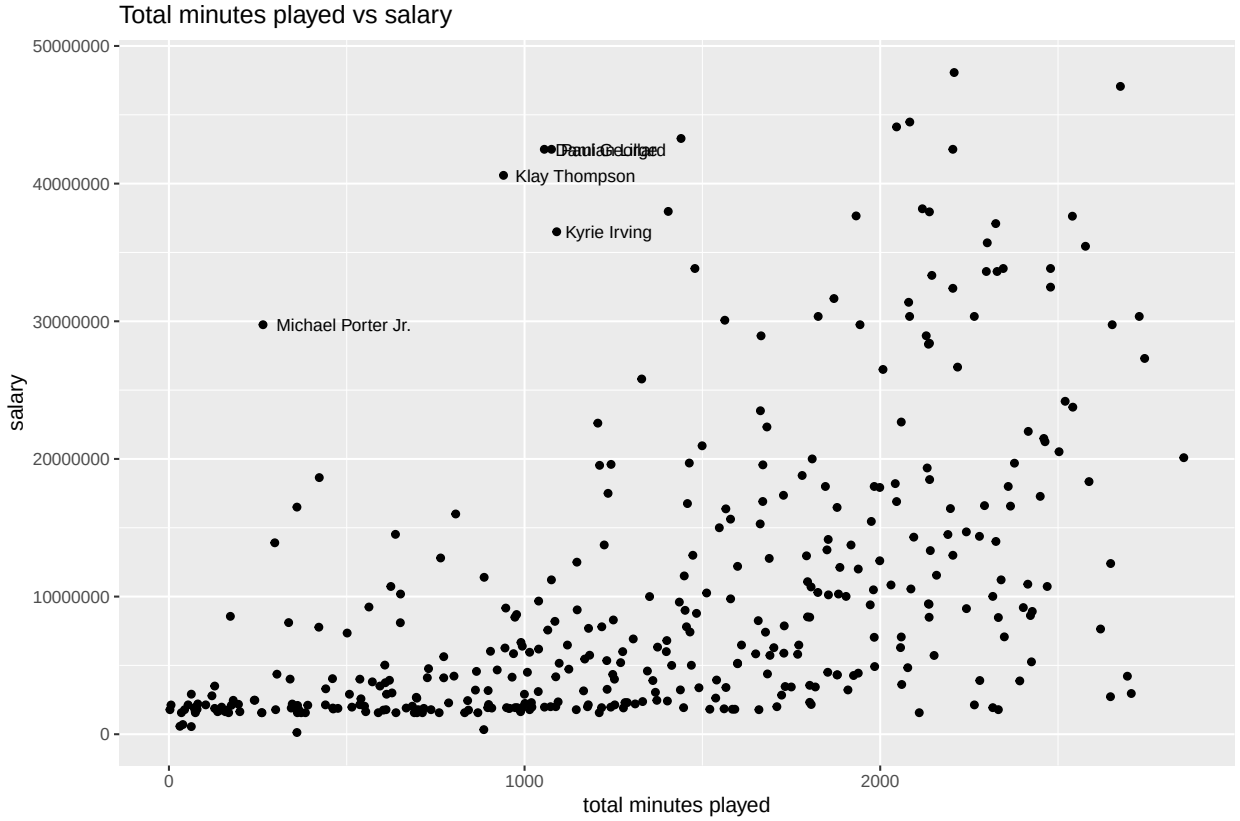
```
##   Estimate Std. Error    t value   Pr(>|t|)
## 0.12996933 0.05214643 2.49239169 0.01314391
```

The weight is 0.13 and show high significance.
The p-value is 1.3% and 2.5 standard error away from 0.
Height has always been a important player attribute throughout the history of basketball. Pace and acceleration has played a more important role in modern NBA, which indirectly reduce the importance of height as height are usually negatively correlated to quickness. However, it is undeniable that height will affect the players finishing ability, defense ability. Shorter players usually have less defensive capability. This decrease the players'overall value and thus given lower salary.

Minutes Played:

```
##    Estimate  Std. Error     t value    Pr(>|t|)
## -0.14304745  0.07854464 -1.82122491  0.06941423
```

The minutes played has weight of -0.143 and shows moderate significance.
The p-value is 6.9% and -1.8 standard away from 0.
This result contradict with the correlation of minutes played and salary of 0.6089823.



This show the players with high salary but low minutes played might be the reason of the result. These players are often injured. This show that more minutes played doesn't necessarily correlated to higher salary. The player salary can also be affected by pass performance and team contribution outside of the court.

Defensive Rebound:

```
##   Estimate Std. Error    t value   Pr(>|t|)
## 2.45200687 1.26576543 1.93717321 0.05351719
```

It has weight of 2.5 and show moderate significance.
It has p-value of 5.4% and 1.9 standard error away from 0.

Securing defensive rebound is important in a game after the opponent missing their field goal attempts. A high defensive rebound represent less second chance point for the opponent and a chance for transition offense. A high defensive rebound can also indirectly indicate the defensive ability of the team, as a defensive rebound is a result of a successful defense. This factor will lead to a higher salary of the player.

Block Percentage:

```
BLK% - Block Percentage (available since the 1973-74 season in the NBA); the formula is 100 * (BLK * (T
```

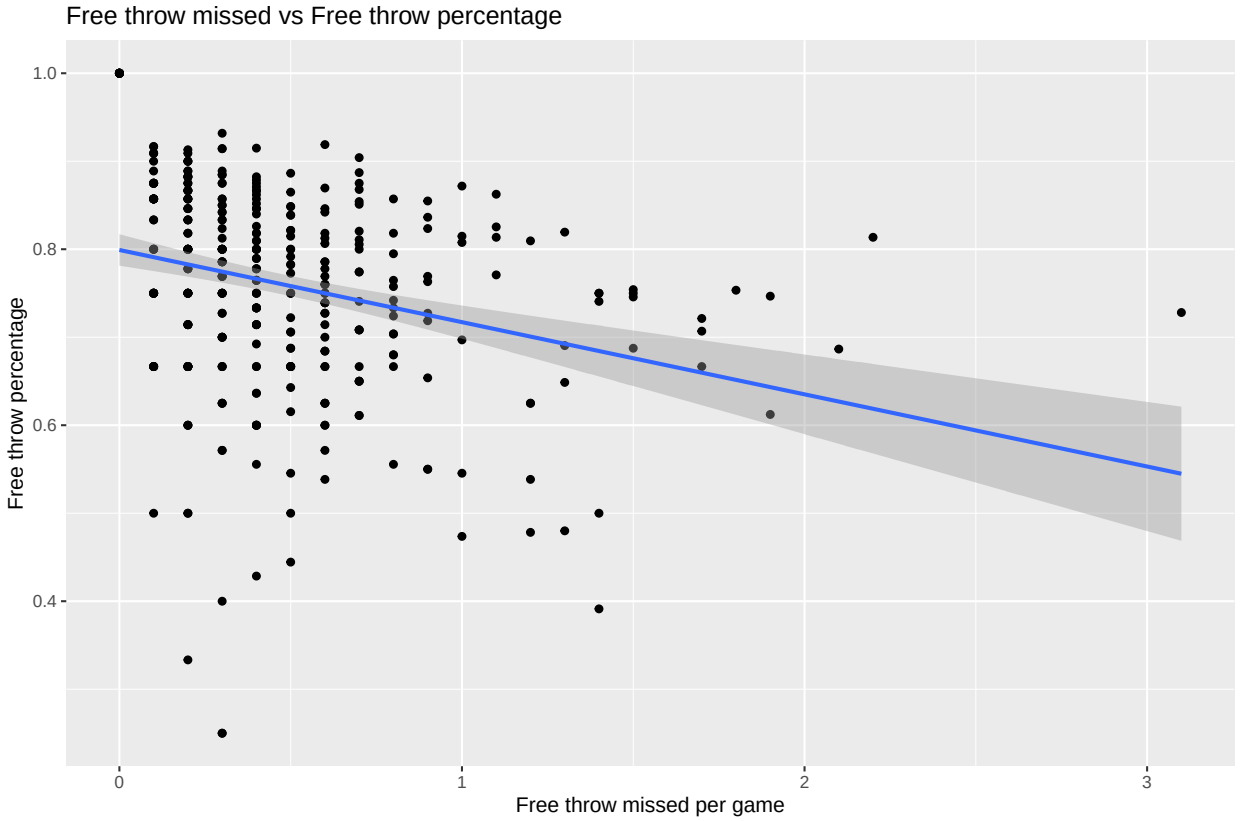
```
##   Estimate Std. Error    t value   Pr(>|t|)
## 2.20188562 1.11965738 1.96657090 0.05001135
```

It has weight of 2.2 and show moderate significance.
It has p-value of 5% and 2 standard error away from 0.
A high block percentage represent a strong defensive ability which increase the value of the player, which results in higher salary.

Free Throw Missed:

```
##   Estimate Std. Error    t value   Pr(>|t|)
## 0.78663950 0.47297942 1.66315798 0.09716335
```

It has a weight of 0.79 and shows moderate significant.
It has p-value of 9.7% and 1.6 standard error away from 0. This is counter intuitive as a better free throw player should be better.



Although it is counter intuitive, the correlation between free throw missed and salary is 0.5263002. This

might be explained by some players like Giannis Antetokounmpo and LeBron James has strong ability but low in free throw percentage.

Adams, L., 2022. NBA minimum salaries for 2022/23.
spotrac, n.d. Ishmail wainright.