Chicago Bulls R Markdown

Lorenzo Scotti
9/1/2021

1. Introduction

In a basketball game, the objective of each team is to score more points than the other team to be the winner. As a team sport consisting of five players, each team player requires default skill set of their specialization.

I am a data analyst for a National basketball team, the Chicago Bulls. My major job is to interpret both on-field and off-field data to help the team make strategic decisions. I perform data and team scores etc, in order to look for meaningful insight and improvement plans. In the last NBA season, my team placed 27th out of 30 and my current project is to carry out data analysis based on the previous season and players related statistics. The winning percentage can be manipulated by reprioritizing data analyses. I also work with the general manager of the Chicago Bulls to analyze the team's performance through player's individual performance data in conjunction with the best winning tactics. The team budget of $118 million is taken into account, in order to form the best performing team in the coming season.

Load required packages

```
library(ggplot2)
library(dplyr)
library(tidyr)
library(stringr)
```

2. Reading and cleaning the raw data

To begin with, the team winning percentage will first be considered. With the five data sets provided, the two 2018-2019 NBA team statistics files are first be read and combined to one data set. All variables from these two files are included at this first stage. No missing values are shown. Instead of using the Wins(W) and Losses(L) data values, Winning Percentage (win_per) is created. Data sets 3 and 4 are read, cleaned and numerous factor variables for further expression. The remaining two data sets of player statistics and player salaries, as well as the expected player contracts budge, will be considered at the later stage.

Data source

```
df_1 <- read.csv("./Capstone Dependencies/2018-19 Team Stats/2018-19_nba_team-statistics_1.csv")
df_2 <- read.csv("./Capstone Dependencies/2018-19 Team Stats/2018-19_nba_team-statistics_2.csv")
team_stats_raw <- rbind(df_1, df_2)
```

```
str(team_stats_raw)
```

```
'data.frame':   30 obs. of  18 variables:
 $ Team    : chr  "Toronto Raptors" "Golden State Warriors" "Houston Rockets" "New Orleans Pelicans" "Boston Celtics" "Milwaukee Bucks" "Denver Nuggets" "Indiana Pacers" ...
 $ Rank    : int  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...
 $ W       : int  58 57 55 50 49 48 46 45 44 42 41 39 37 36 34 33 32 29 28 27 ...
 $ L       : int  24 25 27 32 33 34 36 37 38 40 41 43 45 46 48 49 50 53 54 55 ...
 $ PTS     : int  9704 9519 9267 9476 9396 9553 9553 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 ...
 $ FGM     : int  3616 3479 3394 3463 3433 3488 3488 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 ...
 $ FGA     : int  7267 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 ...
 $ FG%     : chr  "0.498" "0.497" "0.485" "0.495" "0.490" "0.498" "0.498" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" ...
 $ 3PM     : int  1087 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 ...
 $ 3PA     : int  3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 ...
 $ 3P%     : chr  "0.362" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" ...
 $ FTM     : int  1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 ...
 $ FTA     : int  1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 ...
 $ FT%     : chr  "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" ...
 $ OREB    : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ DREB    : int  3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 ...
 $ REB     : int  4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 ...
 $ AST     : int  2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 ...
 $ STL     : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ BLK     : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ TOV     : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
```

```
team_stats_raw
```

```
   W  L                 Team Rank PTS FGM FGA FG% 3PM 3PA 3P% FTM FTA FT%
1 58 24      Toronto Raptors    1 9704 3616 7267 0.498 1087 3000 0.362 1530 1700 0.900
2 57 25 Golden State Warriors    2 9519 3479 7000 0.497 1015 3000 0.338 1530 1700 0.900
3 55 27     Houston Rockets    3 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
4 50 32  New Orleans Pelicans    4 9476 3463 7000 0.495 1015 3000 0.338 1530 1700 0.900
5 49 33      Boston Celtics    5 9396 3433 7000 0.490 1015 3000 0.338 1530 1700 0.900
6 48 34     Milwaukee Bucks    6 9553 3488 7000 0.498 1015 3000 0.338 1530 1700 0.900
7 46 36      Denver Nuggets    7 9553 3488 7000 0.498 1015 3000 0.338 1530 1700 0.900
8 45 37      Indiana Pacers    8 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
9 44 38     Detroit Pistons    9 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
10 42 40     Miami Heat   10 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
11 41 41     Utah Jazz   11 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
12 39 43 Oklahoma City Thunder   12 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
13 37 45 Los Angeles Clippers   13 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
14 36 46 San Antonio Spurs   14 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
15 34 48 Minnesota Timberwolves   15 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
16 33 49 Portland Trail Blazers   16 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
17 32 50 Los Angeles Lakers   17 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
18 29 53 Washington Wizards   18 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
19 28 54 Charlotte Hornets   19 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
20 27 55 Atlanta Hawks   20 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
21 58 24      Toronto Raptors    1 9704 3616 7267 0.498 1087 3000 0.362 1530 1700 0.900
22 57 25 Golden State Warriors    2 9519 3479 7000 0.497 1015 3000 0.338 1530 1700 0.900
23 55 27     Houston Rockets    3 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
24 50 32  New Orleans Pelicans    4 9476 3463 7000 0.495 1015 3000 0.338 1530 1700 0.900
25 49 33      Boston Celtics    5 9396 3433 7000 0.490 1015 3000 0.338 1530 1700 0.900
26 48 34     Milwaukee Bucks    6 9553 3488 7000 0.498 1015 3000 0.338 1530 1700 0.900
27 46 36      Denver Nuggets    7 9553 3488 7000 0.498 1015 3000 0.338 1530 1700 0.900
28 45 37      Indiana Pacers    8 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
29 44 38     Detroit Pistons    9 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
30 42 40     Miami Heat   10 9267 3394 7000 0.485 1015 3000 0.338 1530 1700 0.900
   OREB DREB REB AST STL BLK TOV
1  1000 3000 4000 2000 1000 1000 1000
2  1000 3000 4000 2000 1000 1000 1000
3  1000 3000 4000 2000 1000 1000 1000
4  1000 3000 4000 2000 1000 1000 1000
5  1000 3000 4000 2000 1000 1000 1000
6  1000 3000 4000 2000 1000 1000 1000
7  1000 3000 4000 2000 1000 1000 1000
8  1000 3000 4000 2000 1000 1000 1000
9  1000 3000 4000 2000 1000 1000 1000
10 1000 3000 4000 2000 1000 1000 1000
11 1000 3000 4000 2000 1000 1000 1000
12 1000 3000 4000 2000 1000 1000 1000
13 1000 3000 4000 2000 1000 1000 1000
14 1000 3000 4000 2000 1000 1000 1000
15 1000 3000 4000 2000 1000 1000 1000
16 1000 3000 4000 2000 1000 1000 1000
17 1000 3000 4000 2000 1000 1000 1000
18 1000 3000 4000 2000 1000 1000 1000
19 1000 3000 4000 2000 1000 1000 1000
20 1000 3000 4000 2000 1000 1000 1000
21 1000 3000 4000 2000 1000 1000 1000
22 1000 3000 4000 2000 1000 1000 1000
23 1000 3000 4000 2000 1000 1000 1000
24 1000 3000 4000 2000 1000 1000 1000
25 1000 3000 4000 2000 1000 1000 1000
26 1000 3000 4000 2000 1000 1000 1000
27 1000 3000 4000 2000 1000 1000 1000
28 1000 3000 4000 2000 1000 1000 1000
29 1000 3000 4000 2000 1000 1000 1000
30 1000 3000 4000 2000 1000 1000 1000
```

```
team_stats_raw$win_per <- round W/(W+L)
```

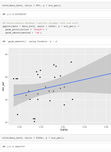
```
str(team_stats_raw)
```

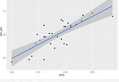
```
'data.frame':   30 obs. of  19 variables:
 $ W       : int  58 57 55 50 49 48 46 45 44 42 41 39 37 36 34 33 32 29 28 27 ...
 $ L       : int  24 25 27 32 33 34 36 37 38 40 41 43 45 46 48 49 50 53 54 55 ...
 $ Team    : chr  "Toronto Raptors" "Golden State Warriors" "Houston Rockets" "New Orleans Pelicans" "Boston Celtics" "Milwaukee Bucks" "Denver Nuggets" "Indiana Pacers" ...
 $ Rank    : int  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...
 $ PTS     : int  9704 9519 9267 9476 9396 9553 9553 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 9267 ...
 $ FGM     : int  3616 3479 3394 3463 3433 3488 3488 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 3394 ...
 $ FGA     : int  7267 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 7000 ...
 $ FG%     : chr  "0.498" "0.497" "0.485" "0.495" "0.490" "0.498" "0.498" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" "0.485" ...
 $ 3PM     : int  1087 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 1015 ...
 $ 3PA     : int  3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 ...
 $ 3P%     : chr  "0.362" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" "0.338" ...
 $ FTM     : int  1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 1530 ...
 $ FTA     : int  1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 ...
 $ FT%     : chr  "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" "0.900" ...
 $ OREB    : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ DREB    : int  3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 3000 ...
 $ REB     : int  4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 4000 ...
 $ AST     : int  2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 ...
 $ STL     : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ BLK     : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
 $ TOV     : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
```

```
team_stats_raw$win_per
```

```
[1] 0.706 0.695 0.671 0.610 0.598 0.585 0.561 0.549 0.537 0.512 0.500 0.476 0.451 0.439 0.415 0.402 0.390 0.354 0.341 0.329
```

```
ggplot(team_stats_raw, aligned = TRUE)
```


Warning: Ignoring unknown aesthetics: aligned
Warning: Ignoring unknown aesthetics: aligned
Warning: Ignoring unknown aesthetics: aligned

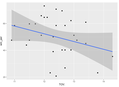


```
KOTOKORA_DATA, verse 7 ~ PPL, 2 * est_gen()
```

```
## (1) G.Hunkel
```

```
Warning messages:
1: Ignoring unknown parameters: method, formula, se
2: Ignoring unknown parameters: method, formula, se
  geom_smooth(method = "lm", formula = y ~ x,
  geom_smooth(method = "lm", formula = y ~ x,
```

```
>> ggsave(device = cairo_pdf, width=7, height=7)
```

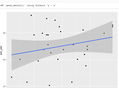


```
KOTOKORA_DATA, verse 7 ~ ORA, 2 * est_gen()
```

```
## (1) G.Hunkel
```

```
Warning messages:
1: Ignoring unknown parameters: method, formula, se
2: Ignoring unknown parameters: method, formula, se
  geom_smooth(method = "lm", formula = y ~ x,
  geom_smooth(method = "lm", formula = y ~ x,
```

```
>> ggsave(device = cairo_pdf, width=7, height=7)
```

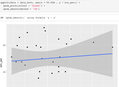


```
KOTOKORA_DATA, verse 7 ~ ORA, 2 * est_gen()
```

```
## (1) G.Hunkel
```

```
Warning messages:
1: Ignoring unknown parameters: method, formula, se
2: Ignoring unknown parameters: method, formula, se
  geom_smooth(method = "lm", formula = y ~ x,
  geom_smooth(method = "lm", formula = y ~ x,
```

```
>> ggsave(device = cairo_pdf, width=7, height=7)
```



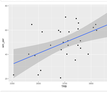
```
KOTOKORA_DATA, verse 7 ~ PPL
```

```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (1) V.S.POINTS
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

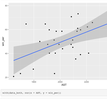



```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (2) V.S.DUNK
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

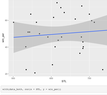



```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (3) V.S.REBOUND
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

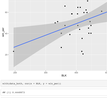



```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (4) V.S.TOV
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

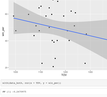



```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (5) V.S.TRB
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

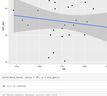



```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (6) V.S.AGE
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

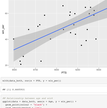



```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (7) V.S.WINNING
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

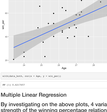



```
ggplot(data = data_bmi, aes(x = BMI, y = win_pct))
```

```
## (8) V.S.WINNING
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = BMI, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
ggplot(data = data_bmi, aes(x = Age, y = win_pct))
```

```
## (9) V.S.WINNING
```

```
## Warning: Ignoring unknown aesthetics: shape
## Warning: Ignoring unknown aesthetics: shape
ggplot(data = data_bmi, aes(x = Age, y = win_pct)) +
  geom_point(colour= "black") +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
ggplot(data = data_bmi, aes(x = Age, y = win_pct))
```

Multiple Linear Regression

By investigating on the above plots, 4 variables with high strength of the winning percentage relationship as well as P-value, are considered for further data modelling.

1) TS. (true shoot percentage)

2) TOV (turn over)

3) TRB (total rebound)

4) Age

```
mlr1 = lm(win_pct ~ TS. + TOV + TRB + Age)
```

```
## Warning: Error in [[<-.data.frame]]: 4 elements in value replacement for 1 column
```

```
## # A tibble: 5 x 2
##   term        estimate std.error  statistic   p.value
##   <chr>          <dbl>     <dbl>      <dbl>     <dbl>
## 1 (Intercept)  -1.00    1.00      -1.00     0.321
## 2 TS.          -0.0000  0.0000    -0.0000   1.000
## 3 TOV          -0.0000  0.0000    -0.0000   1.000
## 4 TRB          -0.0000  0.0000    -0.0000   1.000
## 5 Age          -0.00    0.00      -0.00     1.000
```

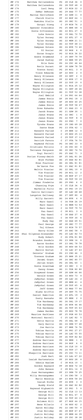
```
summary(mlr1)
```

```
##
## Call:
## lm(formula = win_pct ~ TS. + TOV + TRB + Age)
##
## Residuals:
##  [1]  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000
## [11]  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000
## [21]  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000
##
```

Compare

```
filtered_df = dataset[(dataset_df[ age_pen ] >= 100 ) & (dataset_df[ TOV ] >= 0.5000) & (dataset_df[ TRC ] >= 0.5000) & (dataset_df[ Age ] < 0.7500)]
ugent_pen = filtered_df[ ugent_pen ].dropna()
age_pen = filtered_df[ age_pen ].dropna()
print(ugent_pen.shape, age_pen.shape)
```

```
(8000,) (8000,)
```




```
scipy.stats.ttest_ind(ugent_pen, age_pen, equal_var=False)
```

```
Ttest_indResult(statistic=nan, pvalue=nan)
```

```
df = df.head()
df
```

Players

```
df = pd.read_csv('./Datasets/Basketball/Data/Raw/2016-17_NBA_player-statistics.csv')
df = df.drop_duplicates(subset=['player', 'Tm'], keep='first')
df = df.reset_index(drop=True)
df.to_csv('./Datasets/Basketball/Data/Raw/2016-17_NBA_player-statistics.csv')
```

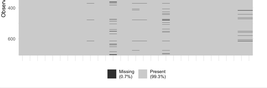
Handling missing data

```
df[df.isnull()].sum()
```

```
Unnamed: 0\n\n\n
```

| | | |

20-09-2020	00	00	0.0000000000
20-09-2020	00	01	0.0000000000
20-09-2020	00	02	0.0000000000
20-09-2020	00	03	0.0000000000
20-09-2020	00




```
pmg[is.na(pmg)] = 0
```

```
sum(is.na(pmg))
```

```
## [1] 0
```

5. Player recommendations

After developing the winning percentage model based on team statistic, the model can next be applied on team player selection by introducing team player statistics. The two provided data sets of team player statistics and team player salaries are first combined. No missing values are found with the chosen data (player_id,Age, TM,GS,POS,PTS,FGS,FTA,TOV,TRB,salary) for interpretation. By considering some players with limited game starts who may give a very high true shooting percentage, only player with game start >41games (half-season) is considered in this model.

filtered player data

```
pmmf <- subset(pmz, select = c('player_name', 'player_id','Age' ,'TM', 'GS', 'Pos', 'PTS','FGA','FTA', 'TOV','TRB
', 'salary'))

# game start more than 41 games will be considered (half of the regular season)
pmaf <- pmmf %>% filter( GS >= 41)

# calculating true shooting%
## TS. = Points/(2*(Field Goals Attempted + 0.44 * Free Throws Attempted))
pmaf <- mutate(pmaf, TS. = PTS/(2*(FGA + 0.44* FTA)))

pmaf <- mutate(pmaf, win_score = -369 + TS. * 426.9 + TOV * -0.04355 + TRB * 0.03569 + Age * 3.73282)
```

```
duplicated(pmaf)
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE
```

```
pmaf <- pmaf%*% distinct(player_name, .keep_all = TRUE)
```

```
pmaf %>% select(player_name, Pos, salary, win_score) %>%
  arrange(desc(win_score)) %>% top_n(10)
```

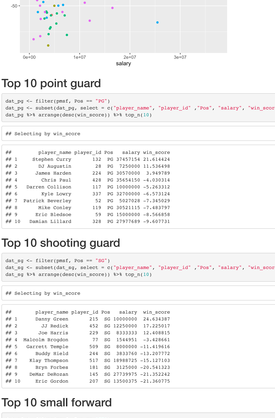
```
## Selecting by win_score
```

```
##                player_name Pos   salary win_score
## 1                Rudy Gobert   C 23493573 50.58856
## 2               DeAndre Jordan   C 22897200 49.45593
## 3                Jarlale McGee   C 2939887 32.69070
## 4                   Taj Gibson  PF 14000000 28.74763
## 5               Clint Capela   C 19793104 27.54803
## 6               Danny Green  PF 15000000 26.92357
## 7  Danilo Gallinari  SF 21587579 26.21374
## 8               Stephen Curry  PG 27457154 21.61462
## 9                 Al Horford   C 28921916 20.51479
## 10               Kevin Durant  SF 20000000 20.38829
```

A table with volumes of player_name, POS, Salary and WIN_score is shown for clearing presentation and comparison among players. The original goal of the analysis is to select the best five starting players that my team can afford, which proving that they are worth it based on the provided team and player statistic. A graph of player salary against win score is then plotted for visualisation comparison. Hence, players of high win score and low salary (left top quarter of the graph) is desirable. The same plots are applied to each of the position (POS) point guard, shooting guard, small forward, power forward and centre to evaluate player selection.

```
player_labels <- if_else(pmmf$win_score >= 0, paste(pmmf$player_id), '')

pmmf %>% ggplot(aes(salary, win_score, color = Pos )) +
  geom_point()+
  geom_text(aes(label= player_labels), nudge_y = 6, nudge_x = 0.25)
```



Top 10 point guard

```
dat_pg <- filter(pmmf, Pos == "PG")
dat_pg <- subset(dat_pg, select = c('player_name', 'player_id' ,'Pos', 'salary', 'win_score'))
dat_pg %>% arrange(desc(win_score)) %>% top_n(10)
```

```
## Selecting by win_score
```

```
##          player_name player_id Pos   salary win_score
## 1       Stephen Curry       132  PG 27457154 21.614624
## 2          DJ Augustin        28  PG  7250000 11.536498
## 3          James Harden       224  PG 30570000  3.949789
## 4          Chris Paul       428  PG 29654150 -4.030314
## 5       Darren Collison       117  PG 10000000 -5.263312
## 6           Kyle Lowry       337  PG 32700000 -6.573124
## 7     Patrick Beverley        52  PG  5027028 -7.345029
## 8          Mike Conley       119  PG 30521115 -7.483797
## 9          Eric Bledsoe        59  PG 15000000 -8.564858
## 10       Damian Lillard       328  PG 27977849 -9.607731
```

Top 10 shooting guard

```
dat_sg <- filter(pmmf, Pos == "SG")
dat_sg <- subset(dat_sg, select = c('player_name', 'player_id' ,'Pos', 'salary', 'win_score'))
dat_sg %>% arrange(desc(win_score)) %>% top_n(10)
```

```
## Selecting by win_score
```

```
##       player_name player_id Pos   salary win_score
## 1       Danny Green       215  SG 15000000  24.634387
## 2          JJ Redick       452  SG 12250000  17.225017
## 3         Joe Harris       229  SG  8319444  11.408915
## 4   Malcolm Brogdon        77  SG  1548951  -3.428661
## 5     Garrett Temple       509  SG  8000000 -11.419616
## 6        Buddy Hield       244  SG  3832760 -13.207772
## 7       Klay Thompson       517  SG 18988725 -15.127503
## 8          Bryn Forbes       181  SG  3125000 -20.541323
## 9       DeMar DeRozan       165  SG 27739975 -21.252242
## 10        Eric Gordon       257  SG 15500375 -21.360775
```

Top 10 small forward

```
dat_sf <- filter(pmmf, Pos == "SF")
dat_sf <- subset(dat_sf, select = c('player_name', 'player_id' ,'Pos', 'salary', 'win_score'))
dat_sf %>% arrange(desc(win_score)) %>% top_n(10)
```

```
## Selecting by win_score
```

```
##                player_name player_id Pos   salary    win_score
## 1  Danilo Gallinari       191  SF 21587579 23.9233714612
## 2          Kevin Durant       160  SF 30000000  20.3883860614
## 3          LeBron James       278  SF 23654150  17.1148289
## 4     Bojan Bogdanovic        42  SF 10500000   7.160704877
## 5          Rashi Leonard       324  SF 2133966  6.9629719973
## 6       Nicolas Batum        39  SF 24000000  -0.0022232873
## 7           Paul George       197  SF 30560700  -2.10890413
## 8        Trevor Ariza        25  SF 15000000  -10.9835351850
## 9          Jimmy Butler        91  SF 19431427  -11.0934565611
## 10   Wesley Matthews       354  SF 19360228  -11.222775168
```

Top 10 power forward

```
dat_pf <- filter(pmmf, Pos == "PF")
dat_pf <- subset(dat_pf, select = c('player_name', 'player_id' ,'Pos', 'salary', 'win_score'))
dat_pf %>% arrange(desc(win_score)) %>% top_n(10)
```

```
## Selecting by win_score
```

```
##                player_name player_id Pos  salary  win_score
## 1                 Taj Gibson     198  PF 14000000 24.7470319
## 2                  Jeff Green     217  PF  2393887 16.7420930
## 3  Giannis Antetokounmpo      20  PF 24157304 15.7646526
## 4                 Paul Millsap     375  PF 29230769 11.6207609
## 5                  Rudy Gay     194  PF 10687200 11.3392548
## 6      Nemanja Bjelica      57  PF 4000000  4.9393397
## 7                  PJ Tucker     525  PF  7959537  2.2134325
## 8              Pascal Siakam     482  PF  1548951  1.0841736
## 9            Thaddeus Young     568  PF 10764845 -0.9554026
## 10                Joe Ingles     266  PF 12545455 -2.3034403
```

Top 10 center

```
dat_c <- filter(pmmf, Pos == "C")
dat_c <- subset(dat_c, select = c('player_name', 'player_id' ,'Pos', 'salary', 'win_score'))
dat_c %>% arrange(desc(win_score)) %>% top_n(10)
```

```
## Selecting by win_score
```

```
##          player_name player_id Pos   salary win_score
## 1       Rudy Gobert       203  C 23493573 50.588564
## 2      DeAndre Jordan       298  C 22897200 49.455932
## 3       Jarlale McGee       361  C  2393887 32.690700
## 4       Clint Capela        96  C 15793104 27.548033
## 5        Al Horford       256  C 28921916 20.514790
## 6  Lamarcus Aldridge        10  C 22347015 20.171455
## 7       Derrick Favors       175  C 14000000 14.288852
## 8       Thomas Bryant        86  C  1378242 10.727755
## 9       Dewayne Dedmon       139  C  4300000  9.456709
## 10         Brook Lopez       335  C  3382000  8.568587
```

Best five starting players

The following 5 players are selected:

1) PG- Point Guard - Stephen Curry

2) SG- Shooting Guard - Danny Green

3) SF- Small Forward - Danilo Gallinari

4) PF- Power Forward - Taj Gibson

5) C- Centre - Rudy Gobert

With total salaries of five starters is 107million, which is under our team next season player contract budget of $118 million.

```
dat_pg_start <- dat_pg %>% arrange(desc(win_score)) %>% top_n(10)%>% slice_max(1)
dat_sg_start <- dat_sg %>% arrange(desc(win_score)) %>% top_n(10)%>% slice_max(1)
dat_sf_start <- dat_sf %>% arrange(desc(win_score)) %>% top_n(10) %>% slice_max(1)
dat_pf_start <- dat_pf %>% arrange(desc(win_score)) %>% top_n(10)%>% slice_max(1)
dat_c_start <- dat_c %>% arrange(desc(win_score)) %>% top_n(10)%>% slice_max(1)

dat_start_five <- bind_rows(dat_pg_start,dat_sg_start, dat_sf_start, dat_pf_start, dat_c_start)
tibble(dat_start_five)
```

```
## # A tibble: 5 x 5
##   player_name       player_id Pos    salary win_score
##   <chr>             <dbl> <chr>     <dbl>     <dbl>
## 1 Stephen Curry       132 PG   27457154  21.6
## 2 Danny Green         215 SG   15000000  24.6
## 3 Danilo Gallinari    191 SF   21587579  23.9
## 4 Taj Gibson          198 PF   14000000  28.7
## 5 Rudy Gobert         203 C    23493573  50.6
```

total salaries of five starters

```
sep_salary <- sum(dat_start_five$salary)
print(sep_salary)
```

```
## [1] 106536206
```

```
#Check the expecting lineup salary
sep_salary <- sum(dat_start_five$salary)
if(sep_salary < 118000000)
print("Within Buget")
```

```
## [1] "Within Buget"
```

6. Summary

The specific analysis of this project is a multiple linear regression model that demonstrate relationship between winning percentage and team/player performance measure. One of the critical parts of this project is to choose which types of performance measures to be selected in order to develop the model. In this project, true shoot percentage, turn over, total rebound, Age are applied. However, the selected five players might have to a total contract cost of 106 million. Therefore, given that the rest of the team requires more budget, other players might have to be considered.

By considering model development limitation, limited set of statistics can be one of the limitations in this modelling project. Multiple years of data would be practical in building stronger relative relationship between winning percentage and other team/player performance variables. Secondly, individual player's statistic such as height, health data, injury history are also factors that should be considered when selecting the team. To conclude, 5 players have been selected based on the winning percentage model within the team specified budget. Further fine tone may be carried out by eliminating the limitation factors.