SportsAnalytics_u3246850

2023-05-07

BASKETBALL REPRODUCIBLE ANALYSIS PROJECT

1.INTRODUCTION:

Basketball is a popular sport worldwide, with millions of fans and players. In the National Basketball Association (NBA), teams compete for the championship title by playing against each other in a series of games. Each basketball team comprises five positions: point guard, shooting guard, small forward, power forward, and centre. Each position has different requirements, and players with distinct skill sets are required to form a balanced and successful team. The Chicago Bulls is an NBA team that finished the 2018-19 season in 27th place out of 30 based on win-loss record, with a player contract budget that ranked 26th out of 30 in the following season (2019-20). The team struggled to put together a competitive team and required a data-driven approach to improve their performance in the upcoming season. As a data analyst, the aim of the project is to use data analysis to identify the best five starting players for the Chicago Bulls for the upcoming NBA season. This project's justification is based on the team's performance, where the Chicago Bulls requires a team that can compete at a high level consistently. By identifying the best players for each position and optimizing the team's budget, the project aims to provide the team with actionable insights that can help improve their win-loss record and compete more effectively in the NBA. The ultimate goal is to assist the team in building a competitive and successful team that can perform well in the league. Additionally, the project will provide insights into areas where the team can improve and make data-driven decisions to maximize their chances of success. The limitations of the project include the unavailability of data on players' injuries and other factors that may impact their performance, which may affect the project's accuracy. Furthermore, the project's scope is limited to the budget allocated to player contracts, and it does not consider other factors that may impact the team's performance, such as coaching strategies and team dynamics. In summary, this project's aim is to identify the best five starting players for the Chicago Bulls for the upcoming NBA season using data analysis, taking into account player statistics from the previous season and their salaries. The project's importance lies in its potential to impact the team's performance and success, which could have far-reaching implications for the team's future. By providing the team with actionable insights, the project aims to assist the team in building a competitive and successful team that can perform well in the NBA

```
#Load required packages
# install.packages("cli")
# library(tidyverse)
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

```
library(reshape2)
```

2.Data reading and cleaning

To complete the task assigned by the general manager of the Chicago Bulls, we need to find the best five starting players (one from each position) the team can afford with the budget of $118 million. We have been provided with five datasets. These are: 2018-19_nba_player-statistics.csv: It contains total statistics for individual NBA players during the 2018-19 season. 2018-19_nba_player-salaries.csv: It contains the salary for individual players during the 2018-19 season. 2019-20_nba_team-payroll.csv: It contains payroll for all teams in the 2019-20 season. 2018-19_nba_team-statistics_1.csv: It contains team statistics for the 2018-19 season. 2018-19_nba_team-statistics_2.csv: It contains advanced team statistics for the 2018-19 season. We will use these datasets to find the best five starting players from the Chicago Bulls' and also from other team who can provide the best performance, keeping in mind the salary budget of $118 million. We will perform exploratory data analysis, data cleaning, and data merging of these datasets. We will also use data modelling, visualization tools to analyze and interpret the data to answer the question of finding the best starting players for Chicago Bulls. Finally, we will produce a reproducible report hosted on GitHub. This process is important in data analysis because it ensures that the data is accurate, consistent, and ready for further analysis. The first step is to load the data into R using the "read.csv" function. The function reads a CSV file and creates a data frame in R. In this code, four datasets are loaded - player statistics, player salaries, team payroll, and team statistics. The "check.names" argument is set to FALSE to avoid any issues with column names that contain spaces or special characters. After loading the data, the code checks for missing values in each dataset using the "colSums" function with the "is.na" function. This provides an idea of the amount and location of missing data, which may need to be imputed or removed before analysis. The "str" function is used to examine the structure of each dataset, which helps to identify the data types and the number of observations and variables. This information is useful for understanding the data and preparing it for analysis. The next step involves changing the data types of certain variables. In this code, the "as.numeric" function is used to convert the "salary" column in the team payroll dataset from character to numeric data type. This is necessary to perform mathematical operations on this column. Finally, the code cleans the data by mapping team names to their corresponding abbreviations or full names. This helps to ensure consistency across datasets, especially when merging datasets based on team names. For example, the player statistics dataset uses team names in their full form while the team payroll dataset uses team names in their abbreviated form. The code maps each abbreviated team name to its full form using the "case_when" function. Similarly, the team payroll dataset is cleaned by mapping the team names to the brief team names used in the brief. In summary, loading and cleaning data is an important step in data analysis, as it ensures the accuracy and consistency of the data. This code provides an example of how to load and clean several datasets related to NBA teams and players. By following the code, one can load and clean other datasets in R as well.

```
#*************2. Reading and cleaning the raw data*************************************
#*
#Load player statistics data
player_stats <- read.csv("2018-19_nba_player-statistics.csv", check.names = FALSE)
```

```r
#Load player salaries data
player_salaries <- read.csv("2018-19_nba_player-salaries.csv", check.names = FALSE)

#Load team payroll data
team_payroll <- read.csv("2019-20_nba_team-payroll.csv", check.names = FALSE)

#Load team statistics data
team_stats1 <- read.csv("2018-19_nba_team-statistics_1.csv", check.names = FALSE, header = TRUE)
team_stats2 <- read.csv("2018-19_nba_team-statistics_2.csv", check.names = FALSE, header = TRUE)


# Check missing value

colSums(is.na(player_stats)) # Found missing values
```

```
## player_name         Pos         Age          Tm           G          GS
##           0           0           0           0           0           0
##          MP          FG         FGA         FG%          3P         3PA
##           0           0           0           6           0           0
##         3P%          2P         2PA         2P%        eFG%          FT
##          47           0           0          15           6           0
##         FTA         FT%         ORB         DRB         TRB         AST
##           0          43           0           0           0           0
##         STL         BLK         TOV          PF         PTS
##           0           0           0           0           0
```

```r
colSums(is.na(player_salaries))
```

```
##   player_id player_name      salary
##           0           0           0
```

```r
colSums(is.na(team_payroll))
```

```
## team_id    team  salary
##       0       0       0
```

```r
colSums(is.na(team_stats1))
```

```
##      Rk    Team     Age       W       L      PW      PL     MOV     SOS     SRS    ORtg
##       0       0       0       0       0       0       0       0       0       0       0
##   DRtg    NRtg    Pace     FTr   3PAr      TS%   eFG%    TOV%    ORB% FT/FGA    DRB%
##       0       0       0       0       0       0       0       0       0       0       0
```

```r
colSums(is.na(team_stats2))
```

```
##   Rk Team    G   MP   FG  FGA  FG%   3P  3PA  3P%   2P  2PA  2P%   FT  FTA  FT%
##    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##  ORB  DRB  TRB  AST  STL  BLK  TOV   PF  PTS
##    0    0    0    0    0    0    0    0    0
```

```r
# Check structure
str(player_stats)
```

```
## 'data.frame':    708 obs. of  29 variables:
##  $ player_name: chr  "Alex Abrines" "Quincy Acy" "Jaylen Adams" "Steven Adams" ...
##  $ Pos        : chr  "SG" "PF" "PG" "C" ...
##  $ Age        : int  25 28 22 25 21 21 25 33 21 23 ...
##  $ Tm         : chr  "OKC" "PHO" "ATL" "OKC" ...
##  $ G          : int  31 10 34 80 82 19 7 81 10 38 ...
##  $ GS         : int  2 0 1 80 28 3 0 81 1 2 ...
##  $ MP         : int  588 123 428 2669 1913 194 22 2687 120 416 ...
##  $ FG         : int  56 4 38 481 280 11 3 684 13 67 ...
##  $ FGA        : int  157 18 110 809 486 36 10 1319 39 178 ...
##  $ FG%        : num  0.357 0.222 0.345 0.595 0.576 0.306 0.3 0.519 0.333 0.376 ...
##  $ 3P         : int  41 2 25 0 3 6 0 10 3 32 ...
##  $ 3PA        : int  127 15 74 2 15 23 4 42 12 99 ...
##  $ 3P%        : num  0.323 0.133 0.338 0 0.2 0.261 0 0.238 0.25 0.323 ...
##  $ 2P         : int  15 2 13 481 277 5 3 674 10 35 ...
##  $ 2PA        : int  30 3 36 807 471 13 6 1277 27 79 ...
##  $ 2P%        : num  0.5 0.667 0.361 0.596 0.588 0.385 0.5 0.528 0.37 0.443 ...
##  $ eFG%       : num  0.487 0.278 0.459 0.595 0.579 0.389 0.3 0.522 0.372 0.466 ...
##  $ FT         : int  12 7 7 146 166 4 1 349 8 45 ...
##  $ FTA        : int  13 10 9 292 226 4 2 412 12 60 ...
##  $ FT%        : num  0.923 0.7 0.778 0.5 0.735 1 0.5 0.847 0.667 0.75 ...
##  $ ORB        : int  5 3 11 391 165 3 1 251 11 3 ...
##  $ DRB        : int  43 22 49 369 432 16 3 493 15 20 ...
##  $ TRB        : int  48 25 60 760 597 19 4 744 26 23 ...
##  $ AST        : int  20 8 65 124 184 5 6 194 13 25 ...
##  $ STL        : int  17 1 14 117 71 1 2 43 1 6 ...
##  $ BLK        : int  6 4 5 76 65 4 0 107 0 6 ...
##  $ TOV        : int  14 4 28 135 121 6 2 144 8 33 ...
##  $ PF         : int  53 24 45 204 203 13 4 179 7 47 ...
##  $ PTS        : int  165 17 108 1108 729 32 7 1727 37 211 ...
```

```r
str(player_salaries)
```

```
## 'data.frame':    576 obs. of  3 variables:
##  $ player_id  : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ player_name: chr  "Alex Abrines" "Quincy Acy" "Steven Adams" "Jaylen Adams" ...
##  $ salary     : int  3667645 213948 24157304 236854 2955840 77250 5285394 77250 2000000 22347015 ...
```

```r
str(team_payroll)
```

```
## 'data.frame':    30 obs. of  3 variables:
##  $ team_id: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ team   : chr  "Miami " "Golden State " "Oklahoma City " "Toronto " ...
##  $ salary : chr  "$153,171,497 " "$146,291,276 " "$144,916,427 " "$137,793,831 " ...
```

```r
str(team_stats1)
```

```
## 'data.frame':    30 obs. of  22 variables:
```

```
##  $ Rk    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Team  : chr  "Milwaukee Bucks" "Golden State Warriors" "Toronto Raptors" "Utah Jazz" ...
##  $ Age   : num  26.9 28.4 27.3 27.3 29.2 26.2 24.9 25.7 25.7 27 ...
##  $ W     : int  60 57 58 50 53 53 54 49 49 48 ...
##  $ L     : int  22 25 24 32 29 29 28 33 33 34 ...
##  $ PW    : int  61 56 56 54 53 51 51 52 50 50 ...
##  $ PL    : int  21 26 26 28 29 31 31 30 32 32 ...
##  $ MOV   : num  8.87 6.46 6.09 5.26 4.77 4.2 3.95 4.44 3.4 3.33 ...
##  $ SOS   : num  -0.82 -0.04 -0.6 0.03 0.19 0.24 0.24 -0.54 0.15 -0.57 ...
##  $ SRS   : num  8.04 6.42 5.49 5.28 4.96 4.43 4.19 3.9 3.56 2.76 ...
##  $ ORtg  : num  114 116 113 111 116 ...
##  $ DRtg  : num  105 110 107 106 111 ...
##  $ NRtg  : num  8.6 6.4 6 5.2 4.8 4.2 4.1 4.4 3.3 3.4 ...
##  $ Pace  : num  103.3 100.9 100.2 100.3 97.9 ...
##  $ FTr   : num  0.255 0.227 0.247 0.295 0.279 0.258 0.232 0.215 0.266 0.242 ...
##  $ 3PAr  : num  0.419 0.384 0.379 0.394 0.519 0.339 0.348 0.381 0.347 0.292 ...
##  $ TS%   : num  0.583 0.596 0.579 0.572 0.581 0.568 0.558 0.567 0.545 0.561 ...
##  $ eFG%  : num  0.55 0.565 0.543 0.538 0.542 0.528 0.527 0.534 0.514 0.53 ...
##  $ TOV%  : num  12 12.6 12.4 13.4 12 12.1 11.9 11.5 11.7 12.4 ...
##  $ ORB%  : num  20.8 22.5 21.9 22.9 22.8 26.6 26.6 21.6 26 21.9 ...
##  $ FT/FGA: num  0.197 0.182 0.198 0.217 0.221 0.21 0.175 0.173 0.19 0.182 ...
##  $ DRB%  : num  80.3 77.1 77.1 80.3 74.4 77.9 78 77 78.2 76.2 ...
```

```
str(team_stats2)
```

```
## 'data.frame':    30 obs. of  25 variables:
##  $ Rk  : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Team: chr  "Milwaukee Bucks" "Golden State Warriors" "New Orleans Pelicans" "Philadelphia 76ers"
##  $ G   : int  82 82 82 82 82 82 82 82 82 82 ...
##  $ MP  : int  19780 19805 19755 19805 19830 19855 19855 19880 19730 19930 ...
##  $ FG  : int  3555 3612 3581 3407 3384 3470 3497 3460 3541 3456 ...
##  $ FGA : int  7471 7361 7563 7233 7178 7427 7706 7305 7637 7387 ...
##  $ FG% : num  0.476 0.491 0.473 0.471 0.471 0.467 0.454 0.474 0.464 0.468 ...
##  $ 3P  : int  1105 1087 842 889 821 904 932 1015 927 930 ...
##  $ 3PA : int  3134 2824 2449 2474 2118 2520 2677 2771 2455 2731 ...
##  $ 3P% : num  0.353 0.385 0.344 0.359 0.388 0.359 0.348 0.366 0.378 0.341 ...
##  $ 2P  : int  2450 2525 2739 2518 2563 2566 2565 2445 2614 2526 ...
##  $ 2PA : int  4337 4537 5114 4759 5060 4907 5029 4534 5182 4656 ...
##  $ 2P% : num  0.565 0.557 0.536 0.529 0.507 0.523 0.51 0.539 0.504 0.543 ...
##  $ FT  : int  1471 1339 1462 1742 1853 1558 1461 1449 1354 1508 ...
##  $ FTA : int  1904 1672 1921 2258 2340 1914 2049 1803 1865 1963 ...
##  $ FT% : num  0.773 0.801 0.761 0.771 0.792 0.814 0.713 0.804 0.726 0.768 ...
##  $ ORB : int  762 797 909 892 796 967 1031 786 906 794 ...
##  $ DRB : int  3316 2990 2969 3025 2936 2968 2911 2920 2819 2679 ...
##  $ TRB : int  4078 3787 3878 3917 3732 3935 3942 3706 3725 3473 ...
##  $ AST : int  2136 2413 2216 2207 1970 1887 1917 2085 2083 2154 ...
##  $ STL : int  615 625 610 606 561 546 766 680 679 683 ...
##  $ BLK : int  486 525 441 432 385 413 425 437 363 379 ...
##  $ TOV : int  1137 1169 1215 1223 1193 1135 1145 1150 1095 1154 ...
##  $ PF  : int  1608 1757 1732 1745 1913 1669 1839 1724 1751 1701 ...
##  $ PTS : int  9686 9650 9466 9445 9442 9402 9387 9384 9363 9350 ...
```

```r
# Changing the variable type for the analysis
team_payroll$salary <- as.numeric(gsub("[\\$,]", "", team_payroll$salary))

# Cleaning player stats team name mapping to abbreviation
player_stats <- player_stats%>%
  mutate(Tm = case_when(
    Tm == "ATL" ~ "Atlanta Hawks",
    Tm == "BOS" ~ "Boston Celtics",
    Tm == "BRK" ~ "Brooklyn Nets",
    Tm == "CHI" ~ "Chicago Bulls",
    Tm == "CHO" ~ "Charlotte Hornets",
    Tm == "CLE" ~ "Cleveland Cavaliers",
    Tm == "DAL" ~ "Dallas Mavericks",
    Tm == "DEN" ~ "Denver Nuggets",
    Tm == "DET" ~ "Detroit Pistons",
    Tm == "GSW" ~ "Golden State Warriors",
    Tm == "HOU" ~ "Houston Rockets",
    Tm == "IND" ~ "Indiana Pacers",
    Tm == "LAC" ~ "Los Angeles Clippers",
    Tm == "LAL" ~ "Los Angeles Lakers",
    Tm == "MEM" ~ "Memphis Grizzlies",
    Tm == "MIA" ~ "Miami Heat",
    Tm == "MIL" ~ "Milwaukee Bucks",
    Tm == "MIN" ~ "Minnesota Timberwolves",
    Tm == "NOP" ~ "New Orleans Pelicans",
    Tm == "NYK" ~ "New York Knicks",
    Tm == "OKC" ~ "Oklahoma City Thunder",
    Tm == "ORL" ~ "Orlando Magic",
    Tm == "PHI" ~ "Philadelphia 76ers",
    Tm == "PHO" ~ "Phoenix Suns",
    Tm == "POR" ~ "Portland Trail Blazers",
    Tm == "SAC" ~ "Sacramento Kings",
    Tm == "SAS" ~ "San Antonio Spurs",
    Tm == "TOR" ~ "Toronto Raptors",
    Tm == "TOT" ~ "Total",
    Tm == "UTA" ~ "Utah Jazz",
    Tm == "WAS" ~ "Washington Wizards",
    TRUE ~ NA_character_
  ))

# Cleaning team payroll data to align with the brief team name
team_payroll$team<-trimws(team_payroll$team)
team_payroll <- team_payroll %>%
  mutate(Team = case_when(
    team == "Atlanta" ~ "Atlanta Hawks",
    team == "Boston" ~ "Boston Celtics",
    team == "Brooklyn" ~ "Brooklyn Nets",
    team == "Chicago" ~ "Chicago Bulls",
    team == "Charlotte" ~ "Charlotte Hornets",
    team == "Cleveland" ~ "Cleveland Cavaliers",
    team == "Dallas" ~ "Dallas Mavericks",
    team == "Denver" ~ "Denver Nuggets",
    team == "Detroit" ~ "Detroit Pistons",
```

```
    team == "Golden State" ~ "Golden State Warriors",
    team == "Houston" ~ "Houston Rockets",
    team == "Indiana" ~ "Indiana Pacers",
    team == "LA Clippers" ~ "Los Angeles Clippers",
    team == "LA Lakers" ~ "Los Angeles Lakers",
    team == "Memphis" ~ "Memphis Grizzlies",
    team == "Miami" ~ "Miami Heat",
    team == "Milwaukee" ~ "Milwaukee Bucks",
    team == "Minnesota" ~ "Minnesota Timberwolves",
    team == "New Orleans" ~ "New Orleans Pelicans",
    team == "New York" ~ "New York Knicks",
    team == "Oklahoma City" ~ "Oklahoma City Thunder",
    team == "Orlando" ~ "Orlando Magic",
    team == "Philadelphia" ~ "Philadelphia 76ers",
    team == "Phoenix" ~ "Phoenix Suns",
    team == "Portland" ~ "Portland Trail Blazers",
    team == "Sacramento" ~ "Sacramento Kings",
    team == "San Antonio" ~ "San Antonio Spurs",
    team == "Toronto" ~ "Toronto Raptors",
    team == "Utah" ~ "Utah Jazz",
    team == "Washington" ~ "Washington Wizards",
    TRUE ~ NA_character_
  ))
```

3.Exploratory data analysis:

Exploratory data analysis (EDA) is a crucial step in data analysis that allows data scientists to gain a deeper understanding of the data and identify any potential issues that need to be addressed before moving on to more complex analysis. In this process, the data is examined in a variety of ways to identify patterns, relationships, and anomalies. First, it reads in five data frames: player_stats, player_salaries, team_payroll, team_stats1, and team_stats2. The function glimpse() is then used to view the structure of each data frame. glimpse() is a useful function for quickly viewing the structure of a data frame, including the variable names, their data type, and the first few observations. Next, the distribution of variables is checked using summary statistics and visualizations. For the player_stats data frame, the distribution of player age and points are visualized using histograms. For the player_salaries and team_payroll data frames, the distribution of salaries is visualized using histograms with a logarithmic scale on the y-axis to better view the distribution. For team_stats1 and team_stats2 data frames, the age and point distributions are visualized respectively using histograms. Next, it checks for relationships between variables or differences between groups. Firstly, the player_stats and player_salaries data frames are merged using the join() function. The complete.cases() function is then used to remove any rows with null values in the player_id column. A scatter plot is then created to understand the relationship between salary and points across players, which shows a positive correlation between the two variables. To further explore relationships between variables, a correlation matrix is calculated for the numeric variables in the player_stats_salaries data frame. The cor() function is used to calculate the correlation matrix, and the melt() function from the reshape2 package is used to convert the matrix into a long-format data frame that can be used to create a heatmap with ggplot2. Finally, a heatmap is created using ggplot2 to visualize the correlation matrix. It merges the player_stats_salaries and team_stats_salary data frames using the full_join() function, which combines the data frames based on a common variable, in this case, the team name. The merged data frame can be used for further analysis. In summary, it demonstrates various data exploration and manipulation techniques, including data structure inspection, variable distribution visualization, scatter plots, correlation matrix calculation, and data frame merging. These techniques are useful for understanding the data, identifying patterns, and gaining insights that can inform further analysis. The decision to develop a data model that is easily reproducible is crucial

for this task as it allows us to modify and update our analysis in the future. As the data analyst for the Chicago Bulls, we need to ensure that we can make informed decisions when selecting players for the team, and a model that is flexible and scalable is essential for achieving this. By capturing relevant data on player statistics, such as performance, age, height, weight, and previous experience, we can ensure that our data model is comprehensive and informative. By incorporating statistical methods, such as regression analysis or clustering, we can identify the best five starting players for each position for the Chicago Bulls. Furthermore, by ensuring that our data model is reproducible, we can easily update it with new data as it becomes available. This means that we can continually assess player performance and make informed decisions based on the latest data. In the future, we can incorporate data from the current NBA season or include data from other sources, such as scouting reports or expert opinions, to improve the accuracy of our model. In conclusion, developing a reproducible data model that captures relevant player statistics and incorporates appropriate statistical methods is essential for identifying the best five starting players for each position for the Chicago Bulls. By creating a model that is flexible and scalable, we can update and modify it in the future to make informed decisions based on the latest data.

```
# ********************** 3. Exploratory analysis ******************************

#****************** 3a checking for errors and missing values within the datasets******************

# Check structure
glimpse(player_stats)
```

```
## Rows: 708
## Columns: 29
## $ player_name <chr> "Alex Abrines", "Quincy Acy", "Jaylen Adams", "Steven Adam~
## $ Pos         <chr> "SG", "PF", "PG", "C", "C", "SF", "SG", "C", "SG", "SG", "~
## $ Age         <int> 25, 28, 22, 25, 21, 21, 25, 33, 21, 23, 20, 26, 28, 25, 25~
## $ Tm          <chr> "Oklahoma City Thunder", "Phoenix Suns", "Atlanta Hawks", ~
## $ G           <int> 31, 10, 34, 80, 82, 19, 7, 81, 10, 38, 80, 19, 81, 48, 43,~
## $ GS          <int> 2, 0, 1, 80, 28, 3, 0, 81, 1, 2, 80, 1, 81, 4, 40, 0, 8, 8~
## $ MP          <int> 588, 123, 428, 2669, 1913, 194, 22, 2687, 120, 416, 2096, ~
## $ FG          <int> 56, 4, 38, 481, 280, 11, 3, 684, 13, 67, 335, 65, 257, 64,~
## $ FGA         <int> 157, 18, 110, 809, 486, 36, 10, 1319, 39, 178, 568, 141, 5~
## $ `FG%`       <dbl> 0.357, 0.222, 0.345, 0.595, 0.576, 0.306, 0.300, 0.519, 0.~
## $ `3P`        <int> 41, 2, 25, 0, 3, 6, 0, 10, 3, 32, 6, 17, 96, 24, 9, 2, 7, ~
## $ `3PA`       <int> 127, 15, 74, 2, 15, 23, 4, 42, 12, 99, 45, 36, 280, 77, 34~
## $ `3P%`       <dbl> 0.323, 0.133, 0.338, 0.000, 0.200, 0.261, 0.000, 0.238, 0.~
## $ `2P`        <int> 15, 2, 13, 481, 277, 5, 3, 674, 10, 35, 329, 48, 161, 40, ~
## $ `2PA`       <int> 30, 3, 36, 807, 471, 13, 6, 1277, 27, 79, 523, 105, 313, 8~
## $ `2P%`       <dbl> 0.500, 0.667, 0.361, 0.596, 0.588, 0.385, 0.500, 0.528, 0.~
## $ `eFG%`      <dbl> 0.487, 0.278, 0.459, 0.595, 0.579, 0.389, 0.300, 0.522, 0.~
## $ FT          <int> 12, 7, 7, 146, 166, 4, 1, 349, 8, 45, 197, 42, 150, 26, 37~
## $ FTA         <int> 13, 10, 9, 292, 226, 4, 2, 412, 12, 60, 278, 54, 173, 35, ~
## $ `FT%`       <dbl> 0.923, 0.700, 0.778, 0.500, 0.735, 1.000, 0.500, 0.847, 0.~
## $ ORB         <int> 5, 3, 11, 391, 165, 3, 1, 251, 11, 3, 191, 8, 112, 24, 48,~
## $ DRB         <int> 43, 22, 49, 369, 432, 16, 3, 493, 15, 20, 481, 43, 498, 60~
## $ TRB         <int> 48, 25, 60, 760, 597, 19, 4, 744, 26, 23, 672, 51, 610, 84~
## $ AST         <int> 20, 8, 65, 124, 184, 5, 6, 194, 13, 25, 110, 76, 104, 23, ~
## $ STL         <int> 17, 1, 14, 117, 71, 1, 2, 43, 1, 6, 43, 16, 68, 22, 54, 1,~
## $ BLK         <int> 6, 4, 5, 76, 65, 4, 0, 107, 0, 6, 120, 4, 33, 13, 37, 0, 1~
## $ TOV         <int> 14, 4, 28, 135, 121, 6, 2, 144, 8, 33, 103, 26, 72, 23, 58~
## $ PF          <int> 53, 24, 45, 204, 203, 13, 4, 179, 7, 47, 184, 46, 143, 48,~
## $ PTS         <int> 165, 17, 108, 1108, 729, 32, 7, 1727, 37, 211, 873, 189, 7~
```

```
glimpse(player_salaries)
```

```
## Rows: 576
## Columns: 3
## $ player_id   <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,~
## $ player_name <chr> "Alex Abrines", "Quincy Acy", "Steven Adams", "Jaylen Adam~
## $ salary      <int> 3667645, 213948, 24157304, 236854, 2955840, 77250, 5285394~
```

```
glimpse(team_payroll)
```

```
## Rows: 30
## Columns: 4
## $ team_id <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,~
## $ team    <chr> "Miami", "Golden State", "Oklahoma City", "Toronto", "Milwauke~
## $ salary  <dbl> 153171497, 146291276, 144916427, 137793831, 130988604, 1302566~
## $ Team    <chr> "Miami Heat", "Golden State Warriors", "Oklahoma City Thunder"~
```

```
glimpse(team_stats1)
```

```
## Rows: 30
## Columns: 22
## $ Rk       <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18~
## $ Team     <chr> "Milwaukee Bucks", "Golden State Warriors", "Toronto Raptors"~
## $ Age      <dbl> 26.9, 28.4, 27.3, 27.3, 29.2, 26.2, 24.9, 25.7, 25.7, 27.0, 2~
## $ W        <int> 60, 57, 58, 50, 53, 53, 54, 49, 49, 48, 51, 48, 48, 42, 42, 3~
## $ L        <int> 22, 25, 24, 32, 29, 29, 28, 33, 33, 34, 31, 34, 34, 40, 40, 4~
## $ PW       <int> 61, 56, 56, 54, 53, 51, 51, 52, 50, 50, 48, 45, 43, 43, 41, 4~
## $ PL       <int> 21, 26, 26, 28, 29, 31, 31, 30, 32, 32, 34, 37, 39, 39, 41, 4~
## $ MOV      <dbl> 8.87, 6.46, 6.09, 5.26, 4.77, 4.20, 3.95, 4.44, 3.40, 3.33, 2~
## $ SOS      <dbl> -0.82, -0.04, -0.60, 0.03, 0.19, 0.24, 0.24, -0.54, 0.15, -0.~
## $ SRS      <dbl> 8.04, 6.42, 5.49, 5.28, 4.96, 4.43, 4.19, 3.90, 3.56, 2.76, 2~
## $ ORtg     <dbl> 113.8, 115.9, 113.1, 110.9, 115.5, 114.7, 113.0, 112.2, 110.3~
## $ DRtg     <dbl> 105.2, 109.5, 107.1, 105.7, 110.7, 110.5, 108.9, 107.8, 107.0~
## $ NRtg     <dbl> 8.6, 6.4, 6.0, 5.2, 4.8, 4.2, 4.1, 4.4, 3.3, 3.4, 2.6, 1.7, 0~
## $ Pace     <dbl> 103.3, 100.9, 100.2, 100.3, 97.9, 99.1, 97.7, 99.6, 102.8, 98~
## $ FTr      <dbl> 0.255, 0.227, 0.247, 0.295, 0.279, 0.258, 0.232, 0.215, 0.266~
## $ `3PAr`   <dbl> 0.419, 0.384, 0.379, 0.394, 0.519, 0.339, 0.348, 0.381, 0.347~
## $ `TS%`    <dbl> 0.583, 0.596, 0.579, 0.572, 0.581, 0.568, 0.558, 0.567, 0.545~
## $ `eFG%`   <dbl> 0.550, 0.565, 0.543, 0.538, 0.542, 0.528, 0.527, 0.534, 0.514~
## $ `TOV%`   <dbl> 12.0, 12.6, 12.4, 13.4, 12.0, 12.1, 11.9, 11.5, 11.7, 12.4, 1~
## $ `ORB%`   <dbl> 20.8, 22.5, 21.9, 22.9, 22.8, 26.6, 26.6, 21.6, 26.0, 21.9, 2~
## $ `FT/FGA` <dbl> 0.197, 0.182, 0.198, 0.217, 0.221, 0.210, 0.175, 0.173, 0.190~
## $ `DRB%`   <dbl> 80.3, 77.1, 77.1, 80.3, 74.4, 77.9, 78.0, 77.0, 78.2, 76.2, 7~
```

```
glimpse(team_stats2)
```

```
## Rows: 30
## Columns: 25
## $ Rk    <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1~
## $ Team  <chr> "Milwaukee Bucks", "Golden State Warriors", "New Orleans Pelican~
## $ G     <int> 82, 82, 82, 82, 82, 82, 82, 82, 82, 82, 82, 82, 82, 82, 82, 82, ~
```

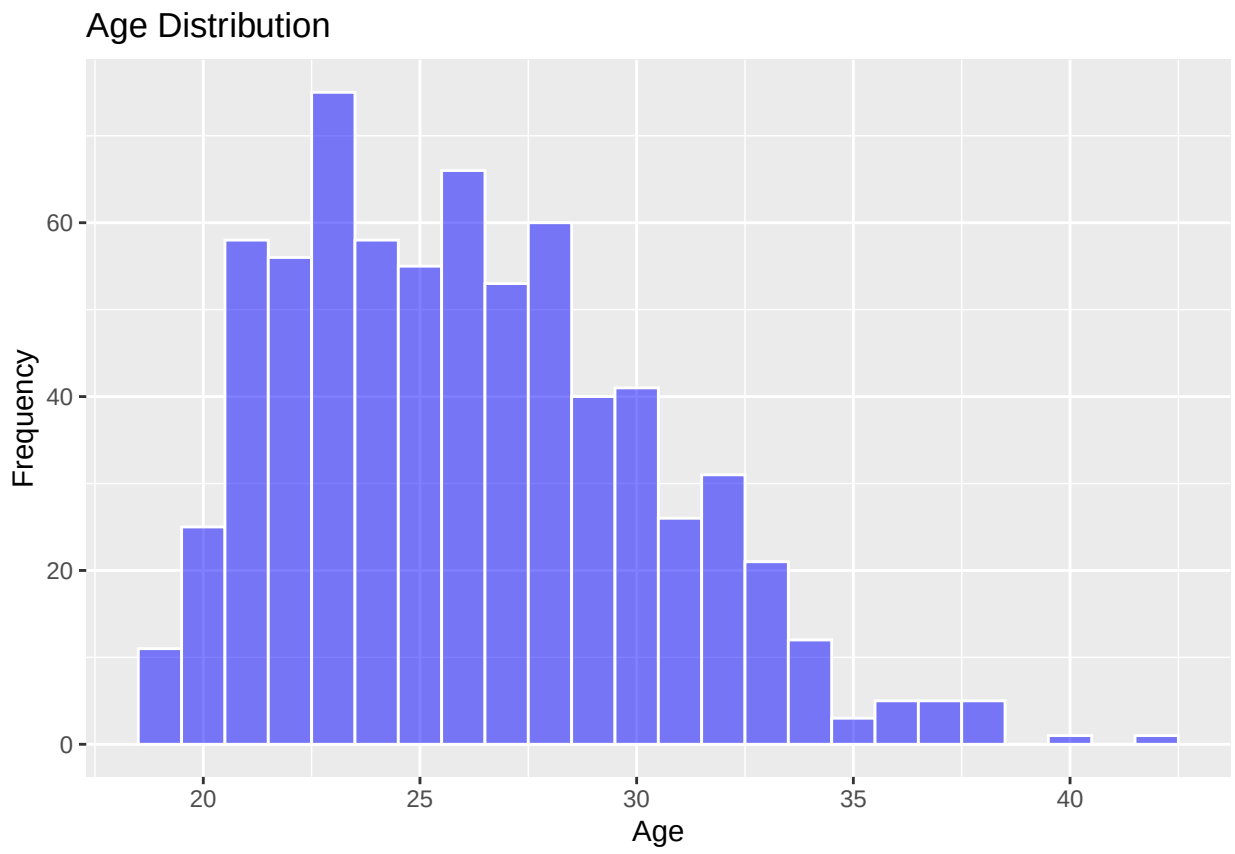
```
## $ MP    <int> 19780, 19805, 19755, 19805, 19830, 19855, 19855, 19880, 19730, 1~
## $ FG    <int> 3555, 3612, 3581, 3407, 3384, 3470, 3497, 3460, 3541, 3456, 3218~
## $ FGA   <int> 7471, 7361, 7563, 7233, 7178, 7427, 7706, 7305, 7637, 7387, 7163~
## $ `FG%` <dbl> 0.476, 0.491, 0.473, 0.471, 0.471, 0.467, 0.454, 0.474, 0.464, 0~
## $ `3P`  <int> 1105, 1087, 842, 889, 821, 904, 932, 1015, 927, 930, 1323, 1067,~
## $ `3PA` <int> 3134, 2824, 2449, 2474, 2118, 2520, 2677, 2771, 2455, 2731, 3721~
## $ `3P%` <dbl> 0.353, 0.385, 0.344, 0.359, 0.388, 0.359, 0.348, 0.366, 0.378, 0~
## $ `2P`  <int> 2450, 2525, 2739, 2518, 2563, 2566, 2565, 2445, 2614, 2526, 1895~
## $ `2PA` <int> 4337, 4537, 5114, 4759, 5060, 4907, 5029, 4534, 5182, 4656, 3442~
## $ `2P%` <dbl> 0.565, 0.557, 0.536, 0.529, 0.507, 0.523, 0.510, 0.539, 0.504, 0~
## $ FT    <int> 1471, 1339, 1462, 1742, 1853, 1558, 1461, 1449, 1354, 1508, 1582~
## $ FTA   <int> 1904, 1672, 1921, 2258, 2340, 1914, 2049, 1803, 1865, 1963, 2001~
## $ `FT%` <dbl> 0.773, 0.801, 0.761, 0.771, 0.792, 0.814, 0.713, 0.804, 0.726, 0~
## $ ORB   <int> 762, 797, 909, 892, 796, 967, 1031, 786, 906, 794, 836, 955, 923~
## $ DRB   <int> 3316, 2990, 2969, 3025, 2936, 2968, 2911, 2920, 2819, 2679, 2613~
## $ TRB   <int> 4078, 3787, 3878, 3917, 3732, 3935, 3942, 3706, 3725, 3473, 3449~
## $ AST   <int> 2136, 2413, 2216, 2207, 1970, 1887, 1917, 2085, 2083, 2154, 1741~
## $ STL   <int> 615, 625, 610, 606, 561, 546, 766, 680, 679, 683, 700, 675, 683,~
## $ BLK   <int> 486, 525, 441, 432, 385, 413, 425, 437, 363, 379, 405, 419, 411,~
## $ TOV   <int> 1137, 1169, 1215, 1223, 1193, 1135, 1145, 1150, 1095, 1154, 1094~
## $ PF    <int> 1608, 1757, 1732, 1745, 1913, 1669, 1839, 1724, 1751, 1701, 1803~
## $ PTS   <int> 9686, 9650, 9466, 9445, 9442, 9402, 9387, 9384, 9363, 9350, 9341~
```

```
##**********3b Checking distribution of variable*****************************888
#using  summary statistics and visualizations by histogram for different data frames: player_stats, pla

# Summary of player
summary(player_stats)
```

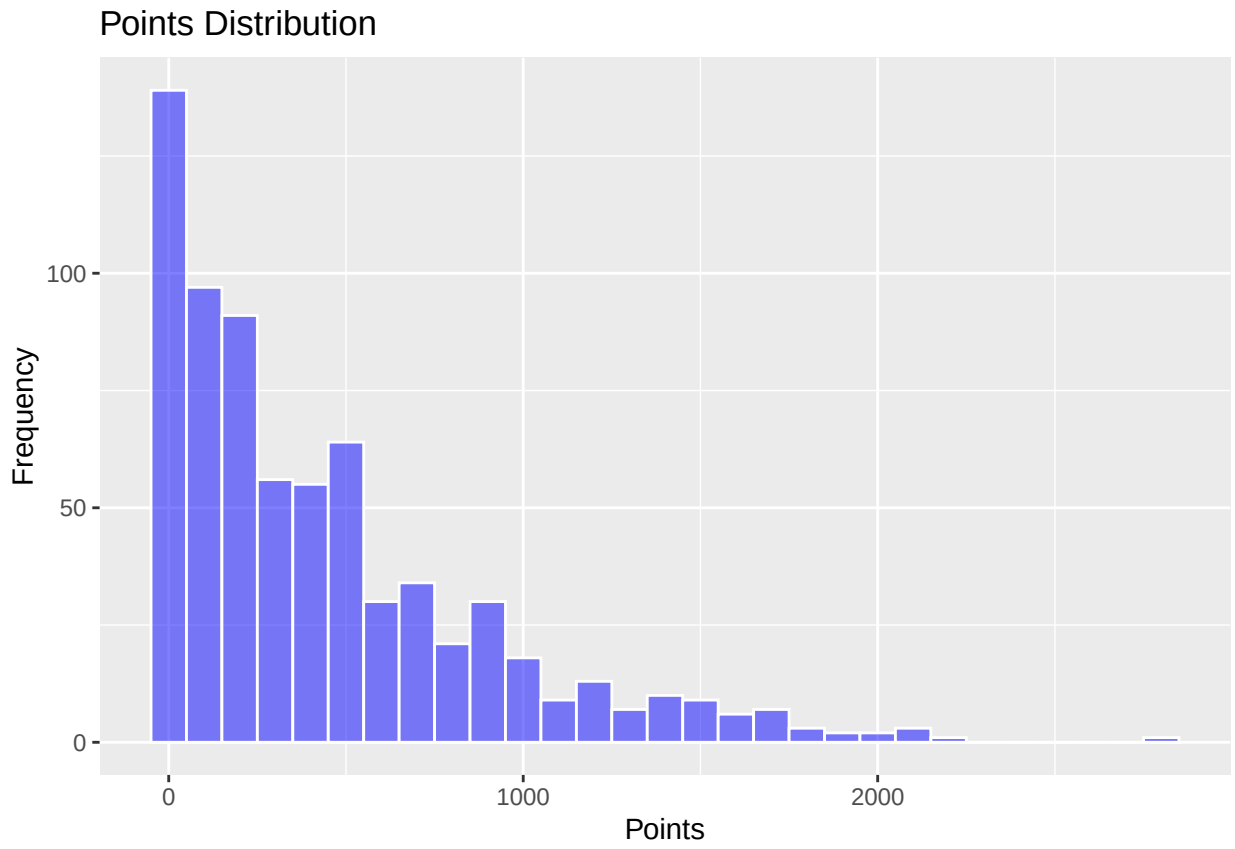
```
##  player_name            Pos                 Age             Tm
##  Length:708         Length:708         Min.   :19.00   Length:708
##  Class :character   Class :character   1st Qu.:23.00   Class :character
##  Mode  :character   Mode  :character   Median :26.00   Mode  :character
##                                        Mean   :26.14
##                                        3rd Qu.:29.00
##                                        Max.   :42.00
##
##        G               GS              MP               FG
##  Min.   : 1.00   Min.   : 0.00   Min.   :   1.0   Min.   :  0.0
##  1st Qu.:19.00   1st Qu.: 0.00   1st Qu.: 245.2   1st Qu.: 32.0
##  Median :44.00   Median : 6.00   Median : 788.0   Median :108.5
##  Mean   :42.88   Mean   :19.85   Mean   : 972.3   Mean   :162.6
##  3rd Qu.:68.00   3rd Qu.:32.00   3rd Qu.:1579.5   3rd Qu.:236.2
##  Max.   :82.00   Max.   :82.00   Max.   :3028.0   Max.   :843.0
##
##       FGA              FG%               3P              3PA
##  Min.   :   0.00   Min.   :0.0000   Min.   :  0.00   Min.   :   0.0
##  1st Qu.:  72.75   1st Qu.:0.4000   1st Qu.:  4.00   1st Qu.:  13.0
##  Median : 256.00   Median :0.4340   Median : 26.00   Median :  79.0
##  Mean   : 355.42   Mean   :0.4373   Mean   : 46.12   Mean   : 130.1
##  3rd Qu.: 526.00   3rd Qu.:0.4850   3rd Qu.: 69.25   3rd Qu.: 200.0
##  Max.   :1909.00   Max.   :1.0000   Max.   :378.00   Max.   :1028.0
##                    NA's   :6
##       3P%              2P             2PA              2P%
```

```
##  Min.   :0.000   Min.   :  0.0   Min.   :   0.0   Min.   :0.0000
##  1st Qu.:0.286   1st Qu.: 18.0   1st Qu.:  40.0   1st Qu.:0.4500
##  Median :0.335   Median : 71.0   Median : 138.0   Median :0.5000
##  Mean   :0.315   Mean   :116.5   Mean   : 225.3   Mean   :0.4923
##  3rd Qu.:0.372   3rd Qu.:164.2   3rd Qu.: 314.5   3rd Qu.:0.5480
##  Max.   :1.000   Max.   :674.0   Max.   :1277.0   Max.   :1.0000
##  NA's   :47                                       NA's   :15
##       eFG%             FT              FTA             FT%
##  Min.   :0.0000   Min.   :  0.00   Min.   :  0.00   Min.   :0.0000
##  1st Qu.:0.4700   1st Qu.: 11.00   1st Qu.: 15.00   1st Qu.:0.6840
##  Median :0.5080   Median : 39.00   Median : 51.00   Median :0.7630
##  Mean   :0.5002   Mean   : 69.88   Mean   : 91.01   Mean   :0.7396
##  3rd Qu.:0.5517   3rd Qu.: 94.00   3rd Qu.:123.00   3rd Qu.:0.8250
##  Max.   :1.5000   Max.   :754.00   Max.   :858.00   Max.   :1.0000
##  NA's   :6                                         NA's   :43
##       ORB              DRB              TRB              AST
##  Min.   :  0.00   Min.   :  0.0   Min.   :   0.00   Min.   :  0.00
##  1st Qu.:  7.00   1st Qu.: 32.0   1st Qu.:  41.75   1st Qu.: 16.00
##  Median : 23.00   Median :102.5   Median : 128.50   Median : 56.00
##  Mean   : 41.01   Mean   :139.1   Mean   : 180.12   Mean   : 96.32
##  3rd Qu.: 54.00   3rd Qu.:199.0   3rd Qu.: 258.00   3rd Qu.:124.25
##  Max.   :423.00   Max.   :809.0   Max.   :1232.00   Max.   :784.00
##
##       STL              BLK              TOV              PF
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   :  0.0
##  1st Qu.:  7.00   1st Qu.:  3.00   1st Qu.: 11.00   1st Qu.: 24.0
##  Median : 21.00   Median : 10.00   Median : 36.00   Median : 73.5
##  Mean   : 30.58   Mean   : 19.29   Mean   : 53.52   Mean   : 84.1
##  3rd Qu.: 46.00   3rd Qu.: 25.00   3rd Qu.: 75.00   3rd Qu.:131.2
##  Max.   :170.00   Max.   :199.00   Max.   :387.00   Max.   :292.0
##
##       PTS
##  Min.   :   0.00
##  1st Qu.:  82.75
##  Median : 294.00
##  Mean   : 441.29
##  3rd Qu.: 634.00
##  Max.   :2818.00
##
```

```r
# Distributon of player age
ggplot(player_stats, aes(x = Age)) +
  geom_histogram(binwidth = 1, color = "white", fill = "blue", alpha = 0.5) +
  labs(title = "Age Distribution", x = "Age", y = "Frequency")
```



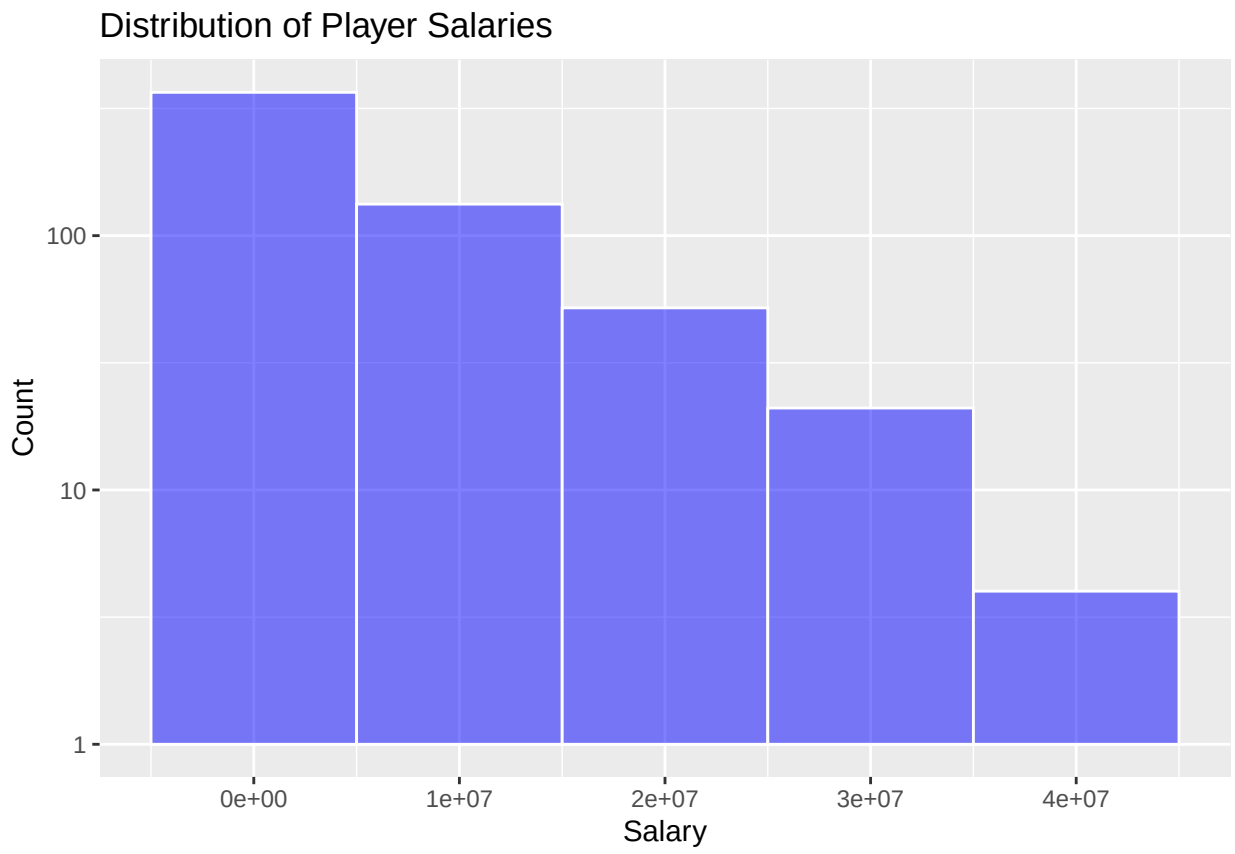

```r
# Distribution of player points
ggplot(player_stats, aes(x = PTS)) +
  geom_histogram(binwidth = 100, color = "white", fill = "blue", alpha = 0.5) +
  labs(title = "Points Distribution", x = "Points", y = "Frequency")
```




```
# Summary of player salary
summary(player_salaries)
```

```
##    player_id     player_name           salary
##  Min.   :  1.0   Length:576         Min.   :   47370
##  1st Qu.:144.8   Class :character   1st Qu.: 1349383
##  Median :288.5   Mode  :character   Median : 2530560
##  Mean   :288.5                      Mean   : 6258149
##  3rd Qu.:432.2                      3rd Qu.: 9000000
##  Max.   :576.0                      Max.   :37457154
```

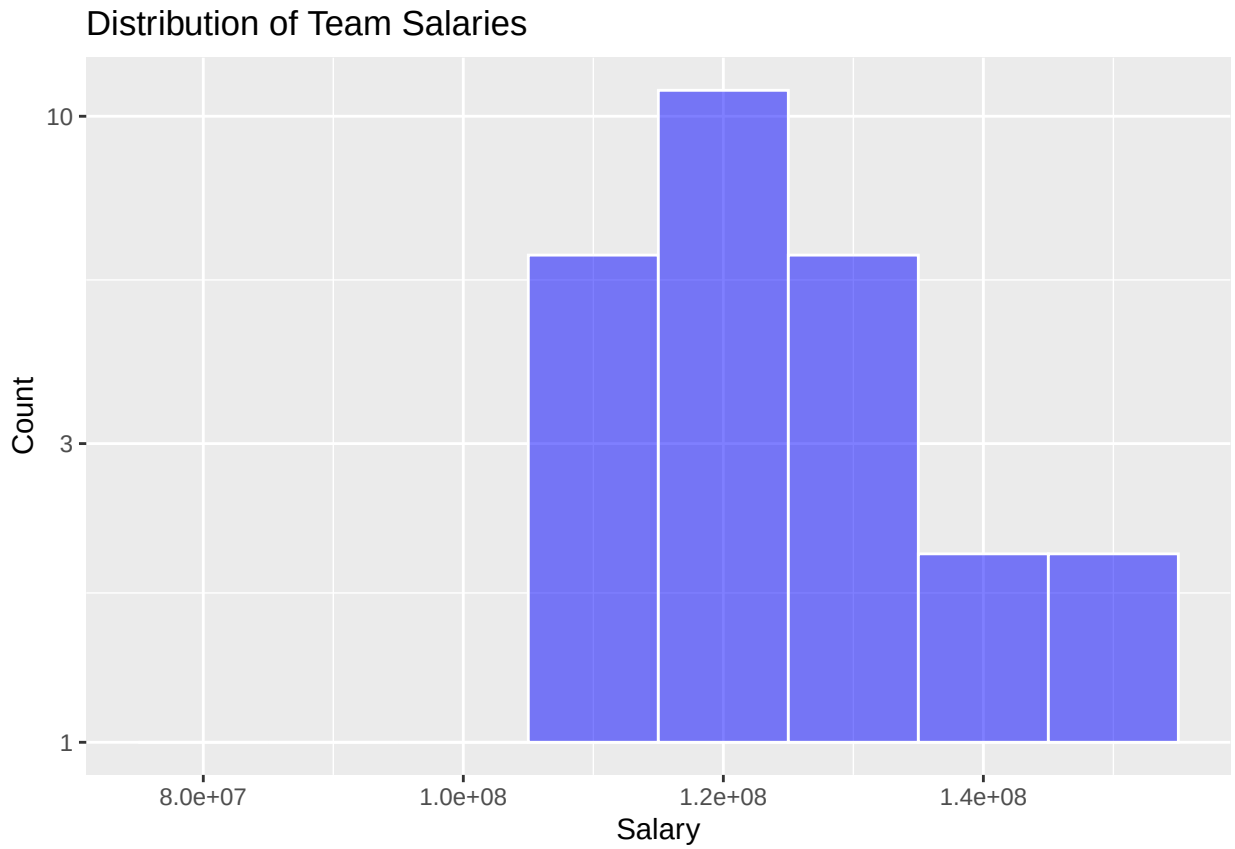
```
# Distribution of player Salary
ggplot(player_salaries, aes(x = salary)) +
  geom_histogram(binwidth = 10000000, color = "white", fill = "blue", alpha = 0.5) +
  scale_y_log10() +
  labs(title = "Distribution of Player Salaries",
       x = "Salary",
       y = "Count")
```




```
# Summary of team salary
summary(team_payroll)
```

```
##     team_id          team               salary              Team
##  Min.   : 1.00   Length:30          Min.   : 79180081   Length:30
##  1st Qu.: 8.25   Class :character   1st Qu.:113968170   Class :character
##  Median :15.50   Mode  :character   Median :121508324   Mode  :character
##  Mean   :15.50                      Mean   :120157121
##  3rd Qu.:22.75                      3rd Qu.:126382440
##  Max.   :30.00                      Max.   :153171497
```

```
# Distribution of team Salary
ggplot(team_payroll, aes(x = salary)) +
  geom_histogram(binwidth = 10000000, color = "white", fill = "blue", alpha = 0.5) +
  scale_y_log10() +
  labs(title = "Distribution of Team Salaries",
       x = "Salary",
       y = "Count")
```

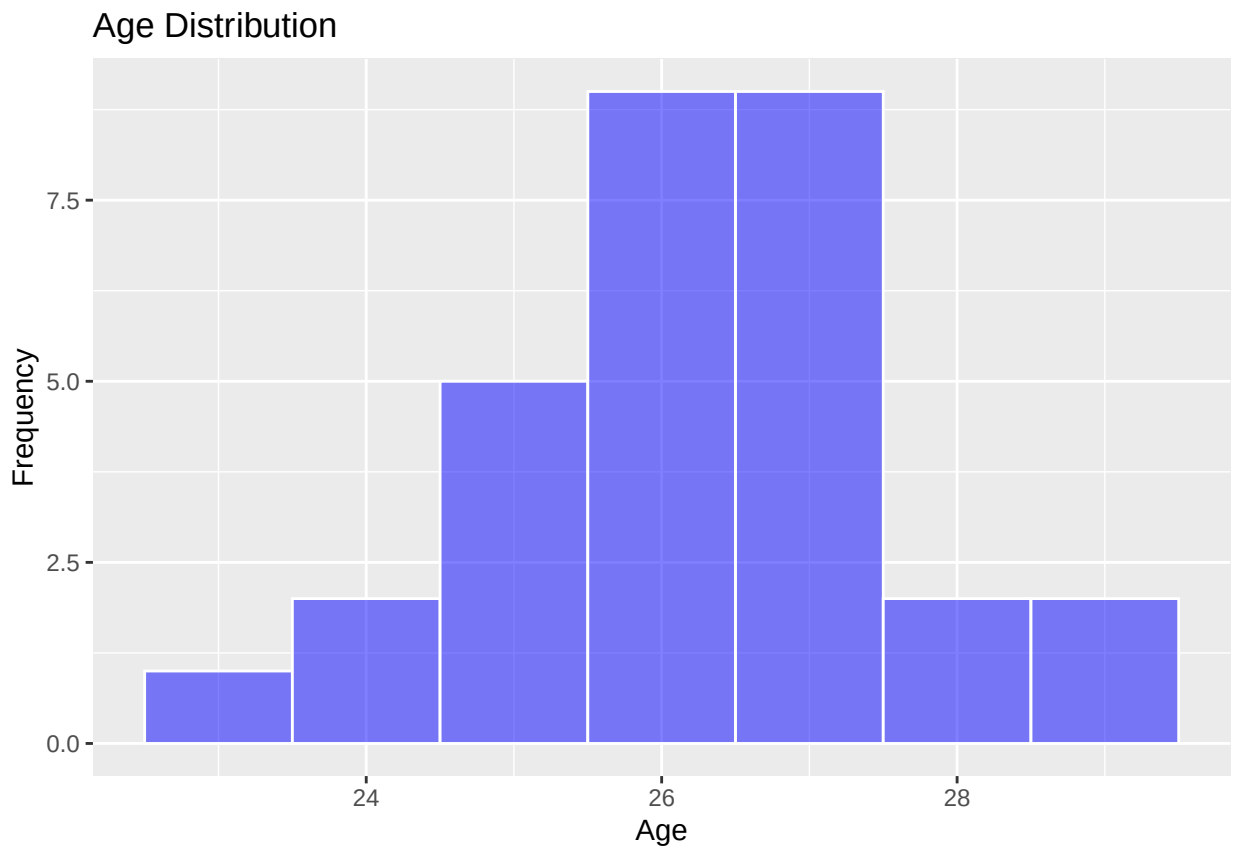



```
summary(team_stats1)
```

```
##        Rk             Team                Age              W         
##  Min.   : 1.00   Length:30          Min.   :23.40   Min.   :17.00  
##  1st Qu.: 8.25   Class :character   1st Qu.:25.48   1st Qu.:33.00  
##  Median :15.50   Mode  :character   Median :26.30   Median :41.50  
##  Mean   :15.50                      Mean   :26.28   Mean   :41.00  
##  3rd Qu.:22.75                      3rd Qu.:27.00   3rd Qu.:49.75  
##  Max.   :30.00                      Max.   :29.20   Max.   :60.00  
##        L               PW              PL              MOV         
##  Min.   :22.00   Min.   :19.00   Min.   :21.00   Min.   :-9.610  
##  1st Qu.:32.25   1st Qu.:37.00   1st Qu.:31.25   1st Qu.:-1.665  
##  Median :40.50   Median :40.50   Median :41.50   Median :-0.150  
##  Mean   :41.00   Mean   :41.10   Mean   :40.90   Mean   : 0.001  
##  3rd Qu.:49.00   3rd Qu.:50.75   3rd Qu.:45.00   3rd Qu.: 3.812  
##  Max.   :65.00   Max.   :61.00   Max.   :63.00   Max.   : 8.870  
##       SOS              SRS                ORtg            DRtg       
##  Min.   :-0.820   Min.   :-9.390000   Min.   :104.5   Min.   :105.2  
##  1st Qu.:-0.325   1st Qu.:-1.327500   1st Qu.:108.3   1st Qu.:108.3  
##  Median : 0.110   Median :-0.425000   Median :110.7   Median :110.2  
##  Mean   :-0.003   Mean   :-0.003333   Mean   :110.4   Mean   :110.4  
##  3rd Qu.: 0.240   3rd Qu.: 3.815000   3rd Qu.:112.5   3rd Qu.:112.6  
##  Max.   : 0.730   Max.   : 8.040000   Max.   :115.9   Max.   :117.6  
##       NRtg                Pace             FTr              3PAr        
##  Min.   :-9.900000   Min.   : 96.60   Min.   :0.2150   Min.   :0.2860  
```

```
##  1st Qu.:-1.650000   1st Qu.: 98.22   1st Qu.:0.2425   1st Qu.:0.3325
##  Median :-0.150000   Median : 99.90   Median :0.2570   Median :0.3475
##  Mean   :-0.003333   Mean   :100.04   Mean   :0.2588   Mean   :0.3588
##  3rd Qu.: 3.925000   3rd Qu.:101.55   3rd Qu.:0.2692   3rd Qu.:0.3832
##  Max.   : 8.600000   Max.   :103.90   Max.   :0.3260   Max.   :0.5190
##       TS%               eFG%             TOV%            ORB%
##  Min.   :0.5290   Min.   :0.4900   Min.   :10.90   Min.   :19.40
##  1st Qu.:0.5505   1st Qu.:0.5140   1st Qu.:11.93   1st Qu.:21.75
##  Median :0.5555   Median :0.5255   Median :12.40   Median :22.60
##  Mean   :0.5596   Mean   :0.5242   Mean   :12.40   Mean   :22.89
##  3rd Qu.:0.5710   3rd Qu.:0.5317   3rd Qu.:12.85   3rd Qu.:24.40
##  Max.   :0.5960   Max.   :0.5650   Max.   :14.30   Max.   :26.60
##      FT/FGA            DRB%
##  Min.   :0.1680   Min.   :72.50
##  1st Qu.:0.1825   1st Qu.:76.25
##  Median :0.1960   Median :77.10
##  Mean   :0.1983   Mean   :77.07
##  3rd Qu.:0.2100   3rd Qu.:77.97
##  Max.   :0.2580   Max.   :80.30
```

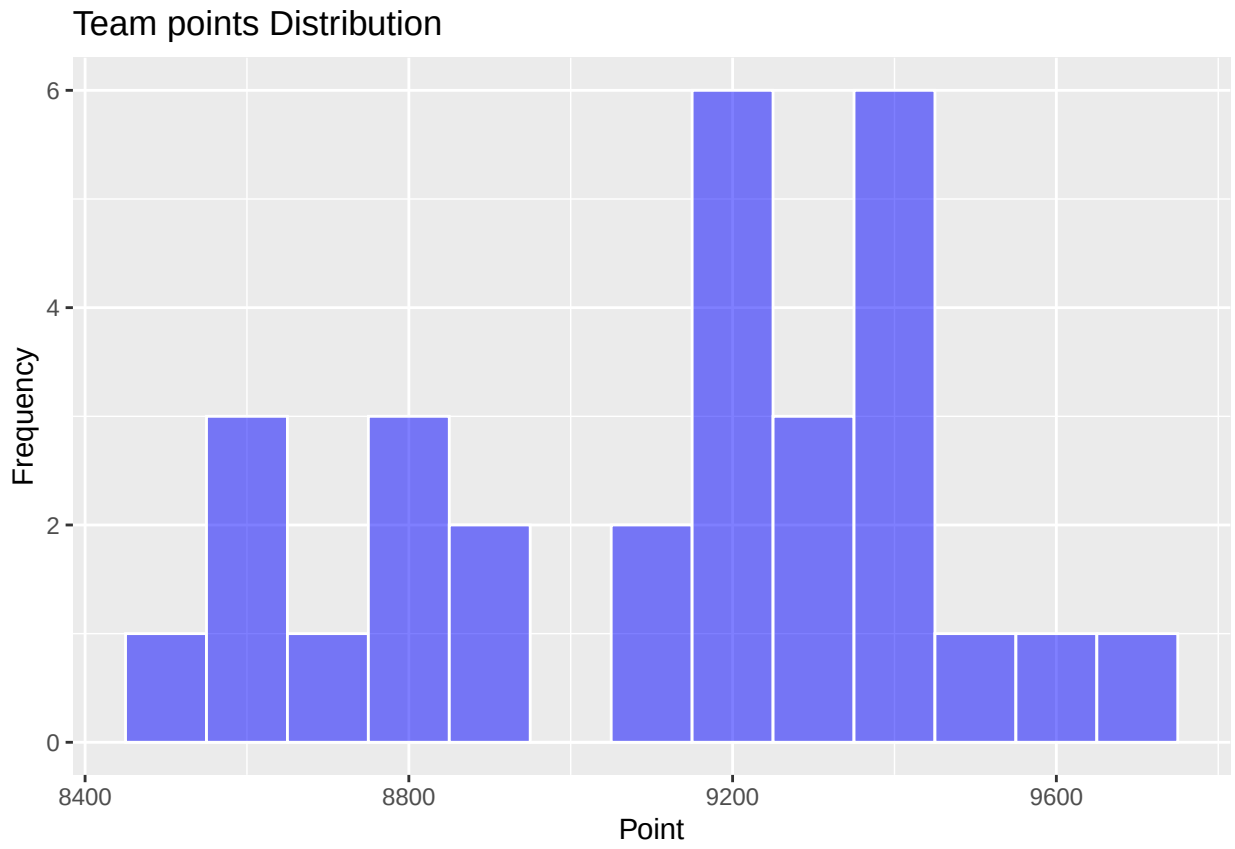
```r
# Distribution of team age
ggplot(team_stats1, aes(x = Age)) +
  geom_histogram(binwidth = 1, color = "white", fill = "blue", alpha = 0.5) +
  labs(title = "Age Distribution", x = "Age", y = "Frequency")
```



```
summary(team_stats2)
```

```
##        Rk             Team                 G             MP              FG
##  Min.   : 1.00   Length:30          Min.   :82   Min.   :19705   Min.   :3113
##  1st Qu.: 8.25   Class :character   1st Qu.:82   1st Qu.:19780   1st Qu.:3272
##  Median :15.50   Mode  :character   Median :82   Median :19805   Median :3391
##  Mean   :15.50                      Mean   :82   Mean   :19815   Mean   :3369
##  3rd Qu.:22.75                      3rd Qu.:82   3rd Qu.:19855   3rd Qu.:3466
##  Max.   :30.00                      Max.   :82   Max.   :19980   Max.   :3612
##       FGA            FG%               3P             3PA
##  Min.   :6924   Min.   :0.4330   Min.   : 745.0   Min.   :2071
##  1st Qu.:7189   1st Qu.:0.4500   1st Qu.: 830.8   1st Qu.:2405
##  Median :7306   Median :0.4615   Median : 927.5   Median :2602
##  Mean   :7315   Mean   :0.4605   Mean   : 931.8   Mean   :2625
##  3rd Qu.:7424   3rd Qu.:0.4708   3rd Qu.:1009.5   3rd Qu.:2815
##  Max.   :7706   Max.   :0.4910   Max.   :1323.0   Max.   :3721
##       3P%               2P            2PA            2P%               FT
##  Min.   :0.3290   Min.   :1895   Min.   :3442   Min.   :0.4790   Min.   :1231
##  1st Qu.:0.3480   1st Qu.:2322   1st Qu.:4535   1st Qu.:0.5070   1st Qu.:1340
##  Median :0.3525   Median :2474   Median :4716   Median :0.5175   Median :1451
##  Mean   :0.3555   Mean   :2437   Mean   :4691   Mean   :0.5202   Mean   :1450
##  3rd Qu.:0.3590   3rd Qu.:2564   3rd Qu.:4998   3rd Qu.:0.5343   3rd Qu.:1532
##  Max.   :0.3920   Max.   :2739   Max.   :5182   Max.   :0.5650   Max.   :1853
##       FTA            FT%              ORB             DRB            TRB
##  Min.   :1575   Min.   :0.6950   Min.   : 718.0   Min.   :2563   Min.   :3311
##  1st Qu.:1741   1st Qu.:0.7482   1st Qu.: 794.5   1st Qu.:2769   1st Qu.:3607
##  Median :1900   Median :0.7715   Median : 833.5   Median :2864   Median :3720
##  Mean   :1892   Mean   :0.7670   Mean   : 848.5   Mean   :2855   Mean   :3704
##  3rd Qu.:1987   3rd Qu.:0.7917   3rd Qu.: 908.2   3rd Qu.:2932   3rd Qu.:3803
##  Max.   :2340   Max.   :0.8190   Max.   :1031.0   Max.   :3316   Max.   :4078
##       AST            STL             BLK             TOV             PF
##  Min.   :1646   Min.   :501.0   Min.   :195.0   Min.   : 992   Min.   :1487
##  1st Qu.:1917   1st Qu.:563.0   1st Qu.:380.5   1st Qu.:1103   1st Qu.:1653
##  Median :2016   Median :621.5   Median :415.5   Median :1148   Median :1712
##  Mean   :2016   Mean   :626.0   Mean   :406.2   Mean   :1155   Mean   :1714
##  3rd Qu.:2132   3rd Qu.:682.2   3rd Qu.:439.2   3rd Qu.:1204   3rd Qu.:1762
##  Max.   :2413   Max.   :766.0   Max.   :525.0   Max.   :1397   Max.   :1932
##       PTS
##  Min.   :8490
##  1st Qu.:8826
##  Median :9184
##  Mean   :9119
##  3rd Qu.:9379
##  Max.   :9686
```

```
ggplot(team_stats2, aes(x = PTS)) +
  geom_histogram(binwidth = 100, color = "white", fill = "blue", alpha = 0.5) +
  labs(title = "Team points Distribution", x = "Point", y = "Frequency")
```

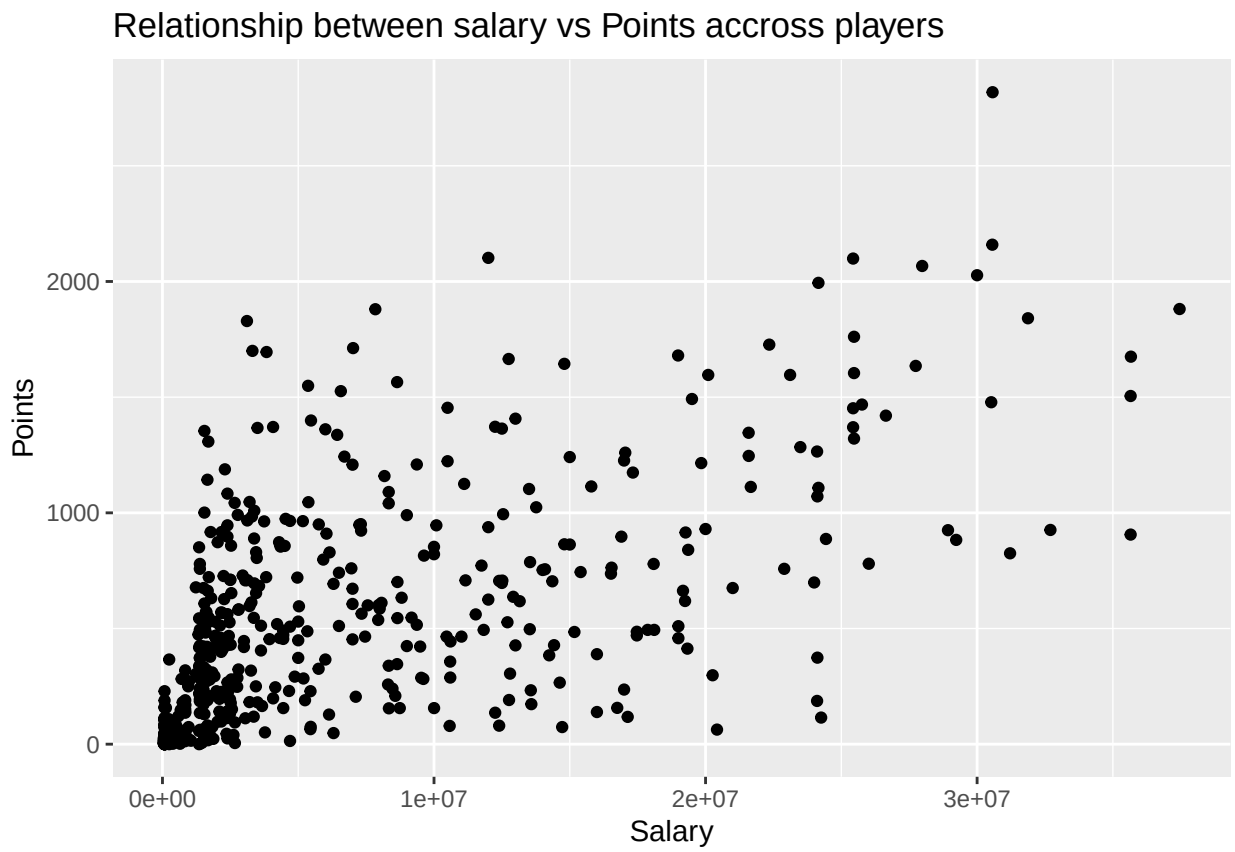



```
########## 3c checking for relationships between variables, or differences between groups#############

# Merge player_stats and player_salaries datasets
player_stats_salaries <- left_join( player_stats,player_salaries, by = "player_name")
colSums(is.na(player_stats_salaries))
```

```
## player_name         Pos         Age          Tm           G          GS
##           0           0           0           0           0           0
##          MP          FG         FGA         FG%          3P         3PA
##           0           0           0           6           0           0
##         3P%          2P         2PA         2P%        eFG%          FT
##          47           0           0          15           6           0
##         FTA         FT%         ORB         DRB         TRB         AST
##           0          43           0           0           0           0
##         STL         BLK         TOV          PF         PTS   player_id
##           0           0           0           0           0          22
##      salary
##          22
```

```
# Removing null values rows in salaries
player_stats_salaries <- player_stats_salaries[complete.cases(player_stats_salaries$player_id), ]

# Removing duplicates which happens because of trading of players in a season
player_stats_salaries<- player_stats_salaries %>%
  group_by(player_id) %>%
```

```
  filter(Tm == "Total" | (!"Total" %in% Tm) | (Tm == "Total" & row_number() == 1)) %>%
  ungroup()

# Scatter plot to understand the relationship between salary and PTS of a player
ggplot(player_stats_salaries, aes(x = salary, y = PTS)) +
  geom_point() +
  labs(title = "Relationship between salary vs Points accross players",x = "Salary", y = "Points")
```




```
# There seems to be an increased positive correlation between salary of a player and points
```

```
# Correlation accross different player statistics

# find numeric variables
player_stats_salaries_omit<-na.omit(player_stats_salaries)
num_vars <- sapply(player_stats_salaries_omit, is.numeric)

# subset dataframe to include only numeric variables
players_corr <- player_stats_salaries_omit[,num_vars]

# calculate correlation matrix
#cor(players_corr) calculates the correlation matrix for the players_corr dataset, which contains the r
cor_matrix <- cor(players_corr)
cor_matrix
```

```
##                    Age           G          GS          MP          FG
```

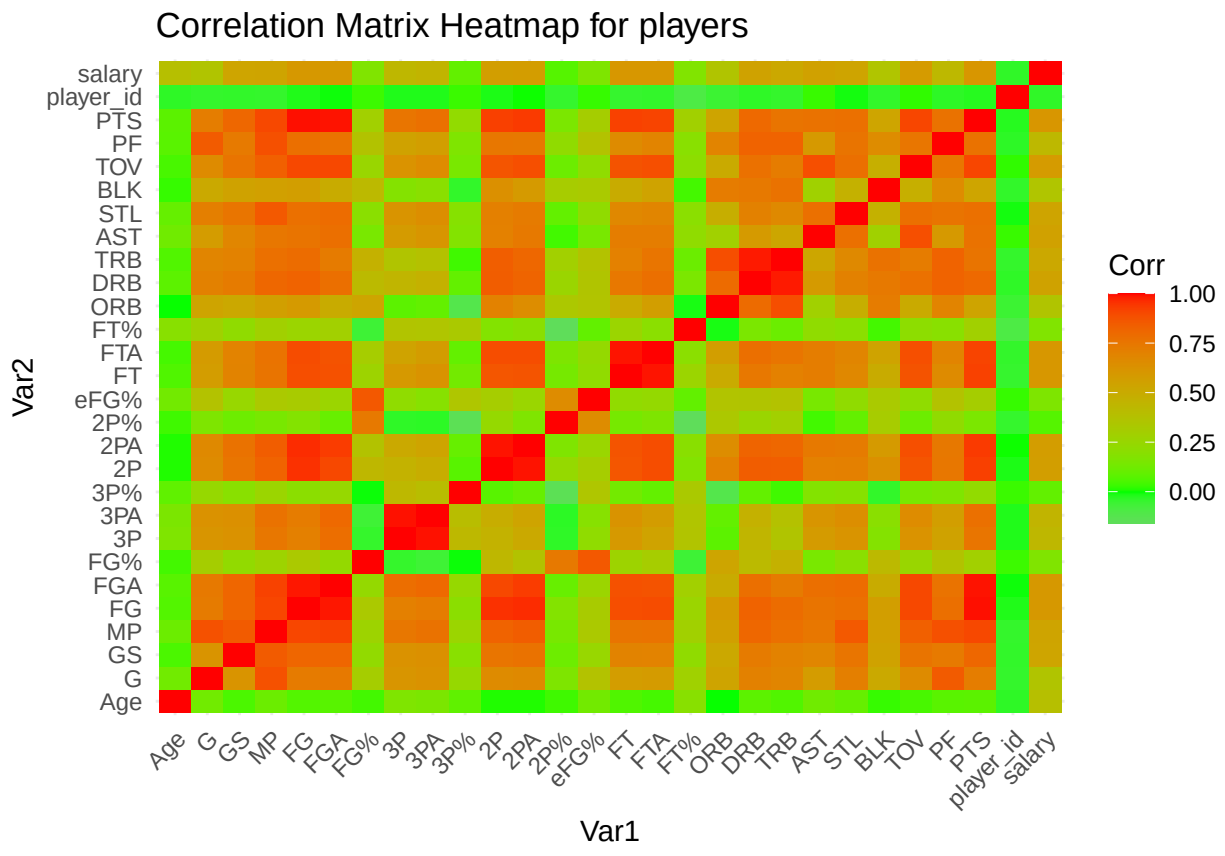
```
## Age        1.000000000  0.12690141  0.05335318  0.11343407  0.06474495
## G          0.126901408  1.00000000  0.61614233  0.88152530  0.73219952
## GS         0.053353178  0.61614233  1.00000000  0.85099246  0.81442669
## MP         0.113434072  0.88152530  0.85099246  1.00000000  0.90999884
## FG         0.064744953  0.73219952  0.81442669  0.90999884  1.00000000
## FGA        0.073515132  0.73834584  0.81317257  0.92056087  0.98678992
## FG%        0.042240024  0.30595920  0.22922003  0.27266733  0.33337583
## 3P         0.165284691  0.61170902  0.62744397  0.74922992  0.70862133
## 3PA        0.154936840  0.62798038  0.63563762  0.76755523  0.72531460
## 3P%        0.089862256  0.24434395  0.19554741  0.26199763  0.20402337
## 2P         0.010693761  0.65841835  0.75499657  0.82301160  0.95343903
## 2PA        0.010606434  0.66811290  0.76969718  0.84231550  0.96188748
## 2P%        0.037705058  0.16348305  0.11642375  0.14478512  0.17871962
## eFG%       0.130358137  0.37469949  0.24950707  0.33337837  0.32149279
## FT         0.060823241  0.57071149  0.69692661  0.75941423  0.89045078
## FTA        0.044862138  0.58310264  0.69814929  0.76568747  0.89399120
## FT%        0.193246167  0.28366537  0.22188086  0.28961542  0.26362572
## ORB        0.001680611  0.53461319  0.51336601  0.56122550  0.58715645
## DRB        0.081297217  0.70656170  0.73247782  0.81103542  0.82270027
## TRB        0.061841659  0.68791273  0.70132483  0.77445871  0.79076422
## AST        0.125619361  0.57699888  0.68290710  0.74897988  0.76082951
## STL        0.099294341  0.71509223  0.75670429  0.85620662  0.77684128
## BLK        0.027573170  0.50953897  0.54530638  0.55533221  0.56694208
## TOV        0.049403486  0.65954180  0.76077081  0.83552969  0.90399597
## PF         0.076112643  0.84984387  0.73715983  0.88466968  0.78125460
## PTS        0.079670718  0.72209624  0.80916173  0.90617655  0.99289872
## player_id -0.032587278 -0.04115463 -0.03796951 -0.04094333 -0.01658034
## salary     0.387870371  0.35808834  0.53436046  0.53855277  0.59877416
##                    FGA          FG%          3P         3PA          3P%
## Age        0.073515132  0.042240024  0.16528469  0.15493684  0.089862256
## G          0.738345842  0.305959199  0.61170902  0.62798038  0.244343950
## GS         0.813172572  0.229220030  0.62744397  0.63563762  0.195547406
## MP         0.920560871  0.272667333  0.74922992  0.76755523  0.261997628
## FG         0.986789923  0.333375833  0.70862133  0.72531460  0.204023374
## FGA        1.000000000  0.235274915  0.78250610  0.80399822  0.243207050
## FG%        0.235274915  1.000000000 -0.04207815 -0.05992484 -0.004104492
## 3P         0.782506095 -0.042078151  1.00000000  0.99116998  0.427440904
## 3PA        0.803998216 -0.059924835  0.99116998  1.00000000  0.400981483
## 3P%        0.243207050 -0.004104492  0.42744090  0.40098148  1.000000000
## 2P         0.905262800  0.436812266  0.46283161  0.48757788  0.073620244
## 2PA        0.933029439  0.370235163  0.51104002  0.53620549  0.102609518
## 2P%        0.102909852  0.741091689 -0.03384297 -0.02980323 -0.143717708
## eFG%       0.261538676  0.861021114  0.22090302  0.18877654  0.351304401
## FT         0.880245186  0.272959725  0.59391822  0.62198707  0.131509819
## FTA        0.874638499  0.310906445  0.54340147  0.57572548  0.094887269
## FT%        0.291875448 -0.059359669  0.36255887  0.35697227  0.329619860
## ORB        0.498759217  0.532059145  0.08063363  0.09440752 -0.116027087
## DRB        0.778278350  0.417235320  0.44695544  0.46849605  0.095594870
## TRB        0.731664901  0.468817648  0.36064747  0.38088916  0.038578632
## AST        0.780518101  0.146917865  0.58089141  0.61106907  0.177363302
## STL        0.792020538  0.198877903  0.61977194  0.64672428  0.186511535
## BLK        0.499283122  0.423345672  0.18097103  0.20108650 -0.037919057
## TOV        0.904828706  0.254611764  0.62279307  0.65626572  0.147384646
## PF         0.764506018  0.370118278  0.54715509  0.56670720  0.165430410
```

```
## PTS        0.990377172  0.291822619  0.75760750  0.77429056  0.227404560
## player_id -0.005035678  0.033818446 -0.01704023 -0.01671297  0.032197879
## salary     0.596166801  0.169524008  0.43640896  0.45196322  0.094148395
##                    2P          2PA         2P%         eFG%          FT
## Age        0.01069376 0.010606434  0.03770506 0.13035814  0.06082324
## G          0.65841835 0.668112902  0.16348305 0.37469949  0.57071149
## GS         0.75499657 0.769697176  0.11642375 0.24950707  0.69692661
## MP         0.82301160 0.842315498  0.14478512 0.33337837  0.75941423
## FG         0.95343903 0.961887479  0.17871962 0.32149279  0.89045078
## FGA        0.90526280 0.933029439  0.10290985 0.26153868  0.88024519
## FG%        0.43681227 0.370235163  0.74109169 0.86102111  0.27295973
## 3P         0.46283161 0.511040020 -0.03384297 0.22090302  0.59391822
## 3PA        0.48757788 0.536205495 -0.02980323 0.18877654  0.62198707
## 3P%        0.07362024 0.102609518 -0.14371771 0.35130440  0.13150982
## 2P         1.00000000 0.990008513  0.23899451 0.30947883  0.86483695
## 2PA        0.99000851 1.000000000  0.16411523 0.25703257  0.87316730
## 2P%        0.23899451 0.164115229  1.00000000 0.65562354  0.14024851
## eFG%       0.30947883 0.257032567  0.65562354 1.00000000  0.22445854
## FT         0.86483695 0.873167305  0.14024851 0.22445854  1.00000000
## FTA        0.89087693 0.893200527  0.16381149 0.23709849  0.98897960
## FT%        0.17623207 0.198324061 -0.15917441 0.09246558  0.26162632
## ORB        0.70319219 0.650871076  0.33706303 0.36293491  0.50166016
## DRB        0.84253601 0.821298473  0.25914406 0.35803759  0.74452190
## TRB        0.83930419 0.808139293  0.29309178 0.37521215  0.70702799
## AST        0.70755900 0.738209677  0.04073689 0.14462573  0.72335170
## STL        0.71105644 0.732963264  0.09581844 0.22231688  0.67446616
## BLK        0.63490981 0.587065397  0.31375483 0.32723688  0.49752672
## TOV        0.86951223 0.887322594  0.11480379 0.22005269  0.87487975
## PF         0.74763918 0.742322817  0.22194197 0.37466007  0.65984902
## PTS        0.92357967 0.937345231  0.15382434 0.30590432  0.92126506
## player_id -0.01354682 0.002964484 -0.04270881 0.02794466 -0.04410382
## salary     0.56572076 0.572792441  0.06897083 0.16256009  0.60154411
##                   FTA         FT%          ORB         DRB         TRB
## Age        0.04486214  0.19324617  0.001680611  0.08129722  0.06184166
## G          0.58310264  0.28366537  0.534613188  0.70656170  0.68791273
## GS         0.69814929  0.22188086  0.513366011  0.73247782  0.70132483
## MP         0.76568747  0.28961542  0.561225500  0.81103542  0.77445871
## FG         0.89399120  0.26362572  0.587156451  0.82270027  0.79076422
## FGA        0.87463850  0.29187545  0.498759217  0.77827835  0.73166490
## FG%        0.31090644 -0.05935967  0.532059145  0.41723532  0.46881765
## 3P         0.54340147  0.36255887  0.080633632  0.44695544  0.36064747
## 3PA        0.57572548  0.35697227  0.094407521  0.46849605  0.38088916
## 3P%        0.09488727  0.32961986 -0.116027087  0.09559487  0.03857863
## 2P         0.89087693  0.17623207  0.703192186  0.84253601  0.83930419
## 2PA        0.89320053  0.19832406  0.650871076  0.82129847  0.80813929
## 2P%        0.16381149 -0.15917441  0.337063034  0.25914406  0.29309178
## eFG%       0.23709849  0.09246558  0.362934908  0.35803759  0.37521215
## FT         0.98897960  0.26162632  0.501660156  0.74452190  0.70702799
## FTA        1.00000000  0.20208859  0.568082175  0.78145438  0.75411741
## FT%        0.20208859  1.00000000 -0.010703813  0.15280930  0.11222932
## ORB        0.56808218 -0.01070381  1.000000000  0.79399227  0.88853625
## DRB        0.78145438  0.15280930  0.793992269  1.00000000  0.98441212
## TRB        0.75411741  0.11222932  0.888536245  0.98441212  1.00000000
## AST        0.72113559  0.22120270  0.287571959  0.58747193  0.52656453
```

```
## STL         0.68870122  0.20193864  0.484789322  0.70388090  0.67147517
## BLK         0.54378553  0.04382575  0.729012574  0.74036181  0.76966276
## TOV         0.88740518  0.21106816  0.505831351  0.77201231  0.72898190
## PF          0.69015860  0.19532482  0.692543199  0.82417516  0.82236634
## PTS         0.91537665  0.28887356  0.537294464  0.80246504  0.76106720
## player_id -0.04016567 -0.09448317 -0.053982945 -0.03398686 -0.04126766
## salary      0.60227687  0.17032871  0.360370573  0.54528659  0.51578810
##                  AST          STL         BLK        TOV          PF
## Age         0.12561936  0.099294341  0.02757317 0.04940349  0.07611264
## G           0.57699888  0.715092230  0.50953897 0.65954180  0.84984387
## GS          0.68290710  0.756704291  0.54530638 0.76077081  0.73715983
## MP          0.74897988  0.856206619  0.55533221 0.83552969  0.88466968
## FG          0.76082951  0.776841279  0.56694208 0.90399597  0.78125460
## FGA         0.78051810  0.792020538  0.49928312 0.90482871  0.76450602
## FG%         0.14691787  0.198877903  0.42334567 0.25461176  0.37011828
## 3P          0.58089141  0.619771936  0.18097103 0.62279307  0.54715509
## 3PA         0.61106907  0.646724280  0.20108650 0.65626572  0.56670720
## 3P%         0.17736330  0.186511535 -0.03791906 0.14738465  0.16543041
## 2P          0.70755900  0.711056440  0.63490981 0.86951223  0.74763918
## 2PA         0.73820968  0.732963264  0.58706540 0.88732259  0.74232282
## 2P%         0.04073689  0.095818436  0.31375483 0.11480379  0.22194197
## eFG%        0.14462573  0.222316880  0.32723688 0.22005269  0.37466007
## FT          0.72335170  0.674466159  0.49752672 0.87487975  0.65984902
## FTA         0.72113559  0.688701224  0.54378553 0.88740518  0.69015860
## FT%         0.22120270  0.201938640  0.04382575 0.21106816  0.19532482
## ORB         0.28757196  0.484789322  0.72901257 0.50583135  0.69254320
## DRB         0.58747193  0.703880903  0.74036181 0.77201231  0.82417516
## TRB         0.52656453  0.671475172  0.76966276 0.72898190  0.82236634
## AST         1.00000000  0.776318454  0.28197840 0.88723457  0.59433717
## STL         0.77631845  1.000000000  0.47019566 0.77959616  0.75884479
## BLK         0.28197840  0.470195664  1.00000000 0.48089978  0.65618143
## TOV         0.88723457  0.779596158  0.48089978 1.00000000  0.75387307
## PF          0.59433717  0.758844790  0.65618143 0.75387307  1.00000000
## PTS         0.76982288  0.776402321  0.53417778 0.90946177  0.76769374
## player_id 0.02983563 -0.007888445 -0.03750786 0.02242358 -0.03142605
## salary      0.55311966  0.538467188  0.35668372 0.58423113  0.42680974
##                  PTS    player_id      salary
## Age         0.07967072 -0.032587278  0.38787037
## G           0.72209624 -0.041154633  0.35808834
## GS          0.80916173 -0.037969512  0.53436046
## MP          0.90617655 -0.040943333  0.53855277
## FG          0.99289872 -0.016580335  0.59877416
## FGA         0.99037717 -0.005035678  0.59616680
## FG%         0.29182262  0.033818446  0.16952401
## 3P          0.75760750 -0.017040229  0.43640896
## 3PA         0.77429056 -0.016712974  0.45196322
## 3P%         0.22740456  0.032197879  0.09414839
## 2P          0.92357967 -0.013546816  0.56572076
## 2PA         0.93734523  0.002964484  0.57279244
## 2P%         0.15382434 -0.042708805  0.06897083
## eFG%        0.30590432  0.027944656  0.16256009
## FT          0.92126506 -0.044103821  0.60154411
## FTA         0.91537665 -0.040165671  0.60227687
## FT%         0.28887356 -0.094483167  0.17032871
```

```
## ORB         0.53729446 -0.053982945  0.36037057
## DRB         0.80246504 -0.033986859  0.54528659
## TRB         0.76106720 -0.041267656  0.51578810
## AST         0.76982288  0.029835635  0.55311966
## STL         0.77640232 -0.007888445  0.53846719
## BLK         0.53417778 -0.037507857  0.35668372
## TOV         0.90946177  0.022423576  0.58423113
## PF          0.76769374 -0.031426052  0.42680974
## PTS         1.00000000 -0.023026079  0.60978811
## player_id -0.02302608  1.000000000 -0.03541524
## salary      0.60978811 -0.035415238  1.00000000
```

```r
#melt() function from the reshape2 package is used to convert the matrix into a long-format data frame

cor_df <- melt(cor_matrix)
colnames(cor_df) <- c("Var1", "Var2", "Corr")

# create heatmap by ggplot()and theme_minimal() function is used to apply a minimal theme to the plot
ggplot(data = cor_df, aes(x = Var1, y = Var2, fill = Corr)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "green", midpoint = 0) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Correlation Matrix Heatmap for players")
```



```r
# Merging player and team statisitcs

# Merge player_stats and team_stats_2 datasets
team_stats <- full_join(team_stats1, team_stats2, by = "Team")

team_payroll <- subset(team_payroll, select = -team)
# team_payroll <- rename(team_payroll, Team = team_full_name)

team_stats_salary<-full_join( team_stats ,team_payroll, by = "Team")
player_stats_salaries <- rename(player_stats_salaries, Team = Tm)

# Merging the player and team stats as a master dataset

# master_player_team<- full_join(player_stats_salaries,team_stats_salary, by = "Team")
#
# colSums(is.na(master_player_team))
#
# master_player_team_omit<-na.omit(master_player_team)
#
# colSums(is.na(master_player_team_omit))

#developing a reproducible data model that captures relevant player statistics and incorporates #approp
```

4. Data modelling and results

This presented is an example of a linear regression model that predicts points (PTS) based on salary, field goals attempted (FGA), and free throws attempted (FTA) in the NBA. The model was created using R programming language, specifically the "lm" function which fits a linear regression model. The data used is player statistics and salaries for the most recent NBA season. The output shows a summary of the linear regression model. The coefficients for FGA and FTA have p-values less than 2e-16, which means they are statistically significant and strongly associated with PTS. However, the coefficient for salary has a p-value of 0.288, which is not statistically significant and suggests that salary is not a good predictor of PTS. The adjusted R-squared value of 0.9922 indicates that the model explains a high proportion of the variance in PTS, which means that the model is a good fit for the data. The F-statistic of 2.915e+04 with a p-value $<$ 2.2e-16 indicates that the overall model is significant, which means that the model is better at predicting PTS than just using the mean of the data. The residual standard error of 40.32 suggests that the model has a moderate level of error in predicting PTS, which means that there is some variability in the data that is not accounted for by the model. However, the normal Q-Q plot and residual vs. fitted plot do not show any major departures from normality or homoscedasticity assumptions, which means that the model assumptions are met. In conclusion, this linear regression model can be used to predict PTS based on FGA and FTA, but not salary. The model is a good fit for the data and meets the assumptions of normality and homoscedasticity. Further analyses could be conducted to improve the model, such as including other variables that may be associated with PTS.

Assumption checking is a crucial step in validating the results of a statistical model. We check for two important assumptions of the linear regression model - homoscedasticity and normality. Homoscedasticity refers to the assumption that the variance of the residuals of the model is constant across all levels of the predictor variables. The plot of residuals vs. fitted values is used to check for homoscedasticity. In the plot generated by the code, the residuals are plotted against the predicted values of the dependent variable (PTS), and we look for a pattern in the points. If the points are randomly scattered around the horizontal line, it indicates that the assumption of homoscedasticity holds. However, if there is a visible pattern in the plot, such as a funnel shape or a curve, it indicates that the model violates the homoscedasticity

assumption. Normality refers to the assumption that the residuals of the model follow a normal distribution. The normality plot (also known as a Q-Q plot) is used to check for normality. In the plot generated by the code, the residuals are plotted against the quantiles of a normal distribution. If the residuals follow a straight line in the plot, it indicates that they are normally distributed. However, if there is a visible deviation from a straight line, such as a curve or a bend, it indicates that the model violates the normality assumption. By checking for these assumptions, we can ensure that the linear regression model is valid and that the results can be trusted. If the assumptions are violated, we may need to modify the model or use a different statistical method to obtain reliable results.

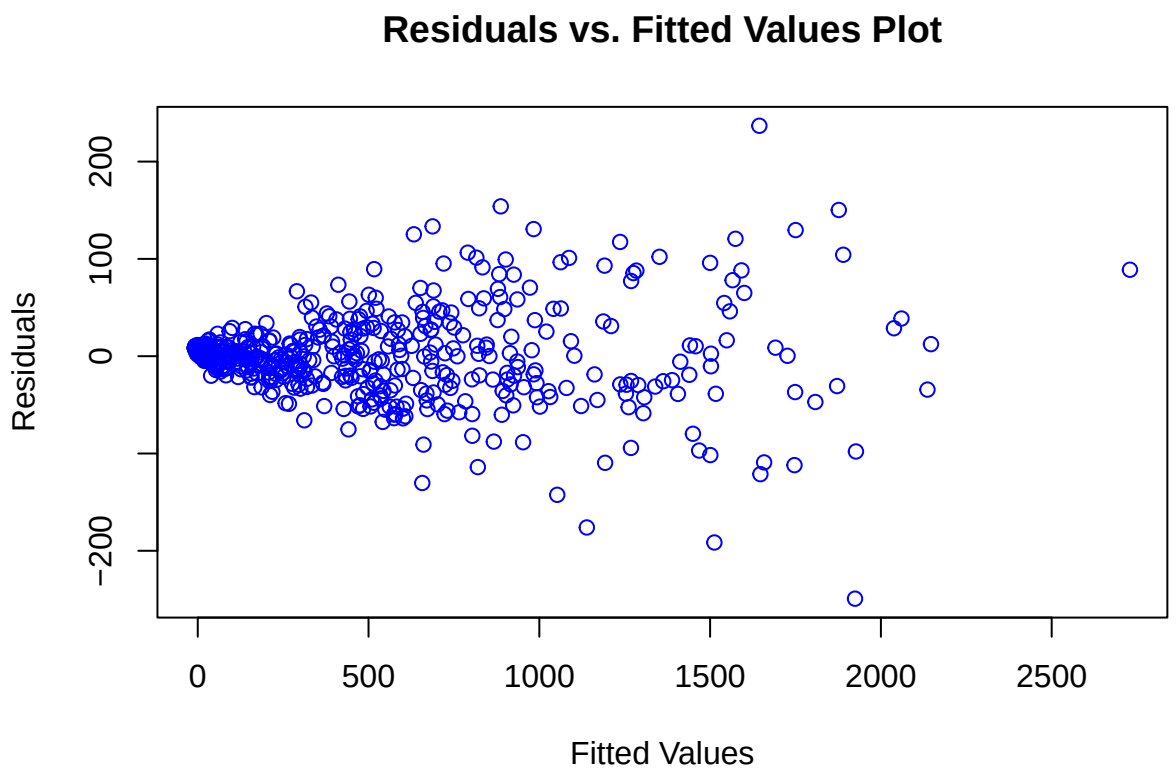
```
# 4.4. Data modelling and results


# Create linear regression model to predict PTS based on salary
lm_model <- lm(PTS ~ salary +  FGA + FTA, data = player_stats_salaries)

# Display model summary
summary(lm_model)
```

```
##
## Call:
## lm(formula = PTS ~ salary + FGA + FTA, data = player_stats_salaries)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -249.209  -20.677    1.635   14.311  236.811
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.539e+00  3.036e+00  -2.813   0.0051 **
## salary       2.726e-07  3.238e-07   0.842   0.4004
## FGA          1.040e+00  1.107e-02  93.958   <2e-16 ***
## FTA          8.669e-01  3.447e-02  25.154   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 45.18 on 508 degrees of freedom
## Multiple R-squared:  0.9916, Adjusted R-squared:  0.9916
## F-statistic: 2.007e+04 on 3 and 508 DF,  p-value: < 2.2e-16
```

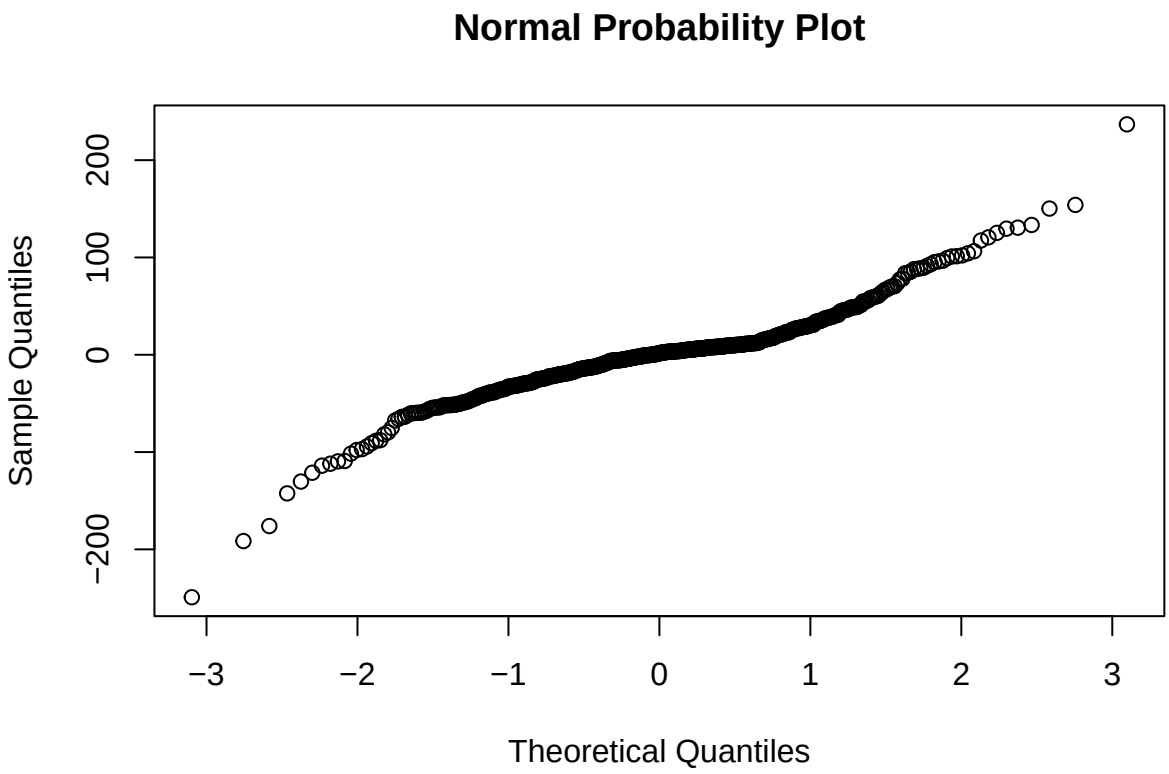
```
# PTS = -8.539 + 2.726e-07(salary) + 1.04(FGA) + 0.8669(FTA)




#***************** Model Interpretation ************************************
# In the linear regression model with PTS as the dependent variable and salary, FGA, and FTA as indepen
```

```
# Assumption checking

# 1. homoscedasticity

# Plot residuals vs. fitted values
plot(lm_model$fitted.values, lm_model$residuals, type = "p", col = "blue",
```

```
     xlab = "Fitted Values", ylab = "Residuals",
     main = "Residuals vs. Fitted Values Plot")
```




```
# 2. Normality plot

#Normal probability plot
qqnorm(lm_model$residuals, main = "Normal Probability Plot")
```



5. Player recommendations

The process analyzes and recommends top players based on various criteria in a basketball dataset. The dataset includes player statistics and salaries for the 2019-2020 NBA season. The script includes four sections, each focusing on different criteria to select the top five players. The first section focuses on selecting the top five players in each position based on cost-effectiveness. The script calculates the cost-effectiveness score for each player, which is the total number of points a player scores in a season divided by their salary. Then, it selects the top player for each position based on their cost-effectiveness score. The output includes the name of the player, their team, position, age, games played, points scored, salary, and cost-effectiveness score. The output also includes the total salary of the selected players. The second section is similar to the first section, but it only considers players from the Chicago Bulls team. The script filters out players from other teams and selects the top player for each position based on their cost-effectiveness score. The output includes the same information as the first section, but only for the Chicago Bulls players. The third section focuses on selecting the top five players in each position based on the total number of points they score in a season, regardless of their salary. The script selects the top player for each position based on their points scored and outputs the same information as the first two sections. The fourth section selects the top five players irrespective of position based on their salaries. The script sorts the dataset by salary in descending order and selects the top five players. The output includes the name of the player, their team, position, age, games played, points scored, and salary. Overall, we provide useful insights and recommendations for basketball team management to help them select the most effective and cost-efficient players. It also demonstrates how R can be used to analyze and visualize data to make data-driven decisions.

The top 5 players based on the analysis are Alex Caruso, Kadeem Allen, Lebron James, Lauri Markkanen, James Harden

```r
# ***** 5. Player recommendation in each position **************************

# Top 5 player with regards to position based on Cost effectiveness

# Calculate cost-effectiveness score
player_stats_salaries$cost_effectiveness <- player_stats_salaries$PTS / player_stats_salaries$salary

# Select top player for each position
point_guard <- player_stats_salaries %>%
  filter(Pos == "PG") %>%
  slice_max(cost_effectiveness)

shooting_guard <- player_stats_salaries %>%
  filter(Pos == "SG") %>%
  slice_max(cost_effectiveness)

small_forward <- player_stats_salaries %>%
  filter(Pos == "SF") %>%
  slice_max(cost_effectiveness)

power_forward <- player_stats_salaries %>%
  filter(Pos == "PF") %>%
  slice_max(cost_effectiveness)

center <- player_stats_salaries %>%
  filter(Pos == "C") %>%
  slice_max(cost_effectiveness)

# Combine selected players into final output
top_five <- bind_rows(
  point_guard %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  shooting_guard %>% select(player_name,Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  small_forward %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  power_forward %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  center %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness)
)

top_five_budget <- sum(top_five$salary)

top_five_budget
```

```
## [1] 606827
```

```r
# Output starting five
top_five
```

```
## # A tibble: 5 x 8
##   player_name        Team      Pos     Age     G   PTS salary cost_effectiveness
##   <chr>              <chr>     <chr> <int> <int> <int>  <int>              <dbl>
## 1 Alex Caruso        Los Ange~ PG       24    25   229  77250            0.00296
## 2 Kadeem Allen       New York~ SG       26    19   189  77250            0.00245
## 3 Danuel House       Houston ~ SF       25    39   366 247827            0.00148
## 4 Alex Poythress     Atlanta ~ PF       25    21   107  77250            0.00139
```

```
## 5 Johnathan Williams Los Ange~ C        23    24   157 127250            0.00123
```

```
# Top 5 player with regards to position based on Cost effectiveness from Chicago Bulls team

chicago_players <- player_stats_salaries %>% filter(Team == "Chicago Bulls")


# Select top player for each position
point_guard <- chicago_players %>%
  filter(Pos == "PG") %>%
  slice_max(cost_effectiveness)

shooting_guard <- chicago_players %>%
  filter(Pos == "SG") %>%
  slice_max(cost_effectiveness)

small_forward <- chicago_players %>%
  filter(Pos == "SF") %>%
  slice_max(cost_effectiveness)

power_forward <- chicago_players %>%
  filter(Pos == "PF") %>%
  slice_max(cost_effectiveness)

center <- chicago_players %>%
  filter(Pos == "C") %>%
  slice_max(cost_effectiveness)

# Combine selected players into final output
chicago_top_five <- bind_rows(
  point_guard %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  shooting_guard %>% select(player_name,Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  small_forward %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  power_forward %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness),
  center %>% select(player_name, Team, Pos, Age, G, PTS, salary, cost_effectiveness)
)

# Output starting five
chicago_top_five
```

```
## # A tibble: 5 x 8
##   player_name      Team        Pos     Age     G   PTS salary cost_effectiveness
##   <chr>            <chr>       <chr> <int> <int> <int>  <int>              <dbl>
## 1 Ryan Arcidiacono Chicago Bu~ PG       24    81   544 1.35e6           0.000403
## 2 Brandon Sampson  Chicago Bu~ SG       21    14    71 7.72e4           0.000919
## 3 JaKarr Sampson   Chicago Bu~ SF       25     4    80 8.55e4           0.000936
## 4 Lauri Markkanen  Chicago Bu~ PF       21    52   974 4.54e6           0.000215
## 5 Wendell Carter   Chicago Bu~ C        19    44   455 4.45e6           0.000102
```

```
chicago_top_five_budget <- sum(chicago_top_five$salary)

chicago_top_five_budget
```

```
## [1] 10495050
```

```r
# Top 5 players based on Points alone, neglecting salary

# Select top player for each position
point_guard_top <- player_stats_salaries %>%
  filter(Pos == "PG") %>%
  slice_max(PTS)

shooting_guard_top <- player_stats_salaries %>%
  filter(Pos == "SG") %>%
  slice_max(PTS)

small_forward_top <- player_stats_salaries %>%
  filter(Pos == "SF") %>%
  slice_max(PTS)

power_forward_top <- player_stats_salaries %>%
  filter(Pos == "PF") %>%
  slice_max(PTS)

center_top <- player_stats_salaries %>%
  filter(Pos == "C") %>%
  slice_max(PTS)

# Combine selected players into final output
top_five_cost <- bind_rows(
  point_guard_top %>% select(player_name, Team, Pos, Age, G, PTS, salary),
  shooting_guard_top %>% select(player_name,Team, Pos, Age, G, PTS, salary),
  small_forward_top %>% select(player_name, Team, Pos, Age, G, PTS, salary),
  power_forward_top %>% select(player_name, Team, Pos, Age, G, PTS, salary),
  center_top %>% select(player_name, Team, Pos, Age, G, PTS, salary)
)

top_five_cost_budget <- sum(top_five_cost$salary)

top_five_cost_budget
```

```
## [1] 118561701
```

```r
# Output starting five not based on cost effectiveness
top_five_cost
```

```
## # A tibble: 5 x 7
##   player_name           Team                  Pos     Age     G   PTS   salary
##   <chr>                 <chr>                 <chr> <int> <int> <int>    <int>
## 1 James Harden          Houston Rockets       PG       29    78  2818 30570000
## 2 Bradley Beal          Washington Wizards    SG       25    82  2099 25434262
## 3 Paul George           Oklahoma City Thunder SF       28    77  2159 30560700
## 4 Giannis Antetokounmpo Milwaukee Bucks       PF       24    72  1994 24157304
## 5 Karl-Anthony Towns    Minnesota Timberwolves C        23    77  1880  7839435
```

```r
# Top 5 irrespective of position

player_stats_salaries[order(-player_stats_salaries$salary),][1:5,] %>% select(player_name, Team, Pos, Ag
```

```
## # A tibble: 5 x 7
##   player_name       Team                  Pos     Age     G   PTS   salary
##   <chr>             <chr>                 <chr> <int> <int> <int>    <int>
## 1 Stephen Curry     Golden State Warriors PG       30    69  1881 37457154
## 2 Russell Westbrook Oklahoma City Thunder PG       30    73  1675 35665000
## 3 LeBron James      Los Angeles Lakers    SF       34    55  1505 35654150
## 4 Chris Paul        Houston Rockets       PG       33    58   906 35654150
## 5 Kyle Lowry        Toronto Raptors       PG       32    65   926 32700000
```

6.Summary

In this project, we aimed to help a general manager of a basketball team build a strong team within a set budget by analyzing NBA player data. Our analysis involved cleaning and processing the data and conducting exploratory data analysis to gain insights into the data. We visualized player salary distributions, player performance metrics, and the relationships between them. To model the relationship between player performance metrics and salary, we used linear regression analysis. We found that performance metrics such as salary, FGA and FTA were significant predictors of player salary. We then used this model to predict salaries for each player in the dataset, identifying overpaid and underpaid players. Using optimization techniques, we selected a starting five of players with the highest combined performance metric score while staying within the budget constraint. We also recommended top players for each position based on their cost-effectiveness, as well as the top five players irrespective of position. Our analysis has provided the general manager with valuable insights into how to build a competitive basketball team within a budget. We identified which players to target, which players are overpaid and underpaid, and which positions to focus on to maximize the team's performance. However, our analysis is limited by the scope of the dataset and the assumptions made in our model. Other factors such as player popularity, team market size, or sponsorship deals may affect player salaries, which we did not consider. Additionally, our model assumes a linear relationship between performance metrics and salary, which may not always be the case in practice. Therefore, it is essential for the general manager to consider these limitations when making decisions based on our analysis. Overall, our project provides a useful framework for analyzing and optimizing a basketball team's performance within a budget constraint. It is crucial to consider the limitations and potential biases in the data when making decisions based on our analysis.

7.Reference

[1] https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/
[2] Baker, M. (2010). Nature Methods: Reproducibility crisis: Blame it on the antibodies. Nature Publishing Group. https://doi.org/10.1038/nmeth.1549