Essentials of Data Analytics Project Report - Cricket Performance Analytics

Surya Prasad S

17/03/2021

Setup

```r
rm(list=ls())
library(dplyr)
```

```
##
## 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
```

```r
library(tidyr)
library(ggplot2)
library(ggpubr)
setwd("G:/VIT/Winter Semester 2020-21/Essentials Of Data Analytics/Project")
getwd()
```

```
## [1] "G:/VIT/Winter Semester 2020-21/Essentials Of Data Analytics/Project"
```

What problem are you trying to solve?

1. Evaluating the impact of players in different phases of the game through a performance index considering the match situation and ground conditions. This would help in teams picking the right players for the right roles in their teams. (Player Selection and Team Recommendation)

2. Score Predictor for the given match. The predictor takes into account the Match situation and Ground Conditions (Score, Overs, Number of Wickets Left, Opposition, Ground). This would help the viewers know who is ahead in terms of winning the match. (Win Predictor)

What data have you chosen?(Chosen Dataset, Source of dataset, Description of dataset, basic commands to describe dataset)

Chosen Dataset: Indian Premier League (Cricket) : Ball-By-Ball Cricket Data

Source of Dataset: Kaggle (https://www.kaggle.com/nowke9/ipldata)

Description of Dataset:

All Indian Premier League Cricket matches between 2008 and 2019.

This is the ball by ball data of all the IPL cricket matches till season 12.

The dataset contains 2 files: deliveries.csv and matches.csv.

matches.csv contains details related to the match such as location, contesting teams, umpires, results, etc.

deliveries.csv is the ball-by-ball data of all the IPL matches including data of the batting team, batsman, bowler, non-striker, runs scored, etc.

```r
deliveries = read.csv("deliveries.csv")
matches = read.csv("matches.csv")
str(deliveries)
```

```
## 'data.frame':    179078 obs. of  21 variables:
##  $ match_id        : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ inning          : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ batting_team    : chr  "Sunrisers Hyderabad" "Sunrisers Hyderabad" "Sunrisers Hyderabad" "Sunrise
##  $ bowling_team    : chr  "Royal Challengers Bangalore" "Royal Challengers Bangalore" "Royal Challeng
##  $ over            : int  1 1 1 1 1 1 1 2 2 2 ...
##  $ ball            : int  1 2 3 4 5 6 7 1 2 3 ...
##  $ batsman         : chr  "DA Warner" "DA Warner" "DA Warner" "DA Warner" ...
##  $ non_striker     : chr  "S Dhawan" "S Dhawan" "S Dhawan" "S Dhawan" ...
##  $ bowler          : chr  "TS Mills" "TS Mills" "TS Mills" "TS Mills" ...
##  $ is_super_over   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ wide_runs       : int  0 0 0 0 2 0 0 0 0 0 ...
##  $ bye_runs        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ legbye_runs     : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ noball_runs     : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ penalty_runs    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ batsman_runs    : int  0 0 4 0 0 0 0 1 4 0 ...
##  $ extra_runs      : int  0 0 0 0 2 0 1 0 0 1 ...
##  $ total_runs      : int  0 0 4 0 2 0 1 1 4 1 ...
##  $ player_dismissed: chr  "" "" "" "" ...
##  $ dismissal_kind  : chr  "" "" "" "" ...
##  $ fielder         : chr  "" "" "" "" ...
```

```r
str(matches)
```

```
## 'data.frame':    756 obs. of  18 variables:
```

```
##  $ id             : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ season         : int  2017 2017 2017 2017 2017 2017 2017 2017 2017 2017 ...
##  $ city           : chr  "Hyderabad" "Pune" "Rajkot" "Indore" ...
##  $ date           : chr  "2017-04-05" "2017-04-06" "2017-04-07" "2017-04-08" ...
##  $ team1          : chr  "Sunrisers Hyderabad" "Mumbai Indians" "Gujarat Lions" "Rising Pune Supergia
##  $ team2          : chr  "Royal Challengers Bangalore" "Rising Pune Supergiant" "Kolkata Knight Ride
##  $ toss_winner    : chr  "Royal Challengers Bangalore" "Rising Pune Supergiant" "Kolkata Knight Ride
##  $ toss_decision  : chr  "field" "field" "field" "field" ...
##  $ result         : chr  "normal" "normal" "normal" "normal" ...
##  $ dl_applied     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ winner         : chr  "Sunrisers Hyderabad" "Rising Pune Supergiant" "Kolkata Knight Riders" "Kin
##  $ win_by_runs    : int  35 0 0 0 15 0 0 0 97 0 ...
##  $ win_by_wickets : int  0 7 10 6 0 9 4 8 0 4 ...
##  $ player_of_match: chr  "Yuvraj Singh" "SPD Smith" "CA Lynn" "GJ Maxwell" ...
##  $ venue          : chr  "Rajiv Gandhi International Stadium, Uppal" "Maharashtra Cricket Associatio
##  $ umpire1        : chr  "AY Dandekar" "A Nand Kishore" "Nitin Menon" "AK Chaudhary" ...
##  $ umpire2        : chr  "NJ Llong" "S Ravi" "CK Nandan" "C Shamshuddin" ...
##  $ umpire3        : chr  "" "" "" "" ...
```

```r
head(deliveries,5)
```

```
##   match_id inning        batting_team                bowling_team over ball
## 1        1      1 Sunrisers Hyderabad Royal Challengers Bangalore    1    1
## 2        1      1 Sunrisers Hyderabad Royal Challengers Bangalore    1    2
## 3        1      1 Sunrisers Hyderabad Royal Challengers Bangalore    1    3
## 4        1      1 Sunrisers Hyderabad Royal Challengers Bangalore    1    4
## 5        1      1 Sunrisers Hyderabad Royal Challengers Bangalore    1    5
##     batsman non_striker   bowler is_super_over wide_runs bye_runs legbye_runs
## 1 DA Warner    S Dhawan TS Mills             0         0        0           0
## 2 DA Warner    S Dhawan TS Mills             0         0        0           0
## 3 DA Warner    S Dhawan TS Mills             0         0        0           0
## 4 DA Warner    S Dhawan TS Mills             0         0        0           0
## 5 DA Warner    S Dhawan TS Mills             0         2        0           0
##   noball_runs penalty_runs batsman_runs extra_runs total_runs player_dismissed
## 1           0            0            0          0          0
## 2           0            0            0          0          0
## 3           0            0            4          0          4
## 4           0            0            0          0          0
## 5           0            0            0          2          2
##   dismissal_kind fielder
## 1
## 2
## 3
## 4
## 5
```

```r
head(matches,5)
```

```
##   id season      city       date                       team1
## 1  1   2017 Hyderabad 2017-04-05         Sunrisers Hyderabad
## 2  2   2017      Pune 2017-04-06              Mumbai Indians
## 3  3   2017    Rajkot 2017-04-07               Gujarat Lions
## 4  4   2017    Indore 2017-04-08      Rising Pune Supergiant
```

```
## 5  5   2017 Bangalore 2017-04-08 Royal Challengers Bangalore
##                         team2                 toss_winner toss_decision result
## 1 Royal Challengers Bangalore Royal Challengers Bangalore         field normal
## 2      Rising Pune Supergiant      Rising Pune Supergiant         field normal
## 3       Kolkata Knight Riders       Kolkata Knight Riders         field normal
## 4             Kings XI Punjab             Kings XI Punjab         field normal
## 5            Delhi Daredevils Royal Challengers Bangalore           bat normal
##   dl_applied                      winner win_by_runs win_by_wickets
## 1          0         Sunrisers Hyderabad          35              0
## 2          0      Rising Pune Supergiant           0              7
## 3          0       Kolkata Knight Riders           0             10
## 4          0             Kings XI Punjab           0              6
## 5          0 Royal Challengers Bangalore          15              0
##   player_of_match                                     venue        umpire1
## 1    Yuvraj Singh Rajiv Gandhi International Stadium, Uppal    AY Dandekar
## 2       SPD Smith   Maharashtra Cricket Association Stadium A Nand Kishore
## 3         CA Lynn    Saurashtra Cricket Association Stadium    Nitin Menon
## 4      GJ Maxwell                    Holkar Cricket Stadium   AK Chaudhary
## 5       KM Jadhav                     M Chinnaswamy Stadium
##         umpire2 umpire3
## 1      NJ Llong
## 2        S Ravi
## 3     CK Nandan
## 4 C Shamshuddin
## 5
```

```r
View(deliveries)
View(matches)
```

Frame your objectives

1. Auction Model for Player Selection and Team Recommendation using Player Performance Index

2. Scoring Pattern Analysis for knowing the batting pattern of teams

3. Win Predictor for knowing who is winning the match

Data Cleaning

Cleaning deliveries.csv

```r
sum(is.na(deliveries))
```

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

```r
deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Royal Challengers Bangalore", "RCB"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Sunrisers Hyderabad", "SRH"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Chennai Super Kings", "CSK"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Mumbai Indians", "MI"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Kolkata Knight Riders", "KKR"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Delhi Daredevils", "DC"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Delhi Capitals", "DC"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Rajasthan Royals", "RR"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Deccan Chargers", "SRH"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Rising Pune Supergiant", "RPS"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Rising Pune Supergiants", "RPS"))

deliveries = deliveries%>%
  mutate(batting_team = replace(batting_team,batting_team == "Pune Warriors", "RPS"))

deliveries= deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Kings XI Punjab", "KXIP"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Gujarat Lions", "GL"))

deliveries = deliveries %>%
  mutate(batting_team = replace(batting_team,batting_team == "Kochi Tuskers Kerala", "KTK"))

unique(deliveries$batting_team)
```

Replacing the names of the teams with their respective abbreviations

```
##  [1] "SRH"  "RCB"  "MI"   "RPS"  "GL"   "KKR"  "KXIP" "DC"   "CSK"  "RR"
## [11] "KTK"
```

```r
deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Royal Challengers Bangalore", "RCB"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Sunrisers Hyderabad", "SRH"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Chennai Super Kings", "CSK"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Mumbai Indians", "MI"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Kolkata Knight Riders", "KKR"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Delhi Daredevils", "DC"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Delhi Capitals", "DC"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Rajasthan Royals", "RR"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Deccan Chargers", "SRH"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Rising Pune Supergiant", "RPS"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Rising Pune Supergiants", "RPS"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Pune Warriors", "RPS"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Kings XI Punjab", "KXIP"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Gujarat Lions", "GL"))

deliveries = deliveries %>%
  mutate(bowling_team = replace(bowling_team,bowling_team == "Kochi Tuskers Kerala", "KTK"))

unique(deliveries$bowling_team)
```

```
##  [1] "RCB"  "SRH"  "RPS"  "MI"   "KKR"  "GL"   "KXIP" "DC"   "CSK"  "RR"
## [11] "KTK"
```

```r
deliveries$dismissal_kind = deliveries$dismissal_kind %>% as.character()
deliveries$wicket = ifelse((deliveries$dismissal_kind=="" | deliveries$dismissal_kind=="run out"), 0, 1)
deliveries$dismissal = ifelse((deliveries$dismissal_kind==""), 0, 1)
deliveries$dot = ifelse((deliveries$total_runs==0), 1, 0)
deliveries$boundary = ifelse((deliveries$total_runs==4 | deliveries$total_runs==6), 1, 0)
deliveries$singles = if_else((deliveries$total_runs==1 | deliveries$total_runs==2 | deliveries$total_ru
```

Adding some more columns to the dataset for clarity of mode of dismissal and runs scored

---

Cleaning matches.csv

```r
sum(is.na(matches))
```

Dropping the Umpire 1 ,2, 3 Column from the dataset since it is not required for analysis

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

```r
sum(is.na(matches$umpire3))
```

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

```r
matches$umpire3 = NULL
matches$umpire1 = NULL
matches$umpire2 = NULL
```

```r
matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Royal Challengers Bangalore", "RCB"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Sunrisers Hyderabad", "SRH"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Chennai Super Kings", "CSK"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Mumbai Indians", "MI"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Kolkata Knight Riders", "KKR"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Delhi Daredevils", "DC"))
```

```r
matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Delhi Capitals", "DC"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Rajasthan Royals", "RR"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Deccan Chargers", "SRH"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Rising Pune Supergiant", "RPS"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Rising Pune Supergiants", "RPS"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Pune Warriors", "RPS"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Kings XI Punjab", "KXIP"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Gujarat Lions", "GL"))

matches = matches %>%
  mutate(team1 = replace(team1,team1 == "Kochi Tuskers Kerala", "KTK"))

unique(matches$team1)
```

Renaming names of teams with their abbreviations

```
##  [1] "SRH"  "MI"   "GL"   "RPS"  "RCB"  "KKR"  "DC"   "KXIP" "CSK"  "RR"
## [11] "KTK"
```

```r
matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Royal Challengers Bangalore", "RCB"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Sunrisers Hyderabad", "SRH"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Chennai Super Kings", "CSK"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Mumbai Indians", "MI"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Kolkata Knight Riders", "KKR"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Delhi Daredevils", "DC"))

matches = matches %>%
```

```r
  mutate(team2 = replace(team2,team2 == "Delhi Capitals", "DC"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Rajasthan Royals", "RR"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Deccan Chargers", "SRH"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Rising Pune Supergiant", "RPS"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Rising Pune Supergiants", "RPS"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Pune Warriors", "RPS"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Kings XI Punjab", "KXIP"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Gujarat Lions", "GL"))

matches = matches %>%
  mutate(team2 = replace(team2,team2 == "Kochi Tuskers Kerala", "KTK"))

unique(matches$team2)
```

```
##  [1] "RCB"  "RPS"  "KKR"  "KXIP" "DC"   "SRH"  "MI"   "GL"   "RR"   "CSK"
## [11] "KTK"
```

```r
matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Royal Challengers Bangalore", "RCB"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Sunrisers Hyderabad", "SRH"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Chennai Super Kings", "CSK"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Mumbai Indians", "MI"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Kolkata Knight Riders", "KKR"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Delhi Daredevils", "DC"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Delhi Capitals", "DC"))

matches = matches %>%
```

```r
  mutate(toss_winner = replace(toss_winner,toss_winner == "Rajasthan Royals", "RR"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Deccan Chargers", "SRH"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Rising Pune Supergiant", "RPS"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Rising Pune Supergiants", "RPS"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Pune Warriors", "RPS"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Kings XI Punjab", "KXIP"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Gujarat Lions", "GL"))

matches = matches %>%
  mutate(toss_winner = replace(toss_winner,toss_winner == "Kochi Tuskers Kerala", "KTK"))

unique(matches$toss_winner)
```

```
##  [1] "RCB"  "RPS"  "KKR"  "KXIP" "SRH"  "MI"   "GL"   "DC"   "CSK"  "RR"
## [11] "KTK"
```

```r
matches = matches %>%
  mutate(winner = replace(winner,winner == "Royal Challengers Bangalore", "RCB"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Sunrisers Hyderabad", "SRH"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Chennai Super Kings", "CSK"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Mumbai Indians", "MI"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Kolkata Knight Riders", "KKR"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Delhi Daredevils", "DC"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Delhi Capitals", "DC"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Rajasthan Royals", "RR"))

matches = matches %>%
```

```r
  mutate(winner = replace(winner,winner == "Deccan Chargers", "SRH"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Rising Pune Supergiant", "RPS"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Rising Pune Supergiants", "RPS"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Pune Warriors", "RPS"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Kings XI Punjab", "KXIP"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Gujarat Lions", "GL"))

matches = matches %>%
  mutate(winner = replace(winner,winner == "Kochi Tuskers Kerala", "KTK"))

matches  = matches %>%
   mutate(winner = replace(winner,winner == "", "None"))

unique(matches$winner)
```

```
##  [1] "SRH"  "RPS"  "KKR"  "KXIP" "RCB"  "MI"   "DC"   "GL"   "CSK"  "RR"
## [11] "KTK"  "None"
```

```r
matches = matches %>%
  mutate(city = replace(city,city == "","Dubai"))
unique(matches$city)
```

Replacing missing value in the City Column

```
##  [1] "Hyderabad"      "Pune"           "Rajkot"         "Indore"
##  [5] "Bangalore"      "Mumbai"         "Kolkata"        "Delhi"
##  [9] "Chandigarh"     "Kanpur"         "Jaipur"         "Chennai"
## [13] "Cape Town"      "Port Elizabeth" "Durban"         "Centurion"
## [17] "East London"    "Johannesburg"   "Kimberley"      "Bloemfontein"
## [21] "Ahmedabad"      "Cuttack"        "Nagpur"         "Dharamsala"
## [25] "Kochi"          "Visakhapatnam"  "Raipur"         "Ranchi"
## [29] "Abu Dhabi"      "Sharjah"        "Dubai"          "Mohali"
## [33] "Bengaluru"
```

Merging 2 datasets

```r
dataset = merge(deliveries,matches,by.x = "match_id", by.y = "id")
View(dataset)
summary(dataset)
```

```
##     match_id         inning      batting_team       bowling_team
##  Min.   :    1   Min.   :1.000   Length:179078      Length:179078
##  1st Qu.:  190   1st Qu.:1.000   Class :character   Class :character
##  Median :  379   Median :1.000   Mode  :character   Mode  :character
##  Mean   : 1802   Mean   :1.483
##  3rd Qu.:  567   3rd Qu.:2.000
##  Max.   :11415   Max.   :5.000
##       over            ball          batsman           non_striker
##  Min.   : 1.00   Min.   :1.000   Length:179078      Length:179078
##  1st Qu.: 5.00   1st Qu.:2.000   Class :character   Class :character
##  Median :10.00   Median :4.000   Mode  :character   Mode  :character
##  Mean   :10.16   Mean   :3.616
##  3rd Qu.:15.00   3rd Qu.:5.000
##  Max.   :20.00   Max.   :9.000
##     bowler           is_super_over         wide_runs          bye_runs
##  Length:179078      Min.   :0.0000000   Min.   :0.00000   Min.   :0.000000
##  Class :character   1st Qu.:0.0000000   1st Qu.:0.00000   1st Qu.:0.000000
##  Mode  :character   Median :0.0000000   Median :0.00000   Median :0.000000
##                     Mean   :0.0004523   Mean   :0.03672   Mean   :0.004936
##                     3rd Qu.:0.0000000   3rd Qu.:0.00000   3rd Qu.:0.000000
##                     Max.   :1.0000000   Max.   :5.00000   Max.   :4.000000
##   legbye_runs       noball_runs        penalty_runs      batsman_runs
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.0e+00   Min.   :0.000
##  1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.0e+00   1st Qu.:0.000
##  Median :0.00000   Median :0.000000   Median :0.0e+00   Median :1.000
##  Mean   :0.02114   Mean   :0.004183   Mean   :5.6e-05   Mean   :1.247
##  3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.0e+00   3rd Qu.:1.000
##  Max.   :5.00000   Max.   :5.000000   Max.   :5.0e+00   Max.   :7.000
##    extra_runs        total_runs     player_dismissed   dismissal_kind
##  Min.   :0.00000   Min.   : 0.000   Length:179078      Length:179078
##  1st Qu.:0.00000   1st Qu.: 0.000   Class :character   Class :character
##  Median :0.00000   Median : 1.000   Mode  :character   Mode  :character
##  Mean   :0.06703   Mean   : 1.314
##  3rd Qu.:0.00000   3rd Qu.: 1.000
##  Max.   :7.00000   Max.   :10.000
##    fielder              wicket          dismissal           dot
##  Length:179078      Min.   :0.00000   Min.   :0.00000   Min.   :0.0000
##  Class :character   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000
##  Mode  :character   Median :0.00000   Median :0.00000   Median :0.0000
##                     Mean   :0.04457   Mean   :0.04933   Mean   :0.3518
##                     3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:1.0000
##                     Max.   :1.00000   Max.   :1.00000   Max.   :1.0000
##     boundary          singles           season         city
##  Min.   :0.0000   Min.   :0.0000   Min.   :2008   Length:179078
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:2011   Class :character
##  Median :0.0000   Median :0.0000   Median :2013   Mode  :character
##  Mean   :0.1605   Mean   :0.4851   Mean   :2013
##  3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:2016
##  Max.   :1.0000   Max.   :1.0000   Max.   :2019
```

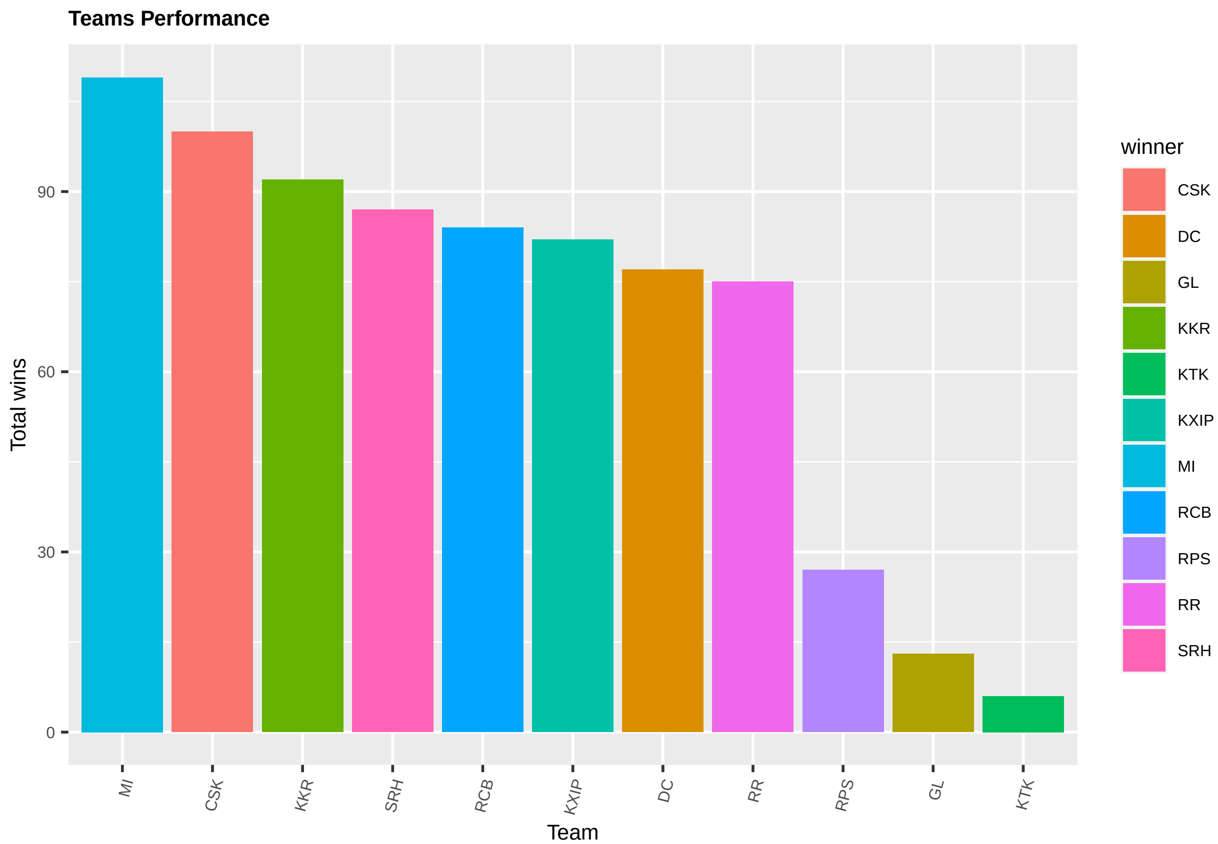
```
##      date               team1              team2            toss_winner
##  Length:179078      Length:179078      Length:179078      Length:179078
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  toss_decision         result            dl_applied         winner
##  Length:179078      Length:179078      Min.   :0.00000   Length:179078
##  Class :character   Class :character   1st Qu.:0.00000   Class :character
##  Mode  :character   Mode  :character   Median :0.00000   Mode  :character
##                                        Mean   :0.01791
##                                        3rd Qu.:0.00000
##                                        Max.   :1.00000
##   win_by_runns    win_by_wickets   player_of_match       venue
##  Min.   :  0.0   Min.   : 0.000   Length:179078      Length:179078
##  1st Qu.:  0.0   1st Qu.: 0.000   Class :character   Class :character
##  Median :  0.0   Median : 3.000   Mode  :character   Mode  :character
##  Mean   : 13.4   Mean   : 3.262
##  3rd Qu.: 19.0   3rd Qu.: 6.000
##  Max.   :146.0   Max.   :10.000
```

Basic Analysis

1. Top 10 Teams with Most Wins

```r
dataset %>%
 filter(result == "normal" | result == "tie") %>%
 group_by(winner) %>%
 summarise(Wins = n_distinct(match_id)) %>%
 ggplot(aes(x = reorder(winner, -Wins), y = Wins))+geom_bar(aes(fill = winner), stat = "identity")+
 labs(title = "Teams Performance", x = "Team", y = "Total wins")+
 theme(axis.text.x=element_text(angle=75, hjust=1), plot.title = element_text(size = 8, face = "bold"))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```



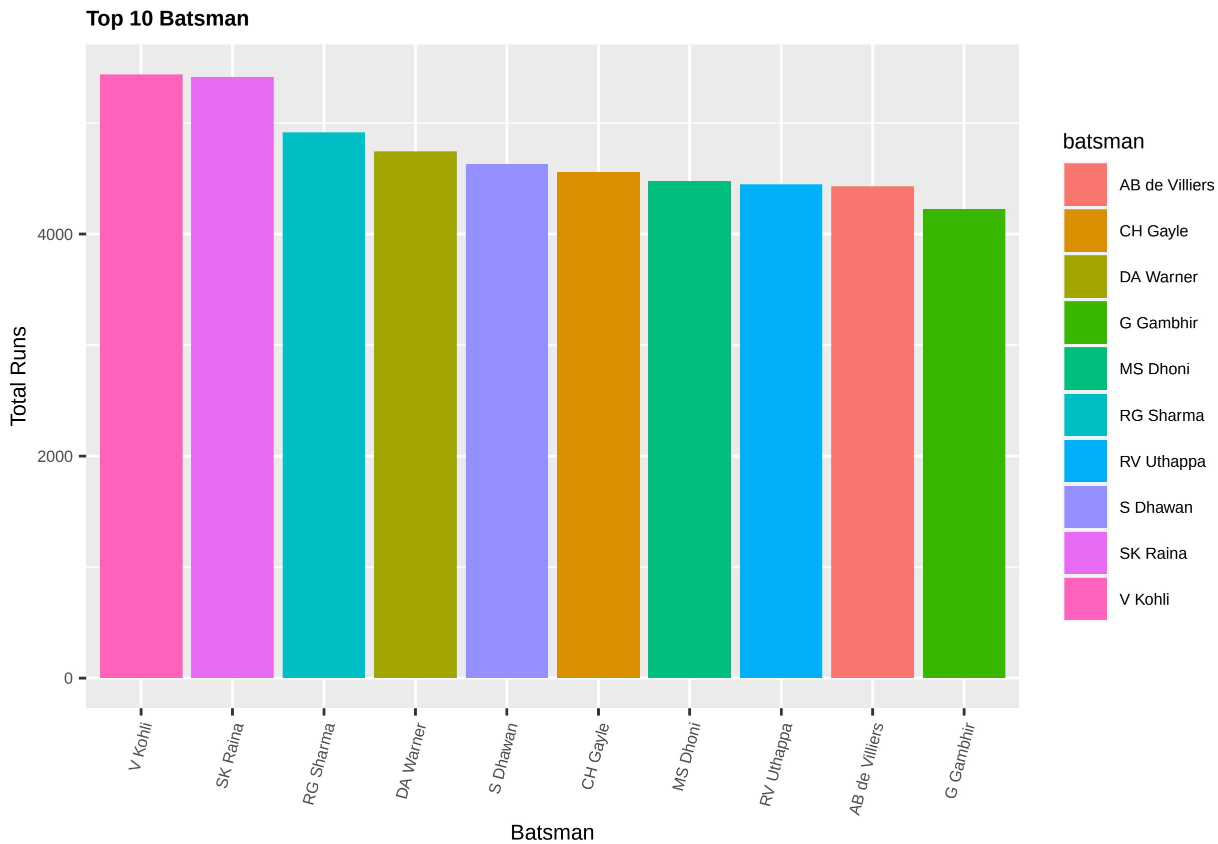
Inference: MI and CSK are the best teams in the competition with more than 80 wins across the 12 years of the tournament.

---

2. Top 10 Batsman with Most Runs

```
dataset %>%
  group_by(batsman) %>%
  summarise(total_runs = sum(batsman_runs)) %>%
  arrange(desc(total_runs)) %>%
  top_n(n = 10, wt = total_runs) %>%
  ggplot(aes(x = reorder(batsman, -total_runs), y = total_runs))+
  geom_bar(aes(fill = batsman),stat = "identity")+
  labs(title = "Top 10 Batsman", x = "Batsman", y = "Total Runs")+
  theme(axis.text.x=element_text(angle=75, hjust=1), plot.title = element_text(size = 8, face = "bold"))
```

```
## 'summarise()' ungrouping output (override with '.groups' argument)
```



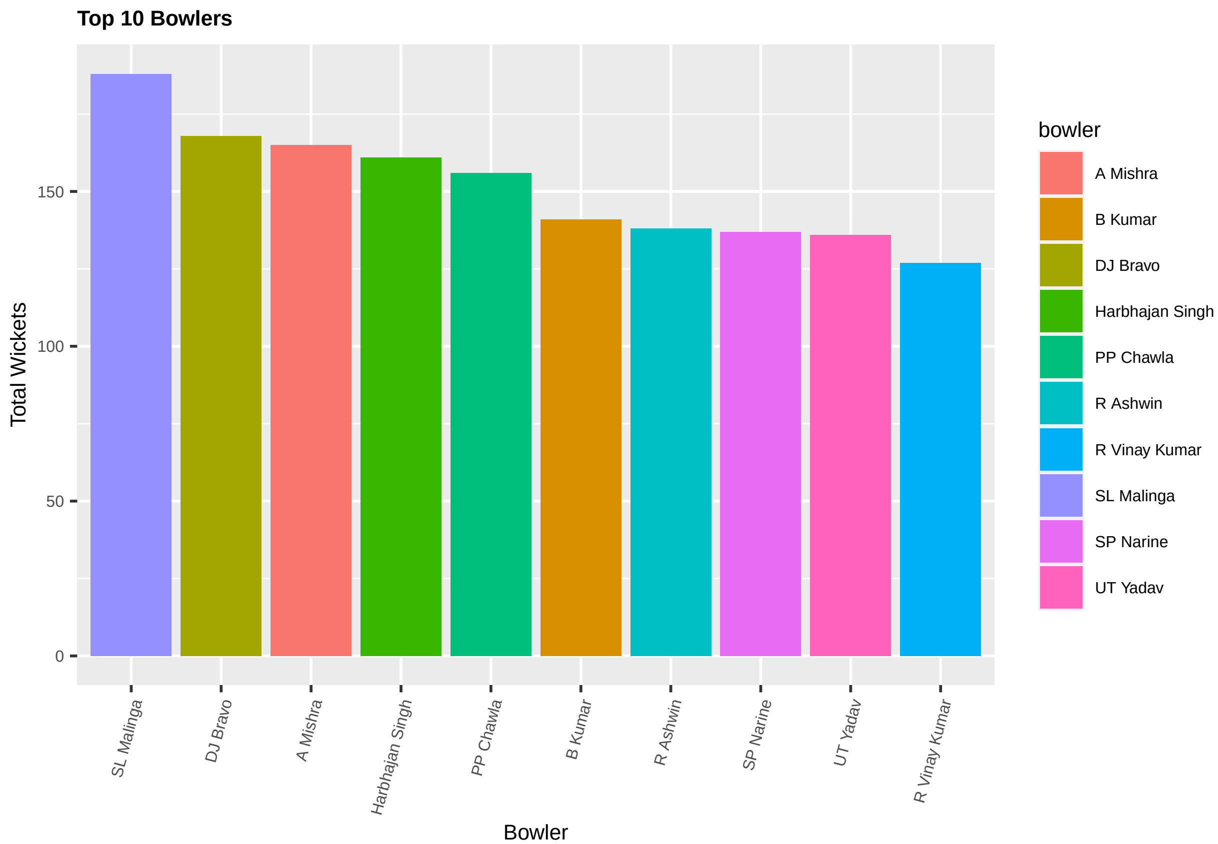
Inference: Virat Kohli and Raina have the best performers with the bat with more than 4000 runs in 12 years of the tournament.

---

3. Top 10 Bowlers with Most Wickets

```
dataset %>%
  group_by(bowler) %>%
  summarise(total_wickets = sum(dismissal)) %>%
  arrange(desc(total_wickets)) %>%
  top_n(n= 10, wt = total_wickets) %>%
  ggplot(aes(x = reorder(bowler,-total_wickets), y= total_wickets))+
  geom_bar(aes(fill= bowler), stat = "identity")+
  labs(title = "Top 10 Bowlers", x = "Bowler", y = "Total Wickets")+
  theme(axis.text.x=element_text(angle=75, hjust=1), plot.title = element_text(size = 8, face = "bold")
```

```
## 'summarise()' ungrouping output (override with '.groups' argument)
```



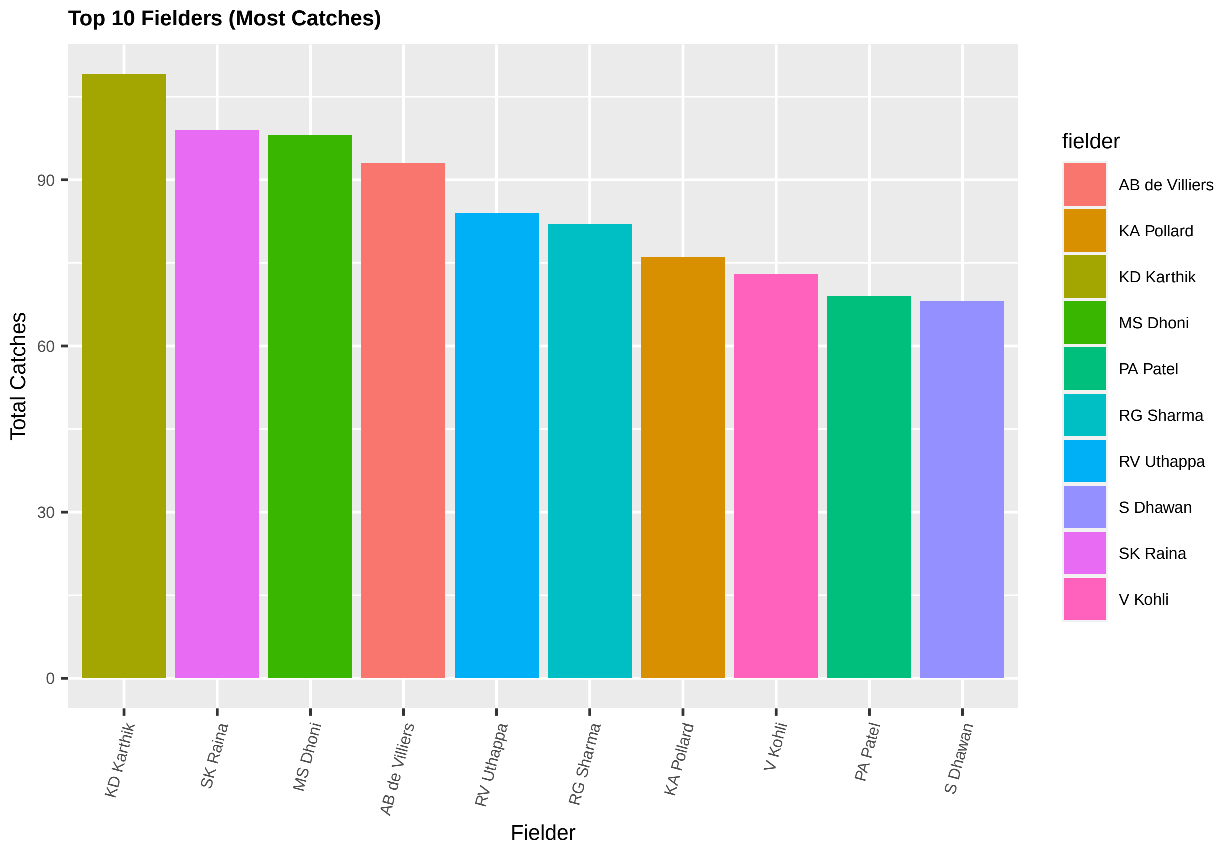
Inference: Malinga and Bravo have been the highest wicket takers in the tournament with more than 150 wickets. Amit Mishra is the highest wicket taker among spinners.

---

4. Top 10 Fielders with Most Catches

```
dataset %>%
group_by(fielder) %>%
summarise(total_catches = length(dismissal_kind[dismissal_kind=="caught"])) %>%
arrange(desc(total_catches)) %>%
top_n(n= 10, wt = total_catches) %>%
ggplot(aes(x = reorder(fielder, -total_catches), y= total_catches))+
geom_bar(aes(fill= fielder), stat = "identity")+
labs(title = "Top 10 Fielders (Most Catches)", x = "Fielder", y = "Total Catches")+
theme(axis.text.x=element_text(angle=75, hjust=1), plot.title = element_text(size = 8, face = "bold"),t
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```



Inference: MS Dhoni and Dinesh Karthik have been the top wicket keepers in the tournament while Raina has been the best fielder in the league with the most catches.

---

5. Team Performance at Home and Away Matches through win percentage

```
t <- dataset %>%
  filter((result=="normal" | result == "tie") & batting_team %in% c("KKR","CSK","DC","MI","SRH","RCB","

kkr_match_played <- t %>%
  filter(batting_team=="KKR") %>%
  mutate(ground_type = if_else(city == "Kolkata","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_match_played = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
kkr_match_won <- t %>%
  filter(batting_team=="KKR" & winner == "KKR") %>%
  mutate(ground_type = if_else(city == "Kolkata","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_win = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
KKR<-merge(kkr_match_played, kkr_match_won, by ="ground_type")

KKR<-KKR %>%
  mutate(winning_perc = (total_win/total_match_played)*100,
         team = "KKR")

csk_match_played<-t %>%
  filter(batting_team=="CSK") %>%
  mutate(ground_type = if_else(city == "Chennai","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_match_played = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
csk_match_won<-t %>%
  filter(batting_team=="CSK" & winner == "CSK") %>%
  mutate(ground_type = if_else(city == "Chennai","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_win = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
CSK<-merge(csk_match_played, csk_match_won, by ="ground_type")

CSK<-CSK %>%
  mutate(winning_perc = (total_win/total_match_played)*100,
         team ="CSK")

mi_match_played<-t %>%
  filter(batting_team=="MI") %>%
  mutate(ground_type = if_else(city == "Mumbai","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_match_played = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
mi_match_won<-t %>%
  filter(batting_team=="MI" & winner == "MI") %>%
  mutate(ground_type = if_else(city == "Mumbai","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_win = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
MI<-merge(mi_match_played, mi_match_won, by ="ground_type")

MI<-MI %>%
  mutate(winning_perc = (total_win/total_match_played)*100,
```

```r
         team= "MI")

KXIP_match_played<-t %>%
  filter(batting_team=="KXIP") %>%
  mutate(ground_type = if_else(city == "Chandigarh","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_match_played = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
KXIP_match_won<-t %>%
  filter(batting_team=="KXIP" & winner == "KXIP") %>%
  mutate(ground_type = if_else(city == "Chandigarh","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_win = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
KXIP<-merge(KXIP_match_played, KXIP_match_won, by ="ground_type")

KXIP<-KXIP %>%
  mutate(winning_perc = (total_win/total_match_played)*100,
         team="KXIP")

RR_match_played<-t %>%
  filter(batting_team=="RR") %>%
  mutate(ground_type = if_else(city == "Jaipur","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_match_played = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
RR_match_won<-t %>%
  filter(batting_team=="RR" & winner == "RR") %>%
  mutate(ground_type = if_else(city == "Jaipur","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_win = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
RR<-merge(RR_match_played, RR_match_won, by ="ground_type")

RR<-RR %>%
  mutate(winning_perc = (total_win/total_match_played)*100,
         team = "RR")

RCB_match_played<-t %>%
  filter(batting_team=="RCB") %>%
  mutate(ground_type = if_else(city == "Bangalore","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_match_played = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
RCB_match_won<-t %>%
  filter(batting_team=="RCB" & winner == "RCB") %>%
  mutate(ground_type = if_else(city == "Bangalore","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_win = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
RCB<-merge(RCB_match_played, RCB_match_won, by ="ground_type")

RCB<-RCB %>%
  mutate(winning_perc = (total_win/total_match_played)*100,
         team ="RCB")

DC_match_played<-t %>%
  filter(batting_team=="DC") %>%
  mutate(ground_type = if_else(city == "Delhi","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_match_played = n_distinct(match_id))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
DC_match_won<-t %>%
  filter(batting_team=="DC" & winner == "DC") %>%
  mutate(ground_type = if_else(city == "Delhi","Home","Away")) %>%
  group_by(ground_type) %>%
  summarise(total_win = n_distinct(match_id))
```

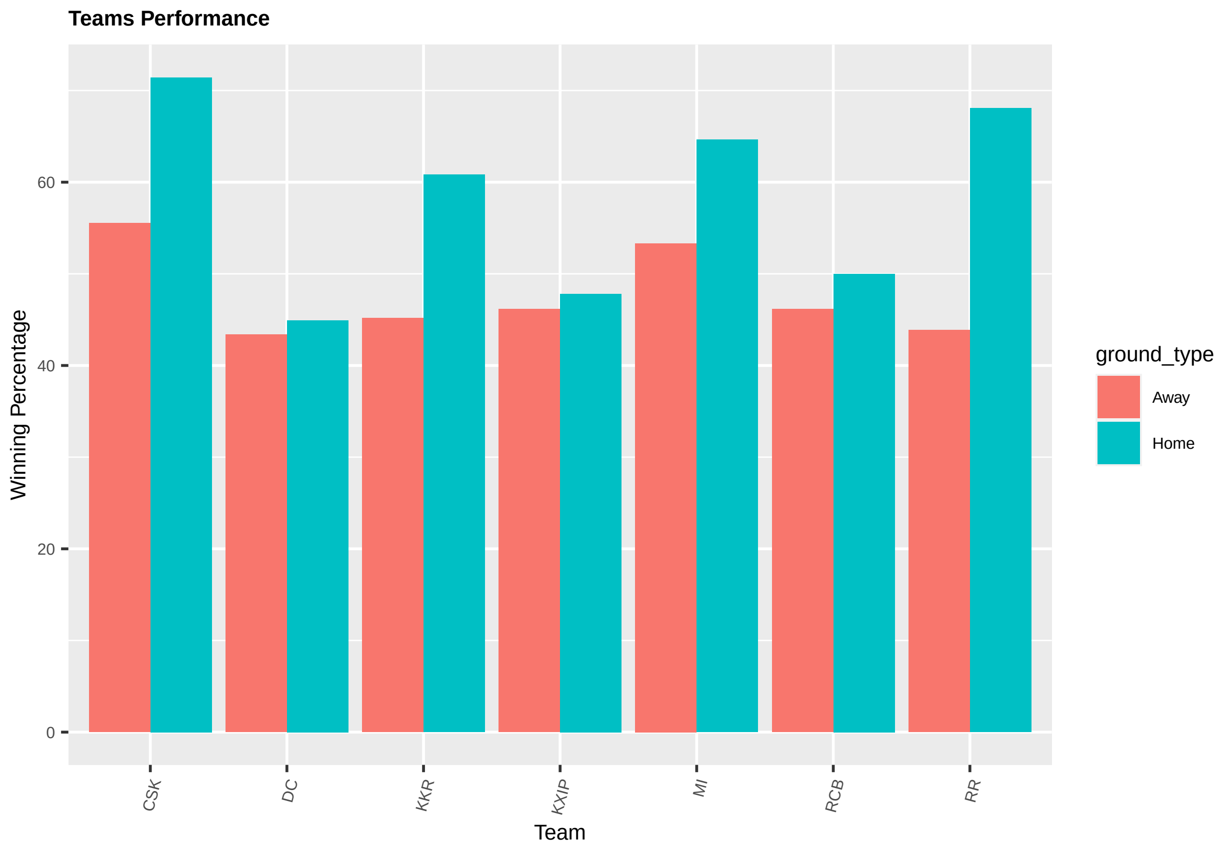
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
DC<-merge(DC_match_played, DC_match_won, by ="ground_type")

DC<-DC %>%
  mutate(winning_perc = (total_win/total_match_played)*100,
         team = "DC")


team_performances<-rbind(CSK, DC,KKR,MI,KXIP,RCB,RR)

team_performances %>%
ggplot(aes(x = team, y =winning_perc,fill = ground_type))+
  geom_bar(stat = "identity", position = "dodge")+
  labs(title = "Teams Performance", x = "Team", y = "Winning Percentage")+
  theme(axis.text.x=element_text(angle=75, hjust=1), plot.title = element_text(size = 8, face = "bold")
```



Inference: CSK and RR have the highest win percentage at their home grounds compared to other teams. CSK and MI have been the best performing teams away from their home grounds. This shows why they are one of the best franchises in the tournament because of their ability to maximise their home advantage and win almost 50 percentage of their away matches as well.

Team Wise Performance Analysis

Phase Wise Analysis of Teams

```
team_pp_runs = dataset %>%
  filter(over<=6, is_super_over == 0) %>%
  group_by(batting_team,match_id) %>%
  summarise(pp_runs = sum(total_runs)) %>%
  arrange(desc(pp_runs))
```

PowerPlay Analysis

```
## `summarise()` regrouping output by 'batting_team' (override with `.groups` argument)
```

```r
team_mean_pp_runs = team_pp_runs %>%
  group_by(batting_team) %>%
  summarise(avg_pp_runs = mean(pp_runs)) %>%
  arrange(desc(avg_pp_runs))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
team_pp_wickets = dataset %>%
  filter(over<=6, is_super_over == 0) %>%
  group_by(bowling_team,match_id) %>%
  summarise(pp_wickets = sum(dismissal)) %>%
  arrange(desc(pp_wickets))
```

```
## `summarise()` regrouping output by 'bowling_team' (override with `.groups` argument)
```

```r
team_mean_pp_wickets = team_pp_wickets %>%
  group_by(bowling_team) %>%
  summarise(avg_pp_wickets = mean(pp_wickets)) %>%
  arrange(desc(avg_pp_wickets))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
team_mean_pp_runs
```

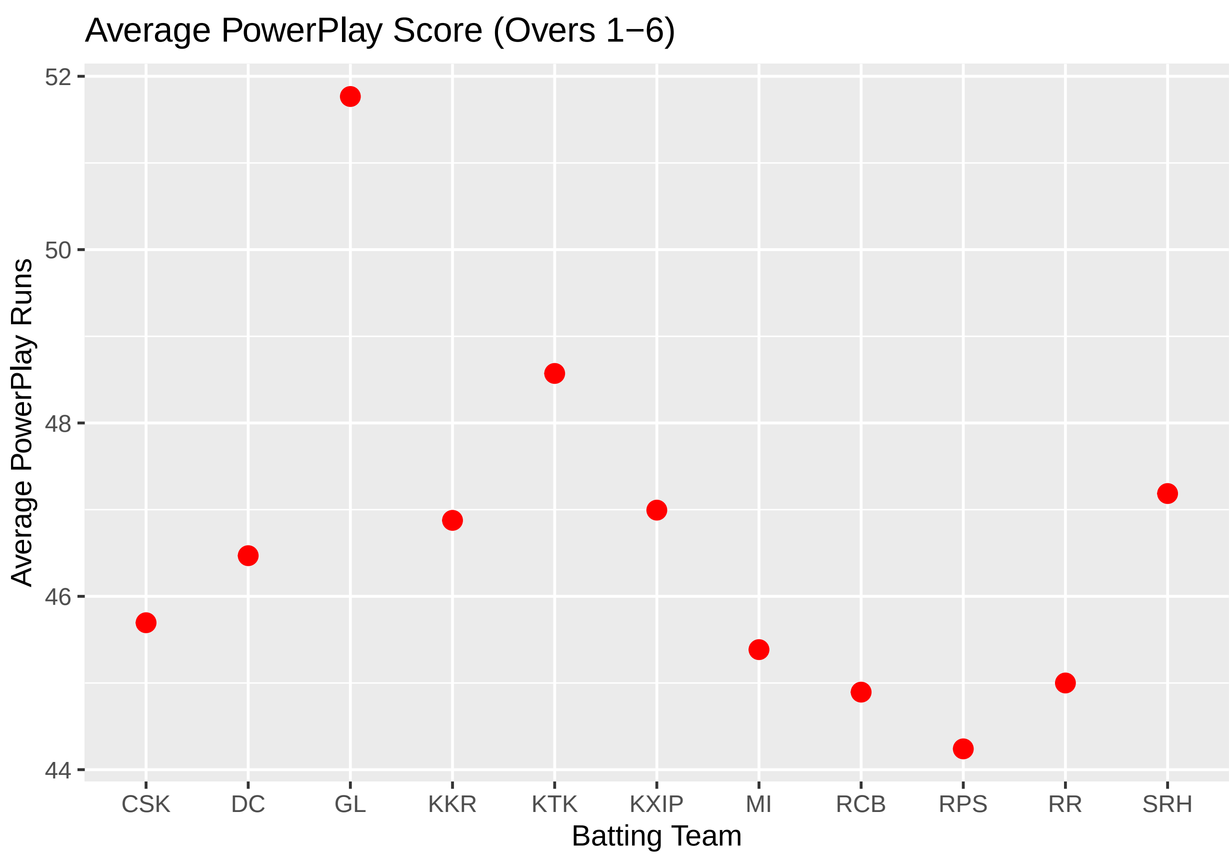
```
## # A tibble: 11 x 2
##    batting_team avg_pp_runs
##    <chr>              <dbl>
##  1 GL                  51.8
##  2 KTK                 48.6
##  3 SRH                 47.2
##  4 KXIP                47.0
##  5 KKR                 46.9
##  6 DC                  46.5
##  7 CSK                 45.7
##  8 MI                  45.4
##  9 RR                  45
## 10 RCB                 44.9
## 11 RPS                 44.2
```

```r
team_mean_pp_wickets
```

```
## # A tibble: 11 x 2
##    bowling_team avg_pp_wickets
##    <chr>                 <dbl>
##  1 RR                     1.58
##  2 KTK                    1.57
##  3 GL                     1.57
##  4 CSK                    1.56
##  5 MI                     1.48
##  6 SRH                    1.44
```

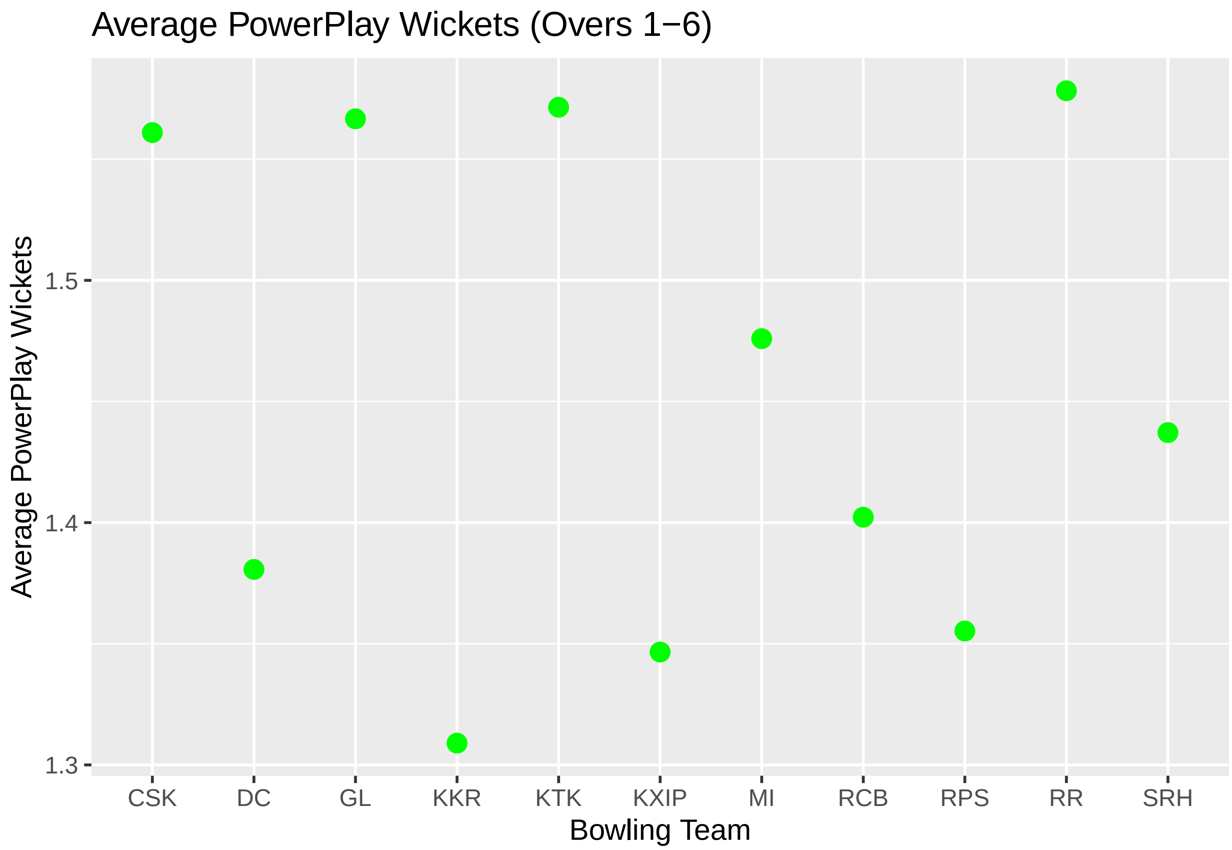
```
##   7 RCB                    1.40
##   8 DC                     1.38
##   9 RPS                    1.36
## 10 KXIP                   1.35
## 11 KKR                    1.31
```

```
ggplot(team_mean_pp_runs,aes(x=batting_team,y=avg_pp_runs))+
  geom_point(color="red",size=3)+labs(x="Batting Team",y="Average PowerPlay Runs",title="Average PowerPl
```




```
ggplot(team_mean_pp_wickets,aes(x=bowling_team,y=avg_pp_wickets))+
  geom_point(color="green",size=3)+labs(x="Bowling Team",y="Average PowerPlay Wickets",title="Average Po
```

Inference: GL and KTK have been the higher scoring teams in Powerplay across seasons.



Inference: KTK,RR,CSK,GL have picked the most wickets in Powerplay across seasons.

```
team_mo_runs = dataset %>%
  filter((over > 6 & over <=15), is_super_over == 0) %>%
  group_by(batting_team,match_id) %>%
  summarise(mo_runs = sum(total_runs)) %>%
  arrange(desc(mo_runs))
```

Middle Overs Analysis

```
## `summarise()` regrouping output by 'batting_team' (override with `.groups` argument)
```

```
team_mean_mo_runs = team_mo_runs %>%
  group_by(batting_team) %>%
  summarise(avg_mo_runs = mean(mo_runs)) %>%
  arrange(desc(avg_mo_runs))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
team_mo_wickets = dataset %>%
  filter((over > 6 & over <=15), is_super_over == 0) %>%
  group_by(bowling_team,match_id) %>%
  summarise(mo_wickets = sum(dismissal)) %>%
  arrange(desc(mo_wickets))
```

```
## `summarise()` regrouping output by 'bowling_team' (override with `.groups` argument)
```

```r
team_mean_mo_wickets = team_mo_wickets %>%
  group_by(bowling_team) %>%
  summarise(avg_mo_wickets = mean(mo_wickets)) %>%
  arrange(desc(avg_mo_wickets))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

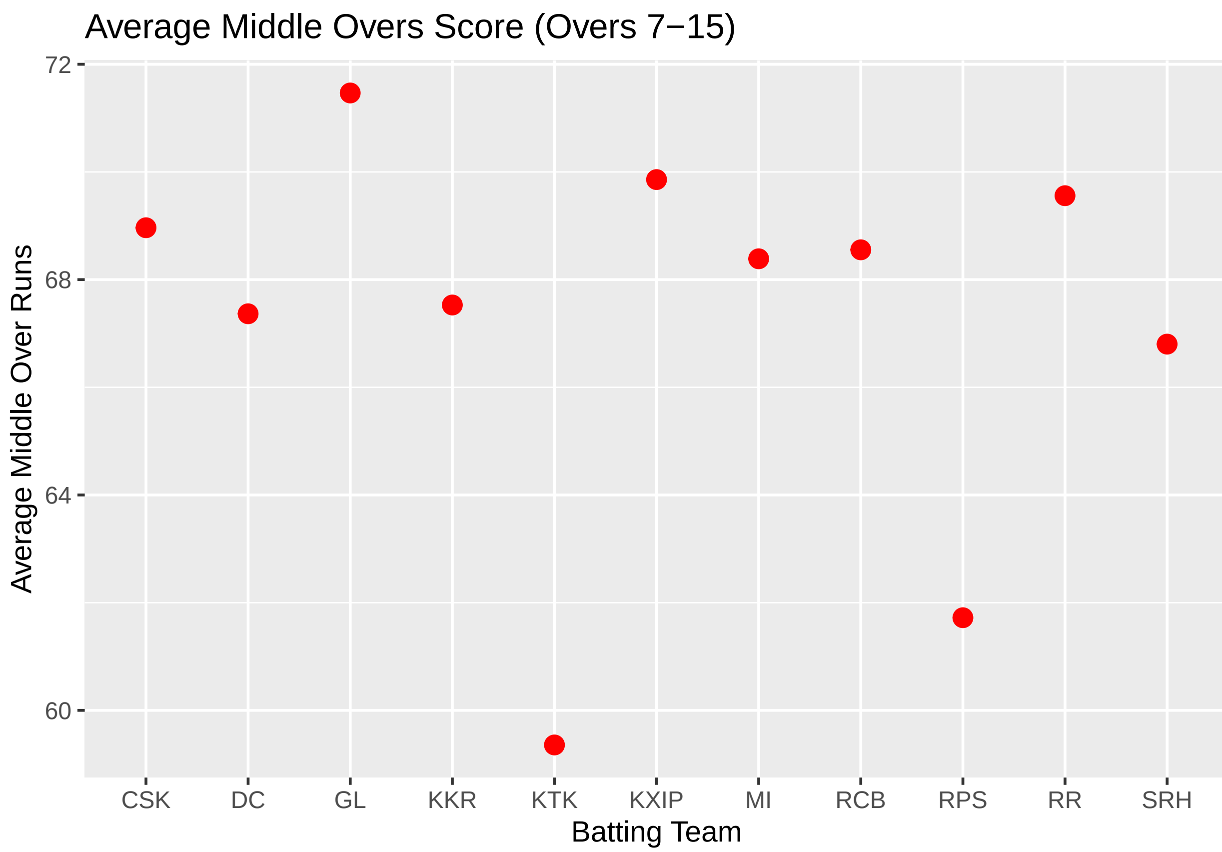
```r
team_mean_mo_runs
```

```
## # A tibble: 11 x 2
##    batting_team avg_mo_runs
##    <chr>              <dbl>
##  1 GL                  71.5
##  2 KXIP                69.9
##  3 RR                  69.6
##  4 CSK                 69.0
##  5 RCB                 68.6
##  6 MI                  68.4
##  7 KKR                 67.5
##  8 DC                  67.4
##  9 SRH                 66.8
## 10 RPS                 61.7
## 11 KTK                 59.4
```

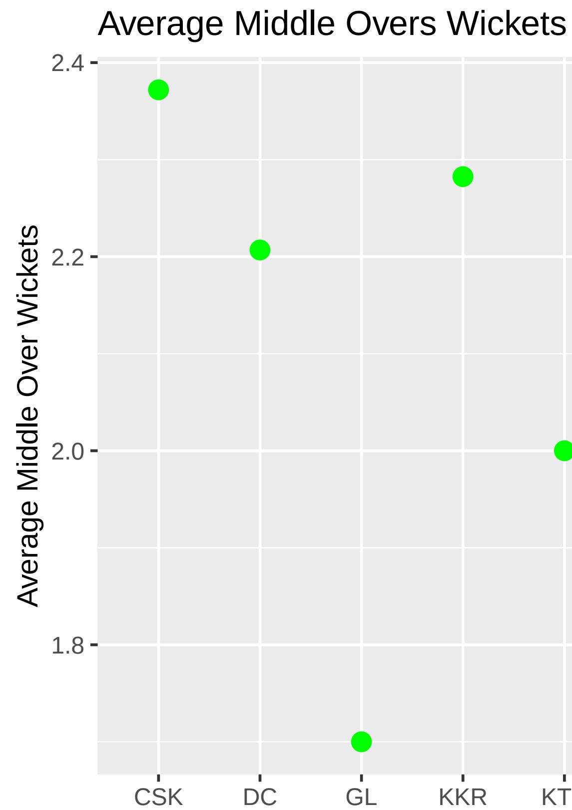
```r
team_mean_mo_wickets
```

```
## # A tibble: 11 x 2
##    bowling_team avg_mo_wickets
##    <chr>                 <dbl>
##  1 CSK                    2.37
##  2 MI                     2.35
##  3 SRH                    2.31
##  4 KKR                    2.28
##  5 RCB                    2.21
##  6 DC                     2.21
##  7 KXIP                   2.18
##  8 RR                     2.14
##  9 RPS                    2.12
## 10 KTK                    2
## 11 GL                     1.7
```

```r
ggplot(team_mean_mo_runs,aes(x=batting_team,y=avg_mo_runs))+
  geom_point(color="red",size=3)+labs(x="Batting Team",y="Average Middle Over Runs",title="Average Middl
```




```r
ggplot(team_mean_mo_wickets,aes(x=bowling_team,y=avg_mo_wickets))+
  geom_point(color="green",size=3)+labs(x="Bowling Team",y="Average Middle Over Wickets",title="Average
```



Inference: GL have the highest middle overs score across seasons

Inference: CSK have picked the most wickets in middle overs across seasons.

```
team_do_runs = dataset %>%
  filter((over > 15  & over <=20), is_super_over == 0) %>%
  group_by(batting_team,match_id) %>%
  summarise(do_runs = sum(total_runs)) %>%
  arrange(desc(do_runs))
```

Death Overs Analysis

```
## `summarise()` regrouping output by 'batting_team' (override with `.groups` argument)
```

```
team_mean_do_runs = team_do_runs %>%
  group_by(batting_team) %>%
  summarise(avg_do_runs = mean(do_runs)) %>%
  arrange(desc(avg_do_runs))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
team_do_wickets = dataset %>%
  filter((over > 15 & over <=20), is_super_over == 0) %>%
  group_by(bowling_team,match_id) %>%
  summarise(do_wickets = sum(dismissal)) %>%
  arrange(desc(do_wickets))
```

```
## 'summarise()' regrouping output by 'bowling_team' (override with '.groups' argument)
```

```r
team_mean_do_wickets = team_do_wickets %>%
  group_by(bowling_team) %>%
  summarise(avg_do_wickets = mean(do_wickets)) %>%
  arrange(desc(avg_do_wickets))
```

```
## 'summarise()' ungrouping output (override with '.groups' argument)
```

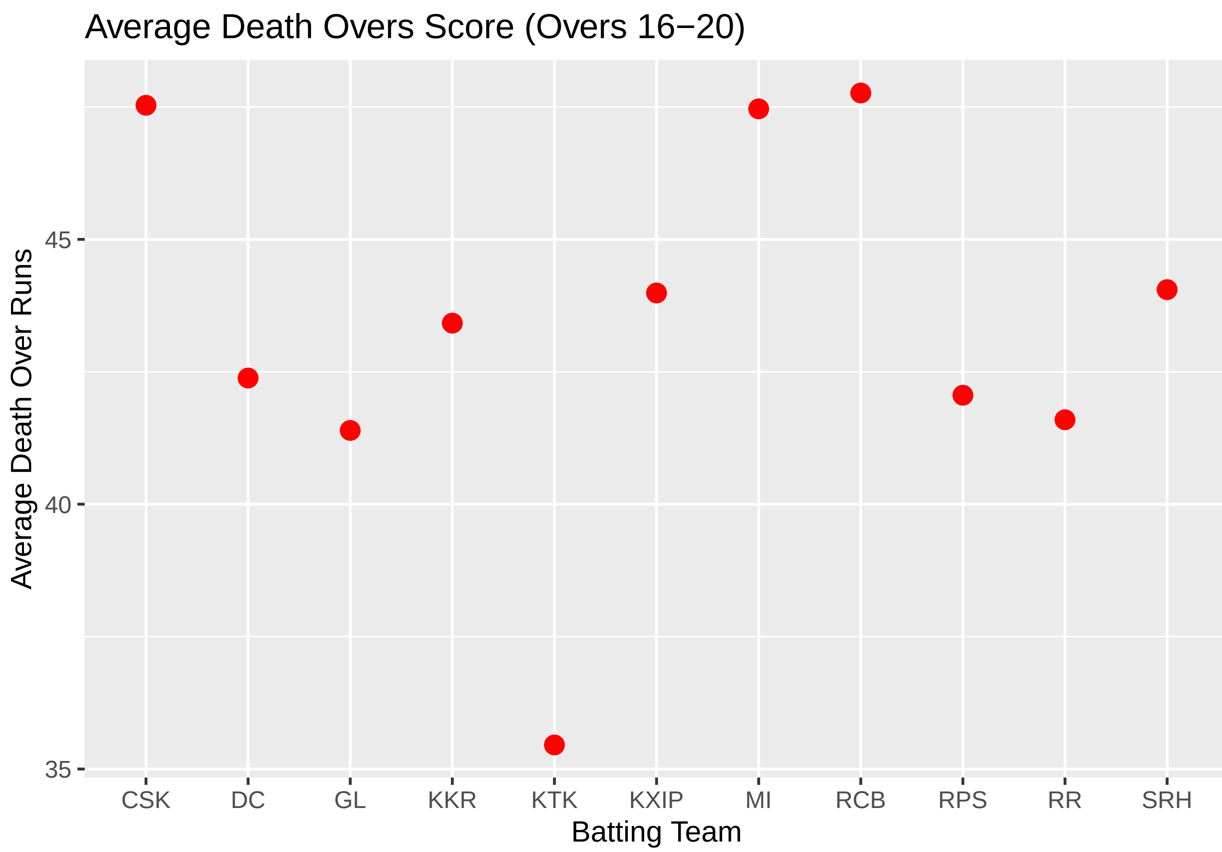
```r
team_mean_do_runs
```

```
## # A tibble: 11 x 2
##    batting_team avg_do_runs
##    <chr>              <dbl>
##  1 RCB                 47.8
##  2 CSK                 47.5
##  3 MI                  47.5
##  4 SRH                 44.1
##  5 KXIP                44.0
##  6 KKR                 43.4
##  7 DC                  42.4
##  8 RPS                 42.1
##  9 RR                  41.6
## 10 GL                  41.4
## 11 KTK                 35.5
```

```r
team_mean_do_wickets
```

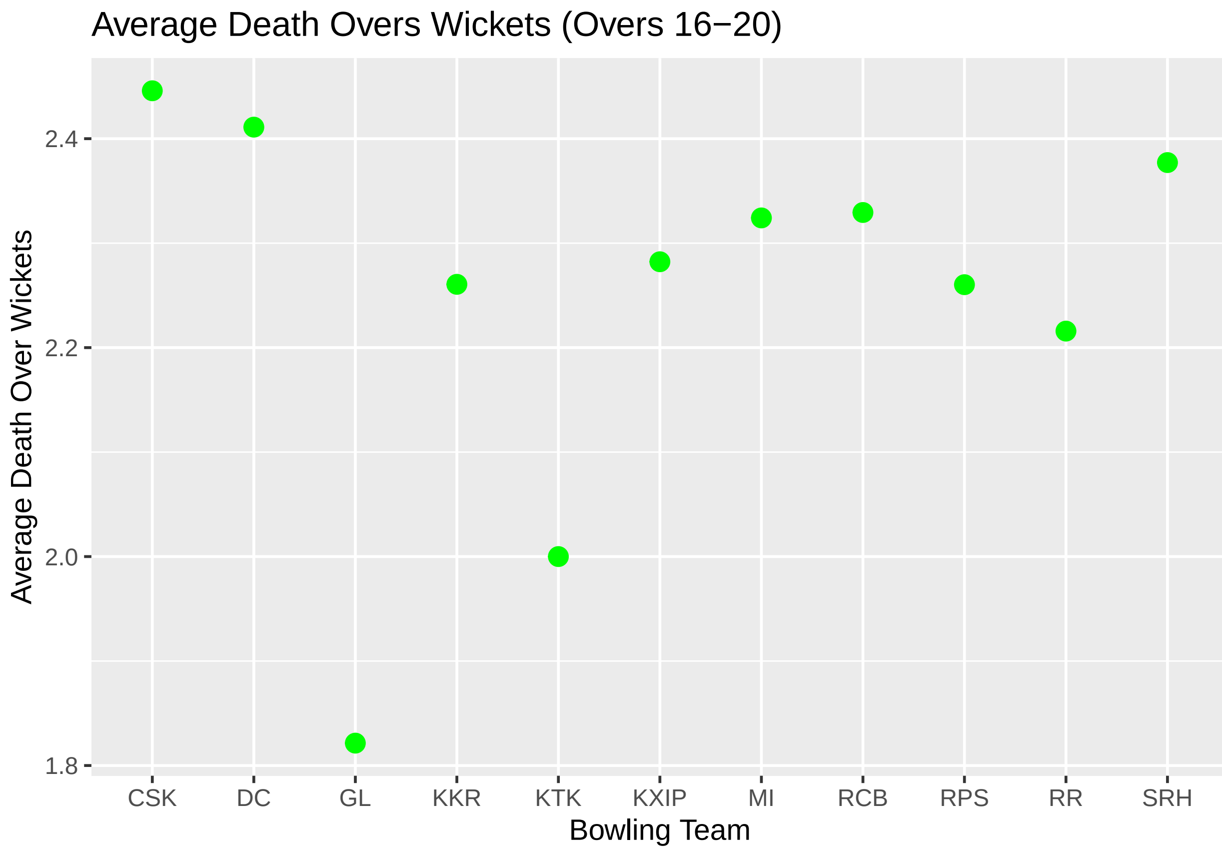
```
## # A tibble: 11 x 2
##    bowling_team avg_do_wickets
##    <chr>                 <dbl>
##  1 CSK                    2.45
##  2 DC                     2.41
##  3 SRH                    2.38
##  4 RCB                    2.33
##  5 MI                     2.32
##  6 KXIP                   2.28
##  7 KKR                    2.26
##  8 RPS                    2.26
##  9 RR                     2.22
## 10 KTK                    2
## 11 GL                     1.82
```

```r
ggplot(team_mean_do_runs,aes(x=batting_team,y=avg_do_runs))+
  geom_point(color="red",size=3)+labs(x="Batting Team",y="Average Death Over Runs",title="Average Death
```




```r
ggplot(team_mean_do_wickets,aes(x=bowling_team,y=avg_do_wickets))+
  geom_point(color="green",size=3)+labs(x="Bowling Team",y="Average Death Over Wickets",title="Average 
```

Inference: CSK, RCB and MI have scored the most runs in this phase across seasons



Inference: CSK have picked the most wickets in this phase across seasons.

Innings Wise Analysis of Teams

```
team_inning1_score = dataset %>%
  filter(inning == 1, is_super_over == 0) %>%
  group_by(match_id,batting_team) %>%
  summarise(first_inning_score = sum(total_runs))
```

```
## `summarise()` regrouping output by 'match_id' (override with `.groups` argument)
```

```
team_inning1_avg_score = team_inning1_score %>%
  group_by(batting_team) %>%
  summarise(first_inning_avg_score = mean(first_inning_score)) %>%
  arrange(desc(first_inning_avg_score))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
team_inning2_score = dataset %>%
  filter(inning == 2, is_super_over == 0) %>%
  group_by(match_id,batting_team) %>%
  summarise(second_inning_score = sum(total_runs))
```

```
## `summarise()` regrouping output by 'match_id' (override with `.groups` argument)
```

```r
team_inning2_avg_score = team_inning2_score %>%
  group_by(batting_team) %>%
  summarise(second_inning_avg_score = mean(second_inning_score)) %>%
  arrange(desc(second_inning_avg_score))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

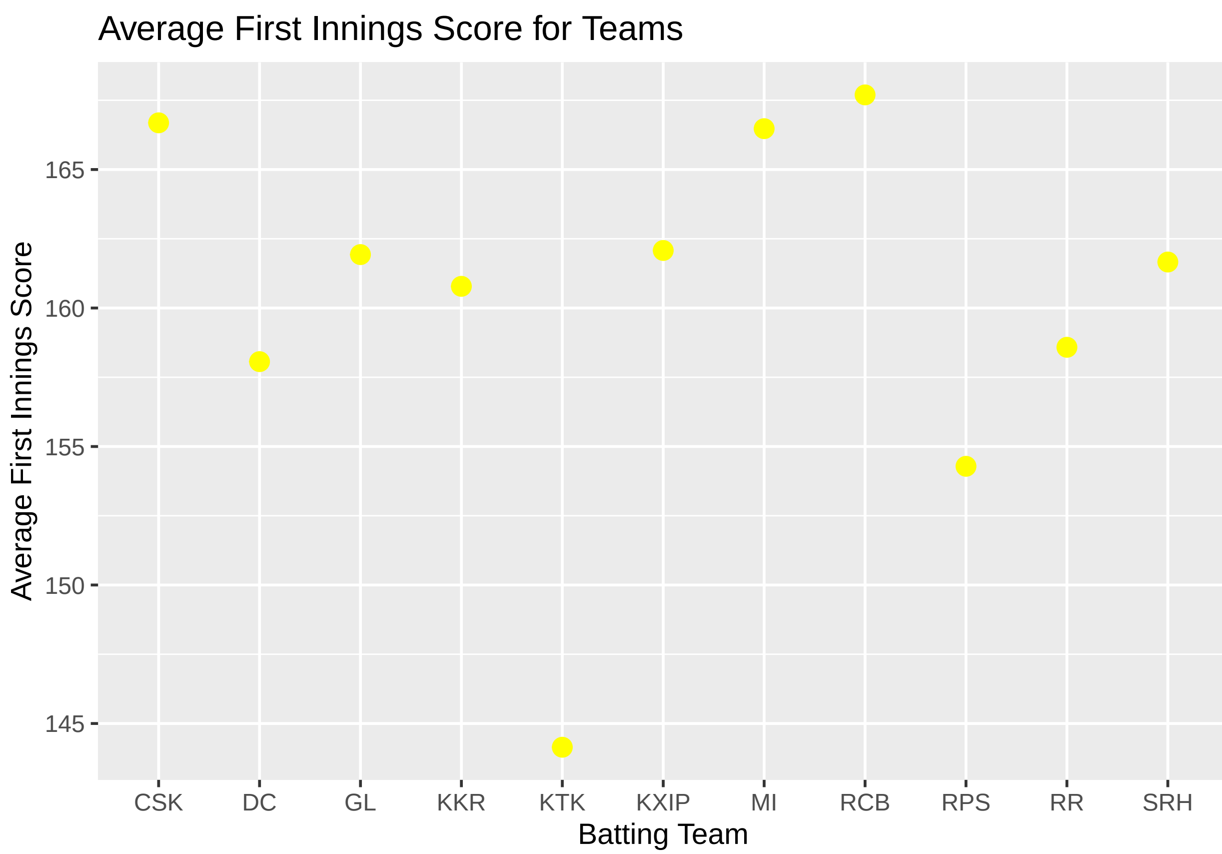
```r
team_inning1_avg_score
```

```
## # A tibble: 11 x 2
##    batting_team first_inning_avg_score
##    <chr>                         <dbl>
##  1 RCB                            168.
##  2 CSK                            167.
##  3 MI                             166.
##  4 KXIP                           162.
##  5 GL                             162.
##  6 SRH                            162.
##  7 KKR                            161.
##  8 RR                             159.
##  9 DC                             158.
## 10 RPS                            154.
## 11 KTK                            144.
```

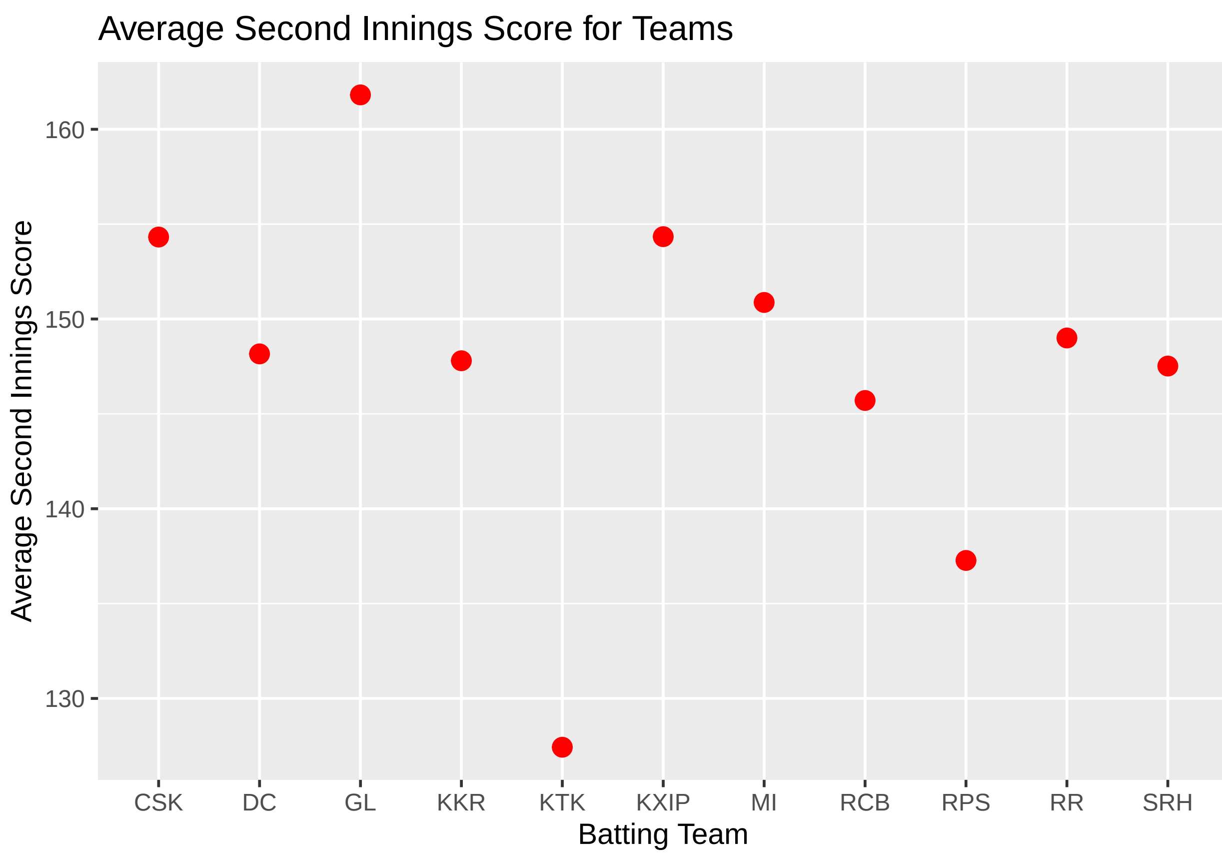
```r
team_inning2_avg_score
```

```
## # A tibble: 11 x 2
##    batting_team second_inning_avg_score
##    <chr>                          <dbl>
##  1 GL                              162.
##  2 KXIP                            154.
##  3 CSK                             154.
##  4 MI                              151.
##  5 RR                              149
##  6 DC                              148.
##  7 KKR                             148.
##  8 SRH                             148.
##  9 RCB                             146.
## 10 RPS                             137.
## 11 KTK                             127.
```

```r
ggplot(team_inning1_avg_score,aes(x=batting_team,y=first_inning_avg_score))+
  geom_point(color="yellow",size=3)+labs(x="Batting Team",y="Average First Innings Score",title="Averag
```




```
ggplot(team_inning2_avg_score,aes(x=batting_team,y=second_inning_avg_score))+
  geom_point(color="red",size=3)+labs(x="Batting Team",y="Average Second Innings Score",title="Average S
```




```
team_inning1_wickets = dataset %>%
  filter(inning == 1, is_super_over == 0) %>%
  group_by(bowling_team,match_id) %>%
  summarise(first_inning_wickets = sum(dismissal))
```

Inference: RCB and GL have the highest average runs scored in the first and second innings of the T20 match.

```
## `summarise()` regrouping output by 'bowling_team' (override with `.groups` argument)
```

```
team_inning1_avg_wickets = team_inning1_wickets %>%
  group_by(bowling_team) %>%
  summarise(first_inning_avg_wickets = mean(first_inning_wickets)) %>%
  arrange(desc(first_inning_avg_wickets))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
team_inning2_wickets = dataset %>%
  filter(inning == 2, is_super_over == 0) %>%
  group_by(bowling_team,match_id) %>%
  summarise(second_inning_wickets = sum(dismissal))
```

```
## `summarise()` regrouping output by 'bowling_team' (override with `.groups` argument)
```

```r
team_inning2_avg_wickets = team_inning2_wickets %>%
  group_by(bowling_team) %>%
  summarise(second_inning_avg_wickets = mean(second_inning_wickets)) %>%
  arrange(desc(second_inning_avg_wickets))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

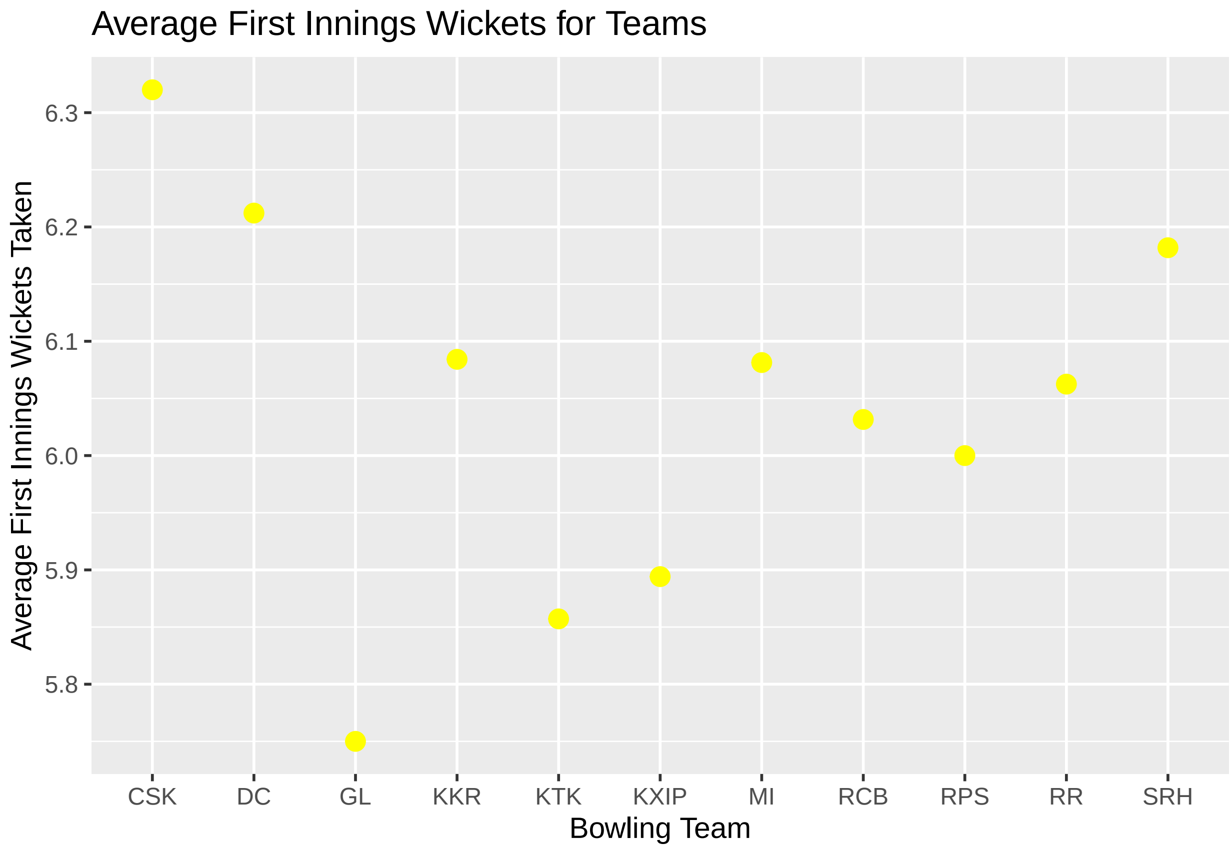
```r
team_inning1_avg_wickets
```

```
## # A tibble: 11 x 2
##    bowling_team first_inning_avg_wickets
##    <chr>                           <dbl>
##  1 CSK                              6.32
##  2 DC                               6.21
##  3 SRH                              6.18
##  4 KKR                              6.08
##  5 MI                               6.08
##  6 RR                               6.06
##  7 RCB                              6.03
##  8 RPS                              6
##  9 KXIP                             5.89
## 10 KTK                              5.86
## 11 GL                               5.75
```

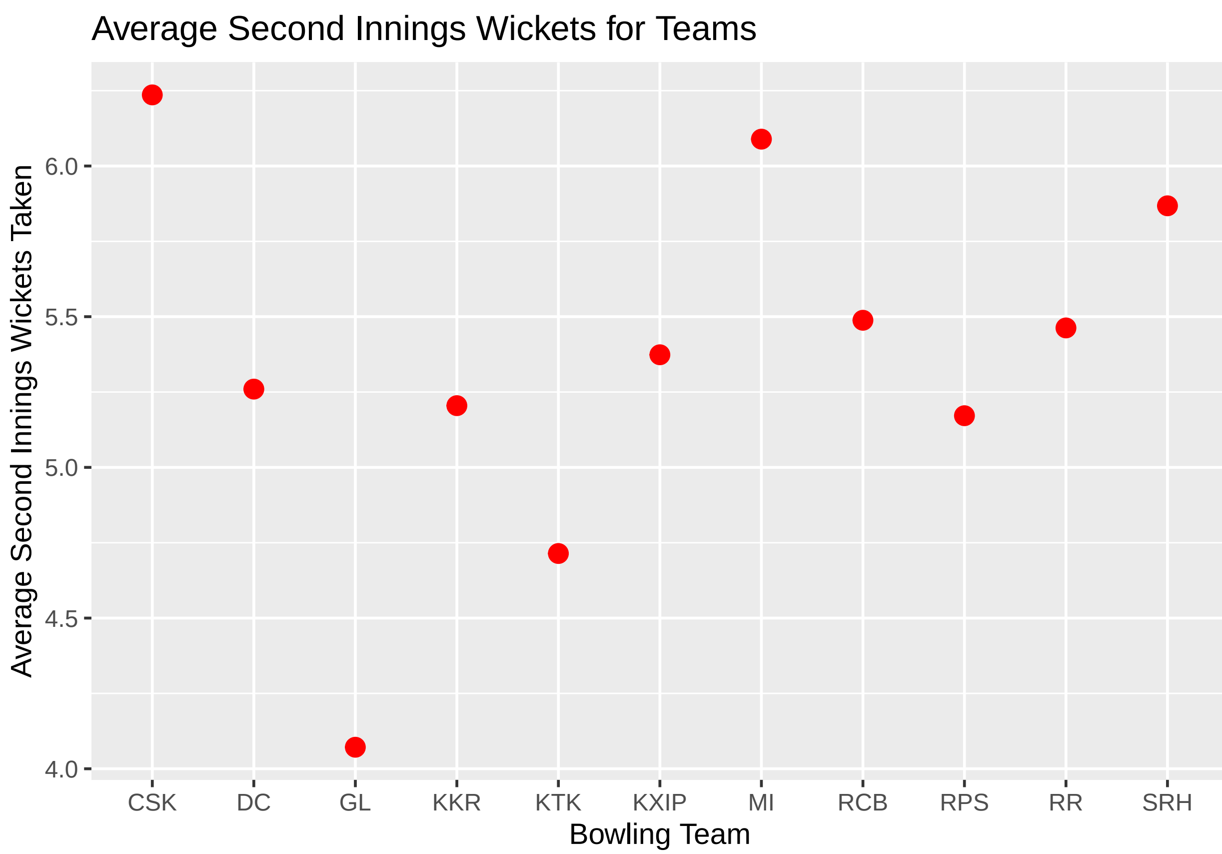
```r
team_inning2_avg_wickets
```

```
## # A tibble: 11 x 2
##    bowling_team second_inning_avg_wickets
##    <chr>                            <dbl>
##  1 CSK                               6.24
##  2 MI                                6.09
##  3 SRH                               5.87
##  4 RCB                               5.49
##  5 RR                                5.46
##  6 KXIP                              5.37
##  7 DC                                5.26
##  8 KKR                               5.20
##  9 RPS                               5.17
## 10 KTK                               4.71
## 11 GL                                4.07
```

```r
ggplot(team_inning1_avg_wickets,aes(x=bowling_team,y=first_inning_avg_wickets))+
  geom_point(color="yellow",size=3)+labs(x="Bowling Team",y="Average First Innings Wickets Taken",title=
```




```
ggplot(team_inning2_avg_wickets,aes(x=bowling_team,y=second_inning_avg_wickets))+
  geom_point(color="red",size=3)+labs(x="Bowling Team",y="Average Second Innings Wickets Taken",title="/
```



Inference: CSK have picked up the most wickets in a T20 match across 12 seasons of the IPL.

Season Wise Analysis of the 2 most successful franchises (CSK AND MI)

```
csk_mi_season_score = dataset %>%
  filter(is_super_over == 0, (batting_team == "CSK" | batting_team == "MI")) %>%
  group_by(match_id,season,batting_team) %>%
  summarise(score = sum(total_runs))
```

Overall Average Score of CSK AND MI

```
## `summarise()` regrouping output by 'match_id', 'season' (override with `.groups` argument)
```

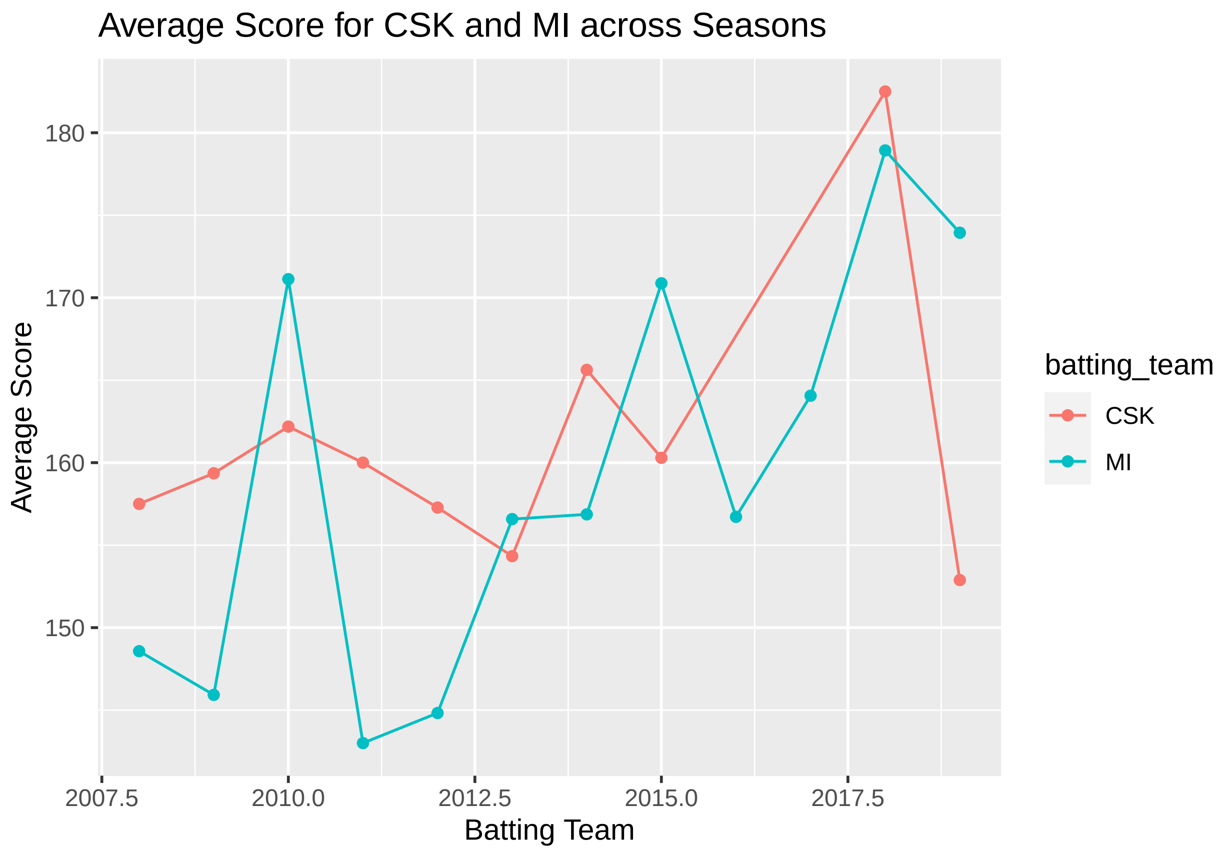
```
csk_mi_season_avg_score = csk_mi_season_score %>%
  group_by(season,batting_team) %>%
  summarise(avg_score = mean(score))
```

```
## `summarise()` regrouping output by 'season' (override with `.groups` argument)
```

```r
csk_mi_season_avg_score
```

```
## # A tibble: 22 x 3
## # Groups:   season [12]
##    season batting_team avg_score
##     <int> <chr>            <dbl>
##  1   2008 CSK               158.
##  2   2008 MI                149.
##  3   2009 CSK               159.
##  4   2009 MI                146.
##  5   2010 CSK               162.
##  6   2010 MI                171.
##  7   2011 CSK               160
##  8   2011 MI                143
##  9   2012 CSK               157.
## 10   2012 MI                145.
## # ... with 12 more rows
```

```r
ggplot(csk_mi_season_avg_score,aes(x = season, y=avg_score, color = batting_team))+geom_point()+
  geom_line()+labs(x="Batting Team",y="Average Score",title="Average Score for CSK and MI across Seasons
```



Inference: CSK have been scoring more runs consistently in a T20 match across the 12 years of IPL compared to their arch-rivals MI.

```r
csk_mi_season_wickets = dataset %>%
  filter(is_super_over == 0, (bowling_team == "CSK" | bowling_team == "MI")) %>%
  group_by(match_id,season,bowling_team) %>%
  summarise(wickets = sum(dismissal))
```

Overall Average Wickets of CSK AND MI

```
## `summarise()` regrouping output by 'match_id', 'season' (override with `.groups` argument)
```

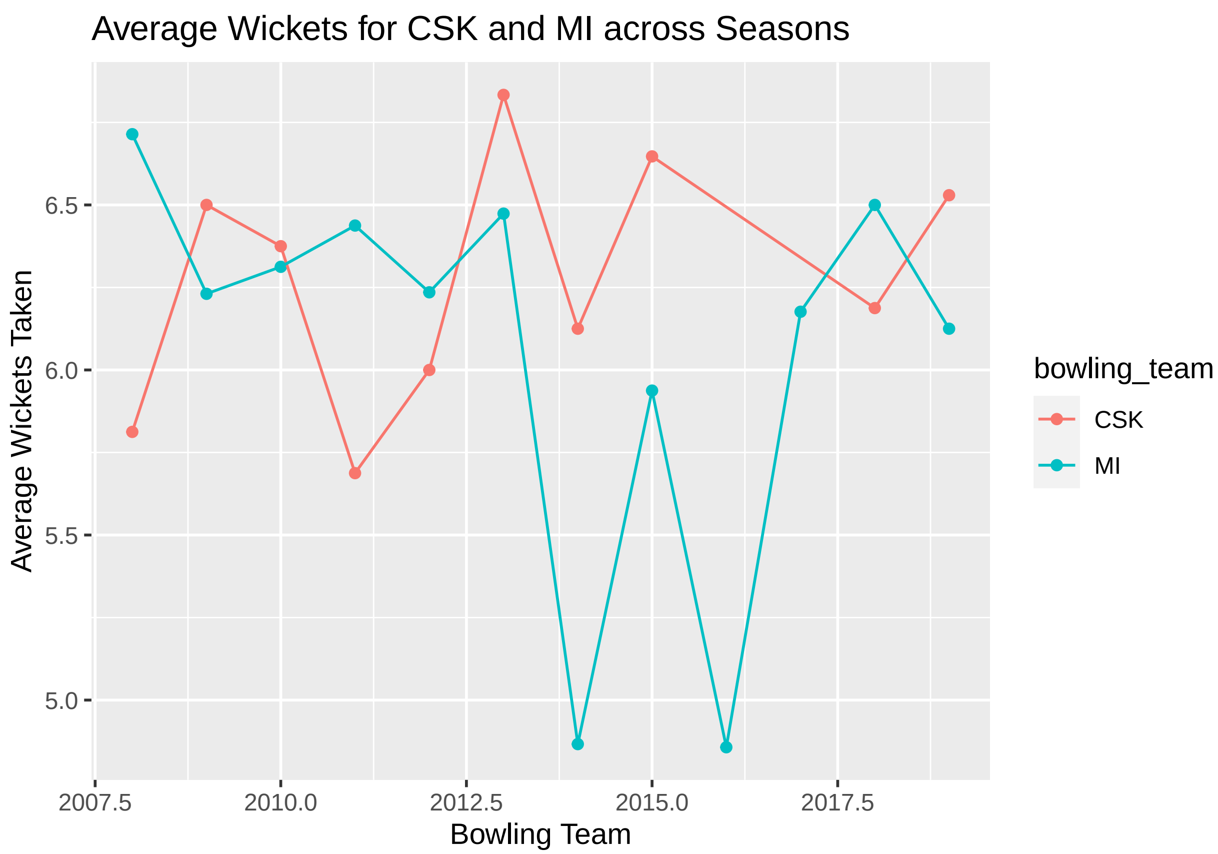
```r
csk_mi_season_avg_wickets = csk_mi_season_wickets %>%
  group_by(season,bowling_team) %>%
  summarise(avg_wickets= mean(wickets))
```

```
## `summarise()` regrouping output by 'season' (override with `.groups` argument)
```

```r
csk_mi_season_avg_wickets
```

```
## # A tibble: 22 x 3
## # Groups:   season [12]
##    season bowling_team avg_wickets
##     <int> <chr>              <dbl>
##  1   2008 CSK                 5.81
##  2   2008 MI                  6.71
##  3   2009 CSK                 6.5
##  4   2009 MI                  6.23
##  5   2010 CSK                 6.38
##  6   2010 MI                  6.31
##  7   2011 CSK                 5.69
##  8   2011 MI                  6.44
##  9   2012 CSK                 6
## 10   2012 MI                  6.24
## # ... with 12 more rows
```

```r
ggplot(csk_mi_season_avg_wickets,aes(x = season, y=avg_wickets, color = bowling_team))+geom_point()+
  geom_line()+labs(x="Bowling Team",y="Average Wickets Taken",title="Average Wickets for CSK and MI acr
```



Inference: CSK are ahead of MI in terms of average wickets picked up in a match across the 12 seasons of the IPL.

Venue Wise Performance Analysis

Analysis is done only for the matches played in Indian Venues

```
indian_venues = dataset %>%
  filter(city == "Mumbai" | city == "Chennai" | city == "Delhi" | city == "Kolkata" | city == "Hyderaba

indian_venues = indian_venues %>%
  mutate(venue = replace(venue,venue == "Feroz Shah Kotla Ground", "Feroz Shah Kotla"))

indian_venues = indian_venues %>%
  mutate(venue = replace(venue,venue == "Dr DY Patil Sports Academy", "Wankhede Stadium"))

indian_venues = indian_venues %>%
  mutate(venue = replace(venue,venue == "Brabourne Stadium", "Wankhede Stadium"))

indian_venues = indian_venues %>%
  mutate(venue = replace(venue,venue == "Subrata Roy Sahara Stadium", "Maharashtra Cricket Association S
```

```
indian_venues = indian_venues %>%
  mutate(venue = replace(venue,venue == "M. A. Chidambaram Stadium", "MA Chidambaram Stadium, Chepauk"))

indian_venues = indian_venues %>%
  mutate(venue = replace(venue,venue == "IS Bindra Stadium", "Punjab Cricket Association IS Bindra Stadi

indian_venues = indian_venues %>%
  mutate(venue = replace(venue,venue == "Rajiv Gandhi Intl. Cricket Stadium", "Rajiv Gandhi Internation


unique(indian_venues$venue)
```

```
##  [1] "Rajiv Gandhi International Stadium, Uppal"
##  [2] "Maharashtra Cricket Association Stadium"
##  [3] "Saurashtra Cricket Association Stadium"
##  [4] "M Chinnaswamy Stadium"
##  [5] "Wankhede Stadium"
##  [6] "Eden Gardens"
##  [7] "Feroz Shah Kotla"
##  [8] "Sawai Mansingh Stadium"
##  [9] "MA Chidambaram Stadium, Chepauk"
## [10] "Nehru Stadium"
## [11] "Punjab Cricket Association IS Bindra Stadium, Mohali"
```

Toss Decisions taken at Venues

```
indian_venues_played = indian_venues %>%
  group_by(venue) %>%
  summarise(matches_played = n_distinct(match_id)) %>%
  arrange(desc(matches_played))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
indian_venues_toss_field = indian_venues %>%
  filter(toss_winner == winner, is_super_over == 0, toss_decision == "field") %>%
  group_by(venue,toss_decision) %>%
  summarise(matches_won = n_distinct(match_id)) %>%
  arrange(desc(matches_won))
```

```
## `summarise()` regrouping output by 'venue' (override with `.groups` argument)
```

```
indian_venues_toss_bat = indian_venues %>%
  filter(toss_winner == winner, is_super_over == 0, toss_decision == "bat") %>%
  group_by(venue,toss_decision) %>%
  summarise(matches_won = n_distinct(match_id)) %>%
  arrange(desc(matches_won))
```

```
## `summarise()` regrouping output by 'venue' (override with `.groups` argument)
```

```
indian_venues_played
```

```
## # A tibble: 11 x 2
##    venue                                              matches_played
##    <chr>                                                       <int>
##  1 Wankhede Stadium                                              101
##  2 Eden Gardens                                                   77
##  3 Feroz Shah Kotla                                               74
##  4 M Chinnaswamy Stadium                                          66
##  5 Rajiv Gandhi International Stadium, Uppal                      64
##  6 MA Chidambaram Stadium, Chepauk                                57
##  7 Sawai Mansingh Stadium                                         47
##  8 Maharashtra Cricket Association Stadium                        38
##  9 Punjab Cricket Association IS Bindra Stadium, Mohali           10
## 10 Saurashtra Cricket Association Stadium                         10
## 11 Nehru Stadium                                                   5
```

```
indian_venues_toss_field
```

```
## # A tibble: 11 x 3
## # Groups:   venue [11]
##    venue                                             toss_decision matches_won
##    <chr>                                             <chr>               <int>
##  1 Wankhede Stadium                                  field                  35
##  2 M Chinnaswamy Stadium                             field                  32
##  3 Eden Gardens                                      field                  31
##  4 Feroz Shah Kotla                                  field                  23
##  5 Sawai Mansingh Stadium                            field                  19
##  6 Rajiv Gandhi International Stadium, Uppal         field                  15
##  7 Maharashtra Cricket Association Stadium           field                  13
##  8 MA Chidambaram Stadium, Chepauk                   field                   8
##  9 Punjab Cricket Association IS Bindra Stadium, Moha~ field                   6
## 10 Saurashtra Cricket Association Stadium            field                   4
## 11 Nehru Stadium                                     field                   1
```

```
indian_venues_toss_bat
```

```
## # A tibble: 10 x 3
## # Groups:   venue [10]
##    venue                                             toss_decision matches_won
##    <chr>                                             <chr>               <int>
##  1 MA Chidambaram Stadium, Chepauk                   bat                    22
##  2 Wankhede Stadium                                  bat                    18
##  3 Feroz Shah Kotla                                  bat                    15
##  4 Eden Gardens                                      bat                    12
##  5 Maharashtra Cricket Association Stadium           bat                    10
##  6 Rajiv Gandhi International Stadium, Uppal         bat                     6
##  7 Sawai Mansingh Stadium                            bat                     6
##  8 M Chinnaswamy Stadium                             bat                     4
##  9 Nehru Stadium                                     bat                     1
## 10 Punjab Cricket Association IS Bindra Stadium, Moha~ bat                     1
```

Inference: Wankhede and Chinnaswamy Stadiums in Mumbai and Bangalore are better chasing grounds while MA Chidambaram Stadium are in Chennai is a better batting defending grounds.

Phase Wise Analaysis

```
venue_pp_score = indian_venues %>%
  filter(is_super_over == 0, over<=6, winner == batting_team) %>%
  group_by(match_id,batting_team,venue,inning) %>%
  summarise(pp_score = sum(total_runs)) %>%
  arrange(desc(pp_score))
```

Average Powerplay Score across Venues

```
## `summarise()` regrouping output by 'match_id', 'batting_team', 'venue' (override with `.groups` argu
```

```
venue_avg_pp_score = venue_pp_score %>%
  group_by(venue) %>%
  summarise(avg_pp_score = mean(pp_score)) %>%
  arrange(desc(avg_pp_score))
```

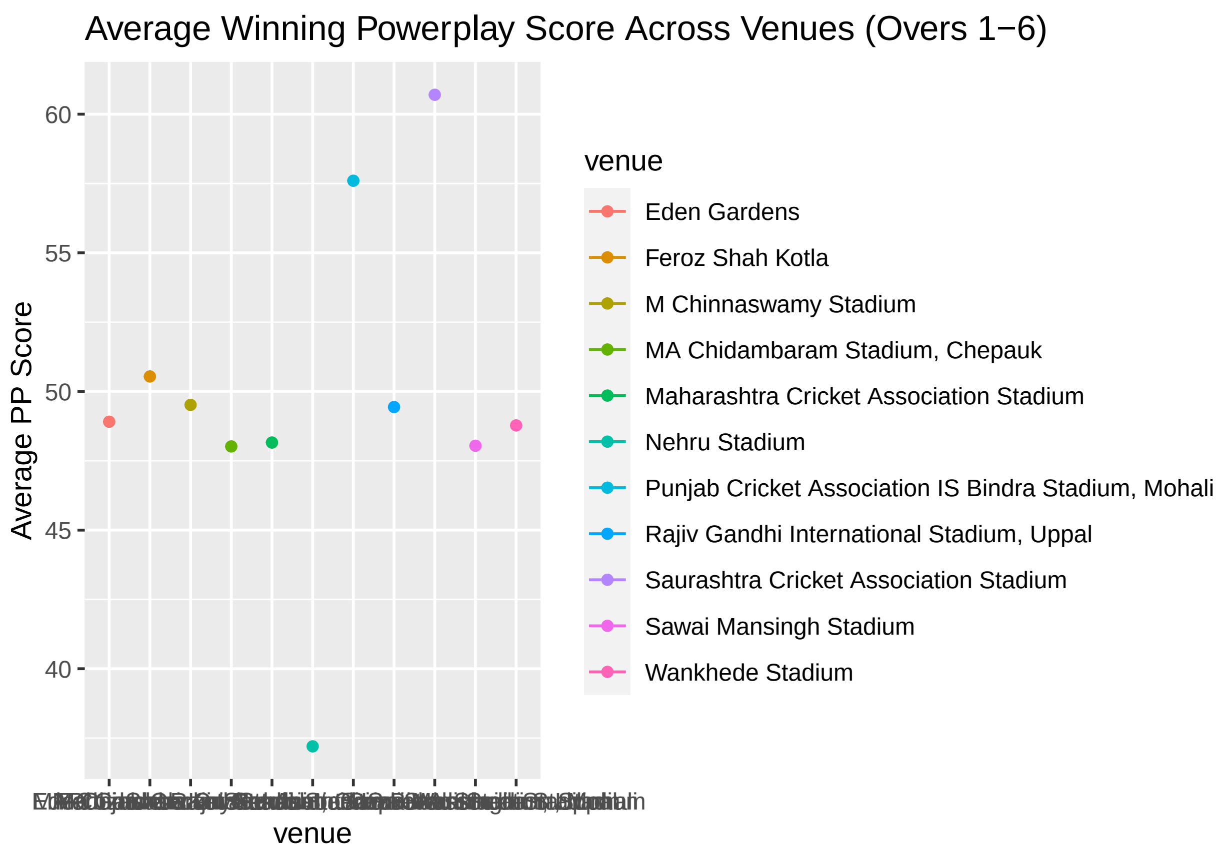
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
venue_avg_pp_score
```

```
## # A tibble: 11 x 2
##    venue                                             avg_pp_score
##    <chr>                                                    <dbl>
##  1 Saurashtra Cricket Association Stadium                    60.7
##  2 Punjab Cricket Association IS Bindra Stadium, Mohali      57.6
##  3 Feroz Shah Kotla                                          50.5
##  4 M Chinnaswamy Stadium                                     49.5
##  5 Rajiv Gandhi International Stadium, Uppal                 49.4
##  6 Eden Gardens                                              48.9
##  7 Wankhede Stadium                                          48.8
##  8 Maharashtra Cricket Association Stadium                   48.2
##  9 Sawai Mansingh Stadium                                    48.0
## 10 MA Chidambaram Stadium, Chepauk                           48.0
## 11 Nehru Stadium                                             37.2
```

```
ggplot(venue_avg_pp_score,aes(x = venue, y=avg_pp_score, color = venue))+geom_point()+
  geom_line()+labs(y="Average PP Score",title="Average Winning Powerplay Score Across Venues (Overs 1-6
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



Inference: Rajkot which is the home ground of GL is the highest scoring ground in the first 6 overs while Kochi which is the home ground of KTK is the least scoring ground in powerplay.

```
venue_mo_score = indian_venues %>%
  filter(is_super_over == 0, over>6 & over<=15, winner == batting_team) %>%
  group_by(match_id,batting_team,venue,inning) %>%
  summarise(mo_score = sum(total_runs)) %>%
  arrange(desc(mo_score))
```

Average Middle Overs Score across Venues

```
## `summarise()` regrouping output by 'match_id', 'batting_team', 'venue' (override with `.groups` argu
```

```
venue_avg_mo_score = venue_mo_score %>%
  group_by(venue) %>%
  summarise(avg_mo_score = mean(mo_score)) %>%
  arrange(desc(avg_mo_score))
```

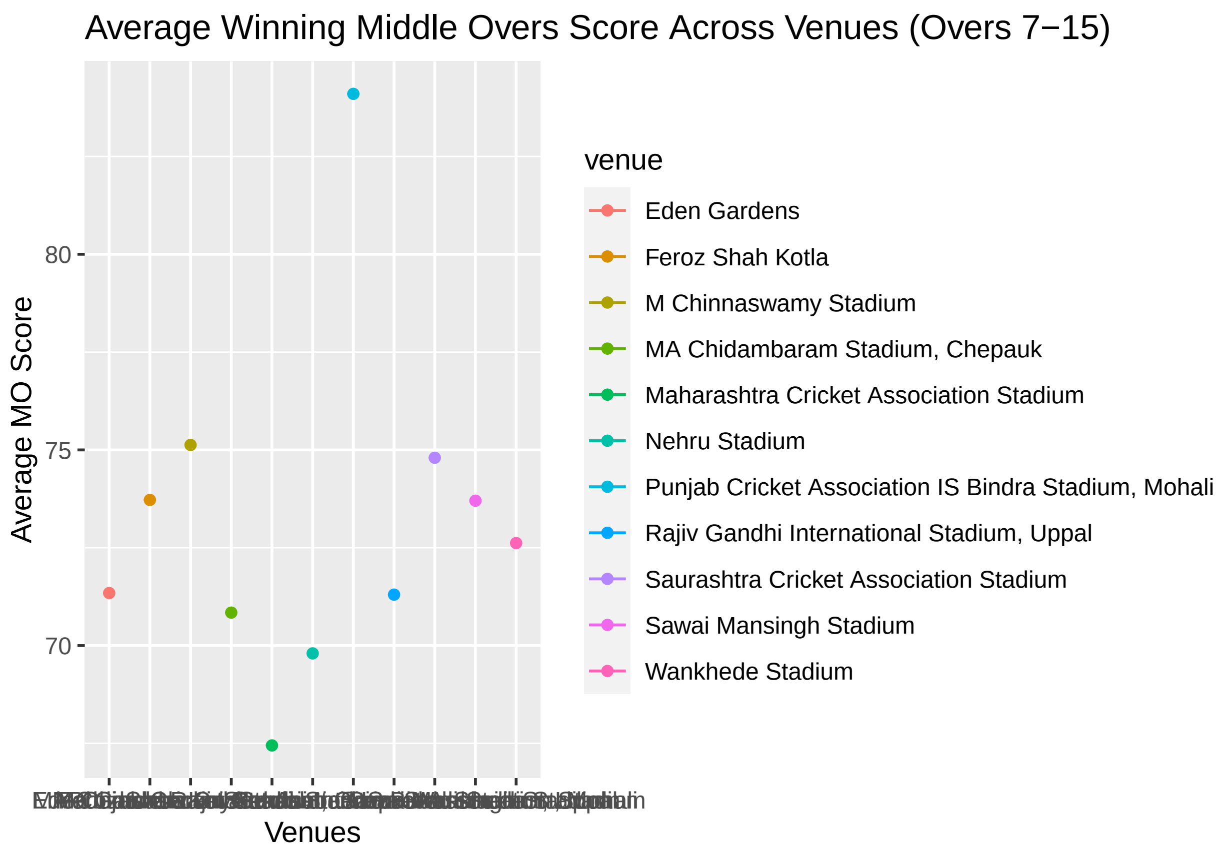
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
venue_avg_mo_score
```

```
## # A tibble: 11 x 2
##    venue                                            avg_mo_score
##    <chr>                                                   <dbl>
##  1 Punjab Cricket Association IS Bindra Stadium, Mohali     84.1
##  2 M Chinnaswamy Stadium                                    75.1
##  3 Saurashtra Cricket Association Stadium                   74.8
##  4 Feroz Shah Kotla                                         73.7
##  5 Sawai Mansingh Stadium                                   73.7
##  6 Wankhede Stadium                                         72.6
##  7 Eden Gardens                                             71.3
##  8 Rajiv Gandhi International Stadium, Uppal                71.3
##  9 MA Chidambaram Stadium, Chepauk                          70.8
## 10 Nehru Stadium                                            69.8
## 11 Maharashtra Cricket Association Stadium                  67.4
```

```r
ggplot(venue_avg_mo_score,aes(x = venue, y=avg_mo_score, color = venue))+geom_point()+
  geom_line()+labs(x="Venues",y="Average MO Score",title="Average Winning Middle Overs Score Across Venu
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



Inference: Mohali, the home ground of KXIP is the highest scoring ground in the middle overs while Kochi, the home ground of KTK is the least scoring ground in the middle overs.

```
venue_do_score = indian_venues %>%
  filter(is_super_over == 0, over>15 & over<=20, winner == batting_team) %>%
  group_by(match_id,batting_team,venue,inning) %>%
  summarise(do_score = sum(total_runs)) %>%
  arrange(desc(do_score))
```

Average Death Overs Score Across Venues

```
## `summarise()` regrouping output by 'match_id', 'batting_team', 'venue' (override with `.groups` argu
```

```
venue_avg_do_score = venue_do_score %>%
  group_by(venue) %>%
  summarise(avg_do_score = mean(do_score)) %>%
  arrange(desc(avg_do_score))
```

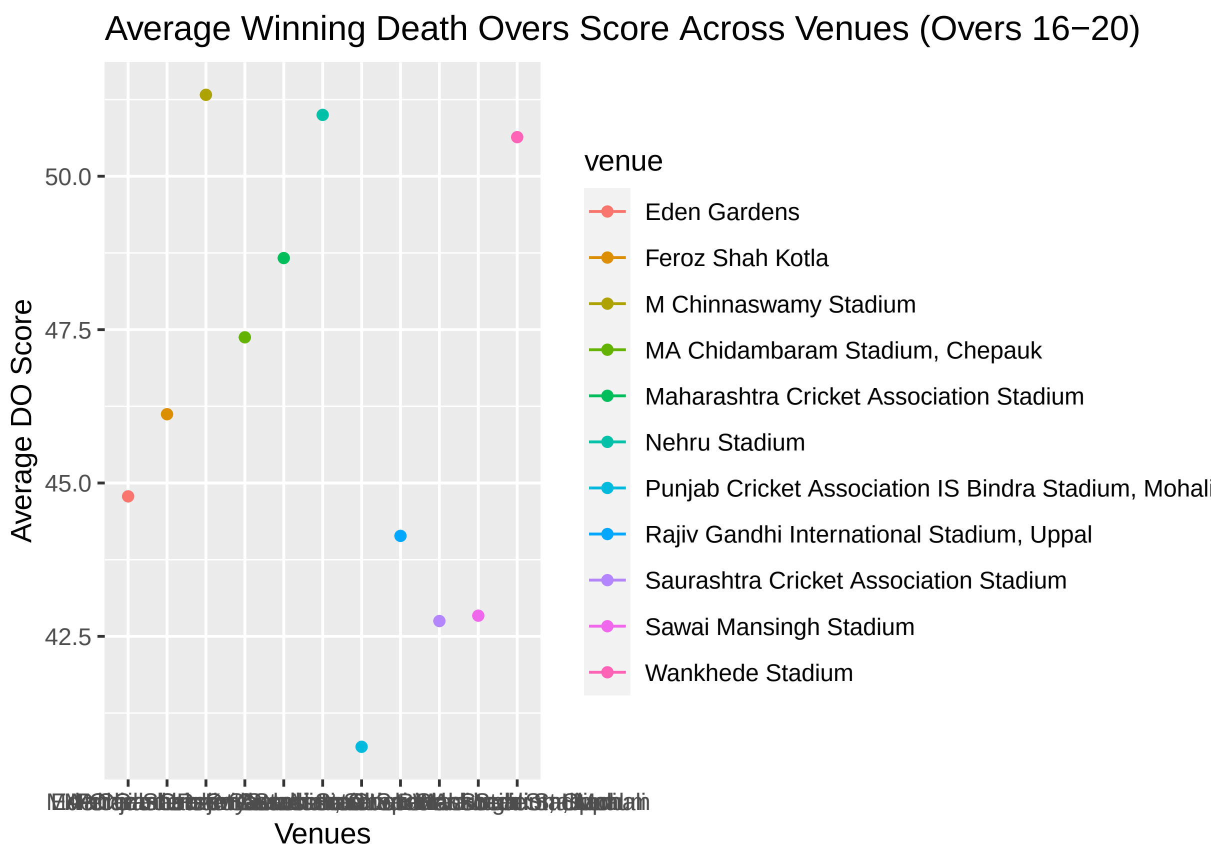
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
venue_avg_do_score
```

```
## # A tibble: 11 x 2
##    venue                                               avg_do_score
##    <chr>                                                      <dbl>
##  1 M Chinnaswamy Stadium                                       51.3
##  2 Nehru Stadium                                               51
##  3 Wankhede Stadium                                            50.6
##  4 Maharashtra Cricket Association Stadium                     48.7
##  5 MA Chidambaram Stadium, Chepauk                             47.4
##  6 Feroz Shah Kotla                                            46.1
##  7 Eden Gardens                                                44.8
##  8 Rajiv Gandhi International Stadium, Uppal                   44.1
##  9 Sawai Mansingh Stadium                                      42.8
## 10 Saurashtra Cricket Association Stadium                      42.8
## 11 Punjab Cricket Association IS Bindra Stadium, Mohali        40.7
```

```
ggplot(venue_avg_do_score,aes(x = venue, y=avg_do_score, color = venue))+geom_point()+
  geom_line()+labs(x="Venues",y="Average DO Score",title="Average Winning Death Overs Score Across Venu
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



Inference: Bangalore, which is home ground of RCB is the best scoring ground in the last stages of an innings while Nehru Stadium in Kochi is the lowest.

Innings Wise Analysis

```
venue_winning_inning1_score = indian_venues %>%
  filter(inning == 1, is_super_over == 0, winner == batting_team) %>%
  group_by(match_id,batting_team,venue) %>%
  summarise(winning_inning1_score = sum(total_runs))
```

Average Winning Score (1st Innings) across Venues

```
## `summarise()` regrouping output by 'match_id', 'batting_team' (override with `.groups` argument)
```

```
venue_winning_avg_inning1_score = venue_winning_inning1_score %>%
  group_by(venue) %>%
  summarise(avg_winning_inning1_score = mean(winning_inning1_score))
```

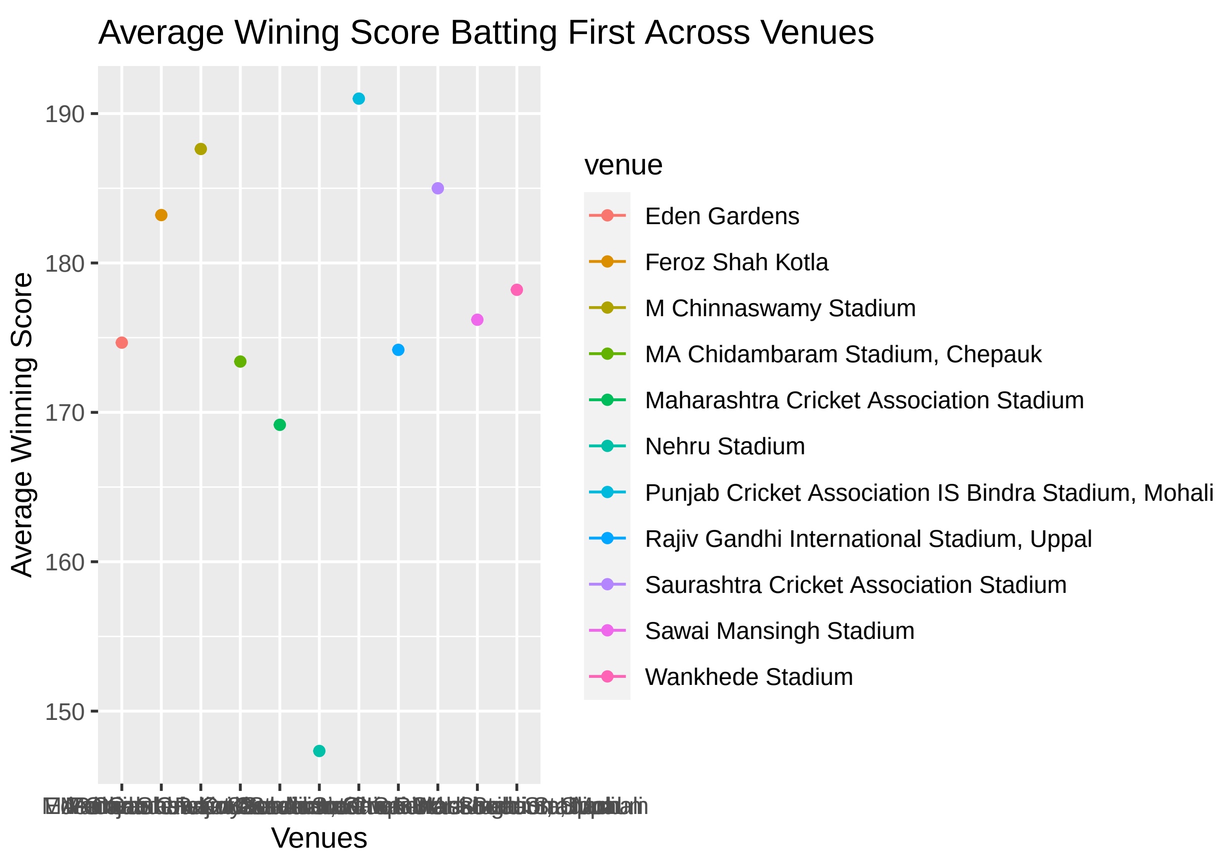
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
venue_winning_avg_inning1_score
```

```
## # A tibble: 11 x 2
##    venue                                                avg_winning_inning1_sco~
##    <chr>                                                                   <dbl>
##  1 Eden Gardens                                                             175.
##  2 Feroz Shah Kotla                                                         183.
##  3 M Chinnaswamy Stadium                                                    188.
##  4 MA Chidambaram Stadium, Chepauk                                          173.
##  5 Maharashtra Cricket Association Stadium                                  169.
##  6 Nehru Stadium                                                            147.
##  7 Punjab Cricket Association IS Bindra Stadium, Mohali                     191
##  8 Rajiv Gandhi International Stadium, Uppal                                174.
##  9 Saurashtra Cricket Association Stadium                                   185
## 10 Sawai Mansingh Stadium                                                   176.
## 11 Wankhede Stadium                                                         178.
```

```r
ggplot(venue_winning_avg_inning1_score,aes(x = venue, y=avg_winning_inning1_score, color = venue))+geom_
  geom_line()+labs(x="Venues",y="Average Winning Score",title="Average Wining Score Batting First Across
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



Inference: Mohali has the highest average winning score in the 1st Innings while Kochi has the lowest in the 1st Innings.

```
venue_winning_inning2_score = indian_venues %>%
  filter(inning == 2, is_super_over == 0, winner == batting_team) %>%
  group_by(match_id,batting_team,venue) %>%
  summarise(winning_inning2_score = sum(total_runs))
```

Average Chasing Score (2nd Innings) across Venues

```
## `summarise()` regrouping output by 'match_id', 'batting_team' (override with `.groups` argument)
```

```
venue_winning_avg_inning2_score = venue_winning_inning2_score %>%
  group_by(venue) %>%
  summarise(avg_winning_inning2_score = mean(winning_inning2_score))
```

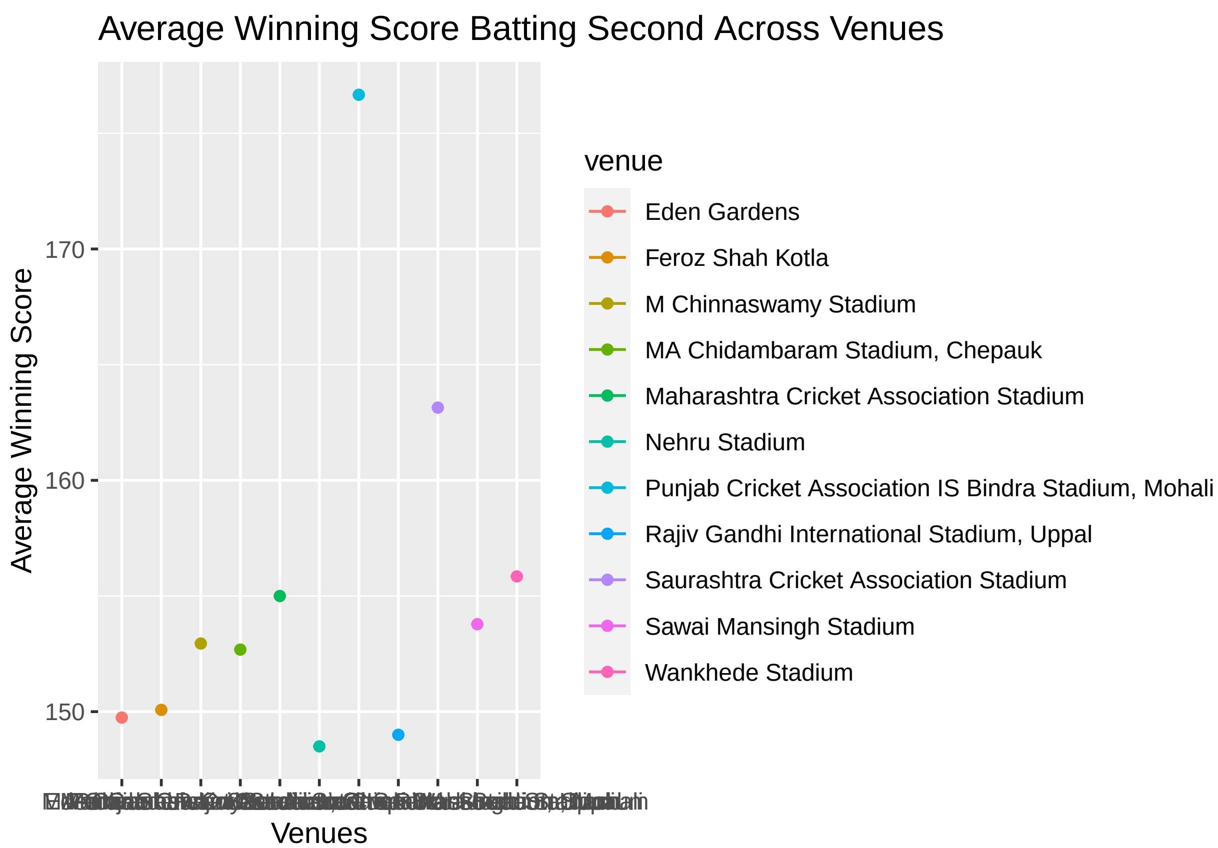
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
venue_winning_avg_inning2_score
```

```
## # A tibble: 11 x 2
##    venue                                              avg_winning_inning2_sco~
##    <chr>                                                                 <dbl>
##  1 Eden Gardens                                                          150.
##  2 Feroz Shah Kotla                                                      150.
##  3 M Chinnaswamy Stadium                                                 153.
##  4 MA Chidambaram Stadium, Chepauk                                       153.
##  5 Maharashtra Cricket Association Stadium                               155
##  6 Nehru Stadium                                                         148.
##  7 Punjab Cricket Association IS Bindra Stadium, Mohali                  177.
##  8 Rajiv Gandhi International Stadium, Uppal                             149
##  9 Saurashtra Cricket Association Stadium                                163.
## 10 Sawai Mansingh Stadium                                                154.
## 11 Wankhede Stadium                                                      156.
```

```
ggplot(venue_winning_avg_inning2_score,aes(x = venue, y=avg_winning_inning2_score, color = venue))+geom_
  geom_line()+labs(x="Venues",y="Average Winning Score",title="Average Winning Score Batting Second Acro
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



Inference: Mohali is a high scoring chasing ground while Kolkata, Delhi, Kochi are low scoring chasing grounds.

Season Wise Analysis of the 2 most successful franchises (CSK AND MI)

```
csk_mi_venue_season_score = indian_venues %>%
  filter(is_super_over == 0, venue == "MA Chidambaram Stadium, Chepauk" | venue == "Wankhede Stadium",wi
  group_by(match_id,season,batting_team,venue,inning) %>%
  summarise(first_inning_score = sum(total_runs))
```

Comparing Chepauk Stadium and Wankhede Stadium (Home Grounds of CSK and MI)

```
## `summarise()` regrouping output by 'match_id', 'season', 'batting_team', 'venue' (override with `.gro
```

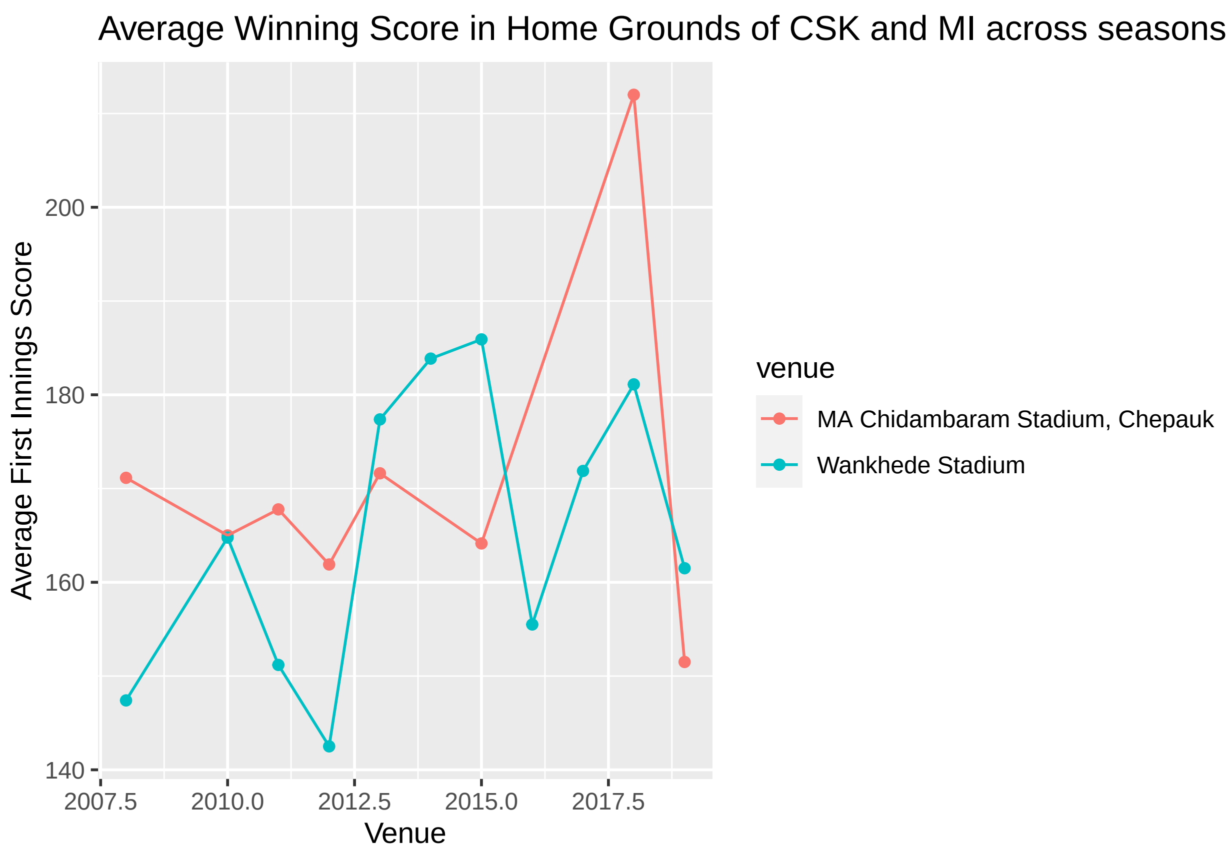
```
csk_mi_venue_avg_season_score = csk_mi_venue_season_score %>%
  group_by(season,venue) %>%
  summarise(avg_inning1_score = mean(first_inning_score))
```

```
## `summarise()` regrouping output by 'season' (override with `.groups` argument)
```

```r
csk_mi_venue_avg_season_score
```

```
## # A tibble: 19 x 3
## # Groups:   season [11]
##    season venue                              avg_inning1_score
##     <int> <chr>                                          <dbl>
##  1   2008 MA Chidambaram Stadium, Chepauk                 171.
##  2   2008 Wankhede Stadium                                147.
##  3   2010 MA Chidambaram Stadium, Chepauk                 165
##  4   2010 Wankhede Stadium                                165.
##  5   2011 MA Chidambaram Stadium, Chepauk                 168.
##  6   2011 Wankhede Stadium                                151.
##  7   2012 MA Chidambaram Stadium, Chepauk                 162.
##  8   2012 Wankhede Stadium                                142.
##  9   2013 MA Chidambaram Stadium, Chepauk                 172.
## 10   2013 Wankhede Stadium                                177.
## 11   2014 Wankhede Stadium                                184.
## 12   2015 MA Chidambaram Stadium, Chepauk                 164.
## 13   2015 Wankhede Stadium                                186.
## 14   2016 Wankhede Stadium                                156.
## 15   2017 Wankhede Stadium                                172.
## 16   2018 MA Chidambaram Stadium, Chepauk                 212
## 17   2018 Wankhede Stadium                                181.
## 18   2019 MA Chidambaram Stadium, Chepauk                 152.
## 19   2019 Wankhede Stadium                                162.
```

```r
ggplot(csk_mi_venue_avg_season_score,aes(x = season, y=avg_inning1_score, color = venue))+geom_point()+
  geom_line()+labs(x="Venue",y="Average First Innings Score",title="Average Winning Score in Home Groun
```



Inference: Home Ground of Chennai and Mumbai have traditionally been high scoring ground across the 12 seasons of the IPL. While the average score in Mumbai have consistently increased over the years from 140 to 180, Chennai has fairly been consistent in scoring around the 160 mark.

Conclusion and Future Work

Based on the above conclusions drawn from venue wise and team wise performance analysis, teams can identify the phases in the game where they are lagging behind and plug those holes by picking appropriate players in the auctions according to their shortcomings and home ground conditions. This will improve their performance and chances of winning the championship.

Appropriate Player based analayis can be carried out on the same dataset in future to identify the best players according to the roles in different phases and venue conditions of the game. Hence, an effective model could be built for the teams to pick the right players in the upcoming auctions and assign roles and startegies to players.

Player Performance Analysis

Create Primary Datasets for Players

```
mat_ds <- matches %>%
  select(
    match_id = id,
    season,
    city,
    team1,
    team2,
    toss_winner,
    toss_dec = toss_decision,
    winner,
    pom = player_of_match,
    venue
  )

del_ds <- deliveries %>%
  select(
    inning,
    match_id,
    over,
    ball,
    batsman,
    bowler,
    runs = batsman_runs,
    bat_team = batting_team,
    bowl_team = bowling_team,
    total_runs,
    dismissal_kind
  ) %>%
  gather(role, player, batsman:bowler) %>%
  mutate(role=as.factor(role))
```

1st Objective - Building a model to rank players by their playing calibre:

A player value depends upon

- **his ability to score quick runs (highest strike rates) and bowl economically (lowest economy rates)**

- **his contribution made to the runs scored by the team and the wickets dismissed by the team in matches that have been both won and lost by his team**

- **his ability to score quick runs against top bowlers (we will consider top 20 bowlers by their economy rate) and to bowl economically against top batsmen (we will consider top 20 batsmen by their strike rates).**

- **all players are rated as a batsman and as a bowler irrespective of their actual or primary domain.**

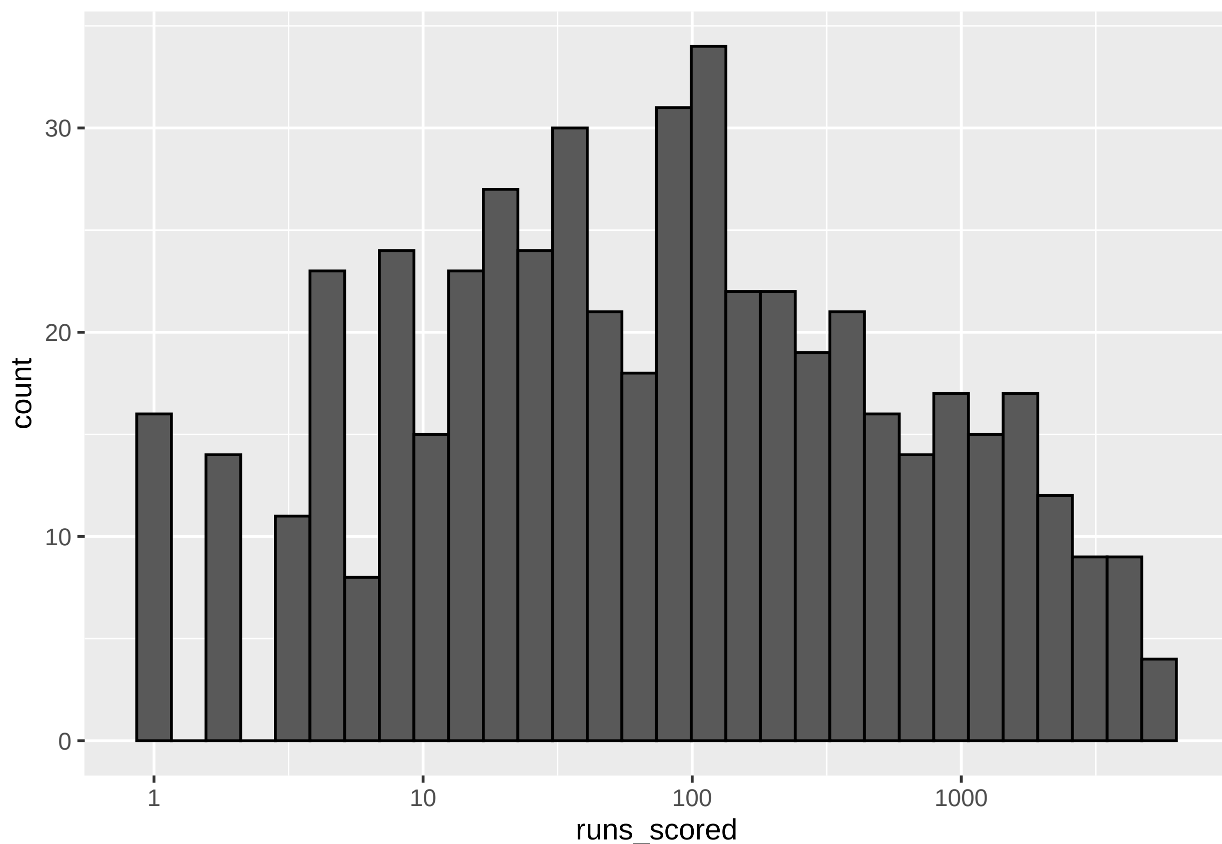
Hence, the nomenclature, "batsman" or "bowler" in the model building refers to all players.

TOP_RATE_PLAYERS:

```
# Distribution of runs scored by batsmen
del_ds %>%
  filter(role == "batsman") %>%
  group_by(player) %>%
  summarize(runs_scored = sum(runs)) %>%
  mutate(runs_scored = runs_scored + 1) %>%
  ggplot(aes(runs_scored)) +
  geom_histogram(aes(), bins=30, colour="black") +
  scale_x_log10()
```

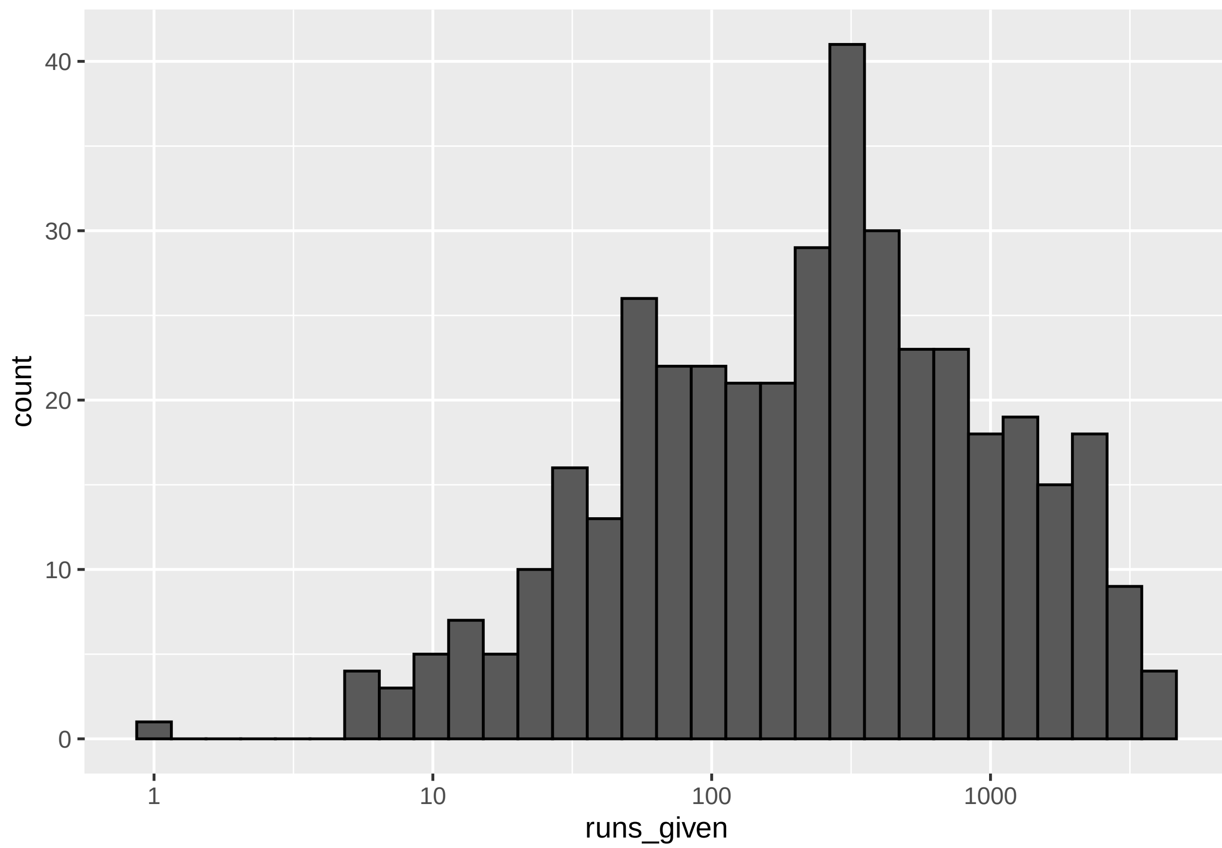
Order of players with best batting striking rates & bowling economy rates

```
## `summarise()` ungrouping output (override with `.groups` argument)
```



```r
# Distribution of runs given by bowlers
del_ds %>%
  filter(role == "bowler") %>%
  group_by(player) %>%
  summarize(runs_given = sum(runs)) %>%
  mutate(runs_given = runs_given + 1) %>%
  ggplot(aes(runs_given)) +
  geom_histogram(aes(), bins=30, colour="black") +
  scale_x_log10()
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```



Inference: The histogram distribution follows normal curve shape, hence we can assume the runs data distribution is normal.

```r
# Batsmen average & median number of balls and runs
batsmen_avgs <- del_ds %>%
  filter(role == "batsman") %>%
  group_by(player) %>%
  summarize(tot_balls = n(), tot_runs = sum(runs)) %>%
  summarize (
    avg_balls = mean(tot_balls),
    median_balls = median(tot_balls),
```

```r
    avg_runs = mean(tot_runs),
    median_runs = median(tot_runs),
    max(tot_runs),
    min(tot_runs),
    max(tot_balls),
    min(tot_balls)
  )
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
t(as.matrix(batsmen_avgs))
```

```
##                      [,1]
## avg_balls        347.0504
## median_balls      70.5000
## avg_runs         432.7248
## median_runs       74.0000
## max(tot_runs)   5434.0000
## min(tot_runs)      0.0000
## max(tot_balls) 4211.0000
## min(tot_balls)     1.0000
```

```r
# Bowler average & median number of balls and runs
bowler_avgs <- del_ds %>%
  filter(role == "bowler") %>%
  group_by(player) %>%
  summarize(tot_balls = n(), tot_runs = sum(runs)) %>%
  summarize (
    avg_balls = mean(tot_balls),
    median_balls = median(tot_balls),
    avg_runs = mean(tot_runs),
    median_runs = median(tot_runs),
    max(tot_runs),
    min(tot_runs),
    max(tot_balls),
    min(tot_balls)
  )
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
t(as.matrix(bowler_avgs))
```

```
##                      [,1]
## avg_balls        442.1679
## median_balls     196.0000
## avg_runs         551.3235
## median_runs      254.0000
## max(tot_runs)   4022.0000
## min(tot_runs)      0.0000
## max(tot_balls) 3451.0000
## min(tot_balls)     1.0000
```

Inference: Now, we will calculate the strike rates for batsmen and economy rates for bowlers using regularization technique. We have earlier seen that the players with highest batting strike rates and best economy rates are not well known for their skills in respective domains (batting or bowling). However, they ended best because of the fact that they played very few balls, resulting in best rates. In order to neutralize this effect, we use penalties to calculate revised batting strike rates or bowling economy rates. From the batsmen and bowler statistics generated above we see the median values are much smaller than the average values. Hence, we use median values as the penalty terms to regularize as this will not effect much the rates of regular, known players in respective domains but will reduce the effects for the players who had batted/ bowled a very few balls. Then we take a look again at the players with highest batting strike rates and lowest economy rates.

```
# Top players with strike rates & economy rates after regularisation using median runs
# and median balls
str_rates <- del_ds %>%
  filter(role == "batsman") %>%
  group_by(player) %>%
  summarize(reg_str_rate = (sum(runs) + batsmen_avgs$median_runs) / (n() +
      batsmen_avgs$median_balls)) %>%
  arrange(desc(reg_str_rate))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
str_rates %>%
  head(20)
```

```
## # A tibble: 20 x 2
##    player         reg_str_rate
##    <chr>                 <dbl>
##  1 AD Russell             1.74
##  2 SP Narine              1.59
##  3 RR Pant                1.59
##  4 GJ Maxwell             1.52
##  5 M Ali                  1.52
##  6 J Bairstow             1.49
##  7 HH Pandya              1.48
##  8 AB de Villiers         1.48
##  9 V Sehwag               1.47
## 10 JC Buttler             1.47
## 11 CH Morris              1.45
## 12 BCJ Cutting            1.45
## 13 CH Gayle               1.45
## 14 K Gowtham              1.42
## 15 KA Pollard             1.40
## 16 KH Pandya              1.40
## 17 N Pooran               1.39
## 18 DA Warner              1.39
## 19 YK Pathan              1.38
## 20 CR Brathwaite          1.38
```

```r
eco_rates <- del_ds %>%
  filter(role == "bowler") %>%
  group_by(player) %>%
  summarize(reg_eco_rate = (sum(runs) + bowler_avgs$median_runs) /
(n() + bowler_avgs$median_balls)) %>%
  arrange(reg_eco_rate)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
eco_rates %>%
  head(20)
```

```
## # A tibble: 20 x 2
##    player           reg_eco_rate
##    <chr>                   <dbl>
##  1 DW Steyn                 1.06
##  2 M Muralitharan           1.07
##  3 R Ashwin                 1.08
##  4 Sohail Tanvir            1.08
##  5 A Kumble                 1.09
##  6 SL Malinga               1.10
##  7 SP Narine                1.10
##  8 SW Tait                  1.11
##  9 DP Nannes                1.12
## 10 MA Starc                 1.13
## 11 Rashid Khan              1.13
## 12 Harbhajan Singh          1.13
## 13 WD Parnell               1.14
## 14 RE van der Merwe         1.14
## 15 J Botha                  1.14
## 16 B Kumar                  1.14
## 17 DL Vettori               1.15
## 18 FH Edwards               1.15
## 19 DE Bollinger             1.15
## 20 A Chandila               1.15
```

Inference: Now as expected we can see that the top players for batting strike rates and bowling economy rates are alltop, regular players in the respective domains of batting and bowling. Next, using the regularized strike rates and economy rates, we construct top rated players, Naturally, we can expect all players who are in batsmen list may not figure in bowler list, and vice versa. This will reintroduce NAs when we try to combine strike rates and economy rates to arrive at player values. We use a similar technique as regularization to replace these NAs. For those players who have never batted, we will assume them to score minimum runs in maximum balls. Hence, we will use median runs and average balls for replacing NAs. Similarly, for players who have never bowled, we will assume them to give away more runs in less balls. Hence, we will use average runs and median balls for replacing NAs. With the above approach, let us see who are our top rated players.

```r
# Top rate players based on strike rates & economy rates
top_rate_players <- str_rates %>%
  full_join(eco_rates, by = "player") %>%
```

```r
  mutate(reg_str_rate = replace_na(
    reg_str_rate,
    batsmen_avgs$median_runs / batsmen_avgs$avg_balls
  )) %>%
  mutate(reg_eco_rate = replace_na(reg_eco_rate, bowler_avgs$avg_runs /
                                     bowler_avgs$median_balls)) %>%
  mutate(player_value = 100 * (reg_str_rate + 1 / reg_eco_rate))

top_rate_players %>%
  arrange(desc(player_value)) %>%
  select(player, player_value) %>%
  mutate(rank = row_number()) %>%
  head(50) %>%
  knitr::kable()
```

player	player_value	rank
SP Narine	249.8296	1
AD Russell	245.4557	2
M Ali	234.0577	3
CH Gayle	228.1677	4
GJ Maxwell	225.5855	5
KH Pandya	225.0858	6
CH Morris	223.7291	7
YK Pathan	222.7373	8
Rashid Khan	221.7830	9
HH Pandya	218.6351	10
SR Watson	218.4355	11
Harbhajan Singh	217.4723	12
K Gowtham	217.2212	13
SK Raina	216.8137	14
KK Cooper	216.5388	15
Mohammad Nabi	216.3234	16
KA Pollard	216.2749	17
BCJ Cutting	216.2111	18
V Sehwag	215.7857	19
MF Maharoof	214.2093	20
JA Morkel	213.8583	21
Shahid Afridi	213.4270	22
Bipul Sharma	212.1187	23
SN Khan	211.7002	24
KP Pietersen	211.6698	25
CR Brathwaite	211.4528	26
RN ten Doeschate	211.2386	27
Ankit Sharma	211.1351	28
KS Williamson	209.8720	29
ST Jayasuriya	209.8612	30
AC Gilchrist	209.3954	31
Umar Gul	209.3522	32
DL Chahar	207.8779	33
RA Tripathi	207.6624	34
N Rana	207.6340	35
RG Sharma	207.5759	36
LJ Wright	207.2914	37

player	player_value	rank
Yuvraj Singh	206.3828	38
M Morkel	206.2108	39
JP Duminy	205.7658	40
JD Ryder	205.6850	41
A Ashish Reddy	205.6455	42
STR Binny	204.9562	43
SM Pollock	204.8870	44
V Kohli	204.3044	45
S Curran	204.2641	46
Shakib Al Hasan	204.2299	47
BA Stokes	204.0291	48
A Symonds	203.9560	49
C de Grandhomme	203.9522	50

Inference: As we could see the list includes some match winning top all round players who are big hitters with high strike rates and bowl tight overs

---

TOP_CONTRI_PLAYERS:

Order of players with best number of highest contributions in won & lost matches

```
# Which teams have won which matches and lost which matches
# Which matches which teams have won
won_t1 <- mat_ds %>%
filter(winner != "") %>%
filter(as.character(winner) == as.character(team1)) %>%
select(match_id, team = team1)
won_t2 <- mat_ds %>%
filter(winner != "") %>%
filter(as.character(winner) == as.character(team2)) %>%
select(match_id, team = team2)
won_matches <- won_t1 %>%
bind_rows(won_t2)
# Which matches which teams have lost
lost_t1 <- mat_ds %>%
filter(winner != "") %>%
filter(as.character(winner) != as.character(team1)) %>%
select(match_id, team = team1, winner)
lost_t2 <- mat_ds %>%
filter(winner != "") %>%
filter(as.character(winner) != as.character(team2)) %>%
select(match_id, team = team2, winner)
lost_matches <- lost_t1 %>%
bind_rows(lost_t2)
```

```
# Batsmen score contribution in won matches
# Top scorer for winning sides
```

```r
batsman_contr_w <- del_ds %>%
full_join(won_matches, by = "match_id") %>%
filter(role == "batsman" & bat_team == team) %>%
group_by(match_id, player) %>%
summarize(batsman_score = sum(runs)) %>%
top_n(1, batsman_score) %>%
full_join(won_matches, by = "match_id")
```

```
## `summarise()` regrouping output by 'match_id' (override with `.groups` argument)
```

```r
batsman_contr_w
```

```
## # A tibble: 767 x 4
## # Groups:   match_id [752]
##    match_id player       batsman_score team
##       <int> <chr>                <int> <chr>
##  1        1 Yuvraj Singh            62 SRH
##  2        2 SPD Smith               84 RPS
##  3        3 CA Lynn                 93 KKR
##  4        4 GJ Maxwell              44 KXIP
##  5        5 KM Jadhav               69 RCB
##  6        6 DA Warner               76 SRH
##  7        7 N Rana                  50 MI
##  8        8 HM Amla                 58 KXIP
##  9        9 SV Samson              102 DC
## 10       10 N Rana                  45 MI
## # ... with 757 more rows
```

```r
# Bowler wicket taking contribution in won matches
# Top wicket taker for winning sides
bowler_contr_w <- del_ds %>%
full_join(won_matches, by = "match_id") %>%
filter(role=="bowler" & bowl_team == team) %>%
filter (dismissal_kind %in% c("bowled", "caught", "caught and bowled", "hit wicket",
"lbw", "stumped")) %>%
select(match_id, team, bowl_team, player, dismissal_kind) %>%
group_by(match_id, player) %>%
summarize(bowler_wckts = n()) %>%
top_n(1, bowler_wckts) %>%
full_join(won_matches, by = "match_id")
```

```
## `summarise()` regrouping output by 'match_id' (override with `.groups` argument)
```

```r
bowler_contr_w
```

```
## # A tibble: 1,246 x 4
## # Groups:   match_id [752]
##    match_id player         bowler_wckts team
##       <int> <chr>                 <int> <chr>
##  1        1 A Nehra                   2 SRH
##  2        1 B Kumar                   2 SRH
```

```
##  3        1 Rashid Khan                2 SRH
##  4        2 Imran Tahir                3 RPS
##  5        3 Kuldeep Yadav              2 KKR
##  6        4 Sandeep Sharma             2 KXIP
##  7        5 B Stanlake                 2 RCB
##  8        5 Iqbal Abdulla              2 RCB
##  9        5 P Negi                     2 RCB
## 10        6 Rashid Khan                3 SRH
## # ... with 1,236 more rows
```

```r
# Top_batsmen on winning sides in the order of highest individual scores
winning_t_scores <- del_ds %>%
full_join(won_matches, by = "match_id") %>%
filter(role == "batsman" & bat_team == team) %>%
group_by(match_id) %>%
summarize(team_score = sum(total_runs)) %>%
full_join(batsman_contr_w, by = "match_id") %>%
arrange(desc(batsman_score))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
winning_t_scores
```

```
## # A tibble: 767 x 5
##    match_id team_score player         batsman_score team
##       <int>      <int> <chr>                  <int> <chr>
##  1      411        263 CH Gayle                 175 RCB
##  2       60        222 BB McCullum              158 KKR
##  3      562        235 AB de Villiers           133 RCB
##  4      620        248 AB de Villiers           129 RCB
##  5      372        215 CH Gayle                 128 RCB
##  6      206        246 M Vijay                  127 CSK
##  7       36        209 DA Warner                126 SRH
##  8      516        226 V Sehwag                 122 KXIP
##  9     7953        187 SR Watson                121 CSK
## 10      243        193 PC Valthaty              120 KXIP
## # ... with 757 more rows
```

```r
# Top_batsmen on winning sides in terms no.of top_scores
win_scores <- winning_t_scores %>%
group_by(player) %>%
summarize(batsman_count = n()) %>%
arrange(desc(batsman_count))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
win_scores
```

```
## # A tibble: 143 x 2
##    player         batsman_count
##    <chr>                  <int>
```

```
##   1 DA Warner                28
##   2 G Gambhir                27
##   3 RG Sharma                26
##   4 CH Gayle                 25
##   5 SK Raina                 21
##   6 AB de Villiers           20
##   7 AM Rahane                20
##   8 AT Rayudu                18
##   9 SR Watson                18
##  10 V Sehwag                 18
## # ... with 133 more rows
```

```
# Top_bowlers on winning sides in terms no.of maximum wickets
win_wickets <- bowler_contr_w %>%
group_by(player) %>%
summarize(bowler_count = n()) %>%
arrange(desc(bowler_count))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
win_wickets
```

```
## # A tibble: 238 x 2
##    player          bowler_count
##    <chr>                  <int>
##  1 SL Malinga                30
##  2 Harbhajan Singh           27
##  3 A Mishra                  26
##  4 DJ Bravo                  25
##  5 R Vinay Kumar             22
##  6 UT Yadav                  22
##  7 PP Chawla                 20
##  8 B Kumar                   19
##  9 SP Narine                 19
## 10 R Ashwin                  18
## # ... with 228 more rows
```

Inference: We have found the top batsman and bowlers who top scored and took the most wickets for their teams the most times in winning causes across 12 seasons of the IPL.

```
# Batsmen score contribution in lost matches
# Top scorer for losing sides
batsman_contr_l <- del_ds %>%
full_join(lost_matches, by = "match_id") %>%
filter(role == "batsman" & bat_team == team) %>%
group_by(match_id, player) %>%
summarize(batsman_score = sum(runs)) %>%
top_n(1, batsman_score) %>%
full_join(lost_matches, by = "match_id") %>%
rename(losing_team=team)
```

```
## `summarise()` regrouping output by 'match_id' (override with `.groups` argument)
```

```r
batsman_contr_l
```

```
## # A tibble: 779 x 5
## # Groups:   match_id [756]
##    match_id player         batsman_score losing_team winner
##       <int> <chr>                  <int> <chr>       <chr>
##  1        1 CH Gayle                  32 RCB         SRH
##  2        2 JC Buttler                38 MI          RPS
##  3        3 SK Raina                  68 GL          KKR
##  4        4 BA Stokes                 50 RPS         KXIP
##  5        5 RR Pant                   57 DC          RCB
##  6        6 DR Smith                  37 GL          SRH
##  7        7 MK Pandey                 81 KKR         MI
##  8        8 AB de Villiers            89 RCB         KXIP
##  9        9 MA Agarwal                20 RPS         DC
## 10       10 DA Warner                 49 SRH         MI
## # ... with 769 more rows
```

```r
# Bowler wicket taking contribution in lost matches
# Top wicket taker for losing sides
bowler_contr_l <- del_ds %>%
full_join(lost_matches, by = "match_id") %>%
filter(role=="bowler" & bowl_team == team) %>%
filter(dismissal_kind %in% c("bowled", "caught", "caught and bowled", "hit wicket",
"lbw", "stumped")) %>%
select(match_id, team, bowl_team, player) %>%
group_by(match_id, player) %>%
summarize(bowler_wckts = n()) %>%
top_n(1, bowler_wckts) %>%
full_join(lost_matches, by = "match_id") %>%
rename(losing_team=team)
```

```
## `summarise()` regrouping output by 'match_id' (override with `.groups` argument)
```

```r
bowler_contr_l
```

```
## # A tibble: 1,240 x 5
## # Groups:   match_id [756]
##    match_id player         bowler_wckts losing_team winner
##       <int> <chr>                 <int> <chr>       <chr>
##  1        1 A Choudhary               1 RCB         SRH
##  2        1 STR Binny                 1 RCB         SRH
##  3        1 TS Mills                  1 RCB         SRH
##  4        1 YS Chahal                 1 RCB         SRH
##  5        2 HH Pandya                 1 MI          RPS
##  6        2 MJ McClenaghan            1 MI          RPS
##  7        2 TG Southee                1 MI          RPS
##  8        4 Imran Tahir               2 RPS         KXIP
##  9        5 CH Morris                 3 DC          RCB
## 10        6 P Kumar                   1 GL          SRH
## # ... with 1,230 more rows
```

```r
# Top_batsmen on losing sides in the order of highest individual scores
losing_t_scores <- del_ds %>%
full_join(lost_matches, by = "match_id") %>%
filter(role == "batsman" & bat_team == team) %>%
group_by(match_id) %>%
summarize(team_score = sum(total_runs)) %>%
full_join(batsman_contr_l, by = "match_id") %>%
arrange(desc(batsman_score))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
losing_t_scores
```

```
## # A tibble: 779 x 6
##    match_id team_score player     batsman_score losing_team winner
##       <int>      <int> <chr>              <int> <chr>       <chr>
##  1     7935        190 RR Pant              130 DC          SRH
##  2       68        214 A Symonds            117 SRH         RR
##  3      517        199 WP Saha              115 KXIP        KKR
##  4    11331        196 AM Rahane            108 RR          DC
##  5    11144        203 SV Samson            106 RR          SRH
##  6    11319        180 CH Gayle             105 KXIP        RCB
##  7       22        198 HM Amla              104 KXIP        MI
##  8       46        189 HM Amla              104 KXIP        GL
##  9    11315        204 KL Rahul             104 KXIP        MI
## 10      410        185 SR Watson            101 RR          CSK
## # ... with 769 more rows
```

```r
# Top_batsmen on losing sides in terms no.of top_scores
loss_scores <- losing_t_scores %>%
group_by(player) %>%
summarize(batsman_count = n()) %>%
arrange(desc(batsman_count))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
loss_scores
```

```
## # A tibble: 166 x 2
##    player          batsman_count
##    <chr>                   <int>
##  1 RV Uthappa                 22
##  2 V Kohli                    22
##  3 DA Warner                  21
##  4 S Dhawan                   20
##  5 CH Gayle                   18
##  6 RG Sharma                  18
##  7 SK Raina                   17
##  8 Yuvraj Singh               17
##  9 AB de Villiers             15
## 10 JP Duminy                  15
## # ... with 156 more rows
```

```r
# Top_bowlers on losing sides in terms no.of maximum wickets
loss_wickets <- bowler_contr_l %>%
group_by(player) %>%
summarize(bowler_count = n()) %>%
arrange(desc(bowler_count))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
loss_wickets
```

```
## # A tibble: 263 x 2
##    player          bowler_count
##    <chr>                  <int>
##  1 B Kumar                   27
##  2 YS Chahal                 23
##  3 A Mishra                  22
##  4 PP Chawla                 22
##  5 DJ Bravo                  21
##  6 DW Steyn                  19
##  7 Harbhajan Singh           19
##  8 R Ashwin                  19
##  9 AB Dinda                  18
## 10 IK Pathan                 18
## # ... with 253 more rows
```

Inference: We have found the top batsman and bowlers who top scored and took the most wickets for their teams the most times in losing causes across 12 seasons of the IPL.

```r
# Top batsmen contribution in won matches & lost matches - arranged by contribution in WON matches
top_contri_batsmen <- win_scores %>%
rename(contribution_in_WON_matches = batsman_count) %>%
full_join(loss_scores, by = "player") %>%
rename(contribution_in_LOST_matches = batsman_count) %>%
arrange(desc(contribution_in_WON_matches))
top_contri_batsmen
```

```
## # A tibble: 195 x 3
##    player         contribution_in_WON_matches contribution_in_LOST_matches
##    <chr>                                <int>                        <int>
##  1 DA Warner                               28                           21
##  2 G Gambhir                               27                           13
##  3 RG Sharma                               26                           18
##  4 CH Gayle                                25                           18
##  5 SK Raina                                21                           17
##  6 AB de Villiers                          20                           15
##  7 AM Rahane                               20                           10
##  8 AT Rayudu                               18                            9
##  9 SR Watson                               18                            6
## 10 V Sehwag                                18                            8
## # ... with 185 more rows
```

```r
# Top batsmen contribution in won matches & lost matches - arranged by contribution in LOST matches
top_contri_batsmen <- win_scores %>%
rename(contribution_in_WON_matches = batsman_count) %>%
full_join(loss_scores, by = "player") %>%
rename(contribution_in_LOST_matches = batsman_count) %>%
arrange(desc(contribution_in_LOST_matches))
top_contri_batsmen
```

```
## # A tibble: 195 x 3
##    player          contribution_in_WON_matches contribution_in_LOST_matches
##    <chr>                                 <int>                        <int>
##  1 V Kohli                                  17                           22
##  2 RV Uthappa                               14                           22
##  3 DA Warner                                28                           21
##  4 S Dhawan                                 17                           20
##  5 RG Sharma                                26                           18
##  6 CH Gayle                                 25                           18
##  7 SK Raina                                 21                           17
##  8 Yuvraj Singh                              6                           17
##  9 AB de Villiers                           20                           15
## 10 JP Duminy                                 2                           15
## # ... with 185 more rows
```

```r
# Top batsmen overall contribution in won matches & lost matches
top_contri_batsmen <- top_contri_batsmen %>%
mutate(batsman_contribution = contribution_in_LOST_matches +
contribution_in_WON_matches) %>%
select(
player,
batsman_contribution,
contribution_in_LOST_matches,
contribution_in_WON_matches
) %>%
arrange(desc(batsman_contribution))
top_contri_batsmen
```

```
## # A tibble: 195 x 4
##    player      batsman_contributi~ contribution_in_LOST_~ contribution_in_WON_m~
##    <chr>                     <int>                  <int>                  <int>
##  1 DA Warner                    49                     21                     28
##  2 RG Sharma                    44                     18                     26
##  3 CH Gayle                     43                     18                     25
##  4 G Gambhir                    40                     13                     27
##  5 V Kohli                      39                     22                     17
##  6 SK Raina                     38                     17                     21
##  7 S Dhawan                     37                     20                     17
##  8 RV Uthappa                   36                     22                     14
##  9 AB de Vill~                  35                     15                     20
## 10 AM Rahane                    30                     10                     20
## # ... with 185 more rows
```

```r
# Top bowlers contribution in won matches and lost matches - arranged by contribution in WON matches
top_contri_bowlers <- win_wickets %>%
rename(contribution_in_WON_matches = bowler_count) %>%
full_join(loss_wickets, by = "player") %>%
rename(contribution_in_LOST_matches = bowler_count) %>%
arrange(desc(contribution_in_WON_matches))
top_contri_bowlers
```

```
## # A tibble: 302 x 3
##    player          contribution_in_WON_matches contribution_in_LOST_matches
##    <chr>                                 <int>                        <int>
##  1 SL Malinga                               30                           18
##  2 Harbhajan Singh                          27                           19
##  3 A Mishra                                 26                           22
##  4 DJ Bravo                                 25                           21
##  5 R Vinay Kumar                            22                           18
##  6 UT Yadav                                 22                           13
##  7 PP Chawla                                20                           22
##  8 B Kumar                                  19                           27
##  9 SP Narine                                19                           18
## 10 R Ashwin                                 18                           19
## # ... with 292 more rows
```

```r
# Top bowlers contribution in won matches and lost matches - arranged by contribution in LOST matches
top_contri_bowlers <- win_wickets %>%
rename(contribution_in_WON_matches = bowler_count) %>%
full_join(loss_wickets, by = "player") %>%
rename(contribution_in_LOST_matches = bowler_count) %>%
arrange(desc(contribution_in_LOST_matches))
top_contri_bowlers
```

```
## # A tibble: 302 x 3
##    player          contribution_in_WON_matches contribution_in_LOST_matches
##    <chr>                                 <int>                        <int>
##  1 B Kumar                                  19                           27
##  2 YS Chahal                                14                           23
##  3 A Mishra                                 26                           22
##  4 PP Chawla                                20                           22
##  5 DJ Bravo                                 25                           21
##  6 Harbhajan Singh                          27                           19
##  7 R Ashwin                                 18                           19
##  8 DW Steyn                                 16                           19
##  9 SL Malinga                               30                           18
## 10 R Vinay Kumar                            22                           18
## # ... with 292 more rows
```

```r
# Top bowlers overall contribution in won matches and lost matches
top_contri_bowlers <- top_contri_bowlers %>%
mutate(bowler_contribution = contribution_in_LOST_matches +
contribution_in_WON_matches) %>%
select(
player,
```

```
bowler_contribution,
contribution_in_LOST_matches,
contribution_in_WON_matches
) %>%
arrange(desc(bowler_contribution))
top_contri_bowlers
```

```
## # A tibble: 302 x 4
##    player       bowler_contributi~ contribution_in_LOST_~ contribution_in_WON_m~
##    <chr>                     <int>                  <int>                  <int>
##  1 A Mishra                     48                     22                     26
##  2 SL Malinga                   48                     18                     30
##  3 B Kumar                      46                     27                     19
##  4 DJ Bravo                     46                     21                     25
##  5 Harbhajan S~                 46                     19                     27
##  6 PP Chawla                    42                     22                     20
##  7 R Vinay Kum~                 40                     18                     22
##  8 YS Chahal                    37                     23                     14
##  9 R Ashwin                     37                     19                     18
## 10 SP Narine                    37                     18                     19
## # ... with 292 more rows
```

Inference: The most contributing players in terms of runs scored and wickets taken across 12 years of the IPL have been ordered in terms of contribution in winning causes. But, the contribution of players depends also depends on the number of matches they play, higher the matches they play, they have higher chances of contributing. Hence, we use median contribution and average matches in arriving at player contributions.

```
# Top_contribution players in won/ lost matches
top_contri_players <- top_contri_batsmen %>%
  full_join(top_contri_bowlers, by="player")

top_contri_players <- as.data.frame(top_contri_players)

top_contri_players[is.na(top_contri_players)] <- 0

top_contri_players <- top_contri_players %>%
  mutate(player_contribution = batsman_contribution + bowler_contribution) %>%
  select(player,player_contribution, batsman_contribution, bowler_contribution)
top_contri_players %>%
  arrange(desc(player_contribution))
```

```
##                player player_contribution batsman_contribution
## 1            DJ Bravo                  56                   10
## 2           SR Watson                  52                   24
## 3           DA Warner                  49                   49
## 4            CH Gayle                  49                   43
## 5           JH Kallis                  48                   23
## 6            A Mishra                  48                    0
## 7          SL Malinga                  48                    0
## 8           RG Sharma                  47                   44
```

```
## 9          SK Raina                 46                   38
## 10    Harbhajan Singh                 46                    0
## 11          B Kumar                 46                    0
## 12         PP Chawla                 42                    0
## 13        G Gambhir                 40                   40
## 14         RA Jadeja                 40                   11
## 15     R Vinay Kumar                 40                    0
## 16           V Kohli                 39                   39
## 17          S Dhawan                 39                   37
## 18         YK Pathan                 38                   25
## 19         SP Narine                 37                    0
## 20         YS Chahal                 37                    0
## 21          R Ashwin                 37                    0
## 22        RV Uthappa                 36                   36
## 23    AB de Villiers                 35                   35
## 24          DW Steyn                 35                    0
## 25          UT Yadav                 35                    0
## 26      Yuvraj Singh                 34                   23
## 27         IK Pathan                 34                    5
## 28        KA Pollard                 32                   16
## 29           Z Khan                 32                    0
## 30        AM Rahane                 30                   30
## 31          P Kumar                 28                    0
## 32         AT Rayudu                 27                   27
## 33        AD Russell                 27                   10
## 34          AR Patel                 27                    6
## 35         JA Morkel                 27                    0
## 36           PP Ojha                 27                    0
## 37          RP Singh                 27                    0
## 38           A Nehra                 27                    0
## 39          SE Marsh                 26                   26
## 40          V Sehwag                 26                   26
## 41          DR Smith                 26                   22
## 42          AB Dinda                 26                    0
## 43       Imran Tahir                 26                    0
## 44    Sandeep Sharma                 26                    0
## 45       JP Faulkner                 25                    3
## 46       DS Kulkarni                 25                    0
## 47          M Morkel                 25                    0
## 48    MJ McClenaghan                 25                    0
## 49          I Sharma                 24                    0
## 50         MM Sharma                 24                    0
## 51        KD Karthik                 23                   23
## 52       BB McCullum                 23                   23
## 53        JD Unadkat                 22                    0
## 54           M Vijay                 21                   21
## 55          PA Patel                 21                   21
## 56          MS Dhoni                 21                   21
## 57     SR Tendulkar                 21                   21
## 58          BJ Hodge                 21                   14
## 59   Shakib Al Hasan                 21                    2
## 60    M Muralitharan                 21                    0
## 61         JJ Bumrah                 21                    0
## 62         JP Duminy                 20                   17
```

```
## 63           MM Patel                   20                     0
## 64           L Balaji                   20                     0
## 65          MK Pandey                   19                    19
## 66          CH Morris                   19                     0
## 67          KV Sharma                   19                     0
## 68             S Kaul                   19                     0
## 69           VR Aaron                   19                     0
## 70           KL Rahul                   18                    18
## 71           R Dravid                   18                    18
## 72          SPD Smith                   18                    18
## 73            RR Pant                   18                    18
## 74       F du Plessis                   18                    18
## 75           R Bhatia                   18                     0
## 76           R Sharma                   18                     0
## 77          SV Samson                   17                    17
## 78           AJ Finch                   17                    17
## 79          HH Pandya                   17                     3
## 80     Mohammed Shami                   17                     0
## 81        Rashid Khan                   17                     0
## 82         SK Trivedi                   17                     0
## 83         MG Johnson                   17                     0
## 84        S Sreesanth                   16                     0
## 85           A Kumble                   16                     0
## 86          Q de Kock                   15                    15
## 87       AC Gilchrist                   15                    15
## 88          BA Stokes                   15                     6
## 89       MC Henriques                   15                     4
## 90           S Nadeem                   15                     0
## 91           HV Patel                   15                     0
## 92            S Gopal                   15                     0
## 93          SB Jakati                   15                     0
## 94            SS Iyer                   14                    14
## 95      KC Sangakkara                   14                    14
## 96         MEK Hussey                   14                    14
## 97   DPMD Jayawardene                   13                    13
## 98            WP Saha                   13                    13
## 99         GJ Maxwell                   13                     9
## 100        AD Mathews                   13                     4
## 101            AJ Tye                   13                     0
## 102         SN Thakur                   13                     0
## 103           CA Lynn                   12                    12
## 104         MK Tiwary                   12                    12
## 105        JC Buttler                   12                    12
## 106         ML Hayden                   12                    12
## 107       LMP Simmons                   12                    12
## 108         DJ Hussey                   12                     9
## 109            N Rana                   12                     8
## 110           J Botha                   12                     4
## 111       NLTC Perera                   12                     2
## 112            P Negi                   12                     0
## 113     Kuldeep Yadav                   12                     0
## 114         RJ Harris                   12                     0
## 115          TA Boult                   12                     0
## 116        TG Southee                   12                     0
```

```
## 117          K Rabada                  12                     0
## 118   NM Coulter-Nile                  12                     0
## 119           P Awana                  12                     0
## 120          MA Starc                  12                     0
## 121          SK Warne                  12                     0
## 122        SC Ganguly                  11                    11
## 123         DA Miller                  11                    11
## 124           KK Nair                  11                    11
## 125         A Symonds                  11                     9
## 126     ST Jayasuriya                  11                     7
## 127        AB Agarkar                  11                     0
## 128         KK Cooper                  11                     0
## 129         KH Pandya                  11                     0
## 130           MS Gony                  11                     0
## 131        WD Parnell                  11                     0
## 132      DT Christian                  11                     0
## 133      DE Bollinger                  11                     0
## 134         S Aravind                  11                     0
## 135       S Badrinath                  10                    10
## 136        TM Dilshan                  10                     8
## 137     Azhar Mahmood                  10                     0
## 138          M Kartik                  10                     0
## 139          J Archer                  10                     0
## 140        PJ Sangwan                  10                     0
## 141         DL Chahar                  10                     0
## 142           NV Ojha                   9                     9
## 143         RS Bopara                   9                     4
## 144  RE van der Merwe                   9                     2
## 145       MF Maharoof                   9                     0
## 146     Anureet Singh                   9                     0
## 147           GB Hogg                   9                     0
## 148           A Singh                   9                     0
## 149         DP Nannes                   9                     0
## 150    Mohammed Siraj                   9                     0
## 151         KM Jadhav                   8                     8
## 152       LRPL Taylor                   8                     8
## 153     KS Williamson                   8                     8
## 154           M Vohra                   8                     8
## 155        DL Vettori                   8                     0
## 156     Iqbal Abdulla                   8                     0
## 157           SW Tait                   8                     0
## 158          AS Yadav                   7                     7
## 159          HH Gibbs                   7                     7
## 160     Mandeep Singh                   7                     7
## 161          AM Nayar                   7                     4
## 162          MR Marsh                   7                     0
## 163         CR Woakes                   7                     0
## 164          K Ahmed                    7                     0
## 165        WPUJC Vaas                   7                     0
## 166    A Ashish Reddy                   7                     0
## 167             B Lee                   7                     0
## 168     KW Richardson                   7                     0
## 169         VY Mahesh                   7                     0
## 170     BW Hilfenhaus                   7                     0
```

```
## 171         SS Tiwary                   6                     6
## 172      KP Pietersen                   6                     6
## 173          GC Smith                   6                     6
## 174     CR Brathwaite                   6                     0
## 175        AS Rajpoot                   6                     0
## 176         R Rampaul                   6                     0
## 177         R Tewatia                   6                     0
## 178    AD Mascarenhas                   6                     0
## 179        PJ Cummins                   6                     0
## 180 Mustafizur Rahman                   6                     0
## 181       M Ur Rahman                   6                     0
## 182          CL White                   5                     5
## 183        EJG Morgan                   5                     5
## 184         GJ Bailey                   5                     5
## 185        MA Agarwal                   5                     5
## 186           HM Amla                   5                     5
## 187       RA Tripathi                   5                     5
## 188         R McLaren                   5                     0
## 189      Bipul Sharma                   5                     0
## 190           A Zampa                   5                     0
## 191          M Ashwin                   5                     0
## 192        M Markande                   5                     0
## 193          R Dhawan                   5                     0
## 194      Basil Thampi                   5                     0
## 195         RD Chahar                   5                     0
## 196     CK Langeveldt                   5                     0
## 197           SE Bond                   5                     0
## 198         STR Binny                   5                     0
## 199     Harmeet Singh                   5                     0
## 200         LR Shukla                   4                     4
## 201          M Manhas                   4                     4
## 202        MV Boucher                   4                     4
## 203          JD Ryder                   4                     4
## 204          MS Bisla                   4                     4
## 205           S Sohal                   4                     4
## 206          TL Suman                   4                     4
## 207          JR Hopes                   4                     0
## 208       AB McDonald                   4                     0
## 209           D Wiese                   4                     0
## 210        MP Stoinis                   4                     0
## 211         AC Thomas                   4                     0
## 212      BE Hendricks                   4                     0
## 213         H Viljoen                   4                     0
## 214         K Gowtham                   4                     0
## 215   Karanveer Singh                   4                     0
## 216      Pankaj Singh                   4                     0
## 217          Umar Gul                   4                     0
## 218          VS Malik                   4                     0
## 219      Ankit Sharma                   4                     0
## 220      CRD Fernando                   4                     0
## 221   Joginder Sharma                   4                     0
## 222           L Ngidi                   4                     0
## 223          AN Ahmed                   4                     0
## 224           BB Sran                   4                     0
```

```
## 225         GD McGrath                    4                     0
## 226           J Theron                    4                     0
## 227           RV Gomez                    4                     0
## 228         YA Abdulla                    4                     0
## 229       Ishan Kishan                    3                     3
## 230       JEC Franklin                    3                     3
## 231        SA Asnodkar                    3                     3
## 232           SA Yadav                    3                     3
## 233        CJ Anderson                    3                     3
## 234             P Shaw                    3                     3
## 235             S Gill                    3                     3
## 236         SP Fleming                    3                     3
## 237          DJG Sammy                    3                     0
## 238           RR Powar                    3                     0
## 239         BAW Mendis                    3                     0
## 240          CJ Jordan                    3                     0
## 241          IC Pandey                    3                     0
## 242           JDP Oram                    3                     0
## 243        Kamran Khan                    3                     0
## 244              M Ali                    3                     0
## 245          P Krishna                    3                     0
## 246   SMSM Senanayake                     3                     0
## 247           BA Bhatt                    3                     0
## 248           H Gurney                    3                     0
## 249          J Suchith                    3                     0
## 250      Mohammad Asif                    3                     0
## 251       S Lamichhane                    3                     0
## 252           S Randiv                    3                     0
## 253         T Thushara                    3                     0
## 254     V Pratap Singh                    3                     0
## 255          VRV Singh                    3                     0
## 256    AA Jhunjhunwala                    2                     2
## 257           B Chipli                    2                     2
## 258           DJ Hooda                    2                     2
## 259         MJ Guptill                    2                     2
## 260        Niraj Patel                    2                     2
## 261     PD Collingwood                    2                     2
## 262            R Parag                    2                     2
## 263        Salman Butt                    2                     2
## 264         SP Goswami                    2                     2
## 265        PC Valthaty                    2                     0
## 266           A Mithun                    2                     0
## 267         B Laughlin                    2                     0
## 268         D du Preez                    2                     0
## 269            I Sodhi                    2                     0
## 270   J Syed Mohammad                     2                     0
## 271        KC Cariappa                    2                     0
## 272            KM Asif                    2                     0
## 273         KP Appanna                    2                     0
## 274          R Sathish                    2                     0
## 275          SB Bangar                    2                     0
## 276            SB Wagh                    2                     0
## 277      Shahid Afridi                    2                     0
## 278      Shoaib Akhtar                    2                     0
```

```
## 279          T Curran                   2                     0
## 280   Y Venugopal Rao                   0                     0
## 281           E Lewis                   0                     0
## 282        AL Menaria                   0                     0
## 283           OA Shah                   0                     0
## 284         BJ Rohrer                   0                     0
## 285      DB Ravi Teja                   0                     0
## 286     LA Pomersbach                   0                     0
## 287        MN van Wyk                   0                     0
## 288           TM Head                   0                     0
## 289         UBT Chand                   0                     0
## 290       YV Takawale                   0                     0
## 291     Anirudh Singh                   0                     0
## 292           AP Dole                   0                     0
## 293       AP Majumdar                   0                     0
## 294           AS Raut                   0                     0
## 295   C de Grandhomme                   0                     0
## 296         CM Gautam                   0                     0
## 297           D Short                   0                     0
## 298            DB Das                   0                     0
## 299         DJ Harris                   0                     0
## 300       JDS Neesham                   0                     0
## 301            K Goel                   0                     0
## 302         M Klinger                   0                     0
## 303          N Pooran                   0                     0
## 304  RN ten Doeschate                   0                     0
## 305          S Curran                   0                     0
## 306          S Vidyut                   0                     0
## 307        SD Chitnis                   0                     0
## 308           SN Khan                   0                     0
## 309       T Henderson                   0                     0
## 310           Y Nagar                   0                     0
## 311        J Bairstow                   0                     0
## 312      BB Samantray                   0                     0
## 313         GH Vihari                   0                     0
## 314       SW Billings                   0                     0
## 315           A Hales                   0                     0
## 316       AC Blizzard                   0                     0
## 317          AC Voges                   0                     0
## 318           AP Tare                   0                     0
## 319          FY Fazal                   0                     0
## 320            JJ Roy                   0                     0
## 321     LA Carseldine                   0                     0
## 322         MD Mishra                   0                     0
## 323      Misbah-ul-Haq                  0                     0
## 324           MJ Lumb                   0                     0
## 325        MN Samuels                   0                     0
## 326           N Saini                   0                     0
## 327          PA Reddy                   0                     0
## 328           RE Levi                   0                     0
## 329         RJ Quiney                   0                     0
## 330        S Anirudha                   0                     0
## 331         S Hetmyer                   0                     0
## 332         SM Katich                   0                     0
```

```
## 333           W Jaffer                   0                    0
## 334                  0                   0                    0
## 335     SJ Srivastava                    0                    0
## 336           S Badree                   0                    0
## 337          S Kaushik                   0                    0
## 338           S Sharma                   0                    0
## 339       Shoaib Ahmed                   0                    0
## 340        A Choudhary                   0                    0
## 341      DP Vijaykumar                   0                    0
## 342    Gurkeerat Singh                   0                    0
## 343          JE Taylor                   0                    0
## 344             P Sahu                   0                    0
## 345           R Shukla                   0                    0
## 346            S Ladda                   0                    0
## 347          SS Mundhe                   0                    0
## 348           TS Mills                   0                    0
## 349           A Uniyal                   0                    0
## 350          AA Chavan                   0                    0
## 351          AA Noffke                   0                    0
## 352           AF Milne                   0                    0
## 353    D Kalyankrishna                   0                    0
## 354           D Willey                   0                    0
## 355      DAJ Bracewell                   0                    0
## 356      DJ Muthuswami                   0                    0
## 357          DNT Zoysa                   0                    0
## 358     Jaskaran Singh                   0                    0
## 359    JJ van der Wath                   0                    0
## 360          JO Holder                   0                    0
## 361        JW Hastings                   0                    0
## 362          KJ Abbott                   0                    0
## 363         L Plunkett                   0                    0
## 364          LJ Wright                   0                    0
## 365          M Santner                   0                    0
## 366           MA Khote                   0                    0
## 367          MJ Clarke                   0                    0
## 368           ND Doshi                   0                    0
## 369              P Raj                   0                    0
## 370   PM Sarvesh Kumar                   0                    0
## 371            R Ninan                   0                    0
## 372            RG More                   0                    0
## 373        RJ Peterson                   0                    0
## 374            RR Raje                   0                    0
## 375             S Mavi                   0                    0
## 376           S Narwal                   0                    0
## 377           SB Joshi                   0                    0
## 378           T Shamsi                   0                    0
## 379     V Chakravarthy                   0                    0
## 380          V Shankar                   0                    0
## 381           PV Tambe                   0                    0
## 382      Sohail Tanvir                   0                    0
## 383        BCJ Cutting                   0                    0
## 384     P Parameswaran                   0                    0
## 385 Washington Sundar                    0                    0
## 386         AG Murtaza                   0                    0
```

```
## 387           B Akhil                  0                     0
## 388        B Stanlake                  0                     0
## 389            K Paul                  0                     0
## 390   KMDN Kulasekara                  0                     0
## 391       M de Lange                   0                     0
## 392          M Ntini                   0                     0
## 393     Mohammad Nabi                  0                     0
## 394       A Chandila                   0                     0
## 395         A Joseph                   0                     0
## 396       Anand Rajan                  0                     0
## 397       Avesh Khan                   0                     0
## 398         CJ McKay                   0                     0
## 399   Gagandeep Singh                  0                     0
## 400          JM Kemp                   0                     0
## 401       K Santokie                   0                     0
## 402         O Thomas                   0                     0
## 403       P Amarnath                   0                     0
## 404     S Kuggeleijn                   0                     0
## 405          S Tyagi                   0                     0
## 406        S Warrier                   0                     0
## 407        SB Styris                   0                     0
## 408        SM Boland                   0                     0
## 409       SM Harwood                   0                     0
## 410       SM Pollock                   0                     0
## 411          WA Mota                   0                     0
##     bowler_contribution
## 1                    46
## 2                    28
## 3                     0
## 4                     6
## 5                    25
## 6                    48
## 7                    48
## 8                     3
## 9                     8
## 10                   46
## 11                   46
## 12                   42
## 13                    0
## 14                   29
## 15                   40
## 16                    0
## 17                    2
## 18                   13
## 19                   37
## 20                   37
## 21                   37
## 22                    0
## 23                    0
## 24                   35
## 25                   35
## 26                   11
## 27                   29
## 28                   16
```

```
## 29                                    32
## 30                                     0
## 31                                    28
## 32                                     0
## 33                                    17
## 34                                    21
## 35                                    27
## 36                                    27
## 37                                    27
## 38                                    27
## 39                                     0
## 40                                     0
## 41                                     4
## 42                                    26
## 43                                    26
## 44                                    26
## 45                                    22
## 46                                    25
## 47                                    25
## 48                                    25
## 49                                    24
## 50                                    24
## 51                                     0
## 52                                     0
## 53                                    22
## 54                                     0
## 55                                     0
## 56                                     0
## 57                                     0
## 58                                     7
## 59                                    19
## 60                                    21
## 61                                    21
## 62                                     3
## 63                                    20
## 64                                    20
## 65                                     0
## 66                                    19
## 67                                    19
## 68                                    19
## 69                                    19
## 70                                     0
## 71                                     0
## 72                                     0
## 73                                     0
## 74                                     0
## 75                                    18
## 76                                    18
## 77                                     0
## 78                                     0
## 79                                    14
## 80                                    17
## 81                                    17
## 82                                    17
```

```
## 83                    17
## 84                    16
## 85                    16
## 86                     0
## 87                     0
## 88                     9
## 89                    11
## 90                    15
## 91                    15
## 92                    15
## 93                    15
## 94                     0
## 95                     0
## 96                     0
## 97                     0
## 98                     0
## 99                     4
## 100                    9
## 101                   13
## 102                   13
## 103                    0
## 104                    0
## 105                    0
## 106                    0
## 107                    0
## 108                    3
## 109                    4
## 110                    8
## 111                   10
## 112                   12
## 113                   12
## 114                   12
## 115                   12
## 116                   12
## 117                   12
## 118                   12
## 119                   12
## 120                   12
## 121                   12
## 122                    0
## 123                    0
## 124                    0
## 125                    2
## 126                    4
## 127                   11
## 128                   11
## 129                   11
## 130                   11
## 131                   11
## 132                   11
## 133                   11
## 134                   11
## 135                    0
## 136                    2
```

```
## 137                         10
## 138                         10
## 139                         10
## 140                         10
## 141                         10
## 142                          0
## 143                          5
## 144                          7
## 145                          9
## 146                          9
## 147                          9
## 148                          9
## 149                          9
## 150                          9
## 151                          0
## 152                          0
## 153                          0
## 154                          0
## 155                          8
## 156                          8
## 157                          8
## 158                          0
## 159                          0
## 160                          0
## 161                          3
## 162                          7
## 163                          7
## 164                          7
## 165                          7
## 166                          7
## 167                          7
## 168                          7
## 169                          7
## 170                          7
## 171                          0
## 172                          0
## 173                          0
## 174                          6
## 175                          6
## 176                          6
## 177                          6
## 178                          6
## 179                          6
## 180                          6
## 181                          6
## 182                          0
## 183                          0
## 184                          0
## 185                          0
## 186                          0
## 187                          0
## 188                          5
## 189                          5
## 190                          5
```

```
## 191                         5
## 192                         5
## 193                         5
## 194                         5
## 195                         5
## 196                         5
## 197                         5
## 198                         5
## 199                         5
## 200                         0
## 201                         0
## 202                         0
## 203                         0
## 204                         0
## 205                         0
## 206                         0
## 207                         4
## 208                         4
## 209                         4
## 210                         4
## 211                         4
## 212                         4
## 213                         4
## 214                         4
## 215                         4
## 216                         4
## 217                         4
## 218                         4
## 219                         4
## 220                         4
## 221                         4
## 222                         4
## 223                         4
## 224                         4
## 225                         4
## 226                         4
## 227                         4
## 228                         4
## 229                         0
## 230                         0
## 231                         0
## 232                         0
## 233                         0
## 234                         0
## 235                         0
## 236                         0
## 237                         3
## 238                         3
## 239                         3
## 240                         3
## 241                         3
## 242                         3
## 243                         3
## 244                         3
```

```
## 245           3
## 246           3
## 247           3
## 248           3
## 249           3
## 250           3
## 251           3
## 252           3
## 253           3
## 254           3
## 255           3
## 256           0
## 257           0
## 258           0
## 259           0
## 260           0
## 261           0
## 262           0
## 263           0
## 264           0
## 265           2
## 266           2
## 267           2
## 268           2
## 269           2
## 270           2
## 271           2
## 272           2
## 273           2
## 274           2
## 275           2
## 276           2
## 277           2
## 278           2
## 279           2
## 280           0
## 281           0
## 282           0
## 283           0
## 284           0
## 285           0
## 286           0
## 287           0
## 288           0
## 289           0
## 290           0
## 291           0
## 292           0
## 293           0
## 294           0
## 295           0
## 296           0
## 297           0
## 298           0
```

```
## 299                                0
## 300                                0
## 301                                0
## 302                                0
## 303                                0
## 304                                0
## 305                                0
## 306                                0
## 307                                0
## 308                                0
## 309                                0
## 310                                0
## 311                                0
## 312                                0
## 313                                0
## 314                                0
## 315                                0
## 316                                0
## 317                                0
## 318                                0
## 319                                0
## 320                                0
## 321                                0
## 322                                0
## 323                                0
## 324                                0
## 325                                0
## 326                                0
## 327                                0
## 328                                0
## 329                                0
## 330                                0
## 331                                0
## 332                                0
## 333                                0
## 334                                0
## 335                                0
## 336                                0
## 337                                0
## 338                                0
## 339                                0
## 340                                0
## 341                                0
## 342                                0
## 343                                0
## 344                                0
## 345                                0
## 346                                0
## 347                                0
## 348                                0
## 349                                0
## 350                                0
## 351                                0
## 352                                0
```

```
## 353                    0
## 354                    0
## 355                    0
## 356                    0
## 357                    0
## 358                    0
## 359                    0
## 360                    0
## 361                    0
## 362                    0
## 363                    0
## 364                    0
## 365                    0
## 366                    0
## 367                    0
## 368                    0
## 369                    0
## 370                    0
## 371                    0
## 372                    0
## 373                    0
## 374                    0
## 375                    0
## 376                    0
## 377                    0
## 378                    0
## 379                    0
## 380                    0
## 381                    0
## 382                    0
## 383                    0
## 384                    0
## 385                    0
## 386                    0
## 387                    0
## 388                    0
## 389                    0
## 390                    0
## 391                    0
## 392                    0
## 393                    0
## 394                    0
## 395                    0
## 396                    0
## 397                    0
## 398                    0
## 399                    0
## 400                    0
## 401                    0
## 402                    0
## 403                    0
## 404                    0
## 405                    0
## 406                    0
```

```
## 407                   0
## 408                   0
## 409                   0
## 410                   0
## 411                   0
```

```r
# Average and median of top contribution by players
stats_contri <- top_contri_players %>%
summarize(avg_contri_pp = mean(player_contribution),
med_contri_pp = median(player_contribution))

# no.of matches played by each player
matches_players <- del_ds %>%
select(match_id, player) %>%
group_by(player, match_id) %>%
slice(1) %>%
ungroup() %>%
count(player) %>%
rename(no_matches_played = n) %>%
arrange(desc(no_matches_played))
matches_players %>% head(150)
```

```
## # A tibble: 150 x 2
##    player          no_matches_played
##    <chr>                       <int>
##  1 SK Raina                      189
##  2 RG Sharma                     182
##  3 MS Dhoni                      170
##  4 RV Uthappa                    170
##  5 V Kohli                       170
##  6 KD Karthik                    162
##  7 RA Jadeja                     162
##  8 YK Pathan                     161
##  9 Harbhajan Singh               160
## 10 S Dhawan                      158
## # ... with 140 more rows
```

```r
# Average and median of matches played by players
stats_matches <- matches_players %>%
summarize(avg_mat_played = mean(no_matches_played),
med_mat_played = median(no_matches_played))

top_contri_players <- top_contri_players %>%
full_join(matches_players, by ="player") %>%
mutate(player_contri_rate=(player_contribution+stats_contri$med_contri_pp)/
(no_matches_played+stats_matches$avg_mat_played))
top_contri_players %>%
select(player, player_contribution, player_contri_rate) %>%
arrange(desc(player_contri_rate)) %>%
head(50) %>%
knitr::kable()
```

player	player_contribution	player_contri_rate
JH Kallis	48	0.4113901
DJ Bravo	56	0.3717456
YS Chahal	37	0.3680407
Imran Tahir	26	0.3597091
CH Gayle	49	0.3477674
SL Malinga	48	0.3457430
SR Watson	52	0.3448261
K Rabada	12	0.3448223
B Kumar	46	0.3438773
MJ McClenaghan	25	0.3435990
DA Warner	49	0.3432627
AD Russell	27	0.3429176
R Vinay Kumar	40	0.3323245
JP Faulkner	25	0.3280516
S Gopal	15	0.3253385
DW Steyn	35	0.3239183
S Kaul	19	0.3133484
VR Aaron	19	0.3133484
AJ Tye	13	0.3124959
Sandeep Sharma	26	0.3112010
SE Marsh	26	0.3080060
BA Stokes	15	0.3044837
R Sharma	18	0.3038644
NM Coulter-Nile	12	0.2996215
A Mishra	48	0.2964640
SP Narine	37	0.2962412
M Morkel	25	0.2947133
MA Starc	12	0.2941138
IK Pathan	34	0.2914094
AB Dinda	26	0.2873544
K Ahmed	7	0.2864530
A Kumble	16	0.2840880
J Archer	10	0.2833967
Z Khan	32	0.2825730
Rashid Khan	17	0.2822553
KK Cooper	11	0.2808951
AR Patel	27	0.2807953
RP Singh	27	0.2807953
LMP Simmons	12	0.2787422
S Sreesanth	16	0.2762404
WD Parnell	11	0.2757317
BJ Hodge	21	0.2735208
Shakib Al Hasan	21	0.2735208
Azhar Mahmood	10	0.2723697
Mohammed Shami	17	0.2713153
DE Bollinger	11	0.2707546
HV Patel	15	0.2698836
MF Maharoof	9	0.2685911
SN Thakur	13	0.2681358
RR Pant	18	0.2669880

Inference: The Top Players with the highest contribution for their teams in winning and losing causes in 12 years of the IPL have been found out.

TOP_EXCEL_PLAYERS:

Order of players with best performance against best bowlers and best batsmen

```r
# Top 20 strike batsmen and top 20 economy bowlers
top_20_batsmen <- str_rates %>%
head(20)
top_20_batsmen
```

```
## # A tibble: 20 x 2
##    player         reg_str_rate
##    <chr>                 <dbl>
##  1 AD Russell             1.74
##  2 SP Narine              1.59
##  3 RR Pant                1.59
##  4 GJ Maxwell             1.52
##  5 M Ali                  1.52
##  6 J Bairstow             1.49
##  7 HH Pandya              1.48
##  8 AB de Villiers         1.48
##  9 V Sehwag               1.47
## 10 JC Buttler             1.47
## 11 CH Morris              1.45
## 12 BCJ Cutting            1.45
## 13 CH Gayle               1.45
## 14 K Gowtham              1.42
## 15 KA Pollard             1.40
## 16 KH Pandya              1.40
## 17 N Pooran               1.39
## 18 DA Warner              1.39
## 19 YK Pathan              1.38
## 20 CR Brathwaite          1.38
```

```r
top_20_bowlers <- eco_rates %>%
head(20)
top_20_bowlers
```

```
## # A tibble: 20 x 2
##    player           reg_eco_rate
##    <chr>                   <dbl>
##  1 DW Steyn                 1.06
##  2 M Muralitharan           1.07
##  3 R Ashwin                 1.08
##  4 Sohail Tanvir            1.08
##  5 A Kumble                 1.09
##  6 SL Malinga               1.10
```

```
##   7 SP Narine               1.10
##   8 SW Tait                 1.11
##   9 DP Nannes               1.12
##  10 MA Starc                1.13
##  11 Rashid Khan             1.13
##  12 Harbhajan Singh         1.13
##  13 WD Parnell              1.14
##  14 RE van der Merwe        1.14
##  15 J Botha                 1.14
##  16 B Kumar                 1.14
##  17 DL Vettori              1.15
##  18 FH Edwards              1.15
##  19 DE Bollinger            1.15
##  20 A Chandila              1.15
```

```r
# batsmen strike rate against top_20 bowlers
sr_vs_t20_bowlers <- deliveries %>%
filter(bowler %in% top_20_bowlers$player) %>%
group_by(player = batsman) %>%
summarize(sr_t20 = (sum(batsman_runs) + batsmen_avgs$median_runs) / (n() +
batsmen_avgs$avg_balls)) %>%
arrange(desc(sr_t20))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
sr_vs_t20_bowlers %>%
head(20) %>%
mutate(rank = row_number())
```

```
## # A tibble: 20 x 3
##    player          sr_t20  rank
##    <chr>            <dbl> <int>
##  1 AB de Villiers   0.870     1
##  2 SR Watson        0.846     2
##  3 DA Warner        0.822     3
##  4 SK Raina         0.806     4
##  5 RV Uthappa       0.790     5
##  6 V Kohli          0.786     6
##  7 YK Pathan        0.783     7
##  8 MS Dhoni         0.765     8
##  9 CH Gayle         0.741     9
## 10 S Dhawan         0.720    10
## 11 RG Sharma        0.691    11
## 12 G Gambhir        0.689    12
## 13 BB McCullum      0.680    13
## 14 AM Rahane        0.659    14
## 15 AC Gilchrist     0.658    15
## 16 KD Karthik       0.658    16
## 17 JP Duminy        0.647    17
## 18 V Sehwag         0.645    18
## 19 AT Rayudu        0.641    19
## 20 DR Smith         0.640    20
```

```r
# bowlers economy rate against top_20 batsmen
er_vs_t20_batsmen <- deliveries %>%
filter(batsman %in% top_20_batsmen$player) %>%
group_by(player = bowler) %>%
summarize(er_t20 = (sum(batsman_runs) + bowler_avgs$avg_runs) / (n() +
bowler_avgs$median_balls)) %>%
arrange(er_t20)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
er_vs_t20_batsmen %>%
head(20) %>%
mutate(rank = row_number())
```

```
## # A tibble: 20 x 3
##    player          er_t20  rank
##    <chr>            <dbl> <int>
##  1 Harbhajan Singh   1.73     1
##  2 R Ashwin          1.74     2
##  3 SP Narine         1.78     3
##  4 DJ Bravo          1.82     4
##  5 JJ Bumrah         1.82     5
##  6 DS Kulkarni       1.85     6
##  7 SL Malinga        1.85     7
##  8 YS Chahal         1.92     8
##  9 P Kumar           1.92     9
## 10 B Kumar           1.92    10
## 11 DW Steyn          1.94    11
## 12 AR Patel          1.94    12
## 13 RA Jadeja         1.94    13
## 14 A Mishra          1.97    14
## 15 UT Yadav          1.97    15
## 16 PP Chawla         1.97    16
## 17 Rashid Khan       1.99    17
## 18 SR Watson         2.01    18
## 19 Sandeep Sharma    2.01    19
## 20 Z Khan            2.01    20
```

Inference: All Top 20 batsman may not have played against the top 20 bowlers, so we normalize by assuming that they would have scored less runs in average number of balls. Similarly, we can assume all bowlers would have given away more runs in less number of balls.

```r
# Top excellence players
top_excel_players <- er_vs_t20_batsmen %>%
full_join(sr_vs_t20_bowlers, by = "player") %>%
mutate(sr_t20 = replace_na(sr_t20, batsmen_avgs$median_runs / batsmen_avgs$avg_balls)) %>%
mutate(er_t20 = replace_na(er_t20, bowler_avgs$avg_runs / bowler_avgs$median_balls))

# Top 50 Excellence Players
top_excel_players %>%
select(player, sr_t20, er_t20) %>%
```

```r
arrange(desc(sr_t20)) %>%
head(50) %>%
knitr::kable()
```

player	sr_t20	er_t20
AB de Villiers	0.8701406	2.812875
SR Watson	0.8464007	2.006728
DA Warner	0.8216833	2.812875
SK Raina	0.8059269	2.330107
RV Uthappa	0.7896125	2.812875
V Kohli	0.7864723	2.810671
YK Pathan	0.7833729	2.207911
MS Dhoni	0.7646577	2.812875
CH Gayle	0.7407404	2.665148
S Dhawan	0.7202782	2.731031
RG Sharma	0.6907171	2.724523
G Gambhir	0.6889001	2.812875
BB McCullum	0.6798980	2.812875
AM Rahane	0.6590200	2.812875
AC Gilchrist	0.6580043	2.812875
KD Karthik	0.6579230	2.812875
JP Duminy	0.6471892	2.411876
V Sehwag	0.6451609	2.812875
AT Rayudu	0.6406005	2.812875
DR Smith	0.6395753	2.458842
SV Samson	0.6348433	2.812875
SE Marsh	0.6279306	2.812875
MK Tiwary	0.6236933	2.814401
DPMD Jayawardene	0.6187604	2.812875
KA Pollard	0.6121924	2.324072
RR Pant	0.6087224	2.812875
KL Rahul	0.6063455	2.812875
JH Kallis	0.6054579	2.030277
WP Saha	0.6048418	2.812875
Yuvraj Singh	0.6044655	2.596451
RA Jadeja	0.6022790	1.943305
MA Agarwal	0.6006772	2.812875
SPD Smith	0.5961683	2.812875
GJ Maxwell	0.5912107	2.576077
MK Pandey	0.5877410	2.812875
PA Patel	0.5752299	2.812875
M Vijay	0.5614641	2.716538
SR Tendulkar	0.5586312	2.823383
BJ Hodge	0.5545283	2.781617
F du Plessis	0.5512560	2.812875
SS Iyer	0.5471879	2.812875
R Dravid	0.5407564	2.812875
KC Sangakkara	0.5323191	2.812875
DA Miller	0.5273600	2.812875
MEK Hussey	0.5253249	2.812875
KM Jadhav	0.5153471	2.812875
IK Pathan	0.5148590	2.259891
AJ Finch	0.5104671	2.882206

player	sr_t20	er_t20
TM Dilshan	0.5097581	2.787679
STR Binny	0.5042926	2.385260

Inference: The Top 50 outstanding players in the league among the best players in the league.

TOP_CALIBER_PLAYERS:

Order of players based on their summarized player values

1. Players by their calibre - strike rate + economy rate,

2. contribution in win/ loss situation and

```
top_calibre_players <- top_rate_players %>%
select(-player_value) %>%
full_join(top_excel_players, by = "player") %>%
mutate(sr_t20 = replace_na(sr_t20, batsmen_avgs$median_runs / batsmen_avgs$avg_balls)) %>%
mutate(er_t20 = replace_na(er_t20, bowler_avgs$avg_runs / bowler_avgs$median_balls)) %>%
full_join(top_contri_players, by="player") %>%
mutate(player_value = 100 * ((reg_str_rate + sr_t20) + 1 /
(reg_eco_rate + er_t20) +
player_contri_rate)) %>%
select(player, player_value) %>%
arrange(desc(player_value)) %>%
mutate(rank = row_number())
top_calibre_players %>%
head(20)
```

3. player's excellence against the best in business

```
## # A tibble: 20 x 3
##    player          player_value  rank
##    <chr>                  <dbl> <int>
##  1 SR Watson               286.     1
##  2 CH Gayle                279.     2
##  3 AD Russell              278.     3
##  4 AB de Villiers          275.     4
##  5 DA Warner               273.     5
##  6 YK Pathan               268.     6
##  7 SK Raina                265.     7
##  8 RR Pant                 264.     8
##  9 SP Narine               264.     9
## 10 V Sehwag                258.    10
## 11 GJ Maxwell              254.    11
```

```
## 12 V Kohli                 253.    12
## 13 KA Pollard              251.    13
## 14 RG Sharma               247.    14
## 15 DR Smith                246.    15
## 16 RV Uthappa              244.    16
## 17 DJ Bravo                243.    17
## 18 S Dhawan                242.    18
## 19 MS Dhoni                241.    19
## 20 SE Marsh                241.    20
```

Since, there are 8 teams in the IPL and they need a list of most valuable players in the league to pick based on the above calculated parameters, we suggested the top 150 MVP players based on their performances in the past years

```
# Top 150 calibre players by player value
top_150_calibre_players <- top_calibre_players %>%
head(150)
top_150_calibre_players%>%
knitr::kable()
```

player	player_value	rank
SR Watson	286.2781	1
CH Gayle	279.4763	2
AD Russell	278.0665	3
AB de Villiers	275.4043	4
DA Warner	273.0908	5
YK Pathan	267.9109	6
SK Raina	265.3342	7
RR Pant	264.2226	8
SP Narine	264.0961	9
V Sehwag	257.7910	10
GJ Maxwell	253.8788	11
V Kohli	253.1660	12
KA Pollard	250.5207	13
RG Sharma	246.7369	14
DR Smith	246.3674	15
RV Uthappa	243.7751	16
DJ Bravo	243.0456	17
S Dhawan	242.2364	18
MS Dhoni	241.1492	19
SE Marsh	240.5851	20
KL Rahul	239.7893	21
AC Gilchrist	239.5414	22
JH Kallis	238.1602	23
JP Duminy	235.7496	24
Yuvraj Singh	235.4209	25
HH Pandya	235.3596	26
JA Morkel	234.4170	27
RA Jadeja	233.7764	28
CH Morris	232.1027	29
BB McCullum	232.0753	30

player	player_value	rank
SPD Smith	231.5587	31
AM Rahane	230.9525	32
G Gambhir	230.7353	33
BJ Hodge	228.0052	34
Harbhajan Singh	227.9050	35
SV Samson	227.6856	36
F du Plessis	227.1501	37
JC Buttler	226.5156	38
CA Lynn	225.9773	39
IK Pathan	225.3059	40
M Ali	224.8685	41
N Rana	224.8264	42
KD Karthik	224.8215	43
AT Rayudu	224.0341	44
JP Faulkner	222.7345	45
AR Patel	222.3960	46
KH Pandya	221.9977	47
AJ Finch	221.5096	48
ML Hayden	219.9535	49
Q de Kock	219.9528	50
WP Saha	219.9048	51
DA Miller	219.4818	52
M Vijay	219.4604	53
BA Stokes	219.1206	54
Shakib Al Hasan	217.7197	55
SR Tendulkar	217.2887	56
KK Cooper	217.1938	57
LMP Simmons	217.1079	58
KP Pietersen	217.0258	59
SS Iyer	216.7518	60
A Symonds	216.7315	61
AD Mathews	216.2824	62
Rashid Khan	214.9284	63
ST Jayasuriya	214.5834	64
MC Henriques	214.2633	65
MK Tiwary	213.8250	66
DPMD Jayawardene	213.8078	67
Azhar Mahmood	212.7629	68
RA Tripathi	211.9048	69
MF Maharoof	211.4654	70
MEK Hussey	210.4893	71
DJ Hussey	210.3496	72
MK Pandey	210.2057	73
STR Binny	209.9834	74
NLTC Perera	209.8843	75
MA Agarwal	209.8140	76
M Vohra	209.6447	77
HM Amla	208.8056	78
CR Brathwaite	208.6980	79
KS Williamson	208.4386	80
PA Patel	207.7451	81
KC Sangakkara	207.5114	82

player	player_value	rank
LRPL Taylor	207.3271	83
K Gowtham	207.2508	84
DL Chahar	205.5721	85
M Morkel	204.9599	86
R Dravid	204.8285	87
JD Ryder	204.3287	88
KM Jadhav	204.2013	89
A Ashish Reddy	203.7592	90
BCJ Cutting	203.6903	91
Mandeep Singh	203.6576	92
HV Patel	203.5207	93
KK Nair	203.2654	94
TM Dilshan	203.0610	95
J Bairstow	202.5690	96
MJ McClenaghan	201.4567	97
P Shaw	201.2977	98
J Archer	200.7274	99
AS Yadav	199.8031	100
CL White	199.6614	101
J Botha	199.3507	102
SA Yadav	199.3059	103
JR Hopes	199.0792	104
AJ Tye	198.7340	105
PP Chawla	198.1781	106
Bipul Sharma	198.1225	107
S Gill	197.9656	108
Umar Gul	197.8405	109
R Vinay Kumar	197.3019	110
DT Christian	196.9072	111
R Ashwin	196.7360	112
DW Steyn	196.5716	113
RN ten Doeschate	196.5103	114
Ishan Kishan	196.4574	115
S Gopal	196.4341	116
SN Khan	196.0900	117
RS Bopara	196.0602	118
V Shankar	195.9876	119
C de Grandhomme	195.8497	120
SN Thakur	195.7367	121
Mohammad Nabi	195.5601	122
KV Sharma	195.5498	123
Ankit Sharma	195.1889	124
Shahid Afridi	194.9222	125
S Curran	194.5918	126
Gurkeerat Singh	193.7496	127
MS Gony	193.4382	128
S Badrinath	192.9870	129
P Negi	192.4278	130
R Bhatia	192.2423	131
N Pooran	192.2009	132
LJ Wright	191.9673	133
SS Tiwary	191.7566	134

player	player_value	rank
GJ Bailey	191.6742	135
PJ Cummins	191.4881	136
EJG Morgan	190.7330	137
TL Suman	190.2680	138
MP Stoinis	190.1025	139
TG Southee	189.7596	140
AM Nayar	189.6730	141
MV Boucher	189.5303	142
OA Shah	189.1976	143
HH Gibbs	189.1440	144
B Kumar	189.1200	145
TM Head	188.6646	146
MJ Guptill	188.6074	147
PD Collingwood	188.3372	148
NV Ojha	188.2495	149
D Wiese	188.0056	150

WIN Predictor

1. Since we have seen venue, batting_turn, toss wins have significant effect on the runs scored and matches won by teams, we will consider "venue", "toss_winner" & "toss_decision" variables along with "team1 (first team to bat)" & "team2 (second team to bat)" as predictor variables to predict the "winner (response variable)" of the match.

2. We will only consider the top 8 teams, which have played the most matches and also the current teams for our model building and predictions.

3. We have 8 different classifiers; 8 different teams that can win matches. Since this is a classification problem and our data is all factors, we use a few machine learning methods ("Naive Bayes", "Regression Tree", "Random Forest", "Multinomial Regression", "Linear Discriminant Analysis" and "K Nearest Neighbours") to train models and predict results.

4. Accuracy will not be the measure of our model prediction strength. This is because, we are not really interested in predicting negatives; rather we are more interested in positive predictions. Besides we have all factor data and many classifiers. Thus our metric of model evaluation will be "F1 Score", which is based on "recall" and "precision".

5. What the models do is to predict the winner of a match given the opponent, venue, toss winner and toss decision.

6. We will create two data sets, "train_set" and "test_set" for training and testing our model

```
# Create the list of top 8 teams from matches_team (teams Vs matches played)

# no.of matches played by each team
played1 <- mat_ds %>%
group_by(team1) %>%
```

```
summarize(count1 = n()) %>%
arrange(team1) %>%
rename(team = team1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
played2 <- mat_ds %>%
group_by(team2) %>%
summarize(count2 = n()) %>%
arrange(team2) %>%
rename(team = team2)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
matches_team <- played1 %>%
full_join(played2, by = "team") %>%
mutate(n_matches_played = count1 + count2) %>%
select(team, n_matches_played) %>% arrange(desc(n_matches_played))


teams <- matches_team %>%
top_n(8) %>%
select (team)
```

```
## Selecting by n_matches_played
```

```
teams
```

```
## # A tibble: 8 x 1
##   team
##   <chr>
## 1 MI
## 2 SRH
## 3 RCB
## 4 KKR
## 5 DC
## 6 KXIP
## 7 CSK
## 8 RR
```

```
# Do required pre-processing and data wrangling
dat_set <- matches %>%
select(first_bat_team = team1,
second_bat_team = team2, winner, venue, toss_winner, toss_decision) %>%
filter(winner != "" & first_bat_team %in% teams$team & second_bat_team %in% teams$team) %>%
mutate(first_bat_team = as.factor(first_bat_team), second_bat_team = as.factor(second_bat_team),
winner = as.factor(winner), venue = as.factor(venue), toss_decision = as.factor(toss_decision),
toss_winner = as.factor(toss_winner))
any(is.na(dat_set))
```

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

```r
summary(dat_set)
```

```
##  first_bat_team second_bat_team     winner
##  SRH    : 97    DC     : 91     MI     :100
##  MI     : 92    RCB    : 88     CSK    : 95
##  CSK    : 83    KKR    : 87     KKR    : 84
##  KXIP   : 83    KXIP   : 78     SRH    : 75
##  KKR    : 76    MI     : 78     KXIP   : 74
##  RCB    : 76    RR     : 76     RCB    : 72
##  (Other):134    (Other):143     (Other):141
##                                         venue      toss_winner  toss_decision
##  Eden Gardens                              : 71   MI     : 89   bat  :251
##  M Chinnaswamy Stadium                     : 64   CSK    : 86   field:390
##  Wankhede Stadium                          : 64   KKR    : 82
##  Feroz Shah Kotla                          : 59   DC     : 81
##  Rajiv Gandhi International Stadium, Uppal: 49   SRH    : 78
##  MA Chidambaram Stadium, Chepauk           : 45   RR     : 76
##  (Other)                                   :289   (Other):149
```

```r
dim(dat_set)
```

```
## [1] 641   6
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```r
options(digits=4)
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```r
# test set will be approx 10% of our dataset
test_index <- createDataPartition(dat_set$winner, times = 1, p = 0.1, list = FALSE)
train_set <- dat_set[-test_index,]
temp_set <- dat_set[test_index,]

# Make sure all variable values in test set are also in train set
test_set <- temp_set %>%
semi_join(train_set, by = "first_bat_team") %>%
semi_join(train_set, by = "second_bat_team") %>%
semi_join(train_set, by = "venue") %>%
semi_join(train_set, by = "toss_decision") %>%
semi_join(train_set, by = "toss_winner")

# Add rows removed from temp set back into train set
removed <- anti_join(temp_set, test_set)
```

```
## Joining, by = c("first_bat_team", "second_bat_team", "winner", "venue", "toss_winner", "toss_decisio
```

```r
train_set <- rbind(train_set, removed)

# Check dimensions & variable names of train_set and test_set
dim(train_set)
```

```
## [1] 573   6
```

```r
dim(test_set)
```

```
## [1] 68  6
```

```r
names(train_set)
```

```
## [1] "first_bat_team"  "second_bat_team" "winner"          "venue"
## [5] "toss_winner"     "toss_decision"
```

```r
names(test_set)
```

```
## [1] "first_bat_team"  "second_bat_team" "winner"          "venue"
## [5] "toss_winner"     "toss_decision"
```

Naive Bayes

```r
# Fit the Model based on "Naive Bayes" method, predict, test, calculate F1 score for all classes
fit_nb <- train(winner ~ ., method = "naive_bayes", data = train_set)
```

```
## Warning: model fit failed for Resample01: usekernel= TRUE, laplace=0, adjust=1 Error in density.defa
##   need at least 2 points to select a bandwidth automatically

## Warning: model fit failed for Resample04: usekernel= TRUE, laplace=0, adjust=1 Error in density.defa
##   need at least 2 points to select a bandwidth automatically

## Warning: predictions failed for Resample06: usekernel= TRUE, laplace=0, adjust=1 Error in stats::app
##   need at least two non-NA values to interpolate

## Warning: predictions failed for Resample07: usekernel= TRUE, laplace=0, adjust=1 Error in stats::app
##   need at least two non-NA values to interpolate

## Warning: model fit failed for Resample11: usekernel= TRUE, laplace=0, adjust=1 Error in density.defa
##   need at least 2 points to select a bandwidth automatically

## Warning: predictions failed for Resample12: usekernel= TRUE, laplace=0, adjust=1 Error in stats::app
##   need at least two non-NA values to interpolate

## Warning: model fit failed for Resample15: usekernel= TRUE, laplace=0, adjust=1 Error in density.defa
##   need at least 2 points to select a bandwidth automatically
```

```
## Warning: model fit failed for Resample21: usekernel= TRUE, laplace=0, adjust=1 Error in density.defau
##   need at least 2 points to select a bandwidth automatically
```

```
## Warning: predictions failed for Resample24: usekernel= TRUE, laplace=0, adjust=1 Error in stats::appr
##   need at least two non-NA values to interpolate
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
pre_nb <- predict(fit_nb, test_set)
F1_nb <- confusionMatrix(pre_nb, test_set$winner)$byClass[,"F1"]
F1_nb <- as.data.frame(t(F1_nb)) %>% mutate(avg_F1_score = rowMeans(.))
F1_nb
```

```
##   Class: CSK Class: DC Class: KKR Class: KXIP Class: MI Class: None Class: RCB
## 1     0.2381        NA         NA          NA    0.5926          NA        0.2
##   Class: RR Class: SRH avg_F1_score
## 1        NA        0.4           NA
```

Inference: As we can see, though Naive Bayes quickly converged in training the model, it did not predict for certain classes, thereby did not produce F1 scores for those classes.

```
# Make column names more readable
colnames(F1_nb) = gsub("Class: ", "", colnames(F1_nb))
# F1 table for different models
F1_table <- data.frame(Model = "Naive Bayes") %>% bind_cols(F1_nb)
F1_table %>% knitr::kable()
```

Model	CSK	DC	KKR	KXIP	MI	None	RCB	RR	SRH	avg_F1_score
Naive Bayes	0.2381	NA	NA	NA	0.5926	NA	0.2	NA	0.4	NA

Decision Tree

```
# Fit the Model based on "rpart (CART)" method, predict, test, calculate F1 score for all classes
fit_rp <- train(winner ~ ., method = "rpart", data = train_set)
pre_rp <- predict(fit_rp, test_set)
F1_rp <- confusionMatrix(pre_rp, test_set$winner)$byClass[,"F1"]
F1_rp <- as.data.frame(t(F1_rp)) %>% mutate(avg_F1_score = rowMeans(.))
F1_rp
```

```
##   Class: CSK Class: DC Class: KKR Class: KXIP Class: MI Class: None Class: RCB
## 1     0.2857        NA     0.6667          NA    0.6667          NA         NA
##   Class: RR Class: SRH avg_F1_score
## 1        NA         NA           NA
```

```r
# Make column names more readable
colnames(F1_rp) = gsub("Class: ", "", colnames(F1_rp))
# Update F1 table - continued.2
F1_table <- bind_rows(F1_table,
data.frame(Model = "CART (rpart)") %>% bind_cols(F1_rp))
F1_table %>% knitr::kable()
```

Model	CSK	DC	KKR	KXIP	MI	None	RCB	RR	SRH	avg_F1_score
Naive Bayes	0.2381	NA	NA	NA	0.5926	NA	0.2	NA	0.4	NA
CART (rpart)	0.2857	NA	0.6667	NA	0.6667	NA	NA	NA	NA	NA

Mulnomial Logistic Regression

```r
# Fit the Model based on "multinom" method, predict, test, calculate F1 score for all classes
fit_mn <- train(winner ~ ., method = "multinom", data = train_set, trace = FALSE)
```

```
## Warning in nnet::multinom(.outcome ~ ., data = dat, decay = param$decay, : group
## 'None' is empty

## Warning in nnet::multinom(.outcome ~ ., data = dat, decay = param$decay, : group
## 'None' is empty

## Warning in nnet::multinom(.outcome ~ ., data = dat, decay = param$decay, : group
## 'None' is empty

## Warning in nnet::multinom(.outcome ~ ., data = dat, decay = param$decay, : group
## 'None' is empty

## Warning in nnet::multinom(.outcome ~ ., data = dat, decay = param$decay, : group
## 'None' is empty

## Warning in nnet::multinom(.outcome ~ ., data = dat, decay = param$decay, : group
## 'None' is empty
```

```r
pre_mn <- predict(fit_mn, test_set)
F1_mn <- confusionMatrix(pre_mn, test_set$winner)$byClass[,"F1"]
F1_mn <- as.data.frame(t(F1_mn)) %>% mutate(avg_F1_score = rowMeans(.))
F1_mn
```

```
##   Class: CSK Class: DC Class: KKR Class: KXIP Class: MI Class: None Class: RCB
## 1        NaN    0.4286        0.8         0.4      0.56          NA     0.4444
##   Class: RR Class: SRH avg_F1_score
## 1       0.5     0.3158           NA
```

```r
# Make column names more readable
colnames(F1_mn) = gsub("Class: ", "", colnames(F1_mn))
```

```r
# Update F1 table - continued.4
F1_table <- bind_rows(F1_table,
data.frame(Model = "Multinom") %>% bind_cols(F1_mn))
F1_table %>% knitr::kable()
```

Model	CSK	DC	KKR	KXIP	MI	None	RCB	RR	SRH	avg_F1_score
Naive Bayes	0.2381	NA	NA	NA	0.5926	NA	0.2000	NA	0.4000	NA
CART (rpart)	0.2857	NA	0.6667	NA	0.6667	NA	NA	NA	NA	NA
Multinom	NaN	0.4286	0.8000	0.4	0.5600	NA	0.4444	0.5	0.3158	NA

Linear Discriminant Analysis

```r
# Fit the Model based on "LDA" method, predict, test, calculate F1 score for all classes
fit_lda <- train(winner ~ ., method = "lda", data = train_set)
```

```
## Warning: model fit failed for Resample01: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 48 appears to be constant within groups

## Warning: model fit failed for Resample02: parameter=none Error in lda.default(x, grouping, ...) :
##   variables 18 28 50 appear to be constant within groups

## Warning: model fit failed for Resample03: parameter=none Error in lda.default(x, grouping, ...) :
##   variables 17 48 appear to be constant within groups

## Warning: model fit failed for Resample04: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 27 appears to be constant within groups

## Warning: model fit failed for Resample05: parameter=none Error in lda.default(x, grouping, ...) :
##   variables 18 48 appear to be constant within groups

## Warning: model fit failed for Resample06: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 48 appears to be constant within groups

## Warning in lda.default(x, grouping, ...): group None is empty

## Warning: model fit failed for Resample07: parameter=none Error in lda.default(x, grouping, ...) :
##   variables 17 48 appear to be constant within groups

## Warning: model fit failed for Resample08: parameter=none Error in lda.default(x, grouping, ...) :
##   variables 17 37 48 appear to be constant within groups

## Warning: model fit failed for Resample09: parameter=none Error in lda.default(x, grouping, ...) :
##   variables 48 50 appear to be constant within groups

## Warning: model fit failed for Resample10: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 48 appears to be constant within groups
```

```
## Warning: model fit failed for Resample11: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 15 appears to be constant within groups

## Warning: model fit failed for Resample12: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 48 appears to be constant within groups

## Warning: model fit failed for Resample14: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 37 appears to be constant within groups

## Warning: model fit failed for Resample15: parameter=none Error in lda.default(x, grouping, ...) :
##   variables 37 50 appear to be constant within groups

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning: model fit failed for Resample18: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 48 appears to be constant within groups

## Warning: model fit failed for Resample19: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 50 appears to be constant within groups

## Warning: model fit failed for Resample20: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 48 appears to be constant within groups

## Warning: model fit failed for Resample21: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 37 appears to be constant within groups

## Warning in lda.default(x, grouping, ...): group None is empty

## Warning: model fit failed for Resample22: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 17 appears to be constant within groups

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning: model fit failed for Resample25: parameter=none Error in lda.default(x, grouping, ...) :
##   variable 48 appears to be constant within groups

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```r
pre_lda <- predict(fit_lda, test_set)
F1_lda <- confusionMatrix(pre_lda, test_set$winner)$byClass[,"F1"]
F1_lda <- as.data.frame(t(F1_lda)) %>% mutate(avg_F1_score = rowMeans(.))
F1_lda
```

```
##   Class: CSK Class: DC Class: KKR Class: KXIP Class: MI Class: None Class: RCB
## 1     0.6316    0.2857     0.6667      0.4615    0.5926         NaN        0.5
##   Class: RR Class: SRH avg_F1_score
## 1    0.7692     0.4286          NaN
```

```r
# Make column names more readable
colnames(F1_lda) = gsub("Class: ", "", colnames(F1_lda))
# Update F1 table - continued.5
F1_table <- bind_rows(F1_table,
data.frame(Model = "LDA") %>% bind_cols(F1_lda))

F1_table %>% knitr::kable()
```

Model	CSK	DC	KKR	KXIP	MI	None	RCB	RR	SRH	avg_F1_score
Naive Bayes	0.2381	NA	NA	NA	0.5926	NA	0.2000	NA	0.4000	NA
CART (rpart)	0.2857	NA	0.6667	NA	0.6667	NA	NA	NA	NA	NA
Multinom	NaN	0.4286	0.8000	0.4000	0.5600	NA	0.4444	0.5000	0.3158	NA
LDA	0.6316	0.2857	0.6667	0.4615	0.5926	NaN	0.5000	0.7692	0.4286	NaN

Random Forest

```r
# Fit the Model based on "rf (Random Forest)" method, predict, test,
# calculate F1 score for all classes
trainctrl <- trainControl(method="cv")
fit_rf <- train(winner ~ ., method = "rf", data = train_set, trControl=trainctrl)
pre_rf <- predict(fit_rf, test_set)
F1_rf <- confusionMatrix(pre_rf, test_set$winner)$byClass[,"F1"]
F1_rf <- as.data.frame(t(F1_rf)) %>% mutate(avg_F1_score = rowMeans(.))
F1_rf
```

```
##   Class: CSK Class: DC Class: KKR Class: KXIP Class: MI Class: None Class: RCB
## 1     0.4444    0.2667     0.8421      0.4615    0.5185          NA     0.5333
##   Class: RR Class: SRH avg_F1_score
## 1    0.6667       0.25           NA
```

```r
# Make column names more readable
colnames(F1_rf) = gsub("Class: ", "", colnames(F1_rf))
# Update F1 table - continued.3
F1_table <- bind_rows(F1_table,
data.frame(Model = "Random Forest (rf)") %>% bind_cols(F1_rf))

F1_table %>% knitr::kable()
```

Model	CSK	DC	KKR	KXIP	MI	None	RCB	RR	SRH	avg_F1_score
Naive Bayes	0.2381	NA	NA	NA	0.5926	NA	0.2000	NA	0.4000	NA
CART (rpart)	0.2857	NA	0.6667	NA	0.6667	NA	NA	NA	NA	NA
Multinom	NaN	0.4286	0.8000	0.4000	0.5600	NA	0.4444	0.5000	0.3158	NA
LDA	0.6316	0.2857	0.6667	0.4615	0.5926	NaN	0.5000	0.7692	0.4286	NaN
Random Forest (rf)	0.4444	0.2667	0.8421	0.4615	0.5185	NA	0.5333	0.6667	0.2500	NA

K-Nearest Neighbours

```r
# Fit the Model based on "KNN" method, predict, test, calculate F1 score for all classes
fit_knn <- train(winner ~ ., method = "knn", data = train_set)
```

```
## Warning: predictions failed for Resample07: k=5 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample07: k=7 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample07: k=9 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample20: k=5 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample20: k=7 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample20: k=9 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample21: k=5 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample21: k=7 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning: predictions failed for Resample21: k=9 Error in dimnames(x) <- dn :
##   length of 'dimnames' [2] not equal to array extent

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```r
pre_knn <- predict(fit_knn, test_set)
F1_knn <- confusionMatrix(pre_knn, test_set$winner)$byClass[,"F1"]
F1_knn <- as.data.frame(t(F1_knn)) %>% mutate(avg_F1_score = rowMeans(.))
F1_knn
```

```
##   Class: CSK Class: DC Class: KKR Class: KXIP Class: MI Class: None Class: RCB
## 1     0.6087    0.3077     0.7778      0.3333    0.5217          NA     0.5333
##   Class: RR Class: SRH avg_F1_score
## 1    0.6154     0.3333           NA
```

```r
# Make column names more readable
colnames(F1_knn) = gsub("Class: ", "", colnames(F1_knn))
# Update F1 table - Final
F1_table <- bind_rows(F1_table,
data.frame(Model = "KNN") %>% bind_cols(F1_knn))

F1_table %>% knitr::kable()
```

Model	CSK	DC	KKR	KXIP	MI	None	RCB	RR	SRH	avg_F1_score
Naive Bayes	0.2381	NA	NA	NA	0.5926	NA	0.2000	NA	0.4000	NA
CART (rpart)	0.2857	NA	0.6667	NA	0.6667	NA	NA	NA	NA	NA
Multinom	NaN	0.4286	0.8000	0.4000	0.5600	NA	0.4444	0.5000	0.3158	NA
LDA	0.6316	0.2857	0.6667	0.4615	0.5926	NaN	0.5000	0.7692	0.4286	NaN
Random Forest (rf)	0.4444	0.2667	0.8421	0.4615	0.5185	NA	0.5333	0.6667	0.2500	NA
KNN	0.6087	0.3077	0.7778	0.3333	0.5217	NA	0.5333	0.6154	0.3333	NA

Inference: LDA and KNN performed much better than other ML Algorithms for predicting the winner of the match in terms of higher F-Score for all 8 teams.

The samples we have for different teams (number of observations) is not similar. Some teams have played much more matches than others. This introduces bias in our data and thus our models may not do effective job in predicting the winners. For example, "Chennai Super Kings", which is top consistent performing team did not play for 2 seasons. This fact could have probably reduced its prediction as the winner by some of the models in favour of other team.

Future Work

1. Add Fielding Component to Player Analysis and Ratings

2. Cluster Best Indian and Foreign Players based on their skill sets and Rating to predict the approximate price in the auction.

3. Fantasy Team Suggestion for Fans for each match to maximise the points earned.

4. Collect Real-Time data such as Wagon Wheel and Pitch Map of Players to formulate individual game plans.
