Patterns of play in women's tennis - Simona Halep

Silvia Dinu

July 31, 2016

Contents

Simona Halep is a Romanian tennis player that had a rapid rise in the rankings in the last 3 years (2012 - Top 50 Player, 2013 - No. 11, 2014 - No. 3, 2015 - No. 2). As tennis is a game of patterns, the goal of this project is to make an analysis of the way Simona Halep has changed her game from a lower ranked player to a Top 5 competitor. The beginning of 2016 finds her at a crossroads. Currently at number 5 in the WTA rankings, having to overcome a series of nagging injuries that did not allow her to perform at the usual capacity, the romanian player tries to regain confidence and compete again at the highest level.Does she have the game to compete with the best on regular basis and if not, what does she have to do in order to maintain herself in the top?

Overview

Short ranking history and number of titles by year

2012 - First Top 50 season; runner-up at Brussels; semifinalist at Fès; quarterfinalist twice

2013 - Near-Top 10 season (finishing No.11); went 7-9 in first nine WTA main draws, reaching 3 round once and 2 round five times but falling 1 round three times and in qualifying once; went 43-8 in next 14 WTA main draws, highlighted by first **six WTA titles** at Nürnberg, 's-Hertogenbosch, Budapest, New Haven, Moscow and Sofia; also semifinalist at Rome and quarterfinalist at Cincinnati; made Top 20 debut on August 26 (after New Haven; rose from No.23 to No.19) and peaked at No.11 on November 4 year-end rankings

2014 - First Top 10 season (finishing No.3); won **two WTA titles** at Doha and Bucharest; runner-up three times at Madrid, Roland Garros and WTA Finals; semifinalist twice at Indian Wells and Wimbledon; quarterfinalist three times at Australian Open, Cincinnati and Beijing;having started the year at No.11, made Top 10 debut on January 27, Top 5 debut on March 17 and set career-high No.2 on August 11 (highest-ranked Romanian in WTA history)

2015 - Best season to date (finishing No.2); won **three WTA titles** at Shenzhen, Dubai and Indian Wells; runner-up twice at Toronto and Cincinnati; semifinalist four times at Miami, Stuttgart, Rome and US Open; quarterfinalist three times at Australian Open, Birmingham and Guangzhou

2016 - Current ranking 5; won **two WTA titles** at Madrid and Bucharest; semifinalist at Sydney; quarterfinalist at Indian Wells, Miami and Wimbledon

The figures presented above show a good trend. In 2013 at 23 years old she won her first six titles, 2 on clay, 1 on grass and 2 on hard courts. Halep became the only player of the year to win titles on clay, grass, hard, and indoor courts, as well as the second most successful player behind Serena Williams in terms of number of titles won. 2014 and 2015 were transitional years in which she tried to get more used with the pressure of being a top player. The beginning of 2016 saw a Simona Halep struggling with injuries and a loss of form. Her first title of the year came in May in Madrid.

The present analysis looks at the numbers on her serve, the winners and the unforced errors, the strategy that she is following, the way her game developed through the last four years and based on the data tries to predict whether or not she is going to win.

Datasets

Csv files used from:

https://github.com/JeffSackmann/tennis_MatchChartingProject

The initial files were named as:

1. overview1
2. forced_errors
3. return_outcomes
4. servebasics
5. servedirection
6. shotdirection
7. surface
8. win_lose
9. games

```
overview1 <- read.csv("~/Learn R/Proiect/overview1.txt")
forced_errors <- read.csv("~/Learn R/Proiect/forced_errors.txt")
return_outcomes <- read.csv("~/Learn R/Proiect/return_outcomes.txt")
servebasics <- read.csv("~/Learn R/Proiect/servebasics.txt")
servedirection <- read.csv("~/Learn R/Proiect/servedirection.txt")
shotdirection <- read.csv("~/Learn R/Proiect/shotdirection.txt")
surface <- read.csv("~/Learn R/Proiect/surface.txt", sep=";")
win_lose <- read.csv("~/Learn R/Proiect/win_lose.txt", sep=";")
games <- read.csv("~/Learn R/Proiect/games.txt", sep="")
```

Initial format of the data with the variables:

overview1

1. match_id
2. player
3. set
4. serve_pts
5. aces
6. dfs
7. first_in
8. first_won
9. second_in
10. second_won
11. bk_pts
12. bp_saved
13. return_pts
14. return_pts_won
15. winners
16. winners_fh
17. winners_bh
18. unforced
19. unforced_fh
20. unforced_bh

```
head(overview1)
```

```
##                                                 match_id player   set
## 1 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1 Total
## 2 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      2 Total
## 3 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1     1
## 4 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      2     1
## 5 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1     2
## 6 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      2     2
##   serve_pts aces dfs first_in first_won second_in second_won bk_pts
## 1        82    1   2       46        24        36         19     13
## 2        76    2  11       50        33        26          6     13
## 3        37    1   0       21        10        16         11      8
## 4        29    2   4       19        13        10          2      4
## 5        24    0   2       14         7        10          2      4
## 6        19    0   4       13        11         6          2      1
##   bp_saved return_pts return_pts_won winners winners_fh winners_bh
## 1        7         76             37      18         11          5
## 2        7         82             39      30         12         13
## 3        6         29             14       8          3          3
## 4        1         37             16      12          2          6
## 5        1         19              6       5          4          1
## 6        1         24             15      11          5          6
##   unforced unforced_fh unforced_bh
## 1       33          13          18
## 2       53          25          17
## 3       11           5           6
## 4       23          10           9
## 5       11           3           6
## 6        9           2           3
```

forced_errors

1. match_id
2. row
3. pts
4. pl1_won
5. pl1_winners
6. pl1_forced
7. pl1_unforced
8. pl2_won
9. pl2_winners
10. pl2_forced
11. pl2_unforced

```
head(forced_errors)
```

```
##                                                 match_id   row pts
## 1 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova Total 158
## 2 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova   1-3  77
## 3 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 1-3-1  38
```

```
## 4 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 1-3-2  39
## 5 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova   4-6  44
## 6 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 4-6-1  25
##   pl1_won pl1_winners pl1_forced pl1_unforced pl2_won pl2_winners
## 1      80          17         10           31      78          27
## 2      46           8          4           13      31           7
## 3      26           6          3            7      12           2
## 4      20           2          1            6      19           5
## 5      18           5          2           12      26          10
## 6       6           1          1            9      19           6
##   pl2_forced pl2_unforced
## 1         18           42
## 2          9           23
## 3          1           17
## 4          8            6
## 5          4           11
## 6          4            4
```

return_outcomes

1. match_id
2. row
3. pts
4. pts_won
5. returnable
6. returnable_won
7. in_play
8. in_play_won
9. winners
10. total_shots

```
head(return_outcomes)
```

```
##                                                match_id player   row
## 1 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1 Total
## 2 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1  v1st
## 3 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1  v2nd
## 4 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1    fh
## 5 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1    bh
## 6 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova      1    gs
##   pts pts_won returnable returnable_won in_play in_play_won winners
## 1  76      37         58             26      54          26       2
## 2  50      17         43             17      40          17       1
## 3  15       9         15              9      14           9       1
## 4  32      15         30             15      28          15       2
## 5  28      11         28             11      26          11       0
## 6  56      24         54             24      50          24       2
##   total_shots
## 1         336
## 2         223
## 3         102
## 4         177
```

```
## 5         143
## 6         302
```

servebasics

1. match_id
2. row
3. pts
4. pts_won
5. aces
6. unret
7. forced_err
8. pts_won_lte_3_shots
9. wide
10. body
11. t

```
head(servebasics)
```

```
##                                                  match_id     row pts
## 1 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 1 Total  82
## 2 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     1 1  46
## 3 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     1 2  36
## 4 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 2 Total  76
## 5 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     2 1  50
## 6 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     2 2  26
##   pts_won aces unret forced_err pts_won_lte_3_shots wide body  t
## 1      43    1     1          2                  21   18   38 26
## 2      24    1     1          2                  13   11   17 18
## 3      19    0     0          0                   8    7   21  8
## 4      39    2     3          2                  14   18   35 23
## 5      33    2     3          2                  13   13   16 21
## 6       6    0     0          0                   1    5   19  2
```

servedirection

1. match_id
2. row
3. deuce_wide
4. deuce_middle
5. deuce_t
6. ad_wide
7. ad_middle
8. ad_t
9. err_net
10. err_wide
11. err_deep
12. err_wide_deep
13. err_foot
14. err_unknown

```
head(servedirection)
```

```
##                                                 match_id     row
## 1 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 1 Total
## 2 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     1 1
## 3 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     1 2
## 4 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 2 Total
## 5 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     2 1
## 6 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     2 2
##   deuce_wide deuce_middle deuce_t ad_wide ad_middle ad_t err_net err_wide
## 1         12           17      15       6        21   11      19        8
## 2          9            8      10       2         9    8       0        0
## 3          3            9       5       4        12    3      19        8
## 4         11           16      12       7        19   11      24        6
## 5          9            9      11       4         7   10       0        0
## 6          2            7       1       3        12    1      24        6
##   err_deep err_wide_deep err_foot err_unknown
## 1       10             1        0           0
## 2        0             0        0           0
## 3       10             1        0           0
## 4        6             1        0           0
## 5        0             0        0           0
## 6        6             1        0           0
```

shotdirection

1. match_id
2. player
3. row
4. crosscourt
5. down_middle
6. down_the_line
7. inside_out
8. inside_in

```
head(servedirection)
```

```
##                                                 match_id     row
## 1 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 1 Total
## 2 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     1 1
## 3 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     1 2
## 4 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova 2 Total
## 5 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     2 1
## 6 20070901-W-US_Open-R32-Agnieszka_Radwanska-Maria_Sharapova     2 2
##   deuce_wide deuce_middle deuce_t ad_wide ad_middle ad_t err_net err_wide
## 1         12           17      15       6        21   11      19        8
## 2          9            8      10       2         9    8       0        0
## 3          3            9       5       4        12    3      19        8
## 4         11           16      12       7        19   11      24        6
## 5          9            9      11       4         7   10       0        0
## 6          2            7       1       3        12    1      24        6
```

```
##   err_deep err_wide_deep err_foot err_unknown
## 1       10             1        0           0
## 2        0             0        0           0
## 3       10             1        0           0
## 4        6             1        0           0
## 5        0             0        0           0
## 6        6             1        0           0
```

surface

```
head(surface)
```

```
##        Tournament Surface
## 1            Rome    clay
## 2       Nuremburg    clay
## 3 s_Hertogenbosch   grass
## 4        Budapest    clay
## 5       New_Haven    hard
## 6         US_Open    hard
```

win_lose

```
head(win_lose)
```

```
##   Win_Lose No_of_sets
## 1        W          2
## 2        L          2
## 3        L          3
## 4        W          2
## 5        W          2
## 6        W          3
```

games

```
head(games)
```

```
##   No_of_games
## 1          18
## 2          15
## 3          23
## 4          17
## 5          18
## 6          29
```

Data wrangling

The datasets were cleaned and combined using the dplyr and tidyr packages.

```
library(dplyr)
library(tidyr)
```

1. Separate match_id into 6 different variables: Date, Gender, Tournament, Round, Player 1 and Player 2

```
overview2 <- separate(overview1, match_id, c("Date", "Gender", "Tournament", "Round", "Player1", "Player
```

```
## Warning: Too many values at 12 locations: 601, 602, 603, 604, 605, 606,
## 7149, 7150, 7151, 7152, 7153, 7154
```

```
forced_errors1 <- separate(forced_errors, match_id, c("Date", "Gender", "Tournament", "Round", "Player1
```

```
## Warning: Too many values at 26 locations: 1152, 1153, 1154, 1155, 1156,
## 1157, 1158, 1159, 1160, 1161, 1162, 1163, 1164, 14032, 14033, 14034, 14035,
## 14036, 14037, 14038, ...
```

```
return_outcomes1 <- separate(return_outcomes, match_id, c("Date", "Gender", "Tournament", "Round", "Play
```

```
## Warning: Too many values at 84 locations: 3685, 3686, 3687, 3688, 3689,
## 3690, 3691, 3692, 3693, 3694, 3695, 3696, 3697, 3698, 3699, 3700, 3701,
## 3702, 3703, 3704, ...
```

```
servebasics1 <- separate(servebasics, match_id, c("Date", "Gender", "Tournament", "Round", "Player1", "
```

```
## Warning: Too many values at 12 locations: 535, 536, 537, 538, 539, 540,
## 6541, 6542, 6543, 6544, 6545, 6546
```

```
servedirection1 <- separate(servedirection, match_id, c("Date", "Gender", "Tournament", "Round", "Player
```

```
## Warning: Too many values at 12 locations: 535, 536, 537, 538, 539, 540,
## 6541, 6542, 6543, 6544, 6545, 6546
```

```
shotdirection1 <- separate(shotdirection, match_id, c("Date", "Gender", "Tournament", "Round", "Player1
```

```
## Warning: Too many values at 16 locations: 695, 696, 697, 698, 699, 700,
## 701, 702, 8616, 8617, 8618, 8619, 8620, 8621, 8622, 8623
```

The first 6 variables in each data set will look like those below:

```
head(overview2)
```

```
##       Date Gender Tournament Round             Player1         Player2
## 1 20070901      W    US_Open   R32 Agnieszka_Radwanska Maria_Sharapova
## 2 20070901      W    US_Open   R32 Agnieszka_Radwanska Maria_Sharapova
## 3 20070901      W    US_Open   R32 Agnieszka_Radwanska Maria_Sharapova
## 4 20070901      W    US_Open   R32 Agnieszka_Radwanska Maria_Sharapova
## 5 20070901      W    US_Open   R32 Agnieszka_Radwanska Maria_Sharapova
## 6 20070901      W    US_Open   R32 Agnieszka_Radwanska Maria_Sharapova
##   player   set serve_pts aces dfs first_in first_won second_in second_won
## 1      1 Total        82    1   2       46        24        36         19
## 2      2 Total        76    2  11       50        33        26          6
```

```
## 3      1     1        37    1   0       21        10        16         11
## 4      2     1        29    2   4       19        13        10          2
## 5      1     2        24    0   2       14         7        10          2
## 6      2     2        19    0   4       13        11         6          2
##   bk_pts bp_saved return_pts return_pts_won winners winners_fh winners_bh
## 1     13        7         76             37      18         11          5
## 2     13        7         82             39      30         12         13
## 3      8        6         29             14       8          3          3
## 4      4        1         37             16      12          2          6
## 5      4        1         19              6       5          4          1
## 6      1        1         24             15      11          5          6
##   unforced unforced_fh unforced_bh
## 1       33          13          18
## 2       53          25          17
## 3       11           5           6
## 4       23          10           9
## 5       11           3           6
## 6        9           2           3
```

2. Create datasets only with data that contains Simona Halep as Player 1 or Player 2 (for servebasics and servedirection also filter data by "Total" - keep only data for the entire match not splitted by sets or by shot)

```
overview_total <- filter(overview2, Player1 == "Simona_Halep" & player == 1 | Player2 == "Simona_Halep"
forced_errors_total <- filter(forced_errors1, Player1 == "Simona_Halep" | Player2 == "Simona_Halep")
return_outcomes_total <- filter(return_outcomes1, Player1 == "Simona_Halep" & player == 1 | Player2 ==
servebasics_total <- filter(servebasics1,  Player1 == "Simona_Halep" & row == "1 Total" | Player2 == "S
servedirection_total <- filter(servedirection1,  Player1 == "Simona_Halep" & row == "1 Total" | Player2
shotdirection_total <- filter(shotdirection1, Player1 == "Simona_Halep" & player == 1| Player2 == "Simo
```

3. Keep only the data that contains "Total" in the row and set columns from the other data sets

```
overview_total <- filter(overview_total, set == "Total")
forced_errors_total <- filter(forced_errors_total, row == "Total")
return_outcomes_total <- filter(return_outcomes_total, row == "Total")
shotdirection_total <- filter(shotdirection_total, row == "Total")
```

4. Transform variable Date in Year/Month/Day, keep only the useful variables and subset data for Year>2012

```
overview_total <- arrange(overview_total, Date)
overview_total$Date <- as.Date(overview_total$Date, "%Y%m%d")
overview_total <- separate(overview_total, Date, c("Year", "Month", "Day"), sep = "-")
overview_total<- subset(overview_total, Year > 2012)
overview_total <- select(overview_total, -Gender, -set, -player, -Player1, -Player2)
```

```
head(overview_total)
```

```
##   Year Month Day      Tournament Round serve_pts aces dfs first_in
## 3 2013    05  16            Rome   R16        48    5   2       37
## 4 2013    05  18            Rome    SF        36    1   1       24
```

```
## 5 2013    06  14       Nuremburg    SF        98    5   3       60
## 6 2013    06  21 s_Hertogenbosch    SF        44    1   0       29
## 7 2013    06  22 s_Hertogenbosch     F        55    3   4       35
## 8 2013    07  14        Budapest     F       102    2   5       70
##   first_won second_in second_won bk_pts bp_saved return_pts return_pts_won
## 3        26        11          8      2        1         60             29
## 4         7        12          6      6        1         53             19
## 5        36        38         18     13        9         58             23
## 6        17        15          9      5        2         57             34
## 7        25        20         10      5        3         55             31
## 8        41        32         15     12        8         95             50
##   winners winners_fh winners_bh unforced unforced_fh unforced_bh
## 3      23          6         12       14           5           7
## 4       4          1          2       17           6          10
## 5      18          8          4       21           9           9
## 6      21          8          9       16          11           5
## 7      20          8          7       24           8          12
## 8      35         24          9       64          28          31
```

```
forced_errors_total <- arrange(forced_errors_total, Date)
forced_errors_total$Date <- as.Date(forced_errors_total$Date, "%Y%m%d")
forced_errors_total<- separate(forced_errors_total, Date, c("Year", "Month", "Day"), sep = "-")
forced_errors_total <- subset(forced_errors_total, Year > 2012)
forced_errors_total <- select(forced_errors_total, -Gender, -row)
forced_errors_total1 <- subset(forced_errors_total, Player1 == "Simona_Halep", select = c(Year:pts,pl1_
forced_errors_total2 <- subset(forced_errors_total, Player2 == "Simona_Halep", select = c(Year:pts,pl2_
names(forced_errors_total1)[9:12] <- c("won", "winners", "forced", "unforced")
names(forced_errors_total2)[9:12] <- c("won", "winners", "forced", "unforced")
x <- bind_rows(forced_errors_total1,forced_errors_total2)
forced_errors_total3 <- arrange(x, Year, Month, Day)
forced_errors_total <- forced_errors_total3
forced_errors_total <- select(forced_errors_total, -Player1, -Player2)
forced_errors_total <- data.frame(forced_errors_total)
```

```
head(forced_errors_total)
```

```
##   Year Month Day      Tournament Round pts won winners forced unforced
## 1 2013    05  16            Rome   R16 108  63      23     12       12
## 2 2013    05  18            Rome    SF  89  32       4     11       16
## 3 2013    06  14       Nuremburg    SF 156  77      17     24       18
## 4 2013    06  21 s_Hertogenbosch    SF 101  60      18     16       16
## 5 2013    06  22 s_Hertogenbosch     F 110  66      18     17       20
## 6 2013    07  14        Budapest     F 197 106      35     28       59
```

```
return_outcomes_total <- arrange(return_outcomes_total, Date)
return_outcomes_total$Date <- as.Date(return_outcomes_total$Date, "%Y%m%d")
return_outcomes_total <- separate(return_outcomes_total, Date, c("Year", "Month", "Day"), sep = "-")
return_outcomes_total <- subset(return_outcomes_total, Year > 2012)
return_outcomes_total <- select(return_outcomes_total, -Gender, -row, -player, -Player1, -Player2)
```

```
head(return_outcomes_total)
```

```
##   Year Month Day      Tournament Round pts pts_won returnable
```

```
## 3 2013    05  16            Rome   R16  60      29         50
## 4 2013    05  18            Rome    SF  53      19         40
## 5 2013    06  14       Nuremburg    SF  58      23         41
## 6 2013    06  21 s_Hertogenbosch    SF  57      34         48
## 7 2013    06  22 s_Hertogenbosch     F  55      31         47
## 8 2013    07  14        Budapest     F  95      50         88
##   returnable_won in_play in_play_won winners total_shots
## 3             28      47          28       2         280
## 4             18      37          18       0         201
## 5             20      41          20       1         216
## 6             32      48          32       1         277
## 7             28      45          28       0         249
## 8             48      85          48       0         717
```

```
servebasics_total <- arrange(servebasics_total, Date)
servebasics_total$Date <- as.Date(servebasics_total$Date, "%Y%m%d")
servebasics_total<- separate(servebasics_total, Date, c("Year", "Month", "Day"), sep = "-")
servebasics_total <- subset(servebasics_total, Year > 2012)
servebasics_total <- select(servebasics_total, -Gender, -row, -Player1, -Player2)
```

```
head(servebasics_total)
```

```
##   Year Month Day      Tournament Round pts pts_won aces unret forced_err
## 3 2013    05  16            Rome   R16  48      34    5     0          4
## 4 2013    05  18            Rome    SF  36      13    1     0          3
## 5 2013    06  14       Nuremburg    SF  98      54    5     1         11
## 6 2013    06  21 s_Hertogenbosch    SF  44      26    1     3          5
## 7 2013    06  22 s_Hertogenbosch     F  55      35    3     2          5
## 8 2013    07  14        Budapest     F 102      56    2     0          8
##   pts_won_lte_3_shots wide body  t
## 3                  16   10   14 24
## 4                   6    6   23  7
## 5                  27   20   50 28
## 6                  10   10   24 10
## 7                  13    8   32 14
## 8                  16   13   66 23
```

```
servedirection_total <- arrange(servedirection_total, Date)
servedirection_total$Date <- as.Date(servedirection_total$Date, "%Y%m%d")
servedirection_total<- separate(servedirection_total, Date, c("Year", "Month", "Day"), sep = "-")
servedirection_total <- subset(servedirection_total, Year > 2012)
servedirection_total <- select(servedirection_total, -Gender, -row, -Player1, -Player2)
```

```
head(servedirection_total)
```

```
##   Year Month Day      Tournament Round deuce_wide deuce_middle deuce_t
## 3 2013    05  16            Rome   R16          5           10      10
## 4 2013    05  18            Rome    SF          4           11       4
## 5 2013    06  14       Nuremburg    SF         11           24      14
## 6 2013    06  21 s_Hertogenbosch    SF          5           10       9
## 7 2013    06  22 s_Hertogenbosch     F          7           15       7
## 8 2013    07  14        Budapest     F          7           32      12
```

```
##   ad_wide ad_middle ad_t err_net err_wide err_deep err_wide_deep err_foot
## 3       5         4   14       2        3        6             2        0
## 4       2        12    3       6        3        4             0        0
## 5       9        26   14      13        5       22             1        0
## 6       5        14    1       5        1        9             0        0
## 7       1        17    7      13        3        8             0        0
## 8       6        34   11      15        5       17             0        0
##   err_unknown
## 3           0
## 4           0
## 5           0
## 6           0
## 7           0
## 8           0
```

```
shotdirection_total <- arrange(shotdirection_total, Date)
shotdirection_total$Date <- as.Date(shotdirection_total$Date, "%Y%m%d")
shotdirection_total<- separate(shotdirection_total, Date, c("Year", "Month", "Day"), sep = "-")
shotdirection_total <- subset(shotdirection_total, Year > 2012)
shotdirection_total <- select(shotdirection_total, -Gender, -row, -player, -Player1, -Player2)
```

```
head(shotdirection_total)
```

```
##   Year Month Day      Tournament Round crosscourt down_middle
## 3 2013    05  16            Rome   R16         65          33
## 4 2013    05  18            Rome    SF         43          26
## 5 2013    06  14       Nuremburg    SF         75          70
## 6 2013    06  21 s_Hertogenbosch    SF         82          29
## 7 2013    06  22 s_Hertogenbosch     F         83          33
## 8 2013    07  14        Budapest     F        214         162
##   down_the_line inside_out inside_in
## 3            31         21         0
## 4            12         11         0
## 5            22         26         2
## 6            21         28         0
## 7            17         39         0
## 8            59         84        10
```

5. Combine dataset overview_total, win_lose, games and surface

```
overview_total_1 <- cbind(overview_total, win_lose, games)
overview_total_1$Tournament <- as.factor(overview_total_1$Tournament)
overview_total_1 <- left_join(overview_total_1, surface, by = "Tournament")
overview_total_1$Year <- as.factor(overview_total_1$Year)
overview_total_1$Round <- as.factor(overview_total_1$Round)
overview_total_1$Month <- as.factor(overview_total_1$Month)
overview_total_1$Win_Lose <- as.factor(overview_total_1$Win_Lose)
overview_total_1$No_of_sets <- as.factor(overview_total_1$No_of_sets)
overview_total_1$No_of_games <- as.factor(overview_total_1$No_of_games)
overview_total_1 <- overview_total_1[,c(1,2,3,4,5,23,24,25,26,6:22)]
```

```
str(overview_total_1)
```

```
## 'data.frame':    152 obs. of  26 variables:
##  $ Year          : Factor w/ 4 levels "2013","2014",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Month         : Factor w/ 11 levels "01","02","03",..: 5 5 6 6 6 7 8 8 8 8 ...
##  $ Day           : chr  "16" "18" "14" "21" ...
##  $ Tournament    : Factor w/ 30 levels "Australian_Open",..: 19 19 17 20 20 5 16 16 28 28 ...
##  $ Round         : Factor w/ 8 levels "F","QF","R128",..: 4 8 8 8 1 1 1 8 3 5 ...
##  $ Win_Lose      : Factor w/ 2 levels "L","W": 2 1 1 2 2 2 2 2 2 2 ...
##  $ No_of_sets    : Factor w/ 2 levels "2","3": 1 1 2 1 1 2 1 1 2 1 ...
##  $ No_of_games   : Factor w/ 24 levels "11","13","14",..: 7 4 12 6 7 18 5 9 17 2 ...
##  $ Surface       : Factor w/ 3 levels "clay","grass",..: 1 1 1 2 2 1 3 3 3 3 ...
##  $ serve_pts     : int  48 36 98 44 55 102 43 61 82 36 ...
##  $ aces          : int  5 1 5 1 3 2 1 0 5 3 ...
##  $ dfs           : int  2 1 3 0 4 5 1 3 5 1 ...
##  $ first_in      : int  37 24 60 29 35 70 32 38 62 29 ...
##  $ first_won     : int  26 7 36 17 25 41 25 26 41 23 ...
##  $ second_in     : int  11 12 38 15 20 32 11 23 20 7 ...
##  $ second_won    : int  8 6 18 9 10 15 9 9 6 3 ...
##  $ bk_pts        : int  2 6 13 5 5 12 4 5 8 1 ...
##  $ bp_saved      : int  1 1 9 2 3 8 4 2 3 1 ...
##  $ return_pts    : int  60 53 58 57 55 95 47 57 92 41 ...
##  $ return_pts_won: int  29 19 23 34 31 50 24 28 47 27 ...
##  $ winners       : int  23 4 18 21 20 35 19 26 15 13 ...
##  $ winners_fh    : int  6 1 8 8 8 24 8 15 9 5 ...
##  $ winners_bh    : int  12 2 4 9 7 9 5 10 1 4 ...
##  $ unforced      : int  14 17 21 16 24 64 6 30 42 14 ...
##  $ unforced_fh   : int  5 6 9 11 8 28 2 17 23 6 ...
##  $ unforced_bh   : int  7 10 9 5 12 31 3 10 14 7 ...
```

Data analysis

The analysis covers 152 matches over a span of four years from May 2013 to June 2016. From the 152 matches, 114 were wins (75%) and 38 were losses(25%).

```
table(overview_total_1$Win_Lose)
```

```
##
##   L   W
##  38 114
```

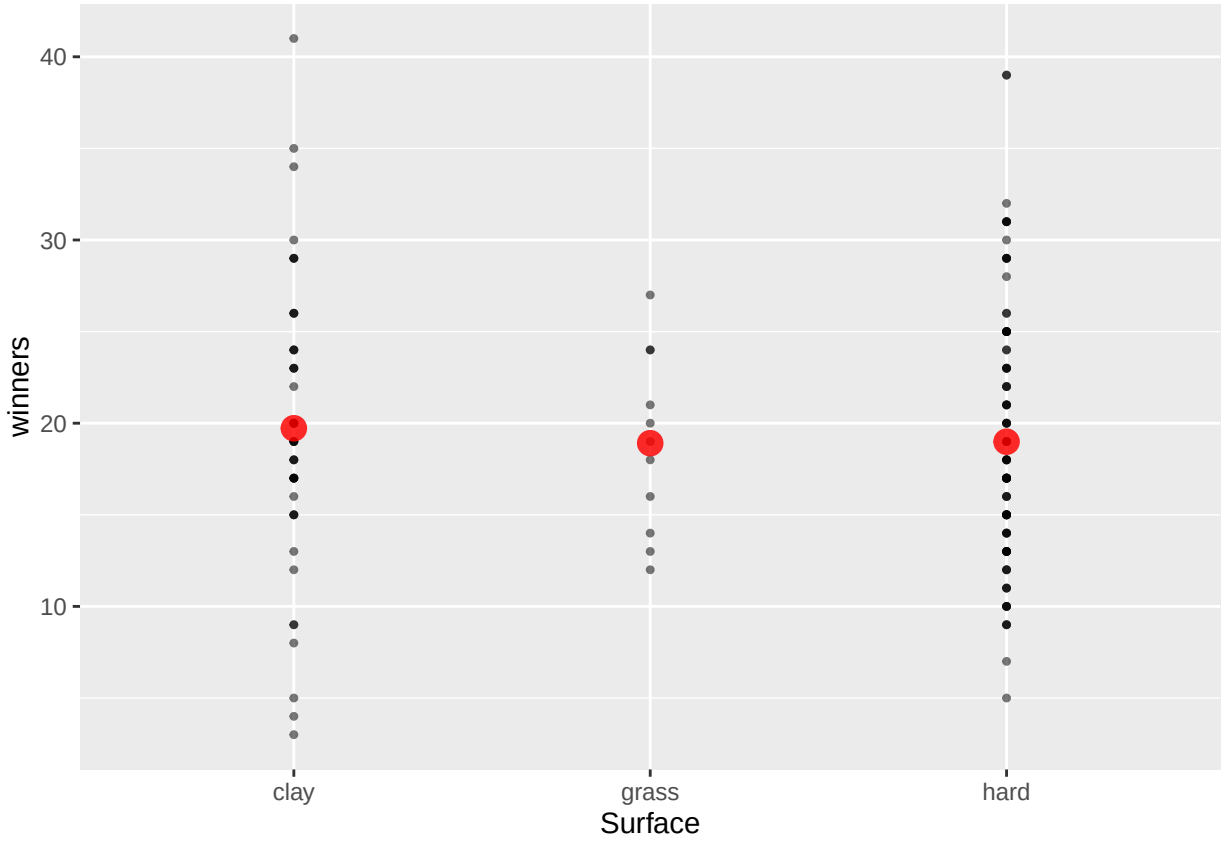
```
library(ggplot2)
```

```
overview_total_1$serve_pts <- as.numeric(overview_total_1$serve_pts)
overview_total_1$aces <- as.numeric(overview_total_1$aces)
overview_total_1$dfs <- as.numeric(overview_total_1$dfs)
overview_total_1$first_in <- as.numeric(overview_total_1$first_in)
overview_total_1$first_won <- as.numeric(overview_total_1$first_won)
overview_total_1$second_in <- as.numeric(overview_total_1$second_in)
overview_total_1$second_won <- as.numeric(overview_total_1$second_won)
overview_total_1$bk_pts <- as.numeric(overview_total_1$bk_pts)
```

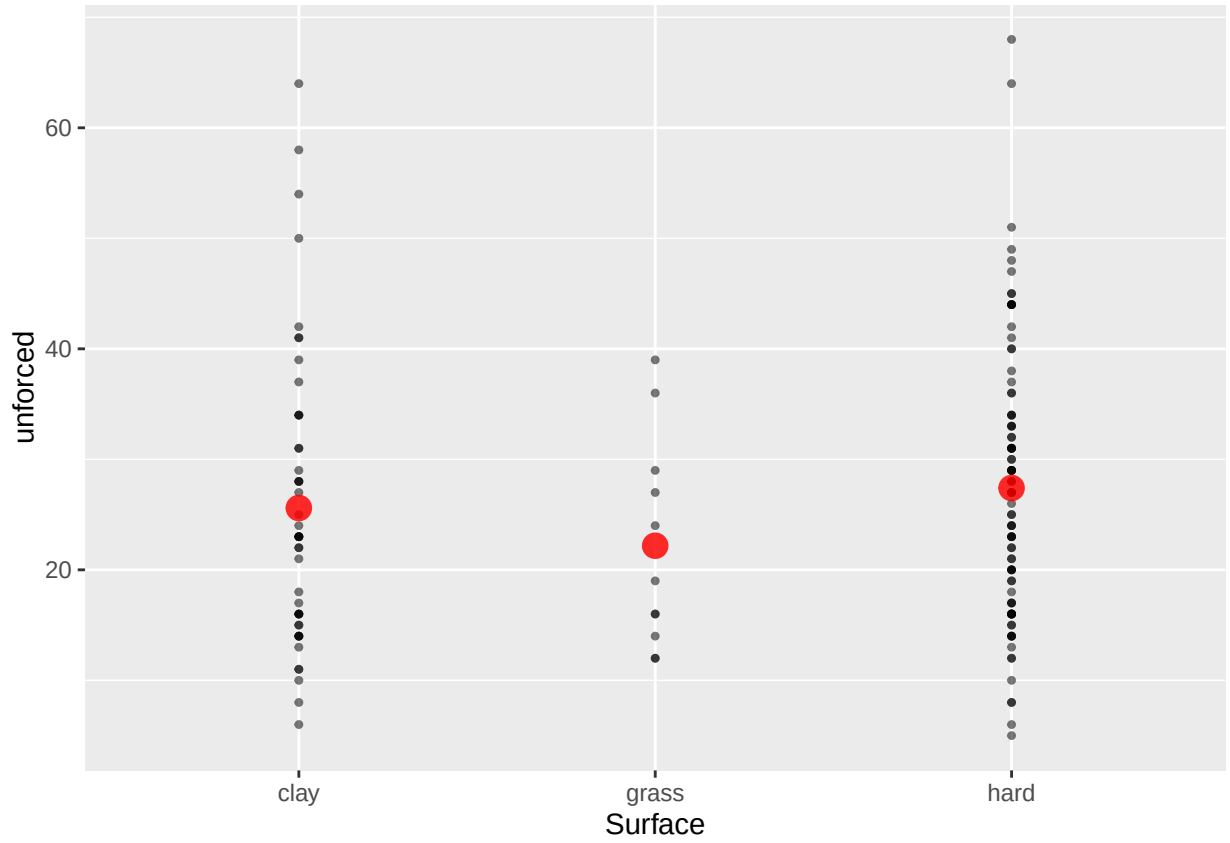
```
overview_total_1$bp_saved <- as.numeric(overview_total_1$bp_saved)
overview_total_1$return_pts <- as.numeric(overview_total_1$return_pts)
overview_total_1$return_pts_won <- as.numeric(overview_total_1$return_pts_won)
overview_total_1$winners <- as.numeric(overview_total_1$winners)
overview_total_1$winners_fh <- as.numeric(overview_total_1$winners_fh)
overview_total_1$winners_bh <- as.numeric(overview_total_1$winners_bh)
overview_total_1$unforced <- as.numeric(overview_total_1$unforced)
overview_total_1$unforced_bh <- as.numeric(overview_total_1$unforced_bh)
overview_total_1$unforced_fh <- as.numeric(overview_total_1$unforced_fh)
```

1. The average number of winners (20 winners is the mean on clay, followed very closely by the mean on grass and hard courts) and unforced errors (the mean between 20 and 28, on grass it tends to be smaller).

```
ggplot() +
  geom_point(aes(x = Surface,y = winners),data=overview_total_1,shape = 20,alpha = 0.5294,position = pos
  stat_summary(aes(x = Surface,y = winners),data=overview_total_1,colour = '#ff0000',size =  4.0,alpha =
```

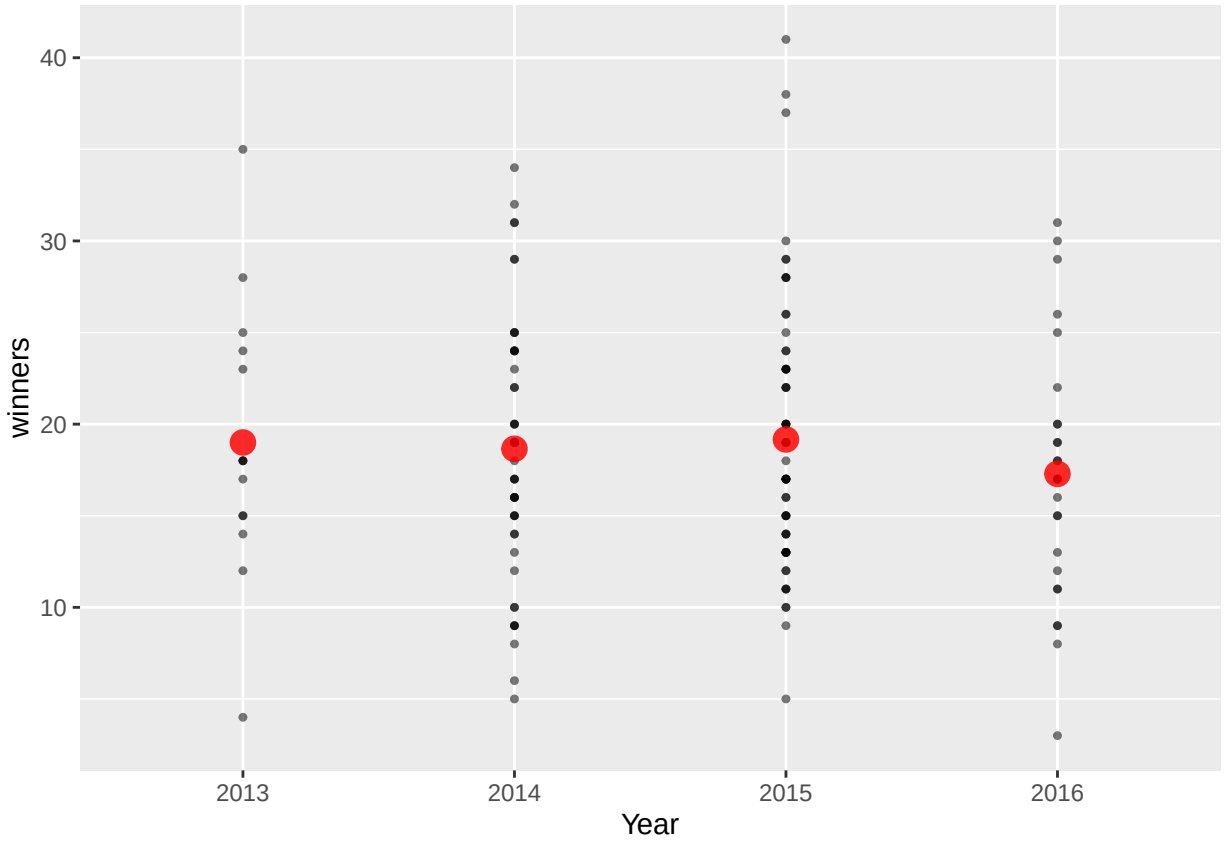



```
ggplot() +
 geom_point(aes(x = Surface,y = unforced),data=overview_total_1,shape = 20,alpha = 0.5294,position = pos
 stat_summary(aes(x = Surface,y = unforced),data=overview_total_1,colour = '#ff0000',size = 4.0,alpha =
```

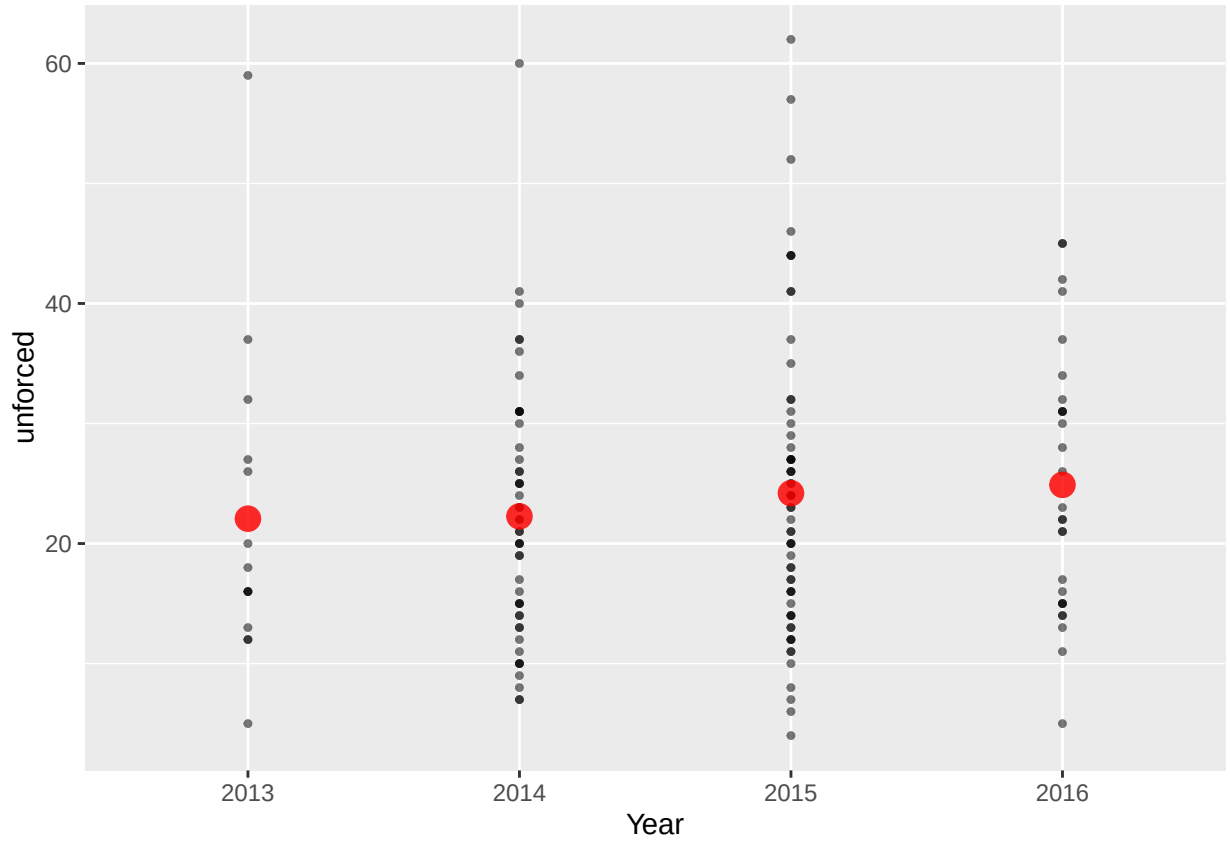


2. The average number of winners, unforced errors and forced errors by Year

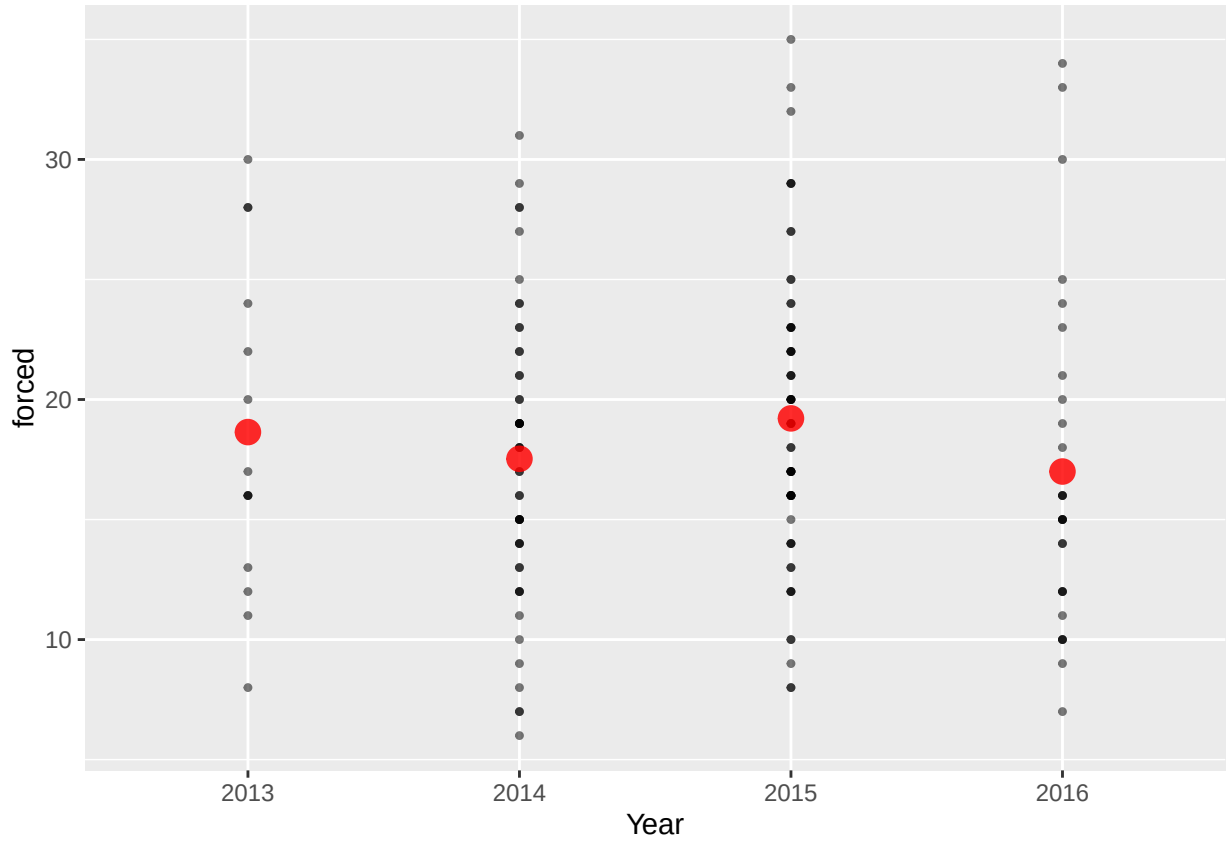
```
ggplot() +
 geom_point(aes(x = Year,y = winners),data=forced_errors_total,shape = 20,alpha = 0.5294,position = posi
 stat_summary(aes(x = Year,y = winners),data=forced_errors_total,colour = '#ff0000',size = 4.0,alpha = 0
```




```
ggplot() +
 geom_point(aes(x = Year,y = unforced),data=forced_errors_total,shape = 20,alpha = 0.5294,position = pos
 stat_summary(aes(x = Year,y = unforced),data=forced_errors_total,colour = '#ff0000',size = 4.0,alpha =
```



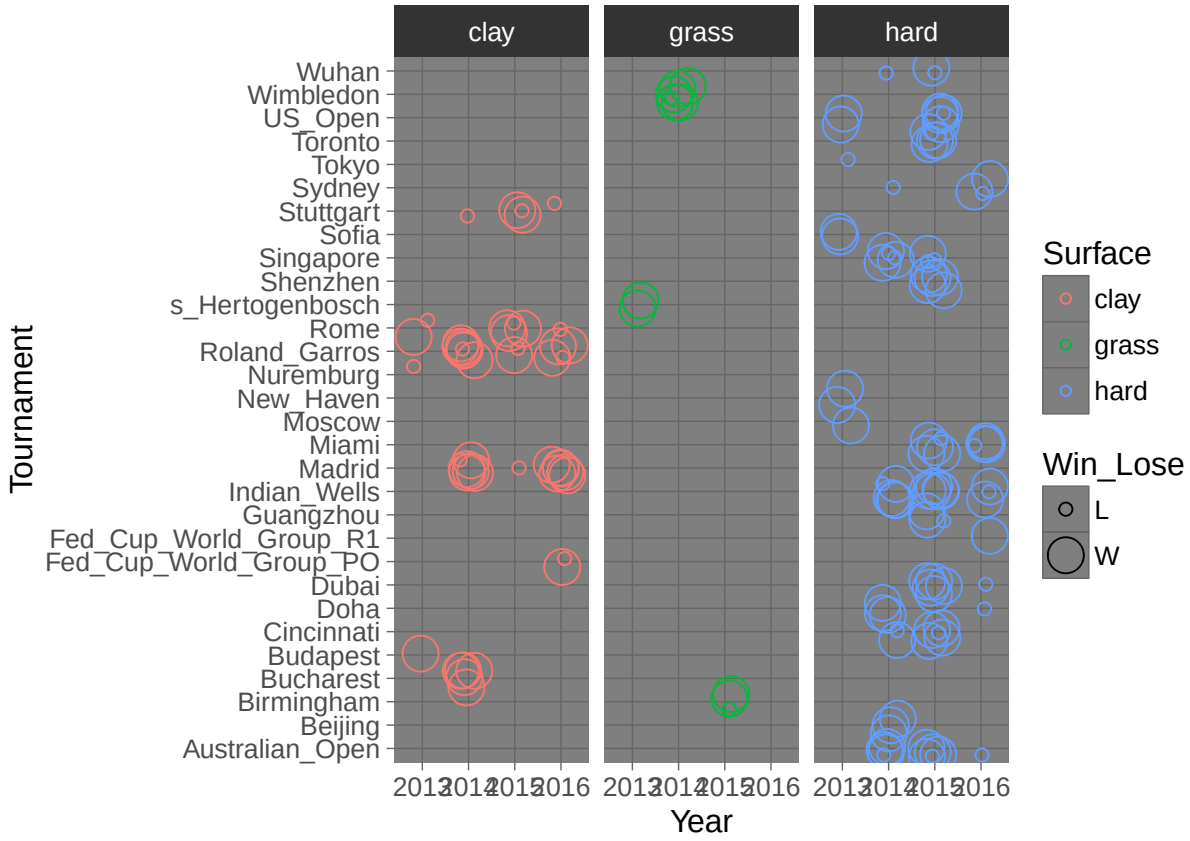

```
ggplot() +
 geom_point(aes(x = Year,y = forced),data=forced_errors_total,shape = 20,alpha = 0.5294,position = posi
 stat_summary(aes(x = Year,y = forced),data=forced_errors_total,colour = '#ff0000',size = 4.0,alpha = 0
```



3. The wins and losses by Tournament, Surface and Year.

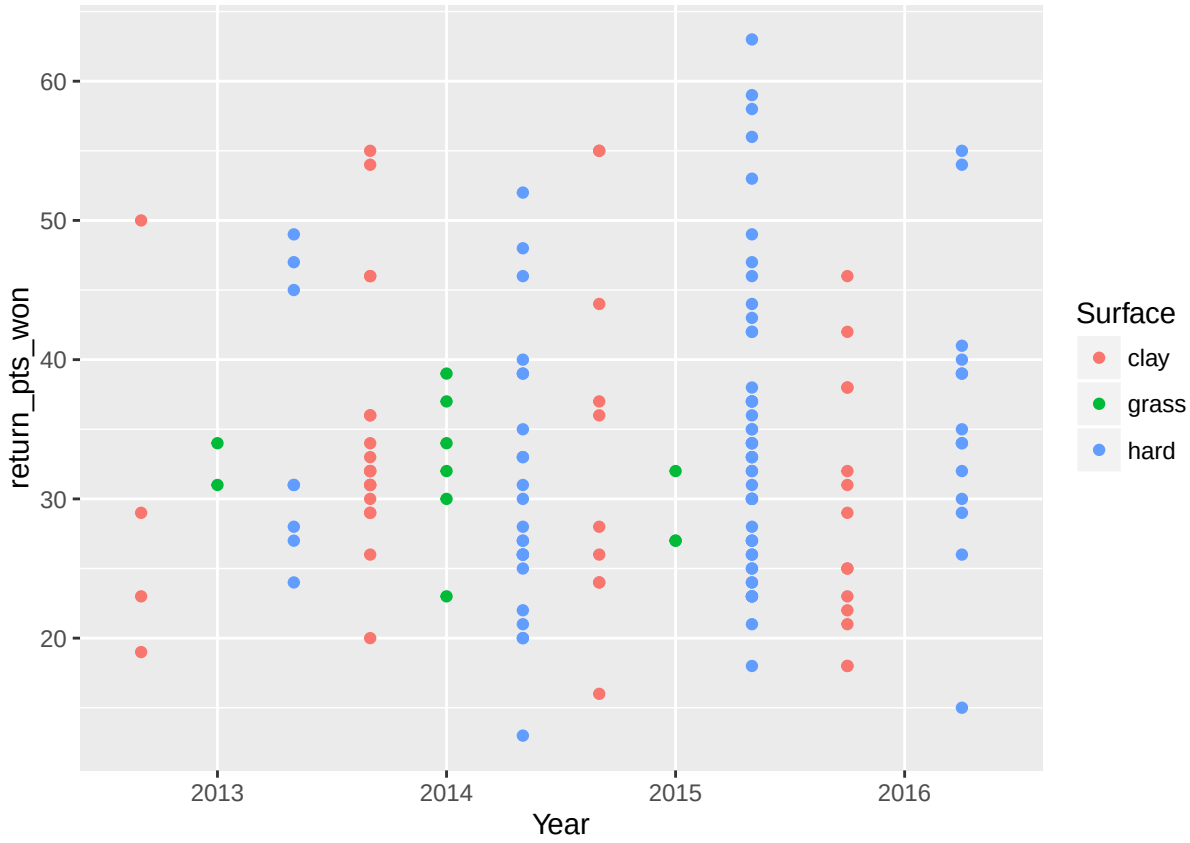
```
ggplot() +
 geom_point(aes(x = Year,y = Tournament,colour = Surface,size = Win_Lose),data=overview_total_1,shape =
 facet_wrap(facets = ~Surface) +
 theme_dark()
```

```
## Warning: Using size for a discrete variable is not advised.
```

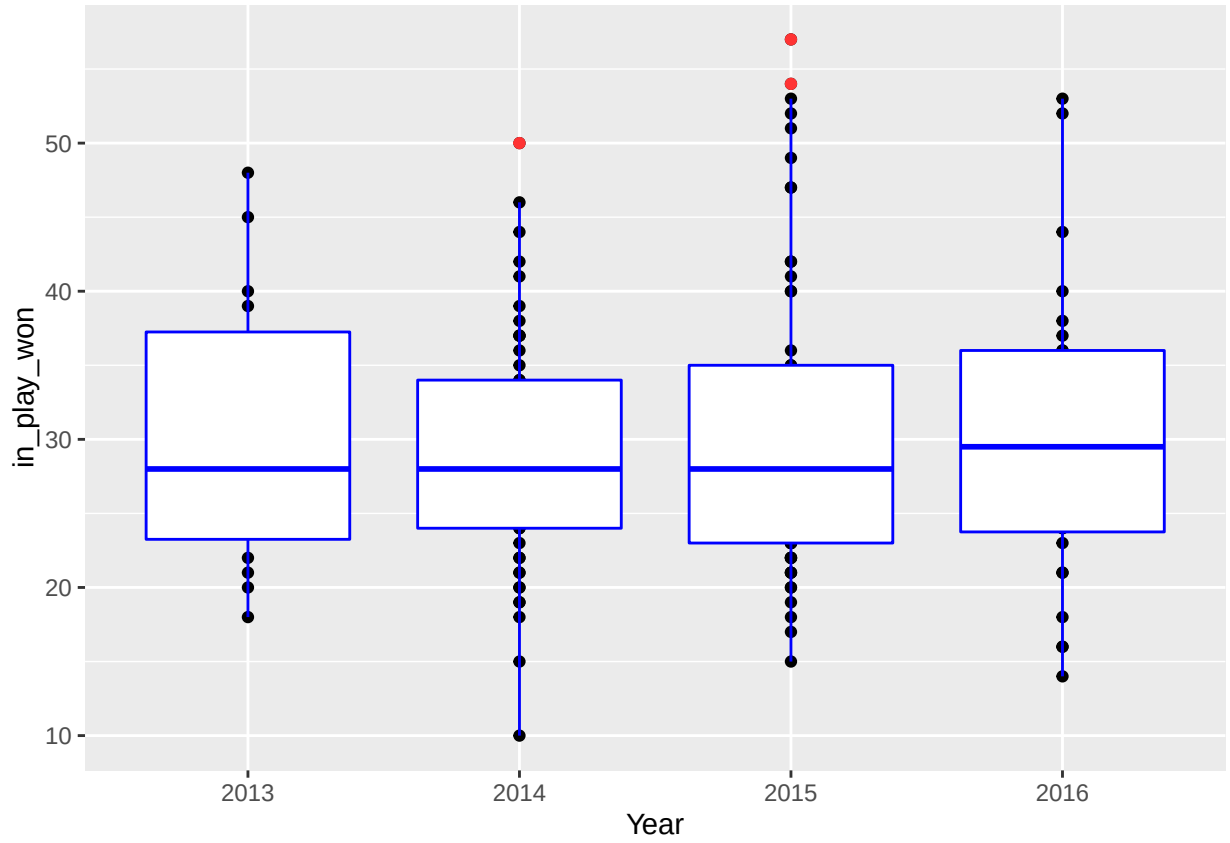


4. Return points won by Year and Surface and return points in play won by Year

```
ggplot() +
 geom_jitter(aes(x = Year,y = return_pts_won,colour = Surface),data=overview_total_1,position = positio
```

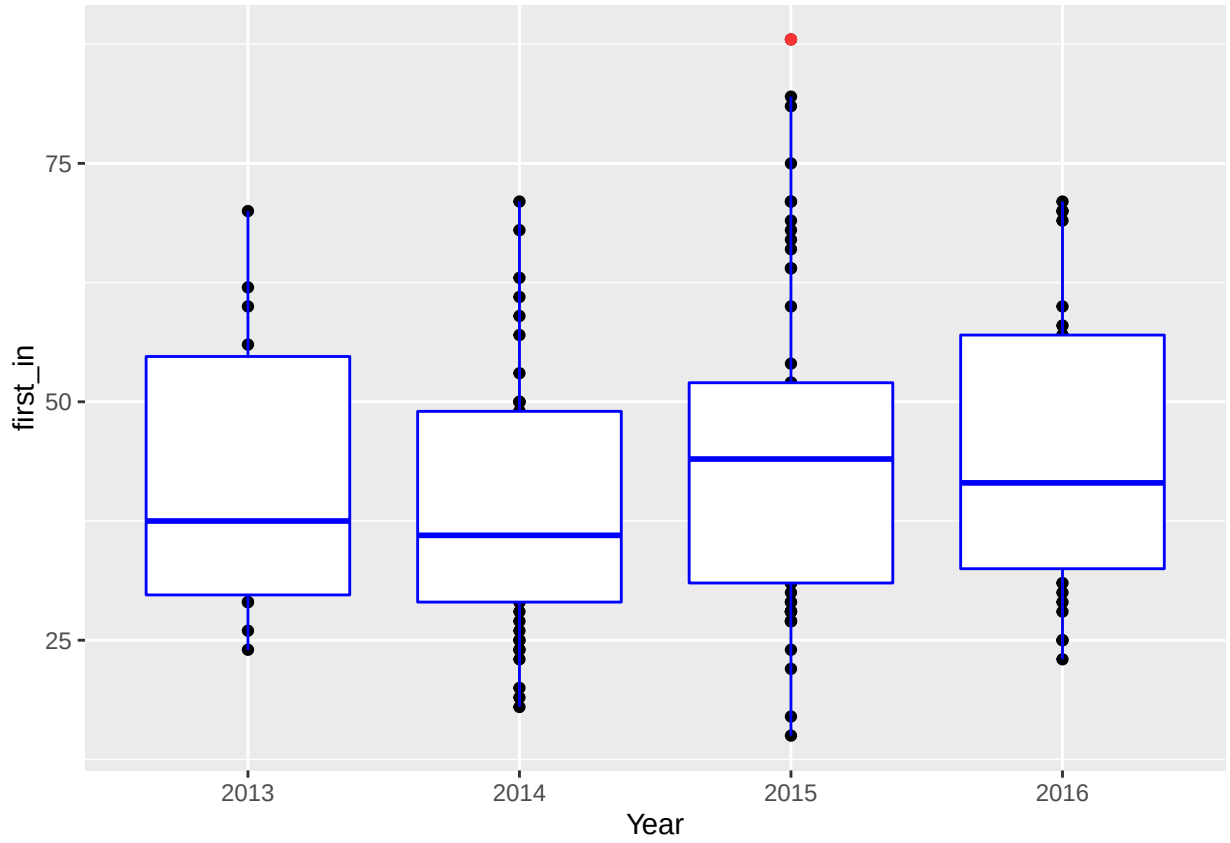



```
ggplot() +
 geom_point(aes(x = Year,y = in_play_won),data=return_outcomes_total) +
 stat_boxplot(aes(x = Year,y = in_play_won),data=return_outcomes_total,colour = '#0000ff',outlier.colou
```

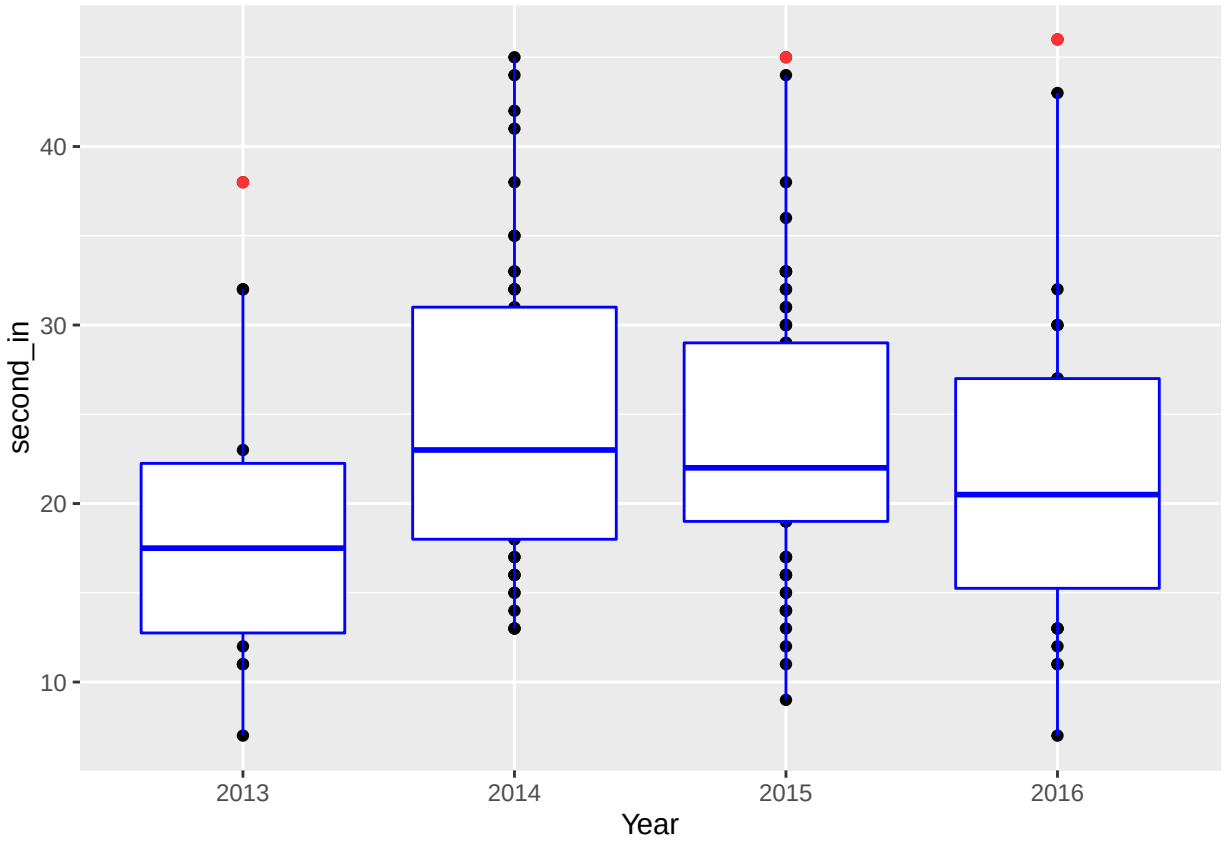


5. First and second serve in by Year

```
ggplot() +
 geom_point(aes(x = Year,y = first_in),data=overview_total_1) +
 stat_boxplot(aes(y = first_in,x = Year),data=overview_total_1,colour = '#0000ff',outlier.colour = '#ff
```

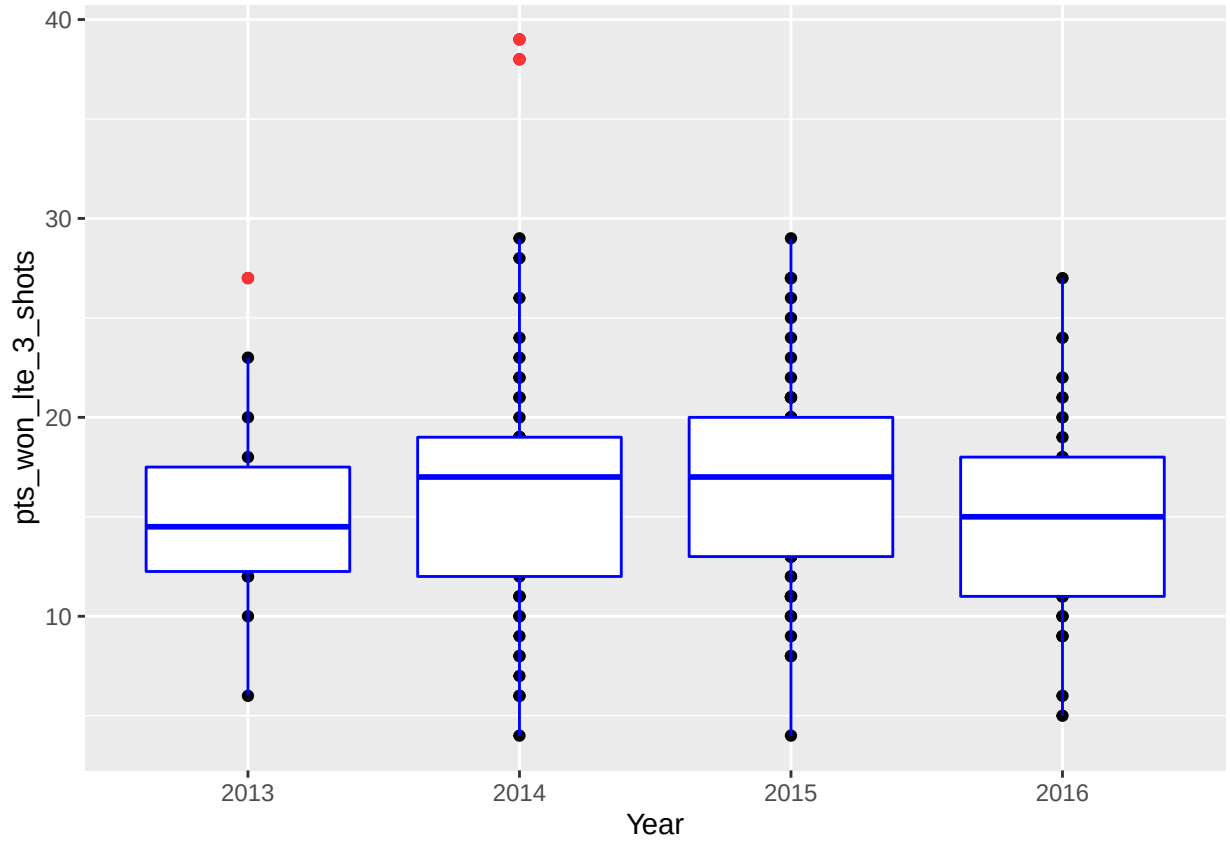



```
ggplot() +
 geom_point(aes(x = Year,y = second_in),data=overview_total_1) +
 stat_boxplot(aes(y = second_in,x = Year),data=overview_total_1,colour = '#0000ff',outlier.colour = '#f
```

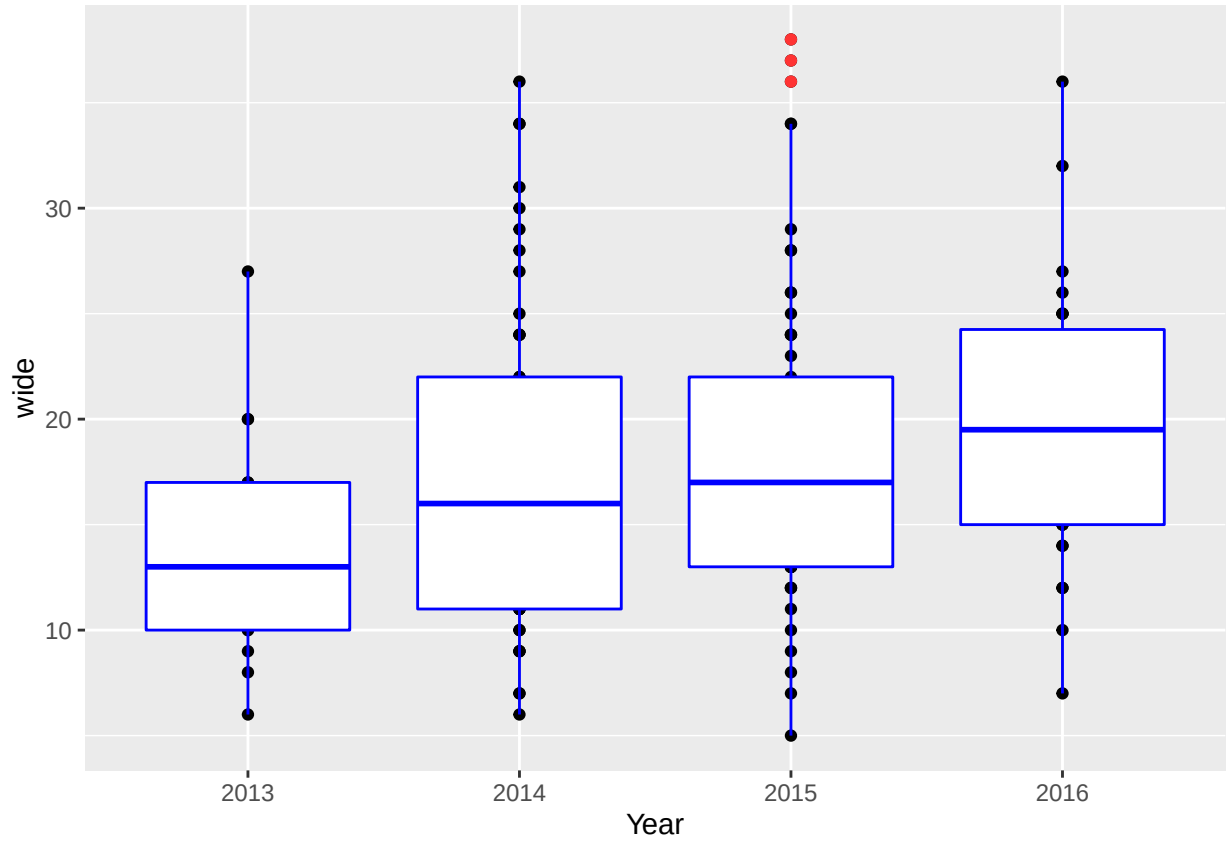


6. Type of serve by Year

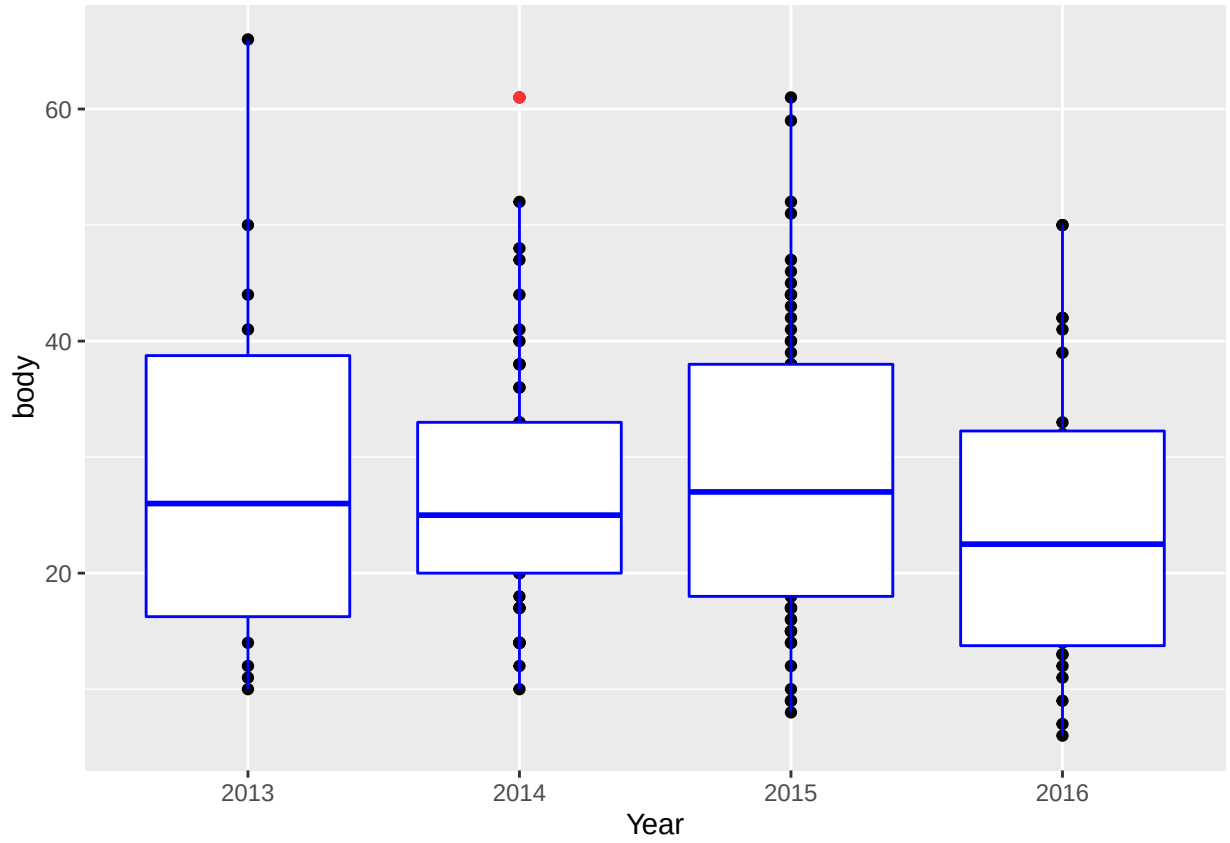
```
ggplot() +
 geom_point(aes(x = Year,y = pts_won_lte_3_shots),data=servebasics_total) +
 stat_boxplot(aes(y = pts_won_lte_3_shots,x = Year),data=servebasics_total,colour = '#0000ff',outlier.c
```



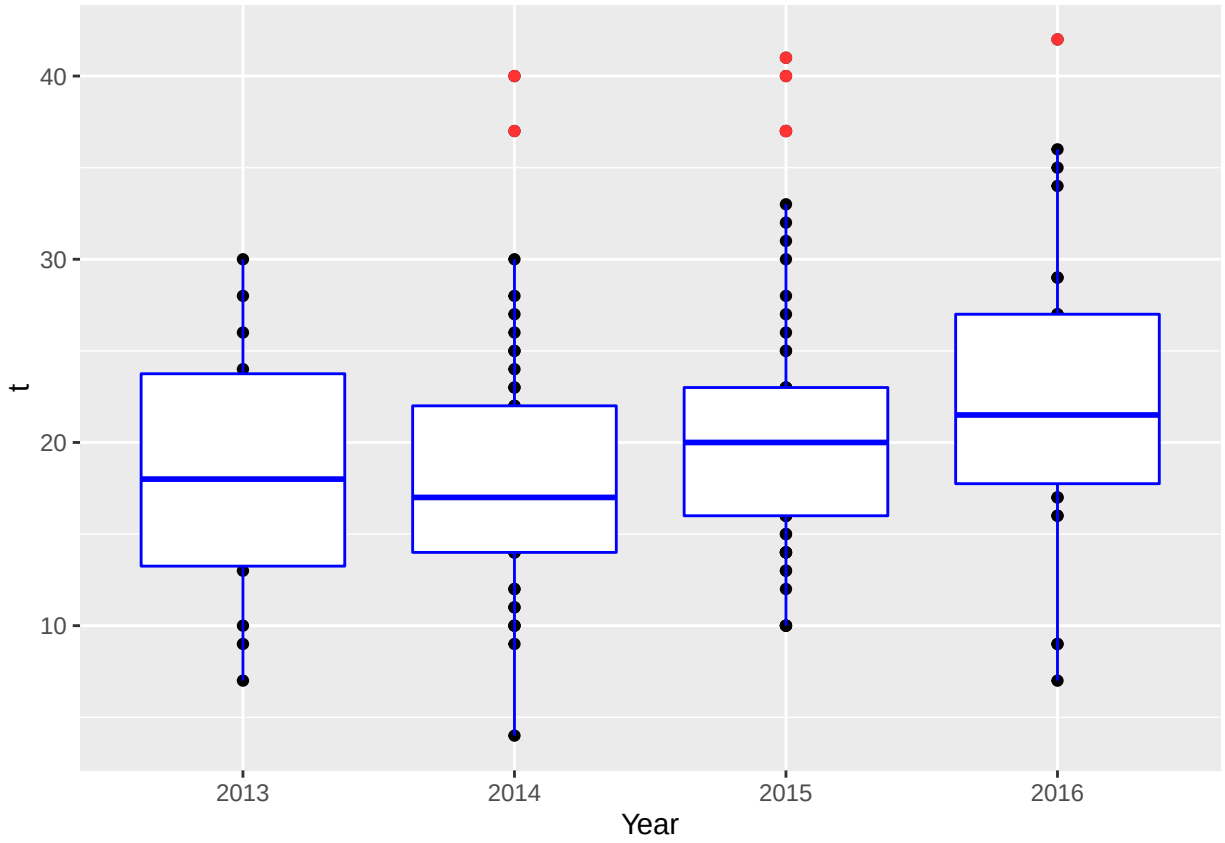

```
ggplot() +
 geom_point(aes(x = Year,y = wide),data=servebasics_total) +
 stat_boxplot(aes(y = wide,x = Year),data=servebasics_total,colour = '#0000ff',outlier.colour = '#ff333
```




```
ggplot() +
 geom_point(aes(x = Year,y = body),data=servebasics_total) +
 stat_boxplot(aes(y = body,x = Year),data=servebasics_total,colour = '#0000ff',outlier.colour = '#ff333
```

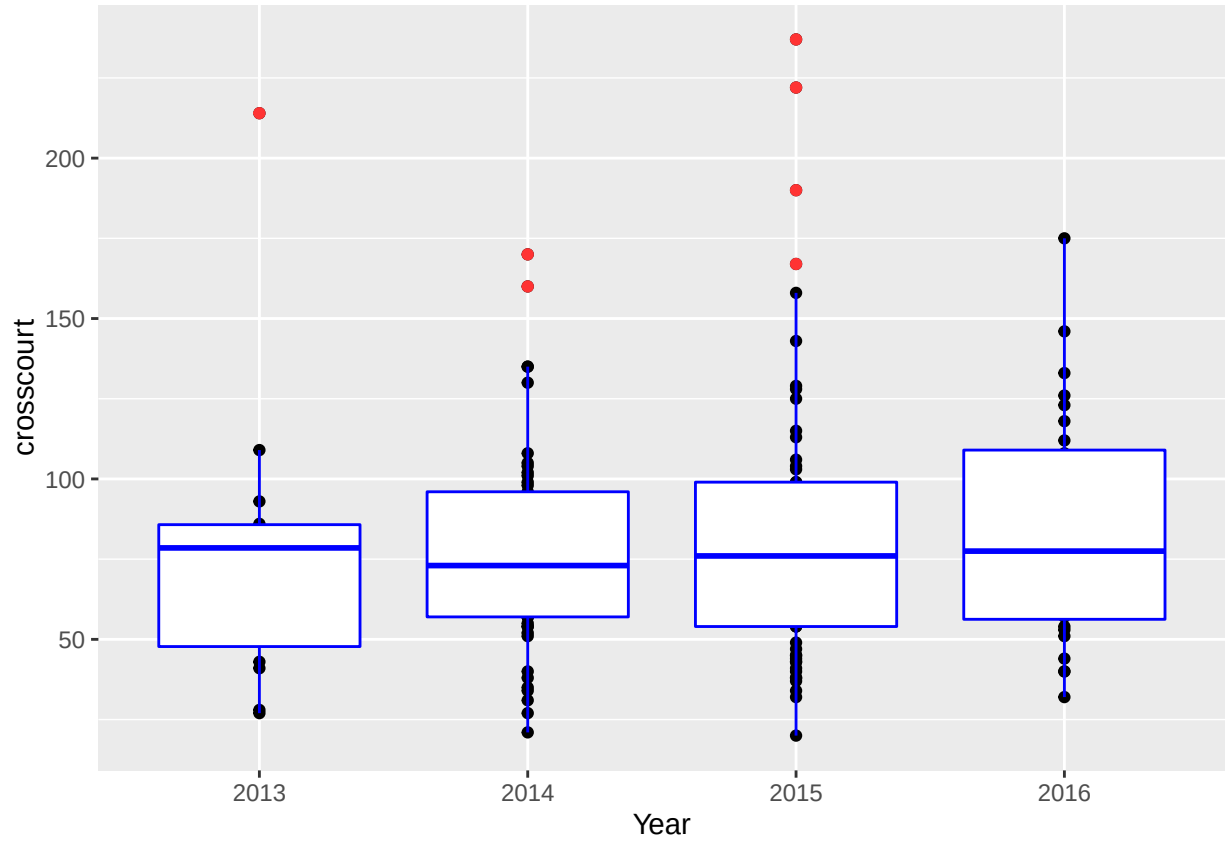



```
ggplot() +
 geom_point(aes(x = Year,y = t),data=servebasics_total) +
 stat_boxplot(aes(y = t,x = Year),data=servebasics_total,colour = '#0000ff',outlier.colour = '#ff3333')
```

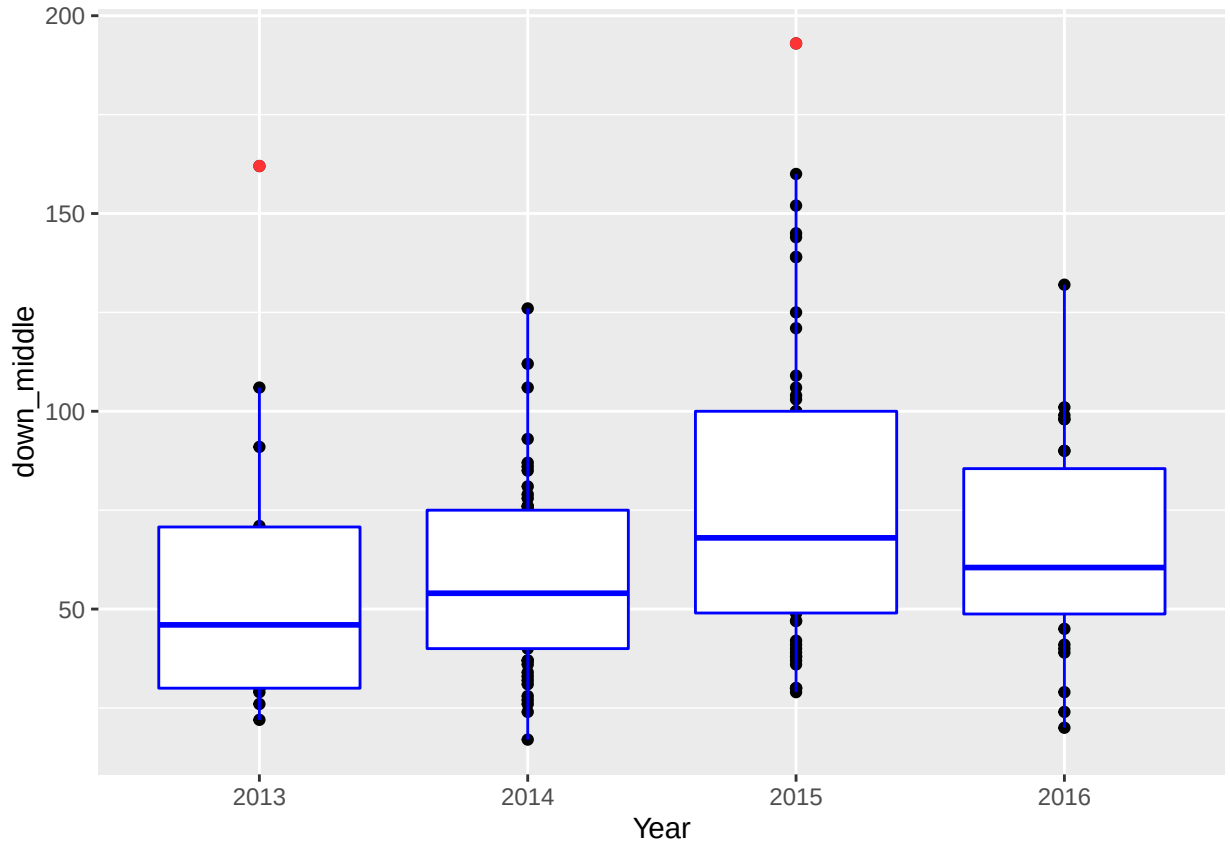


7.Shot direction by Year

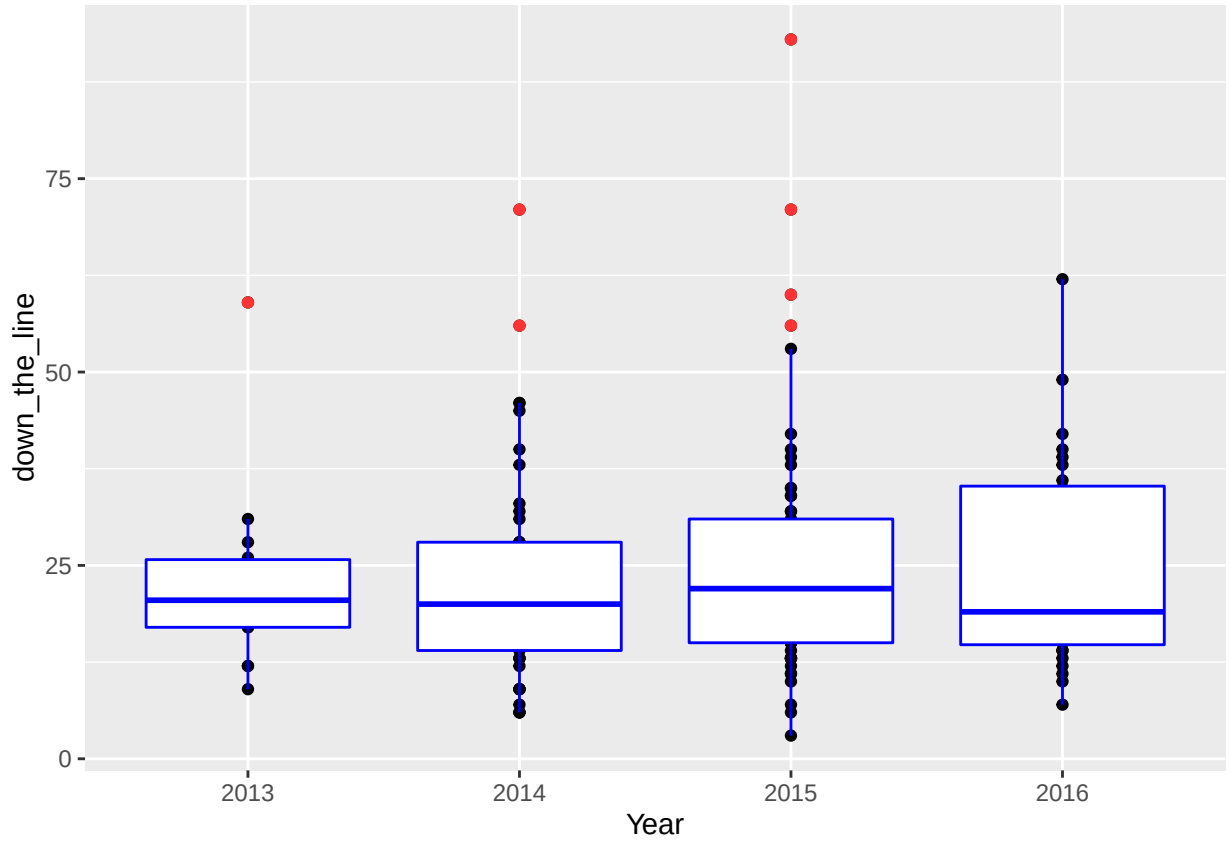
```
ggplot() +
 geom_point(aes(x = Year,y = crosscourt),data=shotdirection_total) +
 stat_boxplot(aes(y = crosscourt,x = Year),data=shotdirection_total,colour = '#0000ff',outlier.colour =
```



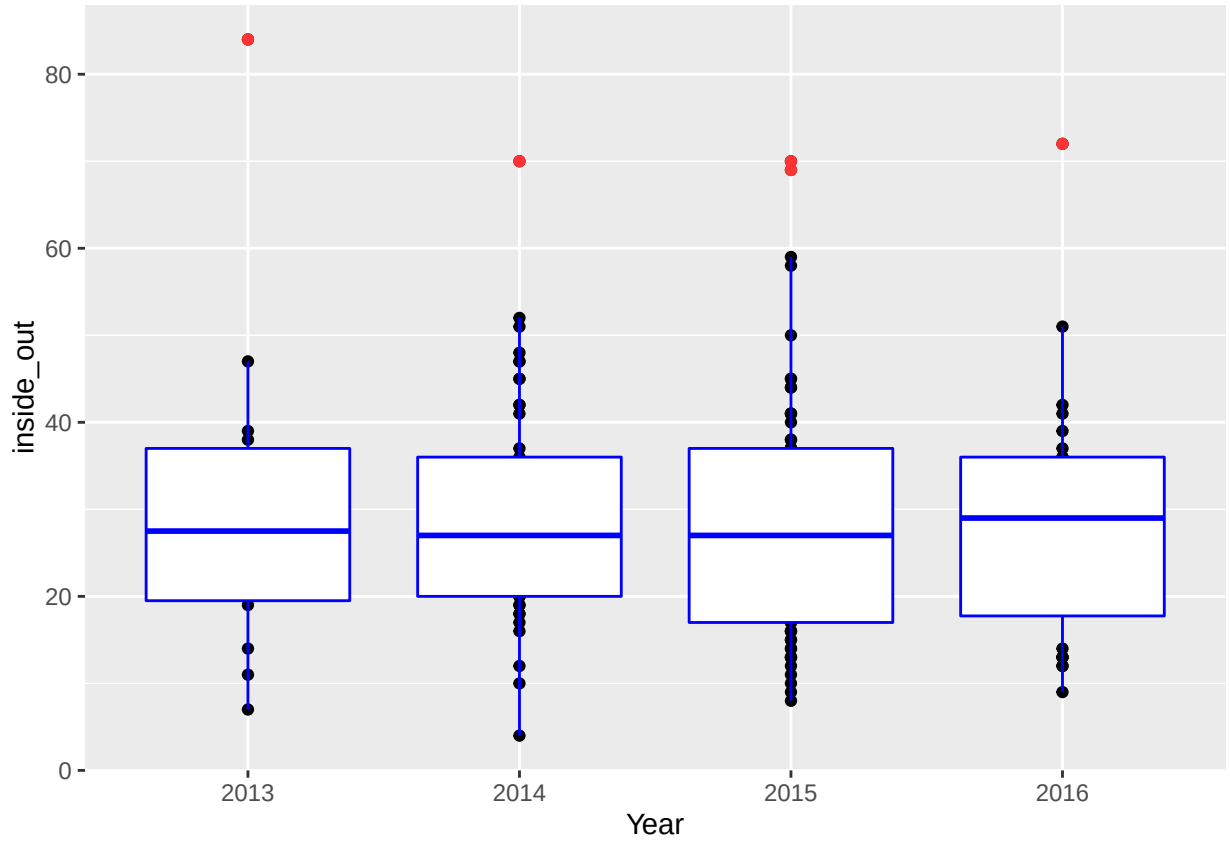

```
ggplot() +
 geom_point(aes(x = Year,y = down_middle),data=shotdirection_total) +
 stat_boxplot(aes(y = down_middle,x = Year),data=shotdirection_total,colour = '#0000ff',outlier.colour =
```



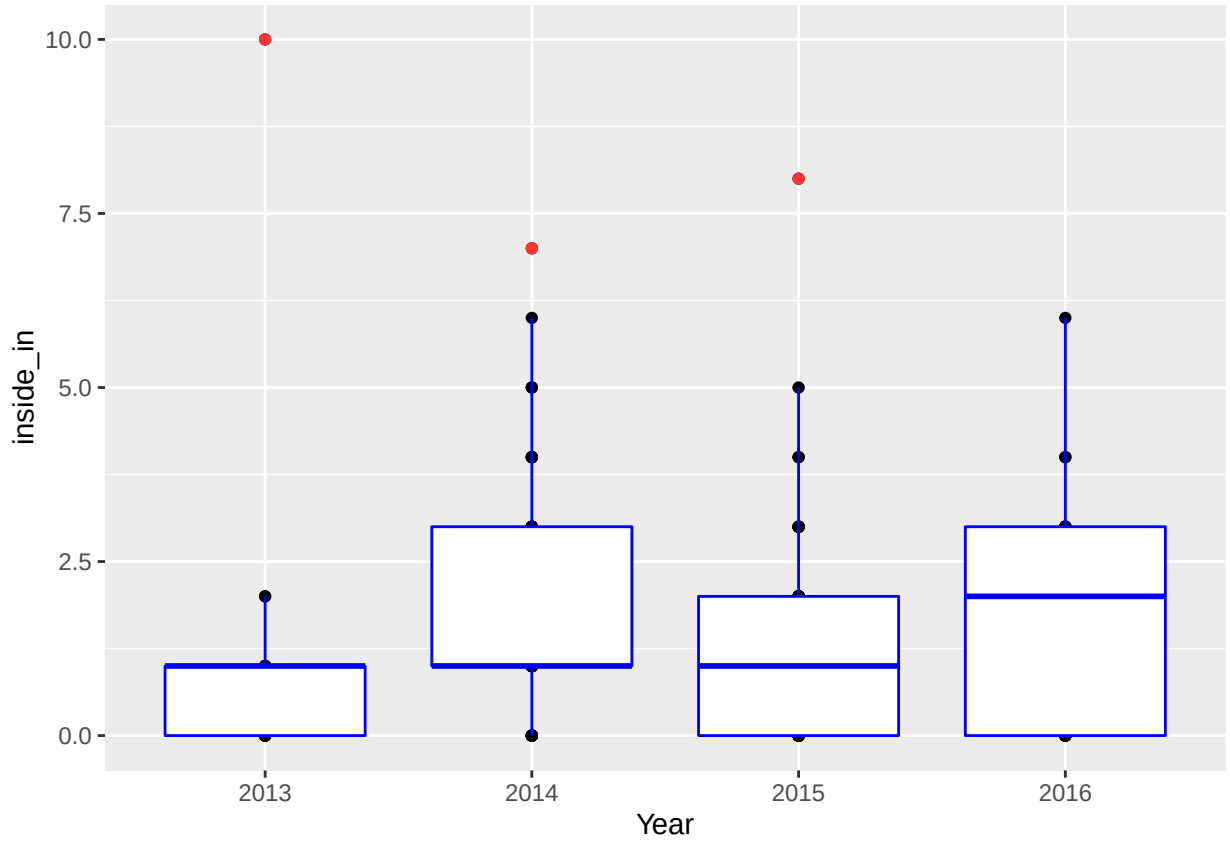

```
ggplot() +
 geom_point(aes(x = Year,y = down_the_line),data=shotdirection_total) +
 stat_boxplot(aes(y = down_the_line,x = Year),data=shotdirection_total,colour = '#0000ff',outlier.colou
```




```
ggplot() +
 geom_point(aes(x = Year,y = inside_out),data=shotdirection_total) +
 stat_boxplot(aes(y = inside_out,x = Year),data=shotdirection_total,colour = '#0000ff',outlier.colour =
```




```
ggplot() +
 geom_point(aes(x = Year,y = inside_in),data=shotdirection_total) +
 stat_boxplot(aes(y = inside_in,x = Year),data=shotdirection_total,colour = '#0000ff',outlier.colour =
```



Data modelling

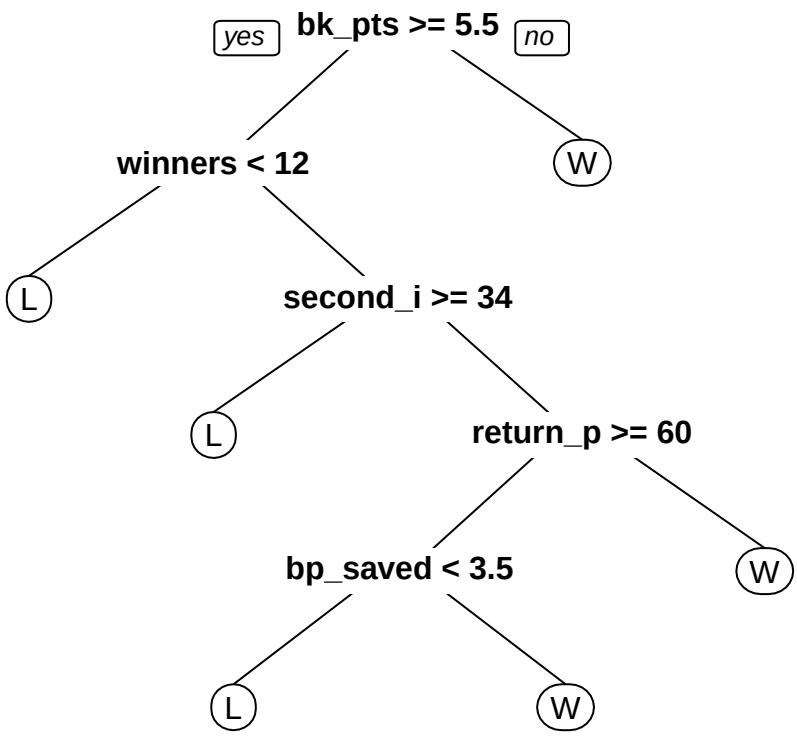
For the prediction a CART model will be used.

```
library(caTools)
```

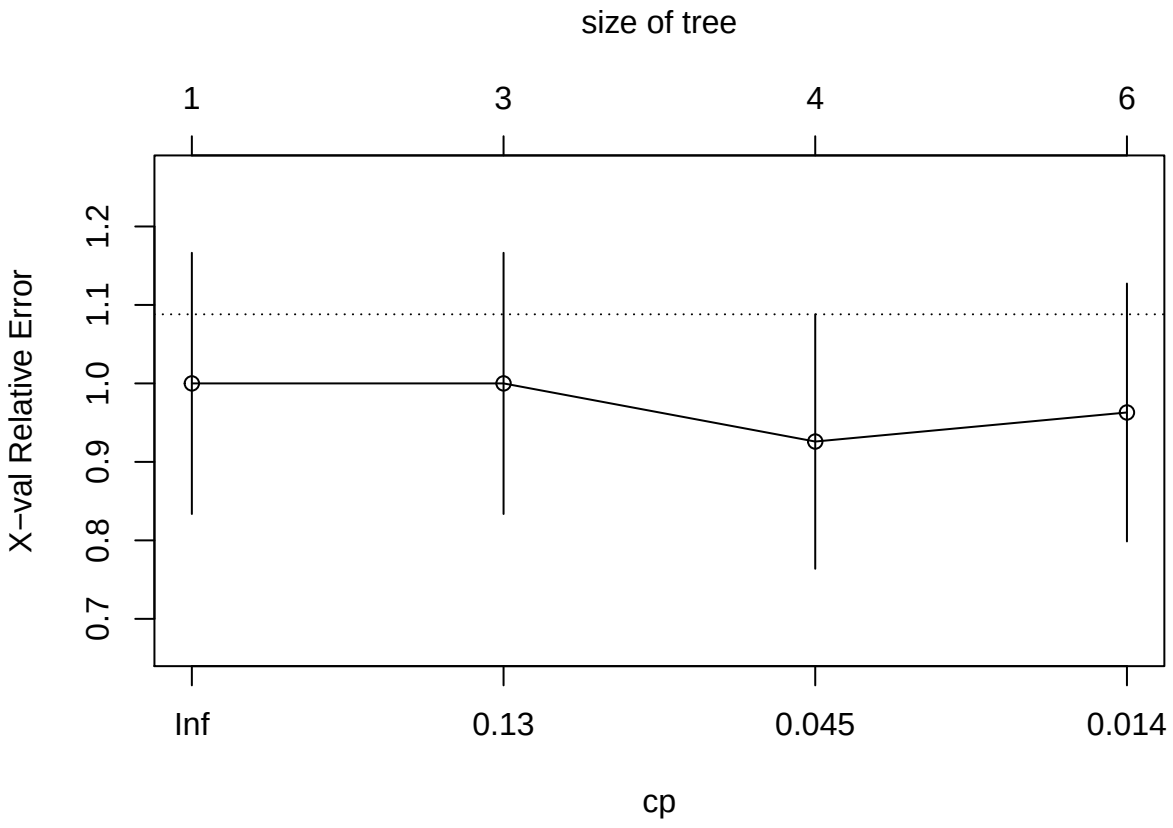
```
set.seed(100)
split <- sample.split(overview_total_1$Win_Lose, SplitRatio = 0.7)
Train <- subset(overview_total_1, split == TRUE)
Test <- subset(overview_total_1, split == FALSE)
```

```
library(rpart)
library(rpart.plot)
```

```
SimonasTree <- rpart(Win_Lose ~ serve_pts + first_in + second_in + bk_pts + bp_saved + return_pts + win
prp(SimonasTree)
```



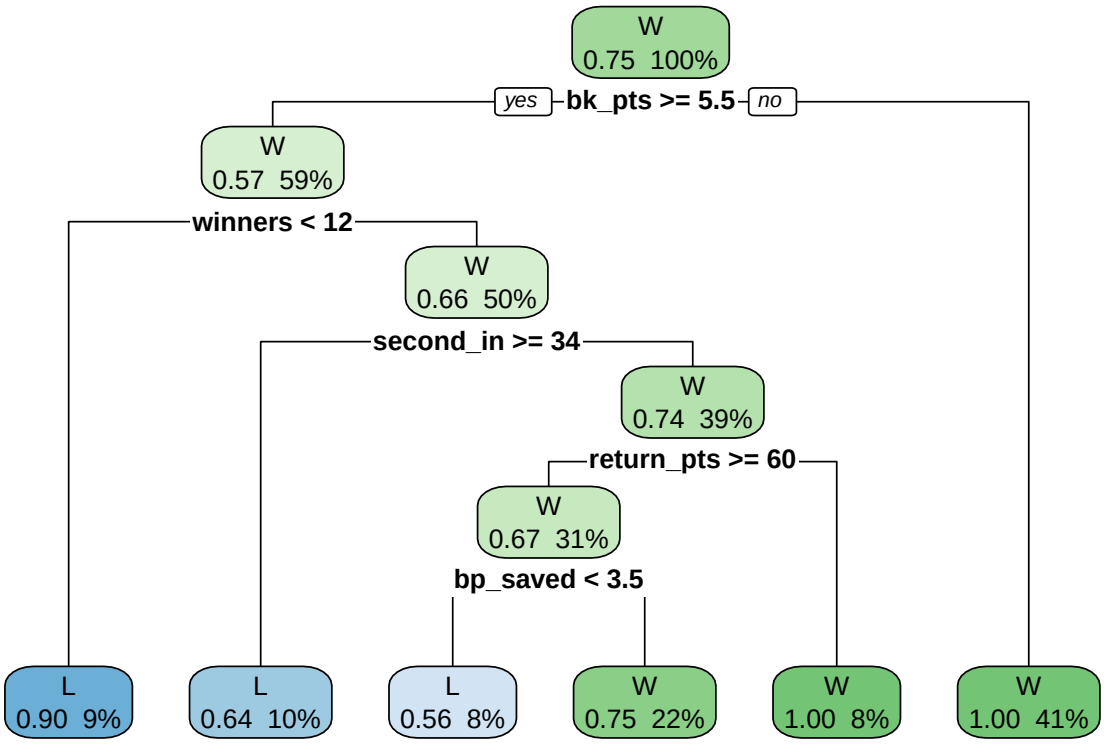

```
plotcp(SimonasTree)
```




```
printcp(SimonasTree)
```

```
##
## Classification tree:
## rpart(formula = Win_Lose ~ serve_pts + first_in + second_in +
##     bk_pts + bp_saved + return_pts + winners + unforced, data = Train,
##     method = "class", control = rpart.control(minbucket = 8))
##
## Variables actually used in tree construction:
## [1] bk_pts     bp_saved   return_pts second_in  winners
##
## Root node error: 27/107 = 0.25234
##
## n= 107
##
##         CP nsplit rel error  xerror    xstd
## 1 0.148148      0   1.00000 1.00000 0.16641
## 2 0.111111      2   0.70370 1.00000 0.16641
## 3 0.018519      3   0.59259 0.92593 0.16211
## 4 0.010000      5   0.55556 0.96296 0.16431
```

```
rpart.plot(SimonasTree, type=2, extra = 108)
```




```
summary(SimonasTree)
```

```
## Call:
## rpart(formula = Win_Lose ~ serve_pts + first_in + second_in +
##     bk_pts + bp_saved + return_pts + winners + unforced, data = Train,
##     method = "class", control = rpart.control(minbucket = 8))
##   n= 107
##
##           CP nsplit rel error    xerror      xstd
## 1 0.14814815      0 1.0000000 1.0000000 0.1664069
## 2 0.11111111      2 0.7037037 1.0000000 0.1664069
## 3 0.01851852      3 0.5925926 0.9259259 0.1621143
## 4 0.01000000      5 0.5555556 0.9629630 0.1643136
##
## Variable importance
##     bk_pts  second_in  serve_pts   bp_saved   unforced   first_in
##         19         17         16         14         10         10
##    winners return_pts
##         10          5
##
## Node number 1: 107 observations,    complexity param=0.1481481
##   predicted class=W  expected loss=0.2523364  P(node) =1
##     class counts:    27    80
##    probabilities: 0.252 0.748
```

```
##   left son=2 (63 obs) right son=3 (44 obs)
##   Primary splits:
##       bk_pts    < 5.5   to the right, improve=9.516689, (0 missing)
##       winners   < 10.5  to the left,  improve=6.617131, (0 missing)
##       serve_pts < 68    to the right, improve=4.482792, (0 missing)
##       first_in  < 47.5  to the right, improve=4.294711, (0 missing)
##       second_in < 26.5  to the right, improve=3.808906, (0 missing)
##   Surrogate splits:
##       bp_saved  < 2.5   to the right, agree=0.850, adj=0.636, (0 split)
##       serve_pts < 57.5  to the right, agree=0.813, adj=0.545, (0 split)
##       first_in  < 38.5  to the right, agree=0.785, adj=0.477, (0 split)
##       second_in < 21.5  to the right, agree=0.785, adj=0.477, (0 split)
##       unforced  < 19.5  to the right, agree=0.785, adj=0.477, (0 split)
##
## Node number 2: 63 observations,    complexity param=0.1481481
##   predicted class=W  expected loss=0.4285714  P(node) =0.588785
##     class counts:    27    36
##    probabilities: 0.429 0.571
##   left son=4 (10 obs) right son=5 (53 obs)
##   Primary splits:
##       winners   < 12.5  to the left,  improve=5.2835580, (0 missing)
##       bp_saved  < 3.5   to the left,  improve=1.6550150, (0 missing)
##       second_in < 34    to the right, improve=1.1508490, (0 missing)
##       bk_pts    < 8.5   to the right, improve=0.5142857, (0 missing)
##       unforced  < 31.5  to the left,  improve=0.4389610, (0 missing)
##   Surrogate splits:
##       serve_pts < 45.5  to the left,  agree=0.889, adj=0.3, (0 split)
##       second_in < 15    to the left,  agree=0.889, adj=0.3, (0 split)
##
## Node number 3: 44 observations
##   predicted class=W  expected loss=0  P(node) =0.411215
##     class counts:     0    44
##    probabilities: 0.000 1.000
##
## Node number 4: 10 observations
##   predicted class=L  expected loss=0.1  P(node) =0.09345794
##     class counts:     9     1
##    probabilities: 0.900 0.100
##
## Node number 5: 53 observations,    complexity param=0.1111111
##   predicted class=W  expected loss=0.3396226  P(node) =0.4953271
##     class counts:    18    35
##    probabilities: 0.340 0.660
##   left son=10 (11 obs) right son=11 (42 obs)
##   Primary splits:
##       second_in  < 34    to the right, improve=2.444581, (0 missing)
##       return_pts < 57.5  to the right, improve=2.173585, (0 missing)
##       serve_pts  < 67.5  to the right, improve=1.780602, (0 missing)
##       first_in   < 46.5  to the right, improve=1.630728, (0 missing)
##       bp_saved   < 3.5   to the left,  improve=1.362046, (0 missing)
##   Surrogate splits:
##       serve_pts  < 103.5 to the right, agree=0.849, adj=0.273, (0 split)
##       return_pts < 120.5 to the right, agree=0.830, adj=0.182, (0 split)
##       bp_saved   < 8.5   to the right, agree=0.811, adj=0.091, (0 split)
```

```
##       unforced   < 49    to the right, agree=0.811, adj=0.091, (0 split)
##
## Node number 10: 11 observations
##   predicted class=L  expected loss=0.3636364  P(node) =0.1028037
##     class counts:     7     4
##    probabilities: 0.636 0.364
##
## Node number 11: 42 observations,    complexity param=0.01851852
##   predicted class=W  expected loss=0.2619048  P(node) =0.3925234
##     class counts:    11    31
##    probabilities: 0.262 0.738
##   left son=22 (33 obs) right son=23 (9 obs)
##   Primary splits:
##       return_pts < 60.5  to the right, improve=1.5714290, (0 missing)
##       bp_saved   < 3.5   to the left,  improve=1.4880950, (0 missing)
##       unforced   < 26    to the right, improve=0.8715263, (0 missing)
##       serve_pts  < 67.5  to the right, improve=0.7714286, (0 missing)
##       second_in  < 26.5  to the right, improve=0.7506811, (0 missing)
##   Surrogate splits:
##       unforced  < 23.5  to the right, agree=0.905, adj=0.556, (0 split)
##       serve_pts < 54    to the right, agree=0.881, adj=0.444, (0 split)
##       second_in < 20.5  to the right, agree=0.881, adj=0.444, (0 split)
##       first_in  < 36    to the right, agree=0.857, adj=0.333, (0 split)
##
## Node number 22: 33 observations,    complexity param=0.01851852
##   predicted class=W  expected loss=0.3333333  P(node) =0.3084112
##     class counts:    11    22
##    probabilities: 0.333 0.667
##   left son=44 (9 obs) right son=45 (24 obs)
##   Primary splits:
##       bp_saved  < 3.5   to the left,  improve=1.2222220, (0 missing)
##       unforced  < 31.5  to the left,  improve=1.0476190, (0 missing)
##       bk_pts    < 10.5  to the left,  improve=0.9166667, (0 missing)
##       serve_pts < 86.5  to the left,  improve=0.7575758, (0 missing)
##       first_in  < 58.5  to the left,  improve=0.7575758, (0 missing)
##   Surrogate splits:
##       bk_pts     < 6.5   to the left,  agree=0.879, adj=0.556, (0 split)
##       first_in   < 38    to the left,  agree=0.848, adj=0.444, (0 split)
##       return_pts < 65.5  to the left,  agree=0.848, adj=0.444, (0 split)
##       serve_pts  < 59    to the left,  agree=0.818, adj=0.333, (0 split)
##       second_in  < 21.5  to the left,  agree=0.758, adj=0.111, (0 split)
##
## Node number 23: 9 observations
##   predicted class=W  expected loss=0  P(node) =0.08411215
##     class counts:     0     9
##    probabilities: 0.000 1.000
##
## Node number 44: 9 observations
##   predicted class=L  expected loss=0.4444444  P(node) =0.08411215
##     class counts:     5     4
##    probabilities: 0.556 0.444
##
## Node number 45: 24 observations
##   predicted class=W  expected loss=0.25  P(node) =0.2242991
```

```
##     class counts:     6    18
##    probabilities: 0.250 0.750
```

Each node shows: + the predicted class (win or lose) + the predicted probability of winning + the percentage of observations in the node

```
PredictCART <- predict(SimonasTree, newdata=Test, type="class")
table(Test$Win_Lose, PredictCART)
```

```
##    PredictCART
##      L  W
##   L  7  4
##   W  6 28
```

Accuracy of the CART model is 0.7777778

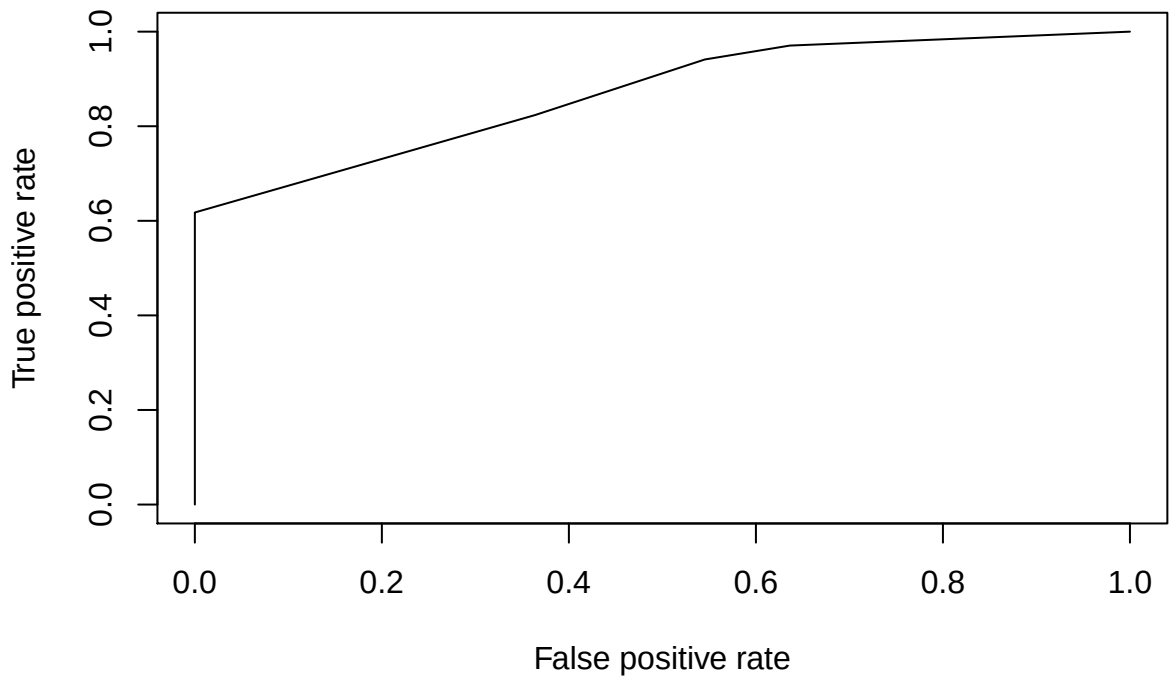
```
library(ROCR)
```

```
PredictROC <- predict(SimonasTree, newdata = Test)
PredictROC
```

```
##             L         W
## 2   0.9000000 0.1000000
## 9   0.5555556 0.4444444
## 11  0.5555556 0.4444444
## 16  0.0000000 1.0000000
## 20  0.0000000 1.0000000
## 27  0.5555556 0.4444444
## 29  0.0000000 1.0000000
## 31  0.6363636 0.3636364
## 35  0.0000000 1.0000000
## 36  0.2500000 0.7500000
## 40  0.0000000 1.0000000
## 43  0.2500000 0.7500000
## 44  0.0000000 1.0000000
## 45  0.0000000 1.0000000
## 49  0.0000000 1.0000000
## 52  0.2500000 0.7500000
## 53  0.0000000 1.0000000
## 56  0.0000000 1.0000000
## 59  0.0000000 1.0000000
## 60  0.9000000 0.1000000
## 61  0.2500000 0.7500000
## 63  0.9000000 0.1000000
## 78  0.5555556 0.4444444
## 85  0.5555556 0.4444444
## 90  0.5555556 0.4444444
## 92  0.2500000 0.7500000
## 95  0.0000000 1.0000000
## 96  0.2500000 0.7500000
## 97  0.0000000 1.0000000
## 99  0.0000000 1.0000000
```

```
## 104 0.2500000 0.7500000
## 105 0.2500000 0.7500000
## 109 0.2500000 0.7500000
## 113 0.0000000 1.0000000
## 114 0.2500000 0.7500000
## 120 0.0000000 1.0000000
## 128 0.9000000 0.1000000
## 131 0.6363636 0.3636364
## 132 0.0000000 1.0000000
## 134 0.9000000 0.1000000
## 143 0.0000000 1.0000000
## 144 0.0000000 1.0000000
## 147 0.0000000 1.0000000
## 149 0.0000000 1.0000000
## 152 0.2500000 0.7500000
```

```
pred <- prediction(PredictROC[,2], Test$Win_Lose)
perf <- performance(pred, "tpr", "fpr")
plot(perf)
```



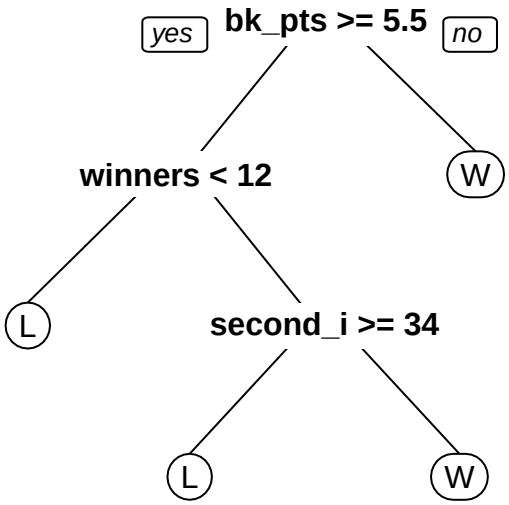

```
library(caret)
library(e1071)
```

```r
fitControl <- trainControl(method="cv", number=10)
cartGrid <- expand.grid(.cp=1:50*0.01)
train(Win_Lose ~ serve_pts + first_in + second_in + bk_pts + bp_saved + return_pts + winners + unforced
```

```
## CART
##
## 107 samples
##  25 predictor
##   2 classes: 'L', 'W'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 97, 96, 97, 96, 96, 96, ...
## Resampling results across tuning parameters:
##
##   cp    Accuracy   Kappa
##   0.01  0.7136364  0.27774029
##   0.02  0.7136364  0.25879292
##   0.03  0.7136364  0.25879292
##   0.04  0.7236364  0.28510871
##   0.05  0.7418182  0.30526760
##   0.06  0.7318182  0.26908339
##   0.07  0.7318182  0.26908339
##   0.08  0.7318182  0.26908339
##   0.09  0.7590909  0.32170476
##   0.10  0.7590909  0.32170476
##   0.11  0.7590909  0.32170476
##   0.12  0.7490909  0.20065213
##   0.13  0.7672727  0.22568957
##   0.14  0.7672727  0.22568957
##   0.15  0.7490909  0.07523585
##   0.16  0.7490909  0.03773585
##   0.17  0.7490909  0.00000000
##   0.18  0.7490909  0.00000000
##   0.19  0.7490909  0.00000000
##   0.20  0.7490909  0.00000000
##   0.21  0.7490909  0.00000000
##   0.22  0.7490909  0.00000000
##   0.23  0.7490909  0.00000000
##   0.24  0.7490909  0.00000000
##   0.25  0.7490909  0.00000000
##   0.26  0.7490909  0.00000000
##   0.27  0.7490909  0.00000000
##   0.28  0.7490909  0.00000000
##   0.29  0.7490909  0.00000000
##   0.30  0.7490909  0.00000000
##   0.31  0.7490909  0.00000000
##   0.32  0.7490909  0.00000000
##   0.33  0.7490909  0.00000000
##   0.34  0.7490909  0.00000000
##   0.35  0.7490909  0.00000000
##   0.36  0.7490909  0.00000000
##   0.37  0.7490909  0.00000000
```

```
##   0.38  0.7490909  0.00000000
##   0.39  0.7490909  0.00000000
##   0.40  0.7490909  0.00000000
##   0.41  0.7490909  0.00000000
##   0.42  0.7490909  0.00000000
##   0.43  0.7490909  0.00000000
##   0.44  0.7490909  0.00000000
##   0.45  0.7490909  0.00000000
##   0.46  0.7490909  0.00000000
##   0.47  0.7490909  0.00000000
##   0.48  0.7490909  0.00000000
##   0.49  0.7490909  0.00000000
##   0.50  0.7490909  0.00000000
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.14.
```

```
SimonasTreeCV <- rpart(Win_Lose ~ serve_pts + first_in + second_in + bk_pts + bp_saved + return_pts + wi
prp(SimonasTreeCV)
```




```
PredictCV <- predict(SimonasTreeCV, newdata=Test, type="class")
table(Test$Win_Lose, PredictCV)
```

```
##    PredictCV
```

```
##      L  W
##   L  5  6
##   W  2 32
```

Accuracy of the Cross Validation model is 0.8222222

Conclusions

1. Simona's favorite surface is the hard court. Her smart aggressive game is very suitable for this surface. As we can see the number of matches won on hard courts is the biggest.
2. The average number of winners and the unforced errors are almost the same on all the three surfaces.
3. We can slight drop in average winners and forced errors in 2016 and a slight increase in unforced errors in 2016. This is can be a sign of trying to be more aggressive on the court.
4. The first serve in and second serve in percentages were very good in 2015 with a short decrease but constant percentage in 2016
5. In the serve basics figures we see a pattern – a decrease in the numbers of the body serve and points longer than 3 shots and an increase in the wide and t serves. This is a sign of a more aggressive pattern of play.
6. From the shot direction figures an increase in the down the middle, inside out and inside in shots are visible. The crosscourt and the down the line shots have a more constant pattern.