ST442 Project

Ryan Mersereau, Aiden Bartlett, Nico Field

2023-12-07

```r
library(readr)
library(tidyverse)
```

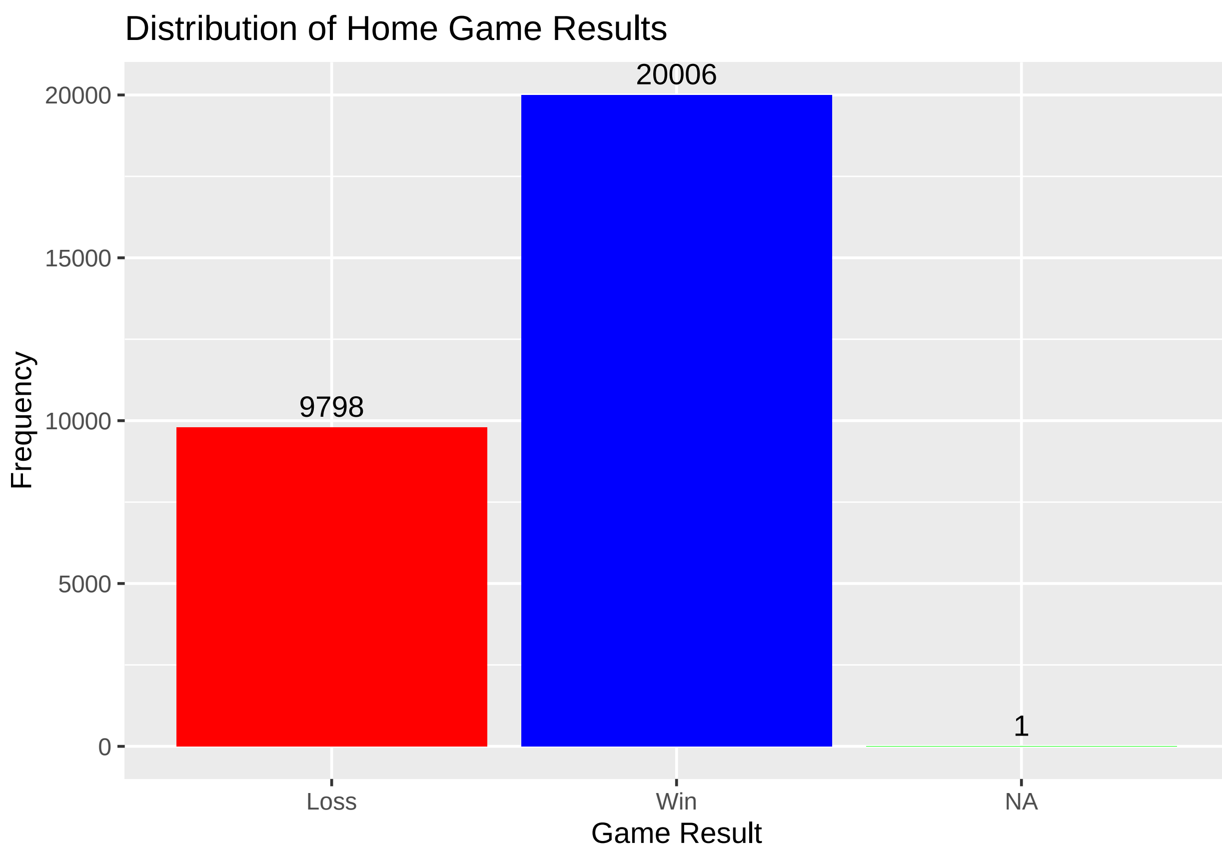
```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v purrr     1.0.2
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
bball <- read_csv("bball.csv")
```

```
## Rows: 29805 Columns: 132
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (50): game_id, status, coverage, scheduled_date, gametime, tournament, ...
## dbl  (78): season, attendance, lead_changes, times_tied, periods, venue_capa...
## lgl   (2): neutral_site, conference_game
## time  (2): h_minutes, a_minutes
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Finding variables that best predict winning for the home team
# Create new variable for point differential and win/loss/tie
library(dplyr)
bball <- bball |>
  mutate(point_differential = h_points_game - a_points_game,
         h_game_result = case_when(
           point_differential > 0 ~ "Win",
           point_differential < 0 ~ "Loss",
         ))
```

```r
# EDA
library(ggplot2)

# Create a bar graph for distribution of results
ggplot(bball, aes(x = h_game_result)) +
  geom_bar(fill = c("red", "blue", "green")) +
  geom_text(stat = "count", aes(label = after_stat(count)), vjust = -0.5) +
  labs(title = "Distribution of Home Game Results", x = "Game Result", y = "Frequency")
```




```
bball |>
  dplyr::select(h_alias, h_points_game, a_alias, a_points_game, point_differential, h_game_result)
```

```
## # A tibble: 29,805 x 6
##    h_alias h_points_game a_alias a_points_game point_differential h_game_result
##    <chr>           <dbl> <chr>           <dbl>              <dbl> <chr>
##  1 CHA                73 UNLV               93                -20 Loss
##  2 CHA                72 KU                123                -51 Loss
##  3 CHA                93 SJU               100                 -7 Loss
##  4 ORST               82 CSF                69                 13 Win
##  5 ORST               76 TLSA               71                  5 Win
##  6 COLO               67 CONN               74                 -7 Loss
##  7 ORST               67 VCU                75                 -8 Loss
##  8 ARIZ               55 WICH               65                -10 Loss
##  9 STAN               55 COLO               56                 -1 Loss
## 10 STAN               77 CAL                71                  6 Win
## # i 29,795 more rows
```

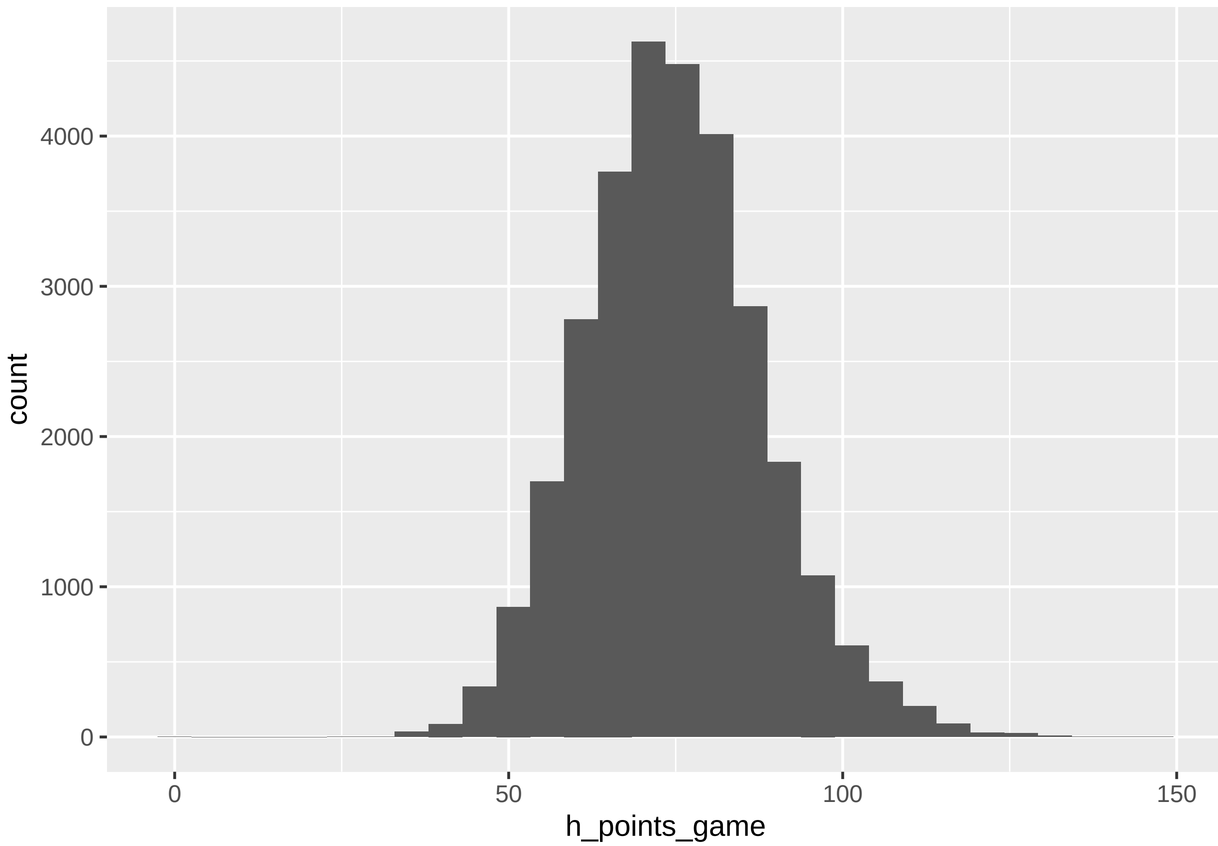
```
bball |>
  dplyr::select(h_points_game, h_points)
```

```
## # A tibble: 29,805 x 2
##    h_points_game h_points
##            <dbl>    <dbl>
##  1            73       73
##  2            72       72
##  3            93       93
```

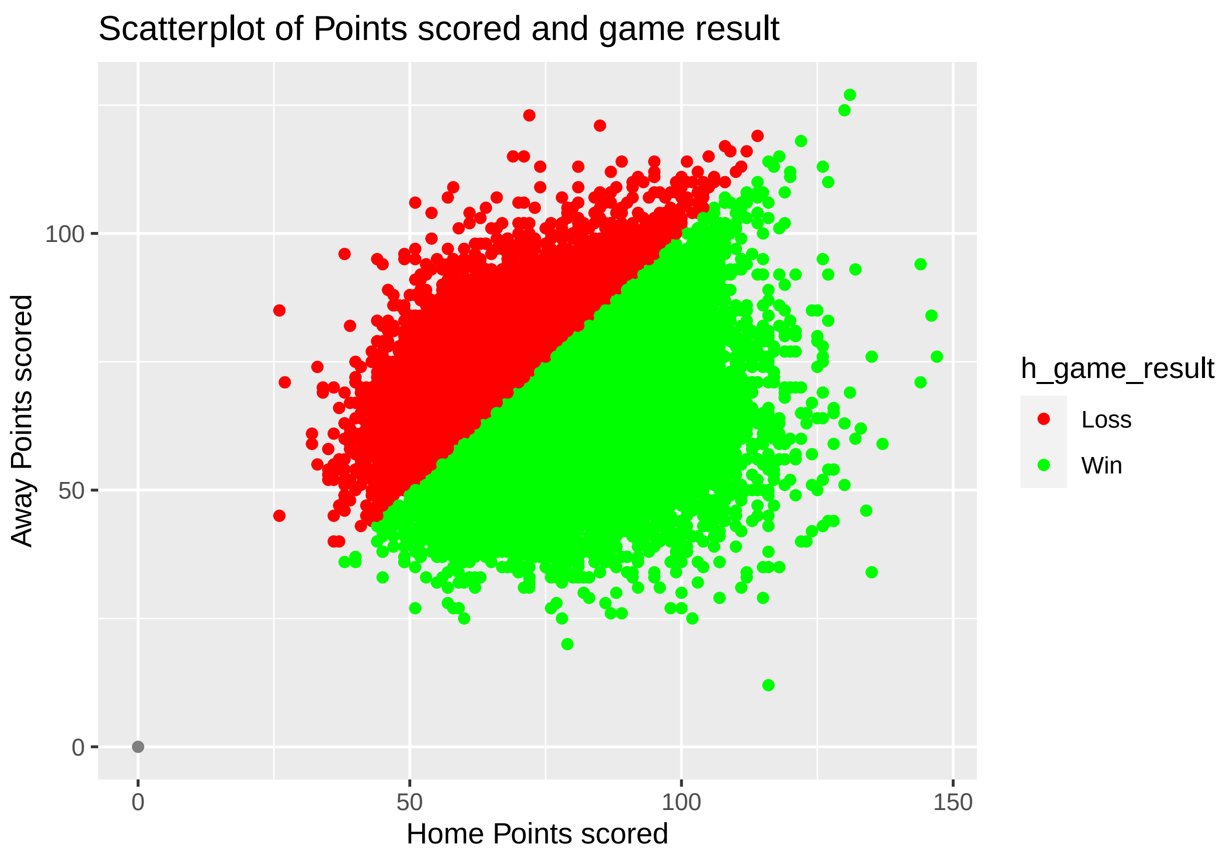
```
## 4           82       82
## 5           76       76
## 6           67       67
## 7           67       67
## 8           55       55
## 9           55       55
## 10          77       77
## # i 29,795 more rows
```

```r
#Histogram of points scored
library(ggplot2)

ggplot(bball, aes(x = h_points_game)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
# Scatter plot of home vs away final score and home result

ggplot(bball, aes(x = h_points_game, y = a_points_game, color = h_game_result)) +
  geom_point() +
  scale_color_manual(values = c("Win" = "green", "Loss" = "red", "Tie" = "blue")) +
  labs(title = "Scatterplot of Points scored and game result",
       x = "Home Points scored",
       y = "Away Points scored")
```



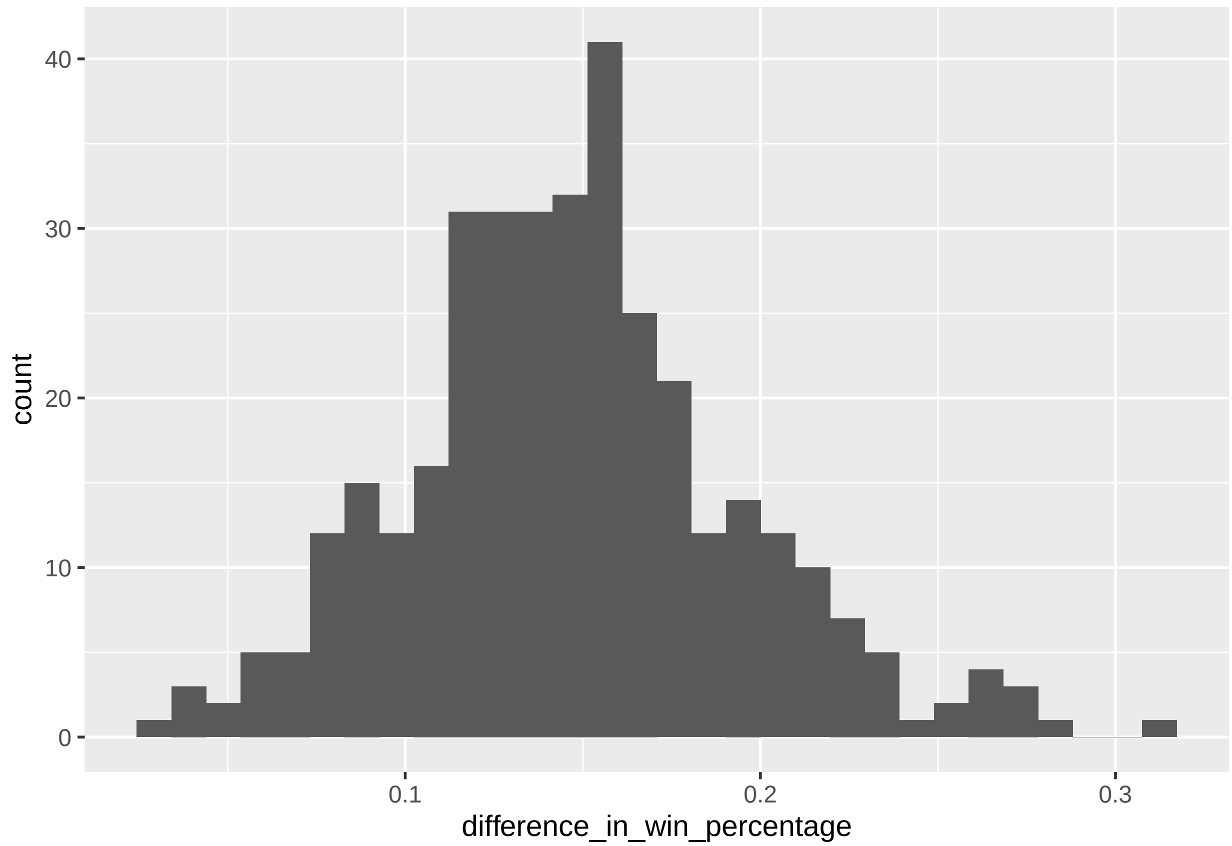

```r
# Predict with logistic regression model
# Include relevant variables
bball_vars <- bball[, c(43:78, 134)] # Including home team stats only
head(bball_vars)
```

```
## # A tibble: 6 x 37
##   h_rank h_minutes h_field_goals_made h_field_goals_att h_field_goals_pct
##    <dbl> <time>                 <dbl>             <dbl>             <dbl>
## 1      0 03:20                     26                61              42.6
## 2      0 03:20                     24                69              34.8
## 3      0 03:20                     34                67              50.7
## 4      0 03:20                     26                54              48.1
## 5      0 03:20                     22                54              40.7
## 6      0 03:20                     23                51              45.1
## # i 32 more variables: h_three_points_made <dbl>, h_three_points_att <dbl>,
## #   h_three_points_pct <dbl>, h_two_points_made <dbl>, h_two_points_att <dbl>,
## #   h_two_points_pct <dbl>, h_blocked_att <dbl>, h_free_throws_made <dbl>,
## #   h_free_throws_att <dbl>, h_free_throws_pct <dbl>,
## #   h_offensive_rebounds <dbl>, h_defensive_rebounds <dbl>, h_rebounds <dbl>,
## #   h_assists <dbl>, h_turnovers <dbl>, h_steals <dbl>, h_blocks <dbl>,
## #   h_assists_turnover_ratio <dbl>, h_personal_fouls <dbl>, ...
```

```r
bball_vars <- bball_vars |>
  dplyr::select(-h_points)

bball_vars <- na.omit(bball_vars)
```

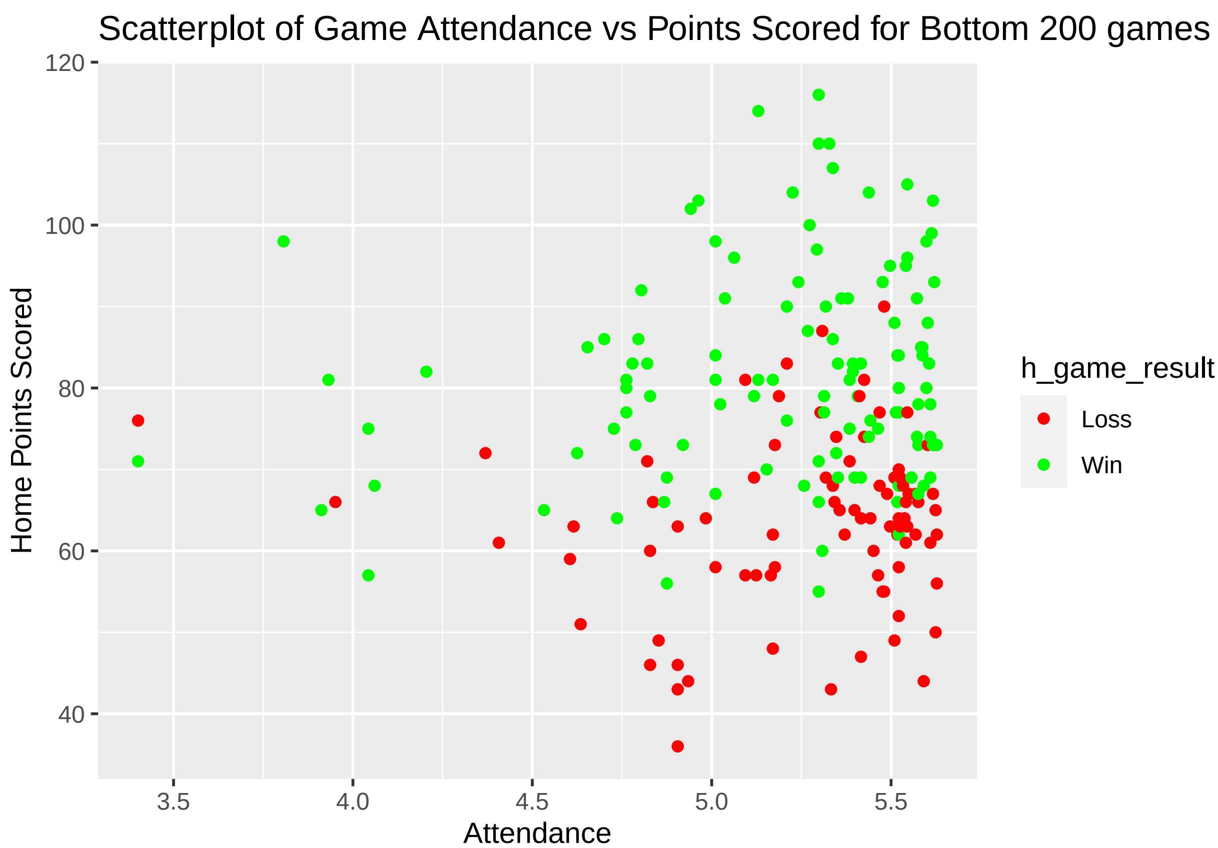
```r
## Let's see what teams' performance changes the most given if they are home or away.

## This is a list of all the teams and their home games / wins.
bball_home_team <- bball |>  group_by(h_name, h_market) |> mutate(home_game_wins_sum = cumsum(h_game_re

## This is a list of all of the same teams and their away games / wins.
bball_away_team <- bball |> group_by(a_name, a_market) |> mutate(away_game_wins_sum = cumsum(h_game_resu


## Now we have to combine the tables in order to tidy it up.
teams_data <- left_join(bball_home_team, bball_away_team, by = join_by(h_name == a_name, h_market == a_

teams_data <- teams_data |> mutate(total_win_percentage = (home_game_wins_sum + away_game_wins_sum) / (

teams_data <- teams_data |> mutate(difference_in_win_percentage = home_win_percentage - total_win_perce

## Shows the distribution of win percentages.
ggplot(teams_data, aes(x = difference_in_win_percentage)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 4 rows containing non-finite values (`stat_bin()`).
```




```r
## Seems strange. How many teams win more on the road than they lose?
teams_above_500_on_the_road <- nrow(filter(teams_data, away_game_wins_sum / away_games_played > 0.5))
## Huh, only 37 out of 359!
```

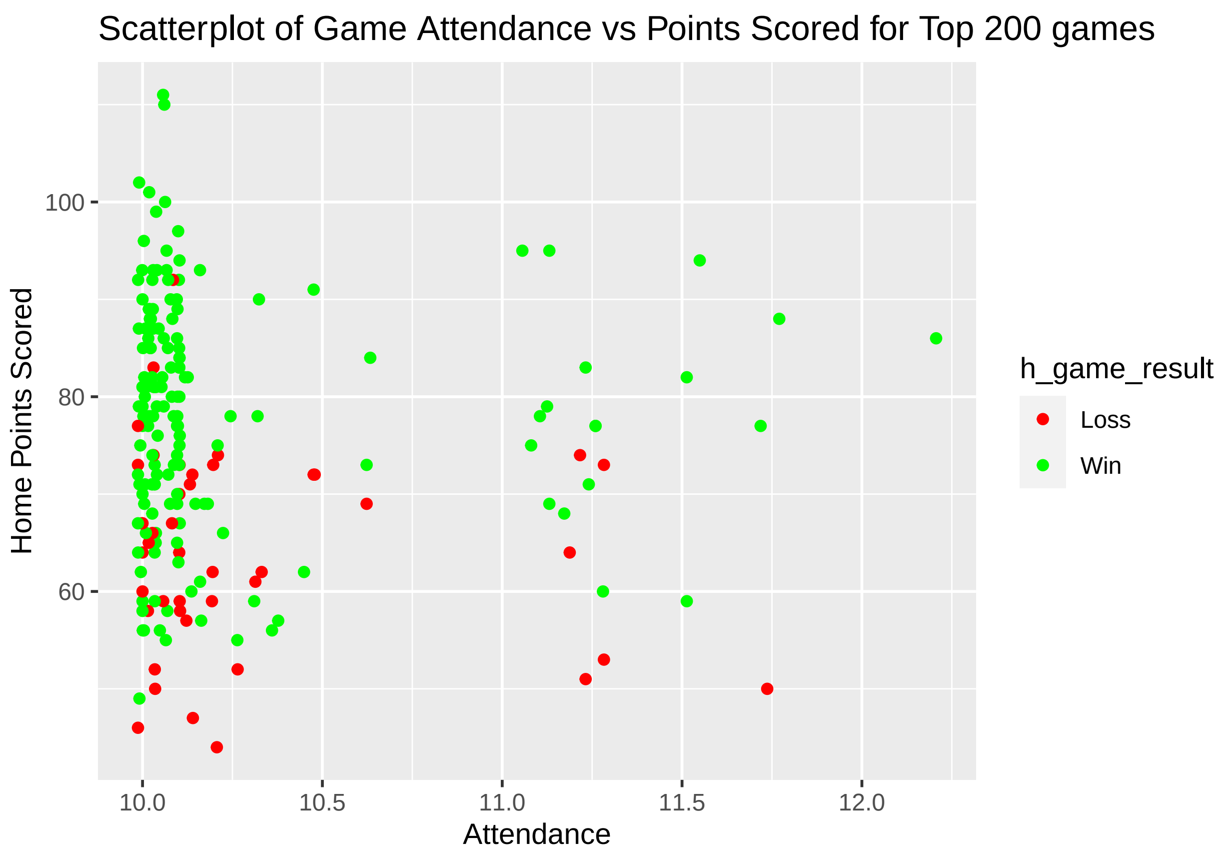
```r
## See if game attendance plays any role in home team success
library(ggplot2)
bball <- bball |>
  mutate(point_differential = h_points_game - a_points_game,
         h_game_result = case_when(
           point_differential > 0 ~ "Win",
           point_differential < 0 ~ "Loss",
         ))

# Remove rows with NA values in the 'attendance' column
bball <- bball %>% filter(!is.na(attendance) & attendance != 0)

# Arrange the dataframe based on attendance in ascending order
bball <- bball %>% arrange(attendance)

# Select the bottom 1000 games with the lowest attendance into a new dataframe
bottom_200_games <- bball %>% slice(1:200)

# Arrange the dataframe based on attendance in ascending order
bball <- bball %>% arrange(desc(attendance))

# Select the bottom 1000 games with the lowest attendance into a new dataframe
top_200_games <- bball %>% slice(1:200)

# Plot with ggplot
ggplot(bottom_200_games, aes(x = log(attendance), y = h_points_game, color = h_game_result)) +
  geom_point() +
  scale_color_manual(values = c("Win" = "green", "Loss" = "red")) +
  labs(title = "Scatterplot of Game Attendance vs Points Scored for Bottom 200 games",
       x = "Attendance",
       y = "Home Points Scored")
```




```r
total_games <- nrow(bball)
# Calculate the number of games won
bottom_games_won <- sum(bottom_200_games$h_game_result == "Win", na.rm = TRUE)

cat("Percentage won for Bottom 200 games:", (bottom_games_won / 200) * 100)
```

```
## Percentage won for Bottom 200 games: 58.5
```

```r
# Plot with ggplot
ggplot(top_200_games, aes(x = log(attendance), y = h_points_game, color = h_game_result)) +
  geom_point() +
  scale_color_manual(values = c("Win" = "green", "Loss" = "red")) +
  labs(title = "Scatterplot of Game Attendance vs Points Scored for Top 200 games",
       x = "Attendance",
       y = "Home Points Scored")
```




```
# Calculate the number of games won
top_games_won <- sum(top_200_games$h_game_result == "Win", na.rm = TRUE)

cat("Percentage won for Top 200 games:", (top_games_won / 200) * 100)
```

```
## Percentage won for Top 200 games: 78.5
```

```
regmodel <- glm(as.factor(h_game_result) ~ h_rank + h_field_goals_made, data = bball_vars, family = bin
summary(regmodel)
```

```
##
## Call:
## glm(formula = as.factor(h_game_result) ~ h_rank + h_field_goals_made,
##     family = binomial, data = bball_vars)
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -4.463608   0.215307 -20.731  < 2e-16 ***
## h_rank              0.045381   0.006485   6.997 2.61e-12 ***
## h_field_goals_made  0.198589   0.008492  23.386  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5648.7  on 4506  degrees of freedom
## Residual deviance: 4818.5  on 4504  degrees of freedom
```

```
## AIC: 4824.5
##
## Number of Fisher Scoring iterations: 4
```

```r
regmodel_full <- glm(as.factor(h_game_result) ~., data = bball_vars, family = binomial)
#summary(regmodel_full)
```

```r
levels(as.factor(bball$h_game_result))
```

```
## [1] "Loss" "Win"
```

```r
# Final model using only 14 most significant variables
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

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

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
final_formula <- as.factor(h_game_result) ~ h_field_goals_made + h_defensive_rebounds + h_points_off_tu

final_model <- glm(final_formula, data = bball_vars, family = binomial)
summary(final_model)
```

```
##
## Call:
## glm(formula = final_formula, family = binomial, data = bball_vars)
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -0.33893    0.65968  -0.514 0.607399
## h_field_goals_made       0.51457    0.02277  22.596  < 2e-16 ***
## h_defensive_rebounds     0.51670    0.02104  24.563  < 2e-16 ***
## h_points_off_turnovers   0.18892    0.01413  13.367  < 2e-16 ***
## h_turnovers             -0.44588    0.02255 -19.776  < 2e-16 ***
## h_offensive_rebounds     0.42862    0.02549  16.812  < 2e-16 ***
## h_team_rebounds          0.44376    0.03483  12.741  < 2e-16 ***
## h_steals                 0.35861    0.02903  12.351  < 2e-16 ***
## h_three_points_made      0.25990    0.02318  11.214  < 2e-16 ***
## h_personal_fouls        -0.14383    0.01669  -8.620  < 2e-16 ***
## h_free_throws_made       0.22963    0.02757   8.329  < 2e-16 ***
## h_field_goals_att       -0.50674    0.02121 -23.893  < 2e-16 ***
## h_free_throws_att       -0.11732    0.02218  -5.289 1.23e-07 ***
## h_rank                   0.03901    0.01016   3.840 0.000123 ***
## h_coach_tech_fouls      -1.36777    0.35877  -3.812 0.000138 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5648.7  on 4506  degrees of freedom
## Residual deviance: 1947.7  on 4492  degrees of freedom
```

```
## AIC: 1977.7
##
## Number of Fisher Scoring iterations: 7
```

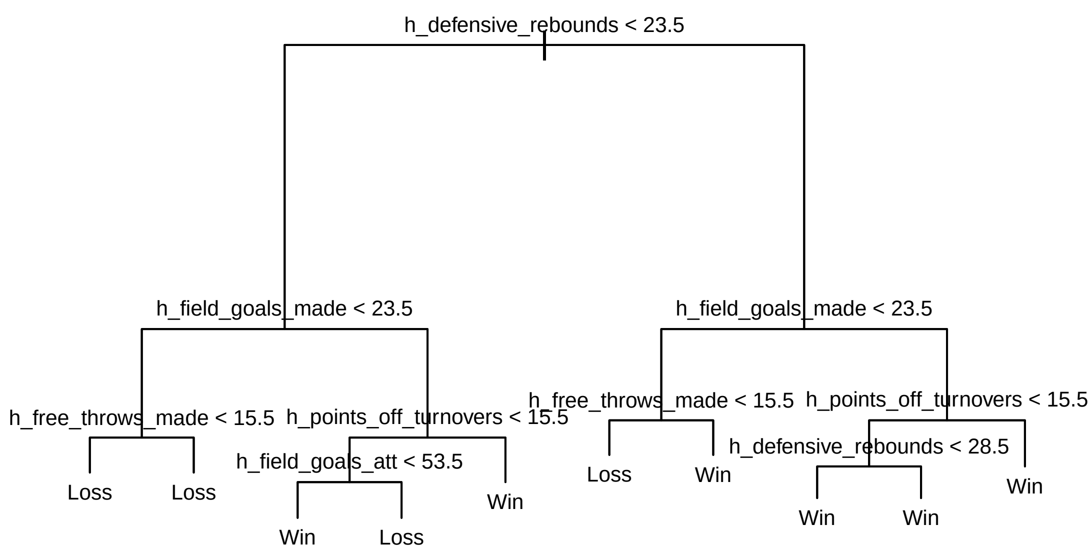
```r
# Classification tree
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.3.2
```

```r
wintree <- tree(final_formula, data = bball_vars)
summary(wintree)
```

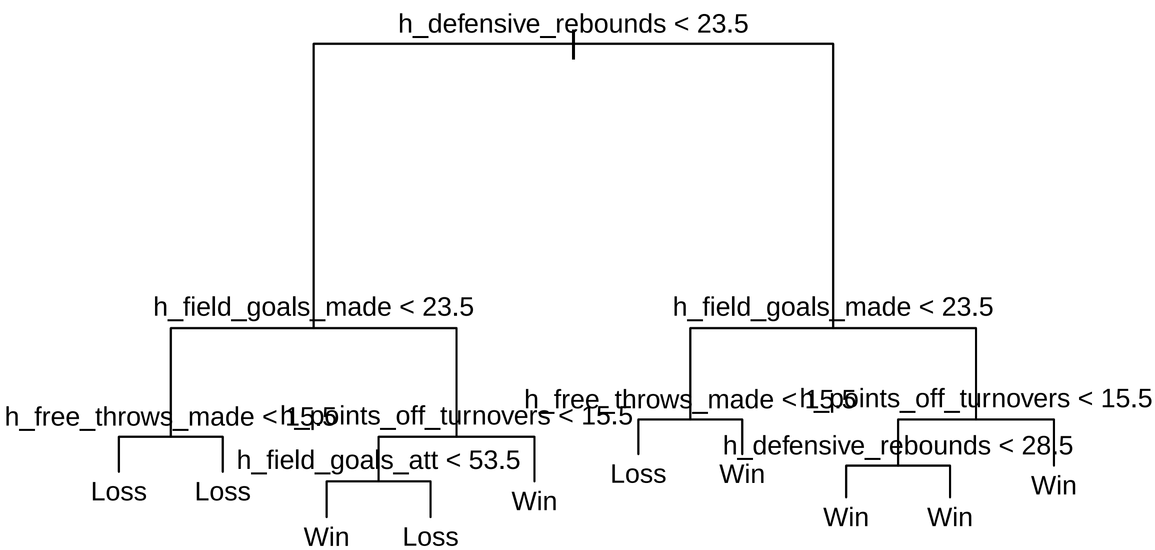
```
##
## Classification tree:
## tree(formula = final_formula, data = bball_vars)
## Variables actually used in tree construction:
## [1] "h_defensive_rebounds"   "h_field_goals_made"     "h_free_throws_made"
## [4] "h_points_off_turnovers" "h_field_goals_att"
## Number of terminal nodes:  10
## Residual mean deviance:  0.9466 = 4257 / 4497
## Misclassification error rate: 0.2205 = 994 / 4507
```

```r
plot(wintree)
text(wintree, pretty = 1, cex = 0.7)
```




```r
# Perform cross-validation
cv_result <- cv.tree(wintree)
```

```r
# Prune the tree based on cross-validation results
pruned_tree <- prune.tree(wintree, best = cv_result$size[which.min(cv_result$dev)])

# Plot the pruned tree
plot(pruned_tree)
text(pruned_tree, pretty = 0, cex = 0.8)
```



Pruning with cross validation resulted in no change to the tree.