Levon_Gevorgyan_DV_Homework_4

2025-03-24

Libraries

```r
library(ggplot2)
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
```

The Dataset

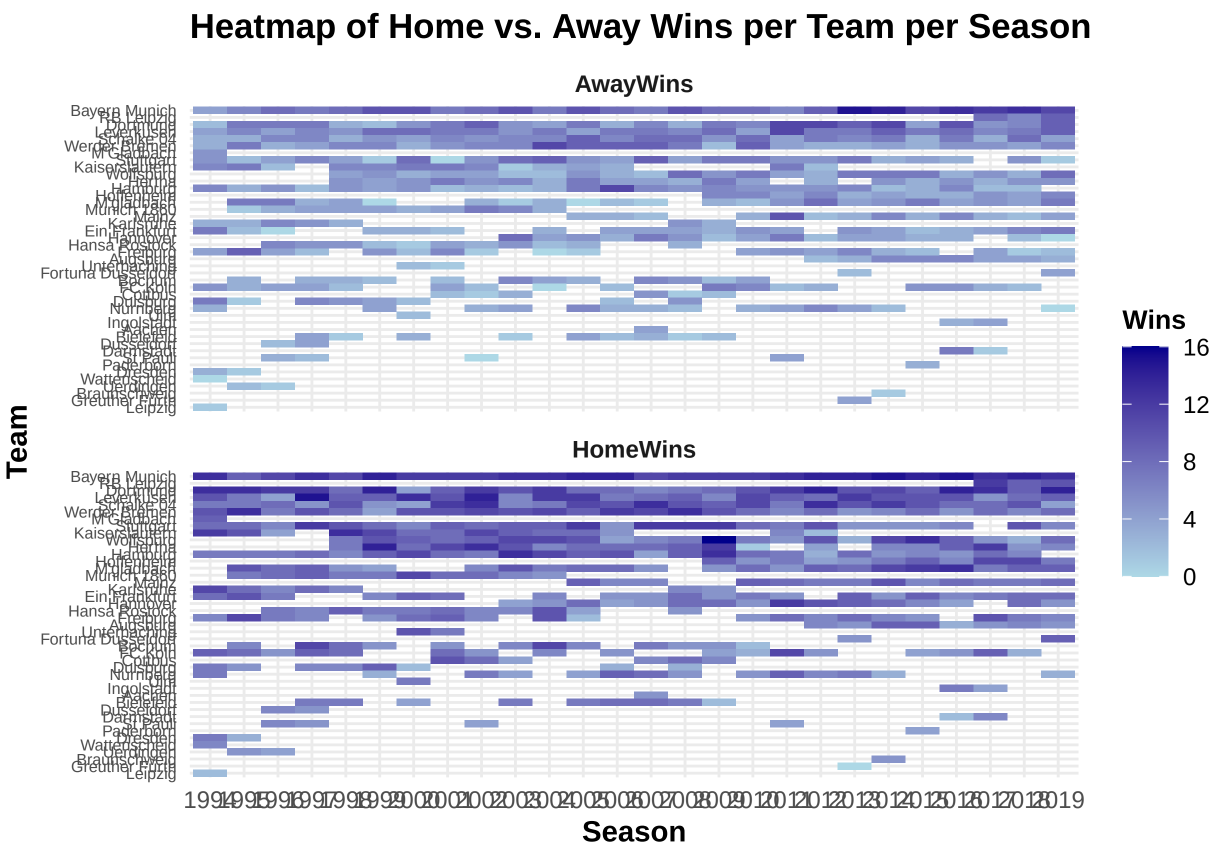
```r
bundesliga <- read.csv("bundesliga.csv")
```

Part 2. Home Advantage Deconstructed

```r
bundesliga <- bundesliga %>%
  mutate(FTR = case_when(
    FTHG > FTAG ~ "H",
    FTHG < FTAG ~ "A",
    TRUE ~ "D"
  ))

home_wins <- bundesliga %>%
  filter(FTR == "H") %>%
  group_by(SEASON, HOMETEAM) %>%
  summarise(HomeWins = n(), .groups = 'drop')

away_wins <- bundesliga %>%
  filter(FTR == "A") %>%
  group_by(SEASON, AWAYTEAM) %>%
  summarise(AwayWins = n(), .groups = 'drop')

team_wins <- full_join(home_wins, away_wins, by = c("SEASON" = "SEASON", "HOMETEAM" = "AWAYTEAM")) %>%
  rename(Team = HOMETEAM) %>%
  mutate(HomeWins = ifelse(is.na(HomeWins), 0, HomeWins),
         AwayWins = ifelse(is.na(AwayWins), 0, AwayWins))
```

1. Heatmap of Home vs. Away Wins per Team per Season

```
team_wins_long <- data.frame()
for (win_type in c("HomeWins", "AwayWins")) {
  temp <- data.frame(
    SEASON = team_wins$SEASON,
    Team = team_wins$Team,
    Wins = team_wins[[win_type]],
    WinType = win_type
  )
  team_wins_long <- rbind(team_wins_long, temp)
}

ggplot(team_wins_long, aes(x = factor(SEASON), y = reorder(Team, Wins), fill = Wins)) +
  geom_tile() +
  facet_wrap(~WinType, ncol = 1) +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +
  theme_minimal() +
  theme(plot.title = element_text(face = "bold"),
        axis.text.y = element_text(size = 6),
        axis.title.x = element_text(face = "bold"),
        axis.title.y = element_text(face = "bold"),
        strip.text = element_text(face = "bold"),
        legend.title = element_text(size = 10, face = "bold")) +
  labs(title = "Heatmap of Home vs. Away Wins per Team per Season",
       x = "Season", y = "Team", fill = "Wins")
```

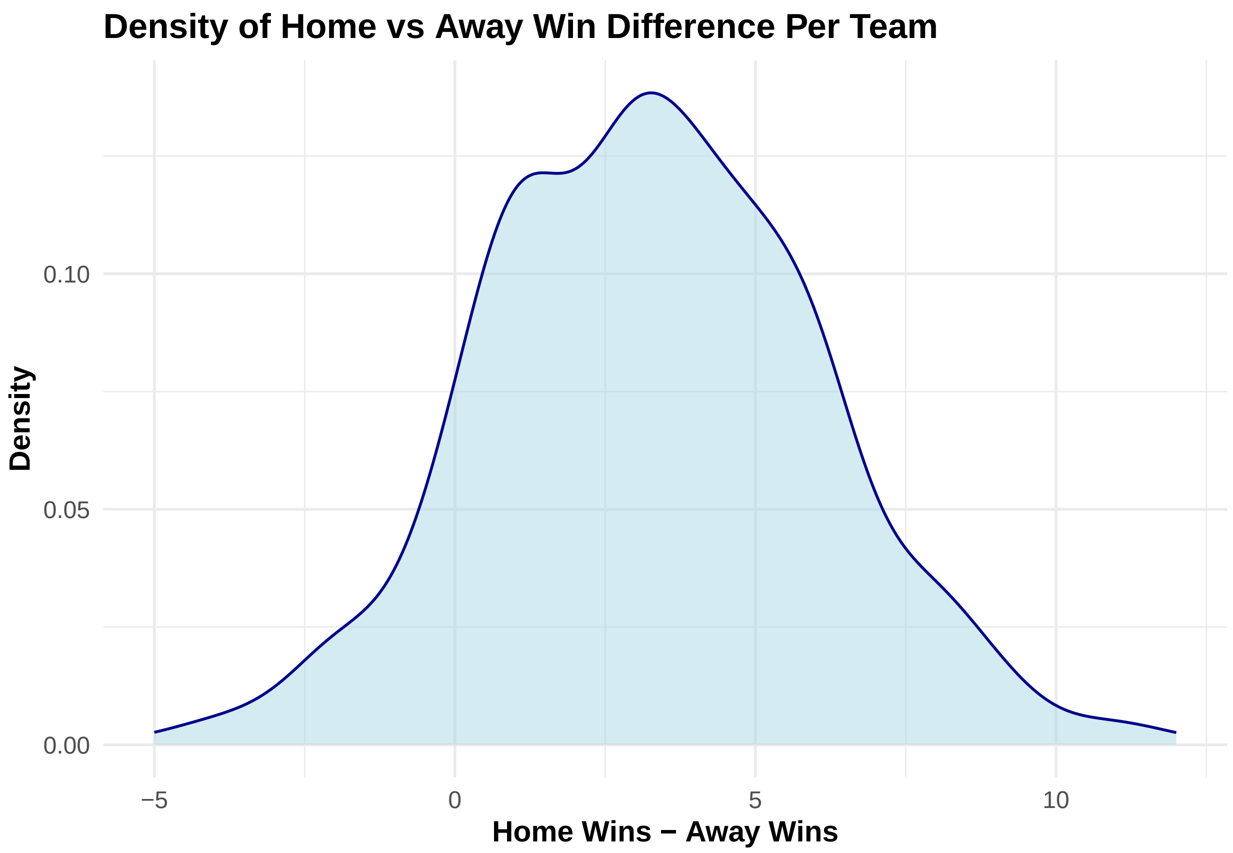


```r
ggsave("home_away_wins_heatmap.pdf", width = 10, height = 12)
```

NOTE: graph is ugly here, but the point is to view at in the **"home__away__wins__heatmap.pdf"** file. There you can see all well-structured and understandable.

2. Point Differential Density

```r
if ("HomeWins" %in% colnames(team_wins) & "AwayWins" %in% colnames(team_wins)) {
  team_wins <- team_wins %>%
    mutate(PointDiff = HomeWins - AwayWins)
} else {
  stop("there are missing columns")
}

ggplot(team_wins, aes(x = PointDiff)) +
  geom_density(fill = "lightblue", color = "darkblue", alpha = 0.5) +
  theme_minimal() +
  theme(
    plot.title = element_text(face = "bold"),
    axis.title.x = element_text(face = "bold"),
    axis.title.y = element_text(face = "bold")
  ) +
  labs(title = "Density of Home vs Away Win Difference Per Team",
       x = "Home Wins - Away Wins", y = "Density")
```



```r
ggsave("point_differential_density.pdf")
```

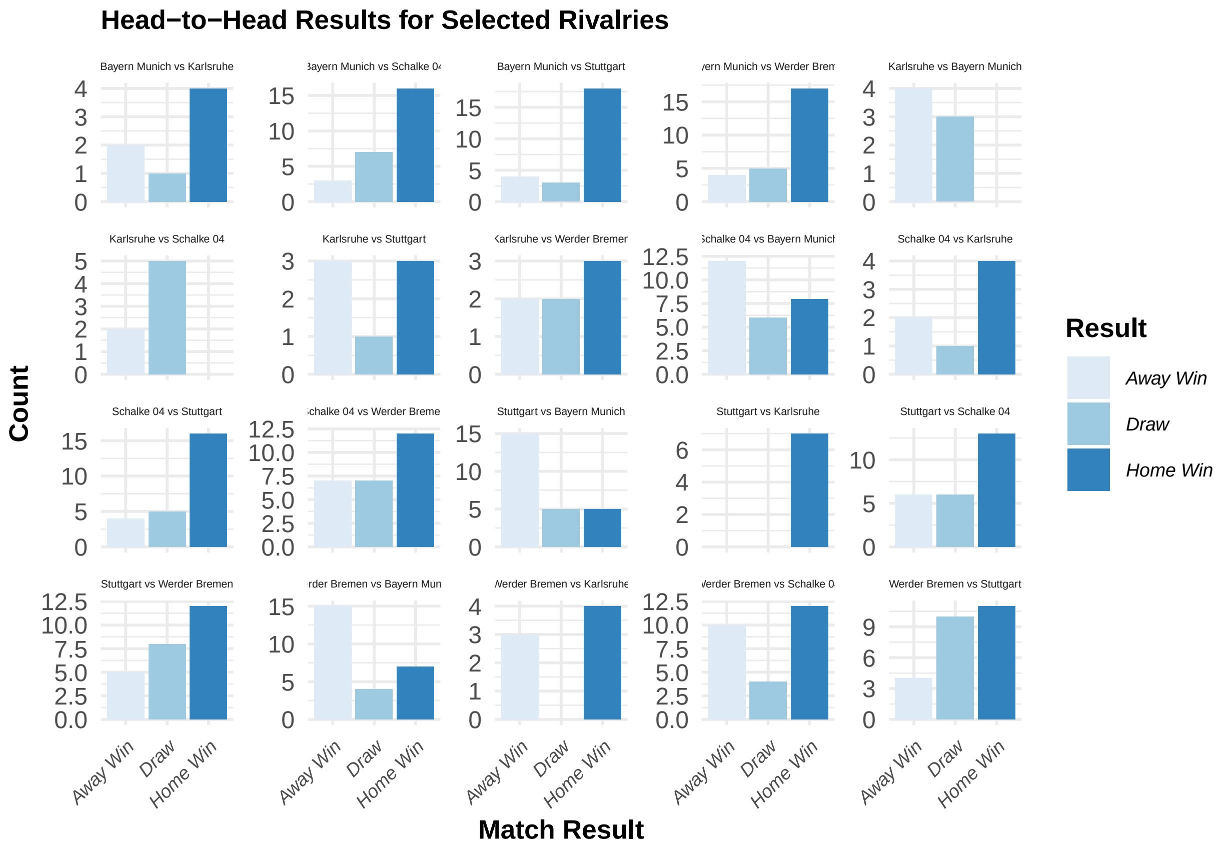
```
## Saving 6.5 x 4.5 in image
```

Part 4. Rivalries & Big Match Patterns

1. Head-to-Head Matrix for Selected Rivalries

```r
rivalires <- list(
  c("Bayern Munich", "Borussia Dortmund"),
  c("Schalke 04", "Borussia Dortmund"),
  c("Hamburger SV", "Werder Bremen"),
  c("Stuttgart", "Karlsruhe"),
  c("Cologne", "Borussia Monchengladbach")
)

rivalry_matches <- bundesliga %>%
  filter((HOMETEAM %in% unlist(rivalires) & AWAYTEAM %in% unlist(rivalires))) %>%
  mutate(Result = case_when(
    FTHG > FTAG ~ "Home Win",
    FTHG < FTAG ~ "Away Win",
    TRUE ~ "Draw"
  ))
```

```r
rivalry_matches$Matchup <- paste(rivalry_matches$HOMETEAM, "vs", rivalry_matches$AWAYTEAM)

ggplot(rivalry_matches, aes(x = Result, fill = Result)) +
  geom_bar() +
  facet_wrap(~Matchup, scales = "free_y") +
  scale_fill_brewer(palette = "Blues") +
  theme_minimal() +
  theme(plot.title = element_text(size = 10, face = "bold"),
        strip.text = element_text(size = 4),
        axis.text.x = element_text(size = 7, angle = 45, hjust = 1, face = "italic"),
        axis.title = element_text(size = 10, face = "bold"),
        legend.title = element_text(size = 10, face = "bold"),
        legend.text = element_text(size = 7, face = "italic")) +
  labs(title = "Head-to-Head Results for Selected Rivalries", x = "Match Result", y = "Count")
```

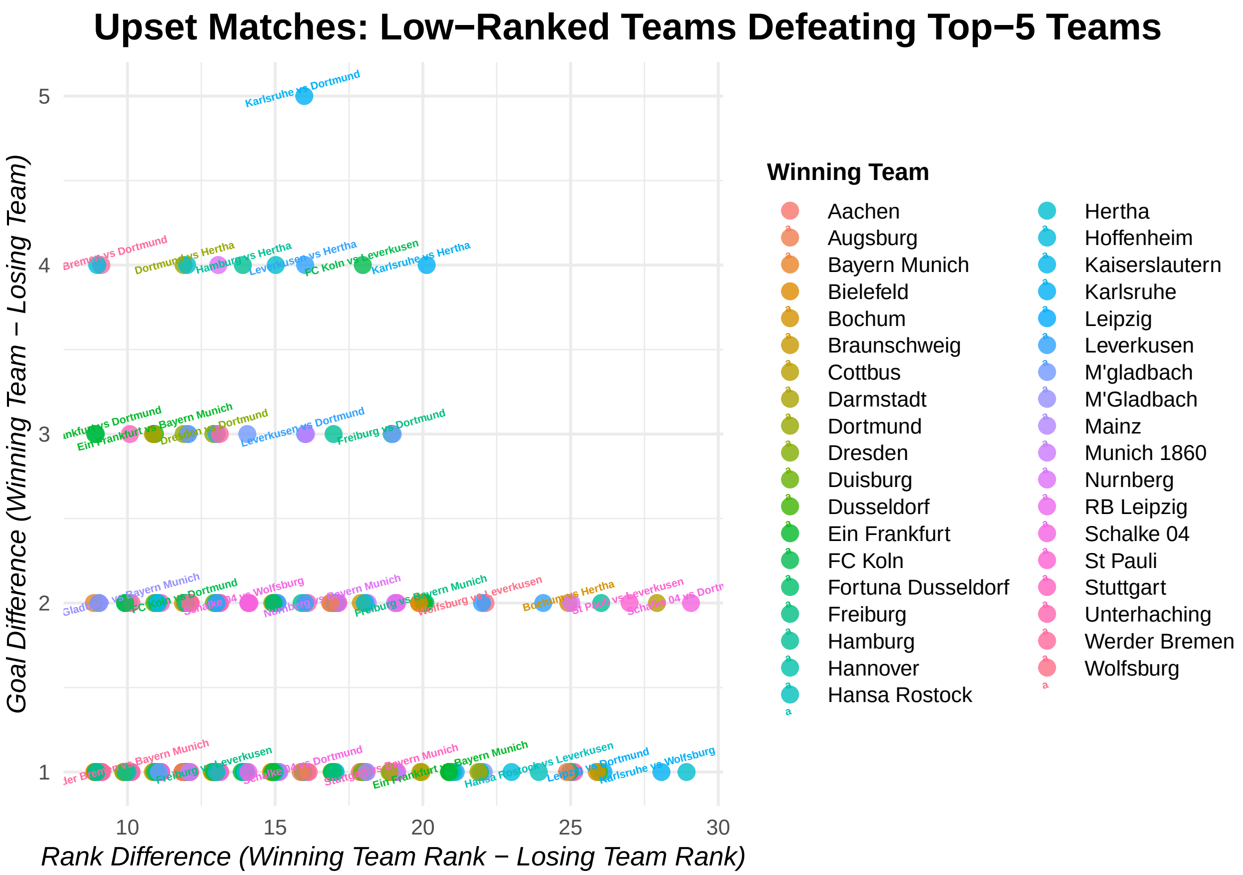


```r
ggsave("rivalry_results.pdf")
```

```
## Saving 6.5 x 4.5 in image
```

2. Upset Visualizer

```r
team_rankings <- bundesliga %>%
  group_by(SEASON, HOMETEAM) %>%
  summarise(
    Points = sum(3 * (FTHG > FTAG) + (FTHG == FTAG), na.rm = TRUE),
    .groups = "drop"
  ) %>%
  arrange(SEASON, desc(Points)) %>%
  mutate(Rank = dense_rank(desc(Points)))

top_5_teams <- team_rankings %>%
  filter(Rank <= 5) %>%
  select(SEASON, HOMETEAM)

match_data <- bundesliga %>%
  left_join(team_rankings, by = c("SEASON", "HOMETEAM")) %>%
  rename(HomeRank = Rank) %>%
  left_join(team_rankings, by = c("SEASON", "AWAYTEAM" = "HOMETEAM")) %>%
  rename(AwayRank = Rank)

upsets <- match_data %>%
  filter(
    (HomeRank > AwayRank + 8 & FTHG > FTAG & AWAYTEAM %in% top_5_teams$HOMETEAM) |
    (AwayRank > HomeRank + 8 & FTHG < FTAG & HOMETEAM %in% top_5_teams$HOMETEAM)
  ) %>%
  mutate(
    RankDifference = abs(HomeRank - AwayRank),
    GoalDifference = abs(FTHG - FTAG),
    WinningTeam = ifelse(FTHG > FTAG, HOMETEAM, AWAYTEAM),
    LosingTeam = ifelse(FTHG > FTAG, AWAYTEAM, HOMETEAM)
  )

famous_upsets <- upsets %>%
  filter(GoalDifference >= 3 | RankDifference >= 10)

ggplot(upsets, aes(x = RankDifference, y = GoalDifference, color = WinningTeam)) +
  geom_point(position = position_dodge(width = 0.3), alpha = 0.8, size = 2.5) +
  geom_text(data = famous_upsets, aes(label = paste(WinningTeam, "vs", LosingTeam)),
            hjust = 0.5, vjust = -0.5, size = 1.5, angle = 15, fontface = "bold", check_overlap = TRUE)
  labs(
    title = "Upset Matches: Low-Ranked Teams Defeating Top-5 Teams",
    x = "Rank Difference (Winning Team Rank - Losing Team Rank)",
    y = "Goal Difference (Winning Team - Losing Team)",
    color = "Winning Team"
  ) +
  theme_minimal() +
  theme(
    legend.position = "right",
    legend.text = element_text(size = 8),
    legend.key.height = unit(0.7, "lines"),
    legend.title = element_text(size = 9, face = "bold"),
    plot.title = element_text(size = 14, face = "bold", hjust = 0.5),
    plot.title.position = "plot",
    plot.margin = margin(1, 1, 1, 1),
    axis.title = element_text(size = 10, face = "italic"),
    axis.text = element_text(size = 8)
  )
```



```
  #scale_color_discrete(labels = function(x) gsub("a", "", x))  I tried to remove
  #"a" from the legend part (I don't know why I have this tho), but this couldn't
  #help me to do so.

ggsave("upset_visualizer.pdf", width = 10, height = 6)
```

Part 5. Overall performance

```
season_points <- data.frame()

for (match in 1:nrow(bundesliga)) {
  home_row <- data.frame(
    SEASON = bundesliga$SEASON[match],
    Team = bundesliga$HOMETEAM[match],
    Points = ifelse(bundesliga$FTHG[match] > bundesliga$FTAG[match], 3,
                    ifelse(bundesliga$FTHG[match] == bundesliga$FTAG[match], 1, 0))
  )

  away_row <- data.frame(
```

```r
    SEASON = bundesliga$SEASON[match],
    Team = bundesliga$AWAYTEAM[match],
    Points = ifelse(bundesliga$FTHG[match] < bundesliga$FTAG[match], 3,
                    ifelse(bundesliga$FTHG[match] == bundesliga$FTAG[match], 1, 0))
  )

  season_points <- rbind(season_points, home_row, away_row)
}

season_points <- season_points %>%
  group_by(SEASON, Team) %>%
  summarise(TotalPoints = sum(Points, na.rm = TRUE), .groups = "drop") %>%
  arrange(SEASON, desc(TotalPoints))

team_colors <- season_points %>%
  distinct(SEASON, Team) %>%
  group_by(SEASON) %>%
  mutate(Color = scales::hue_pal()(n()))

season_points <- season_points %>%
  left_join(team_colors, by = c("SEASON", "Team"))

season_winners <- season_points %>%
  group_by(SEASON) %>%
  slice_max(order_by = TotalPoints, n = 1) %>%
  ungroup()

plot_list <- list()
seasons <- unique(season_points$SEASON)

for (s in seasons) {
  season_data <- season_points %>% filter(SEASON == s)
  winner <- season_winners %>% filter(SEASON == s)

  p <- ggplot(season_data, aes(x = reorder(Team, TotalPoints), y = TotalPoints, fill = Team)) +
    geom_col() +
    geom_col(data = winner, aes(fill = Team), color = "black", linewidth = 1) +
    coord_flip() +
    scale_fill_manual(values = setNames(season_data$Color, season_data$Team)) +
    labs(title = paste("Season", s, "Total Points"), x = "Team", y = "Points") +
    theme_minimal() +
    theme(legend.position = "none")

  plot_list[[as.character(s)]] <- p
}

pdf("seasonal_team_points.pdf", width = 10, height = 6)
for (p in plot_list) {
  print(p)
}
dev.off()
```

```
## pdf
## 2
```

Part 6. Monte Carlo simulation

```r
unique(bundesliga$HOMETEAM)
```

```
##  [1] "Bayern Munich"      "Dortmund"           "Duisburg"
##  [4] "FC Koln"            "Hamburg"            "Leipzig"
##  [7] "M’Gladbach"         "Wattenscheid"       "Werder Bremen"
## [10] "Dresden"            "Ein Frankfurt"      "Freiburg"
## [13] "Kaiserslautern"     "Karlsruhe"          "Leverkusen"
## [16] "Nurnberg"           "Schalke 04"         "Stuttgart"
## [19] "Uerdingen"          "Bochum"             "Munich 1860"
## [22] "M’gladbach"         "Hansa Rostock"      "St Pauli"
## [25] "Dusseldorf"         "Bielefeld"          "Hertha"
## [28] "Wolfsburg"          "Ulm"                "Unterhaching"
## [31] "Cottbus"            "Hannover"           "Mainz"
## [34] "Aachen"             "Hoffenheim"         "Augsburg"
## [37] "Greuther Furth"     "Fortuna Dusseldorf" "Braunschweig"
## [40] "Paderborn"          "Darmstadt"          "Ingolstadt"
## [43] "RB Leipzig"
```

```r
unique(bundesliga$AWAYTEAM)
```

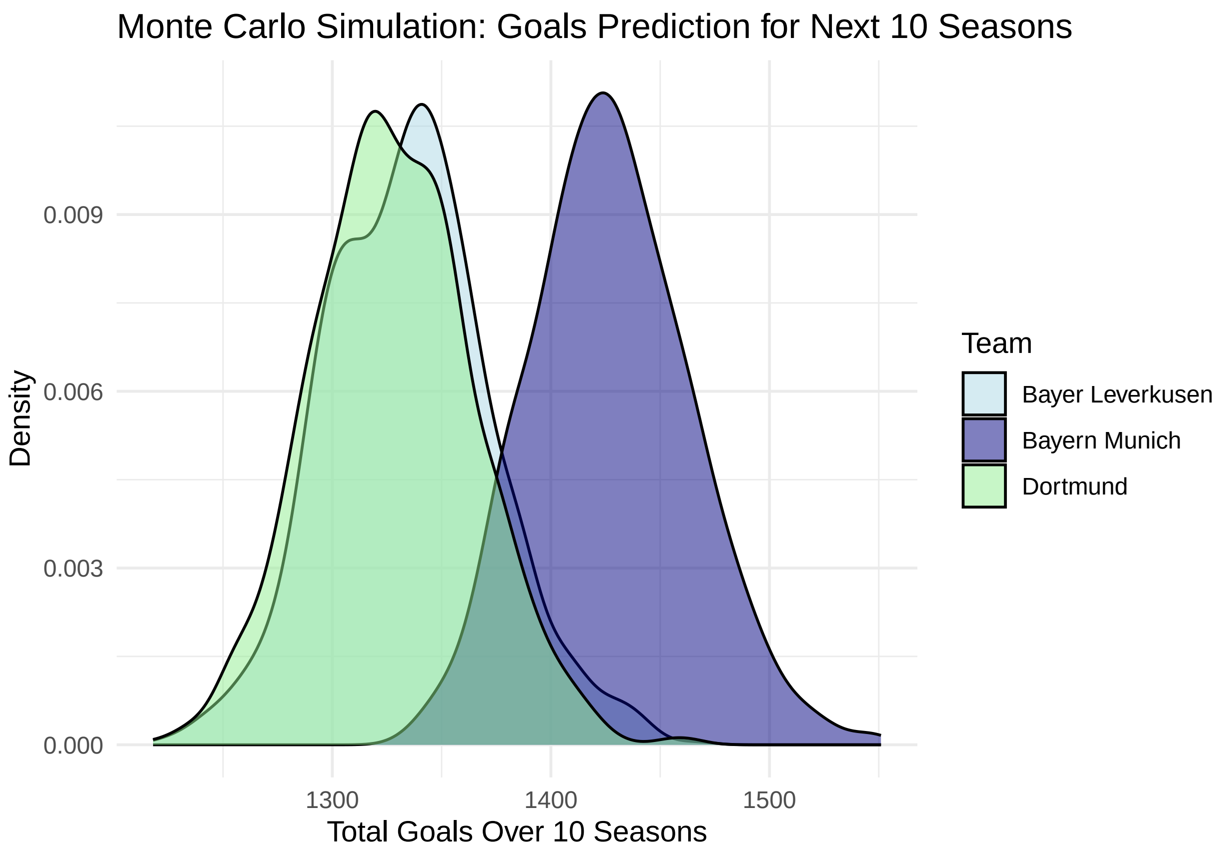
```
##  [1] "Freiburg"           "Karlsruhe"          "Leverkusen"
##  [4] "Kaiserslautern"     "Nurnberg"           "Dresden"
##  [7] "Ein Frankfurt"      "Schalke 04"         "Stuttgart"
## [10] "Duisburg"           "Werder Bremen"      "Wattenscheid"
## [13] "M’Gladbach"         "Hamburg"            "Bayern Munich"
## [16] "FC Koln"            "Dortmund"           "Leipzig"
## [19] "M’gladbach"         "Bochum"             "Munich 1860"
## [22] "Uerdingen"          "Dusseldorf"         "St Pauli"
## [25] "Hansa Rostock"      "Bielefeld"          "Wolfsburg"
## [28] "Hertha"             "Unterhaching"       "Ulm"
## [31] "Cottbus"            "Hannover"           "Mainz"
## [34] "Aachen"             "Hoffenheim"         "Augsburg"
## [37] "Fortuna Dusseldorf" "Greuther Furth"     "Braunschweig"
## [40] "Paderborn"          "Ingolstadt"         "Darmstadt"
## [43] "RB Leipzig"
```

I do this preprocessing step, to see whether I have unique teams playing for bundesliga. As I noticed the data contains *"Bayern Munich"* and *"Bayern Munchen"* names, which as far as my german knowledge gives the knowledge, they are the same. Also for *Dortmund* and *Leverkusen* I change the format (and for Dortmund I consider all the teams that contain "Dortmund")

```r
bundesliga_clean <- bundesliga %>%
  mutate(
    HOMETEAM = case_when(
      HOMETEAM == "Bayern Munchen" ~ "Bayern Munich",
      HOMETEAM == "Borussia Dortmund" ~ "Dortmund",
      HOMETEAM == "Leverkusen" ~ "Bayer Leverkusen",
      TRUE ~ HOMETEAM
    ),
    AWAYTEAM = case_when(
      AWAYTEAM == "Bayern Munchen" ~ "Bayern Munich",
      AWAYTEAM == "Borussia Dortmund" ~ "Dortmund",
      AWAYTEAM == "Leverkusen" ~ "Bayer Leverkusen",
      TRUE ~ AWAYTEAM
    )
  )
```

here I do the conversion process

```r
set.seed(1)

simulate_goals <- function(team_name, seasons = 10, simulations = 1000) {
  past_goals <- bundesliga_clean %>%
    filter(HOMETEAM == team_name | AWAYTEAM == team_name) %>%
    summarise(avg_goals_home = mean(FTHG[FTHG > 0]),
              avg_goals_away = mean(FTAG[FTAG > 0], na.rm = TRUE)) %>%
    mutate(avg_goals = avg_goals_home + avg_goals_away) %>%
    pull(avg_goals)

  sim_results <- replicate(simulations, sum(rpois(seasons * 34, lambda = past_goals)))

  data.frame(Team = team_name, Goals = sim_results)
}

bayern_goals <- simulate_goals("Bayern Munich")
leverkusen_goals <- simulate_goals("Bayer Leverkusen")
dortmund_goals <- simulate_goals("Dortmund")

goal_predictions <- bind_rows(bayern_goals, leverkusen_goals, dortmund_goals)

goal_predictions_clean <- goal_predictions %>%
  filter(is.finite(Goals))

ggplot(goal_predictions_clean, aes(x = Goals, fill = Team)) +
  geom_density(alpha = 0.5) +
  theme_minimal() +
  labs(title = "Monte Carlo Simulation: Goals Prediction for Next 10 Seasons",
       x = "Total Goals Over 10 Seasons", y = "Density") +
  scale_fill_manual(values = c("lightblue", "navy", "lightgreen"))
```



Now, coming to the insights I get from monte carlo simulation above. Let's break to 3 points (teams) and analyze separately all three of them:

1. Bayer Leverkusen - the goals are concentrated in 1320-1340 goals, and it overlaps with Dortmund which suggests that Dortmund as well will accept the same-ish number of goals.

2. Bayern Munich - the predicted goal is centered at 1430-1440 goals, which is by the way the highest centered distirbution out of all three teams. we can notice, that it is wider than the others, which means that the variability of the predicted goals is diverse as well. So, Bayern Munich is expected to have the highest number of goals.

3. Dortmund - centered at 1310-1315 and it is lowest centered graph, which suggests that Dortmund will expect the lowest number of goals during the next 10 seasons.