Homework 4

Lilit Ivanyan

2025-03-29

Part 1: Trend Analysis

1. Analyse trend of goals per season. For example total goals per match, average goals per match.

```
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
```

```
library(tidyr)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths
```

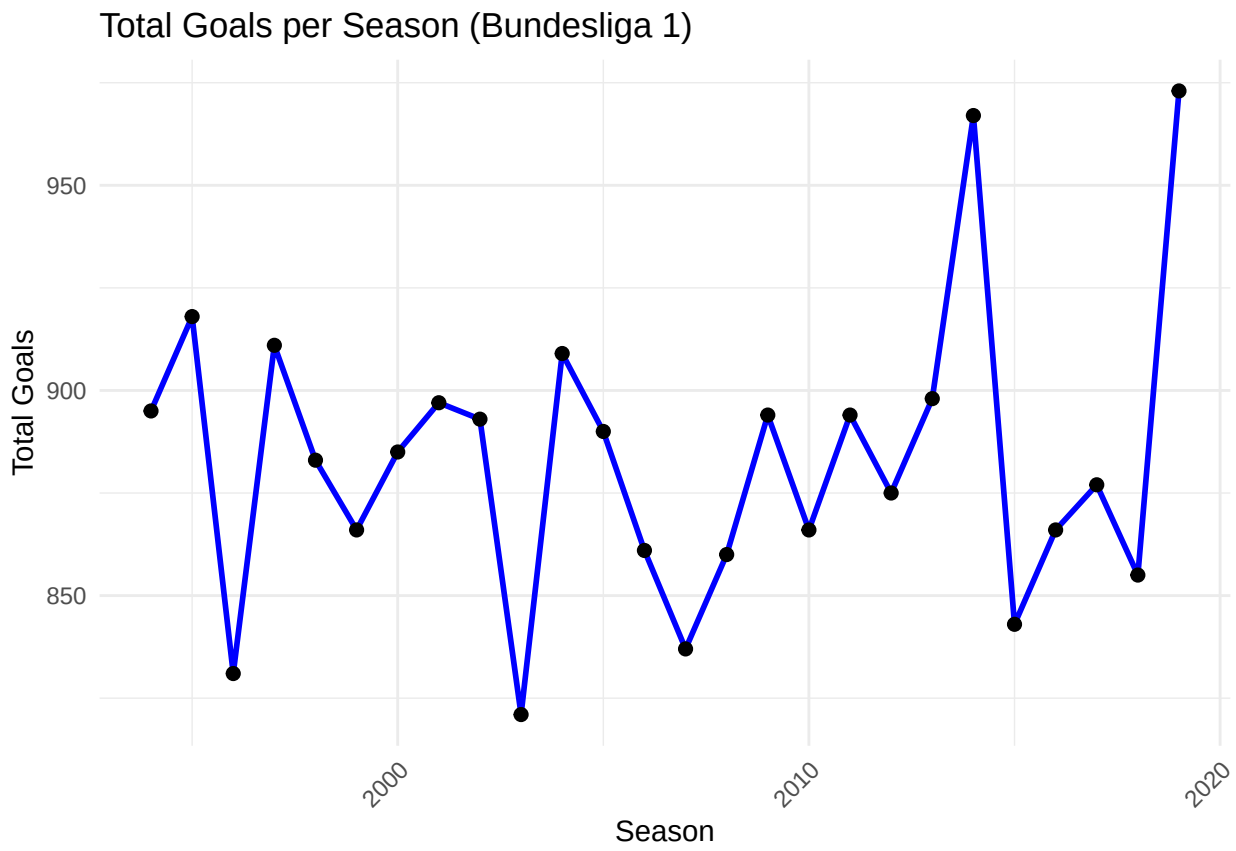
```
bundesliga <- read.csv("bundesliga.csv")
head(bundesliga)
```

```
##   SEASON       LEAGUE       DATE      HOMETEAM       AWAYTEAM FTSC FTHG FTAG
## 1   1994 Bundesliga 1 1993-08-07 Bayern Munich       Freiburg  3-1    3    1
## 2   1994 Bundesliga 1 1993-08-07      Dortmund      Karlsruhe  2-1    2    1
## 3   1994 Bundesliga 1 1993-08-07      Duisburg     Leverkusen  2-2    2    2
## 4   1994 Bundesliga 1 1993-08-07       FC Koln Kaiserslautern  0-2    0    2
## 5   1994 Bundesliga 1 1993-08-07       Hamburg       Nurnberg  5-2    5    2
## 6   1994 Bundesliga 1 1993-08-07       Leipzig        Dresden  3-3    3    3
```

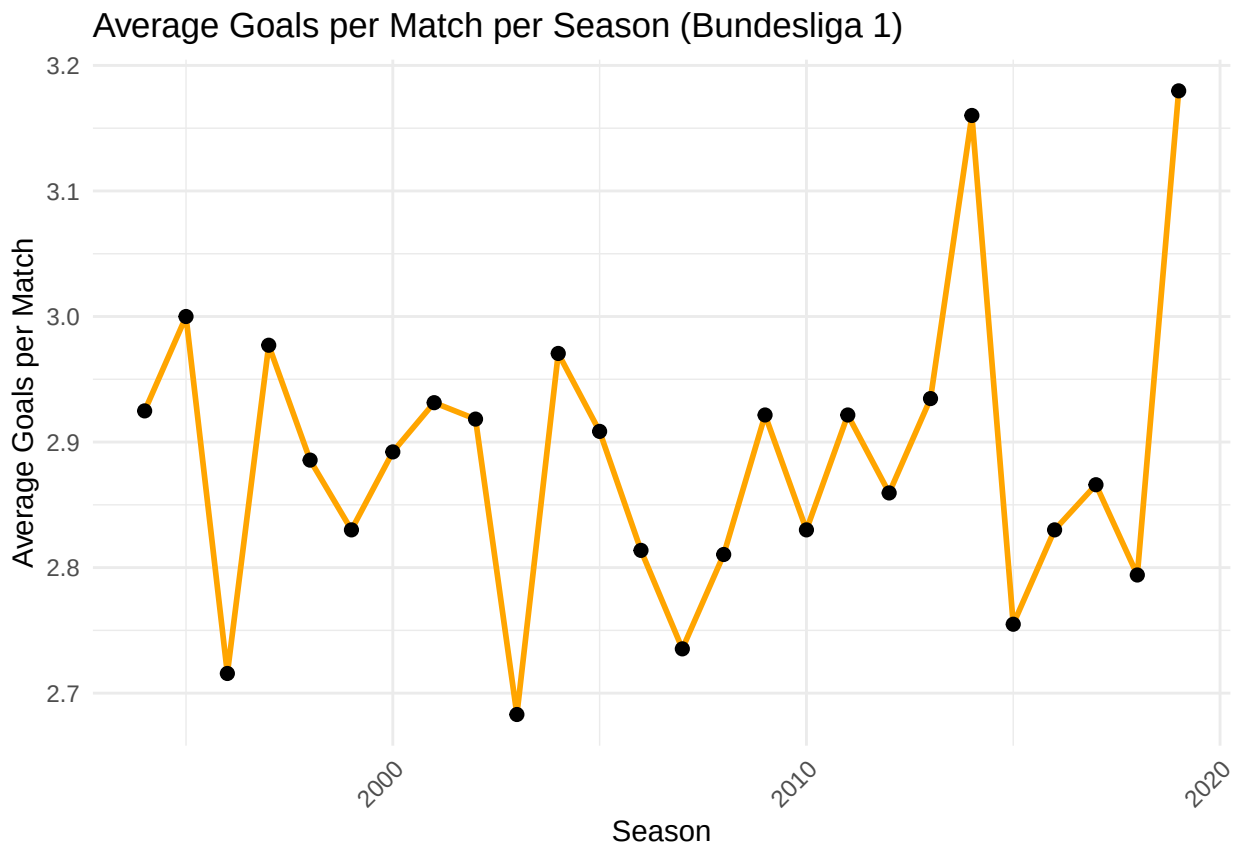
```
##   FTTG
## 1    4
## 2    3
## 3    4
## 4    2
## 5    7
## 6    6
```

```r
goals_per_season <- aggregate(FTTG ~ SEASON, data = bundesliga, sum)
avg_goals_per_match <- aggregate(FTTG ~ SEASON, data = bundesliga, mean)

ggplot(goals_per_season, aes(x = SEASON, y = FTTG)) +
  geom_line(color = "blue", size = 1) +
  geom_point(size = 2) +
  ggtitle("Total Goals per Season (Bundesliga 1)") +
  xlab("Season") +
  ylab("Total Goals") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

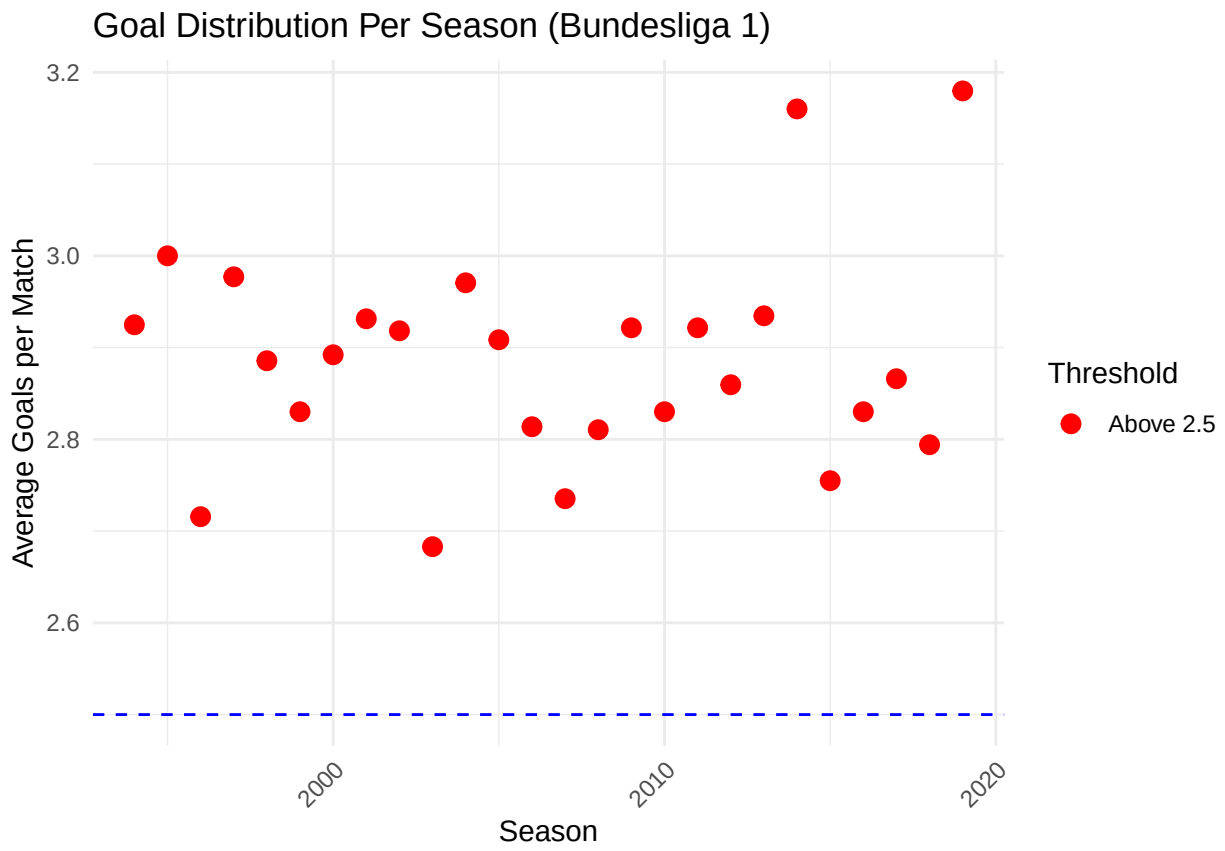


```r
ggplot(avg_goals_per_match, aes(x = SEASON, y = FTTG)) +
  geom_line(color = "orange", size = 1) +
  geom_point(size = 2) +
  ggtitle("Average Goals per Match per Season (Bundesliga 1)") +
  xlab("Season") +
  ylab("Average Goals per Match") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



2. Goal Distribution Per Season. Use appropriate type of graphs for goals per match, year-wise. Colorcode by whether average is above or below 2.5 (over/under bet threshold).

```r
avg_goals_per_match <- aggregate(FTTG ~ SEASON, data = bundesliga, mean)

avg_goals_per_match$Above2.5 <- avg_goals_per_match$FTTG > 2.5

ggplot(avg_goals_per_match, aes(x = SEASON, y = FTTG, color = Above2.5)) +
  geom_point(size = 3) +
  geom_hline(yintercept = 2.5, linetype = "dashed", color = "blue") +
  scale_color_manual(values = c("red", "green"), labels = c("Above 2.5", "Under 2.5")) +
  labs(title = "Goal Distribution Per Season (Bundesliga 1)",
       x = "Season", y = "Average Goals per Match", color = "Threshold") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



3. Create line charts for each season. Visualize trend of goals for each team that played in that season. Highlight only Bayern Munchen with red color. Rest should be gray. Add appropriate title that will contain information about season and total scored goals. Add footnote mentioning total number of goals scored by Bayern Munchen for that season.

```
visualize_goals_per_season <- function(season) {
    season_data <- bundesliga[bundesliga$SEASON == season, ]

    home_goals <- aggregate(FTHG ~ HOMETEAM, data = season_data, sum)
    away_goals <- aggregate(FTAG ~ AWAYTEAM, data = season_data, sum)
    colnames(home_goals) <- c('Team', 'Goals')
    colnames(away_goals) <- c('Team', 'Goals')
    team_goals <- rbind(home_goals, away_goals)
    team_goals <- aggregate(Goals ~ Team, data = team_goals, sum)

    team_goals$Color <- ifelse(team_goals$Team == "Bayern Munich", "red", "gray")

    total_goals <- sum(team_goals$Goals)
    bayern_goals <- if ("Bayern Munich" %in% team_goals$Team) {
        team_goals$Goals[team_goals$Team == "Bayern Munich"]
    } else {
        0
    }

    ggplot(team_goals, aes(x = reorder(Team, Goals), y = Goals, color = Color)) +
```

```
        geom_line(aes(group = 1), size = 1, alpha = 0.8) +
        geom_point(size = 3) +
        scale_color_manual(values = c("gray", "red")) +
        labs(title = paste(season, "Bundesliga Goals Trend (Total Goals:", total_goals, ")"),
             x = "Team",
             y = "Goals",
             color = "Team Highlight") +
        theme_minimal() +
        theme(axis.text.x = element_text(angle = 90, hjust = 1),
              plot.title = element_text(size = 16, face = "bold"),
              plot.caption = element_text(size = 10),
              legend.position = "none") +
        annotate(
            "text",
            x = 3,
            y = 0,
            label = paste("Total goals scored by Bayern Munich:", bayern_goals),
            hjust = -0.3,
            vjust = 1.5,
            size = 4,
            color = "black",
            fontface = "italic"
        )
}

output_pdf <- "part1-3-r.pdf"
pdf(file = output_pdf, width = 12, height = 6)

unique_seasons <- unique(bundesliga$SEASON)
for (season in unique_seasons) {
    print(visualize_goals_per_season(season))
}

dev.off()
```

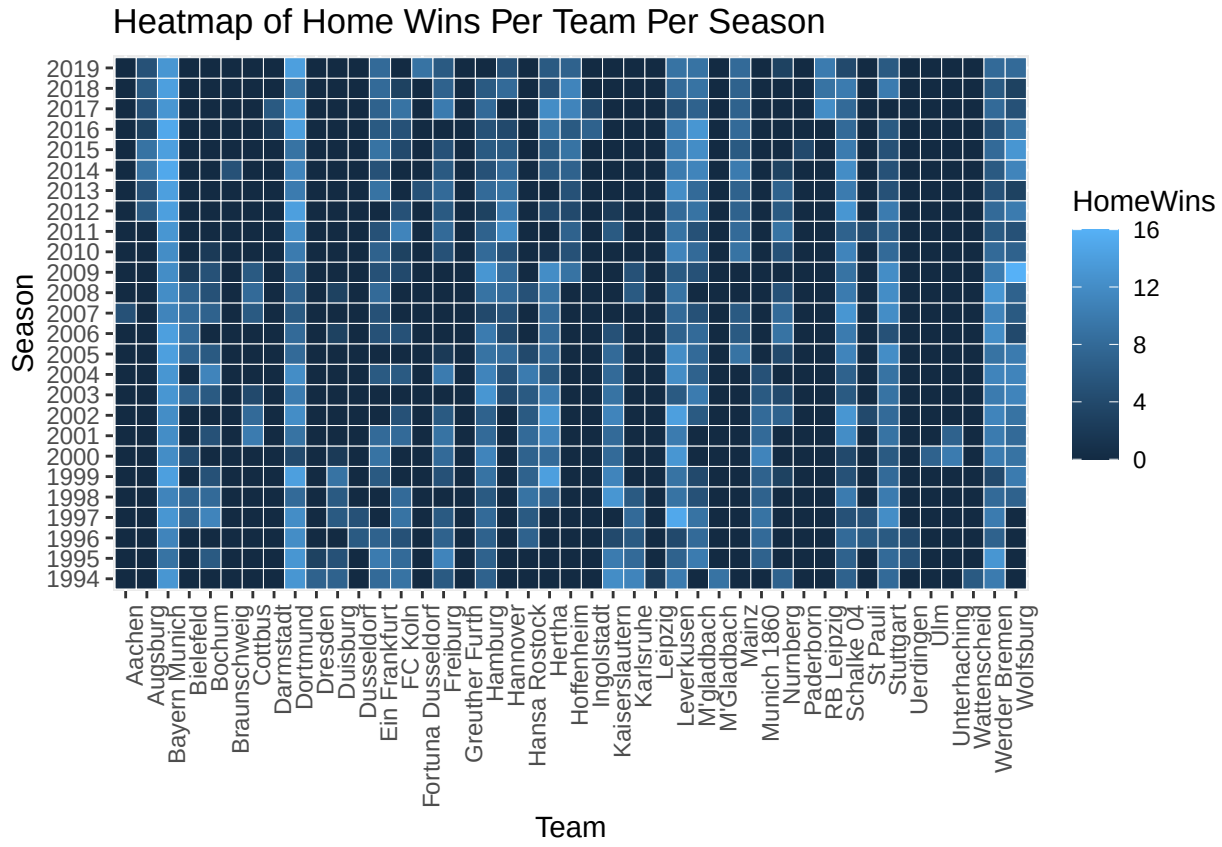
```
## pdf
##   2
```

Part 2 Home Advantage Deconstructed

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

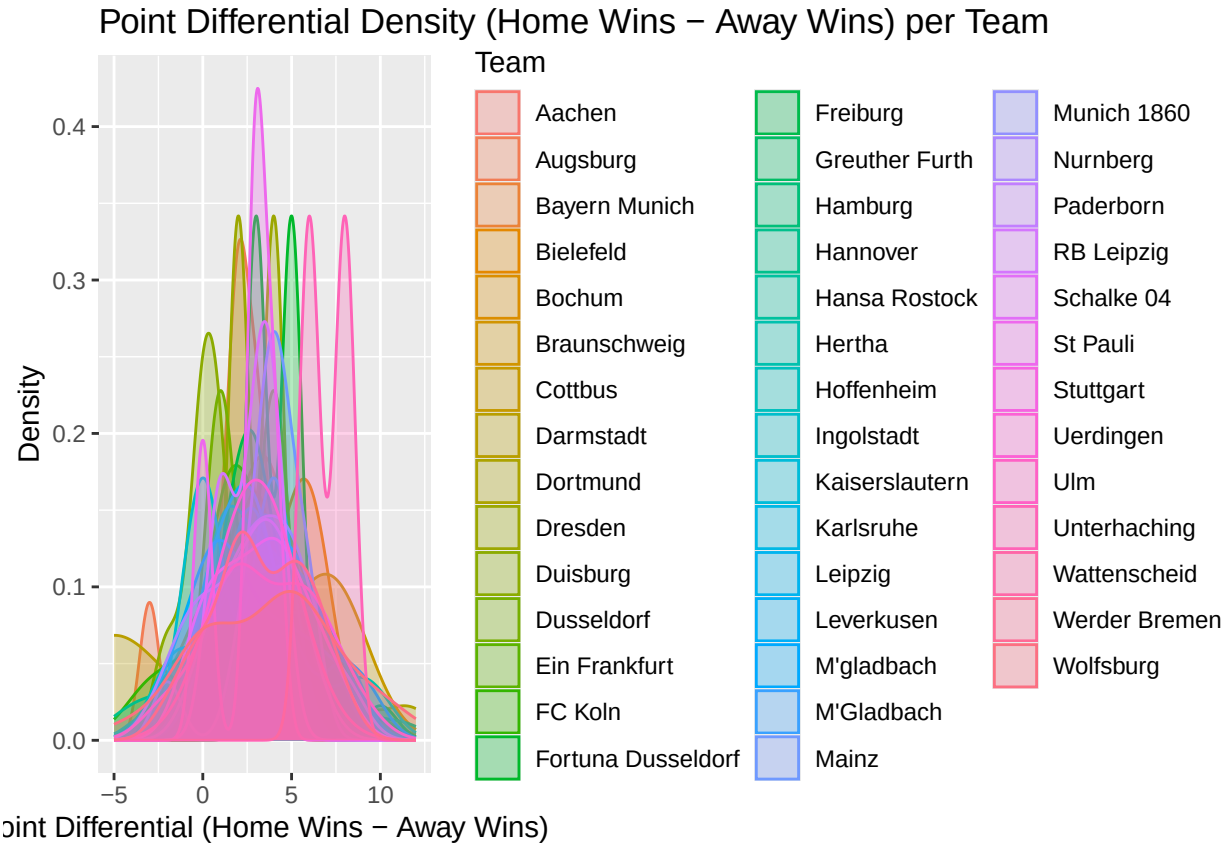
```
process_wins <- function(df) {
  home_wins <- df %>%
    filter(as.numeric(sub("-.*", "", FTSC)) > as.numeric(sub(".*-", "", FTSC))) %>%
    group_by(SEASON, HOMETEAM) %>%
    summarise(HomeWins = n(), .groups = 'drop')

  away_wins <- df %>%
    filter(as.numeric(sub(".*-", "", FTSC)) > as.numeric(sub("-.*", "", FTSC))) %>%
    group_by(SEASON, AWAYTEAM) %>%
    summarise(AwayWins = n(), .groups = 'drop')
```

```r
  colnames(home_wins) <- c("SEASON", "Team", "HomeWins")
  colnames(away_wins) <- c("SEASON", "Team", "AwayWins")

  wins <- full_join(home_wins, away_wins, by = c("SEASON", "Team")) %>%
    replace_na(list(HomeWins = 0, AwayWins = 0))
  return(wins)
}

wins <- process_wins(bundesliga)

wins_matrix <- dcast(wins, SEASON ~ Team, value.var = "HomeWins", fill = 0)

wins_long <- melt(wins_matrix, id.var = "SEASON", variable.name = "Team", value.name = "HomeWins")

ggplot(wins_long, aes(x = Team, y = as.factor(SEASON), fill = HomeWins)) +
  geom_tile(color = "white") +
  labs(title = "Heatmap of Home Wins Per Team Per Season",
       x = "Team",
       y = "Season") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



2. Point Differential Density: Create visualizations that will show difference per team for home and away game wins

```r
process_point_differential <- function(df) {
  home_wins <- df %>%
    filter(as.numeric(sub("-.*", "", FTSC)) > as.numeric(sub(".*-", "", FTSC))) %>%
    group_by(SEASON, HOMETEAM) %>%
    summarise(HomeWins = n(), .groups = 'drop')

  away_wins <- df %>%
    filter(as.numeric(sub(".*-", "", FTSC)) > as.numeric(sub("-.*", "", FTSC))) %>%
    group_by(SEASON, AWAYTEAM) %>%
    summarise(AwayWins = n(), .groups = 'drop')

  home_wins <- rename(home_wins, Team = HOMETEAM)
  away_wins <- rename(away_wins, Team = AWAYTEAM)

  wins <- full_join(home_wins, away_wins, by = c("SEASON", "Team")) %>%
    replace_na(list(HomeWins = 0, AwayWins = 0))

  wins <- wins %>%
    mutate(PointDifferential = HomeWins - AwayWins)

  return(wins)
}

wins <- process_point_differential(bundesliga)

ggplot(wins, aes(x = PointDifferential, color = Team, fill = Team)) +
  geom_density(alpha = 0.3) +
  labs(title = "Point Differential Density (Home Wins - Away Wins) per Team",
       x = "Point Differential (Home Wins - Away Wins)",
       y = "Density")
```



Part 3

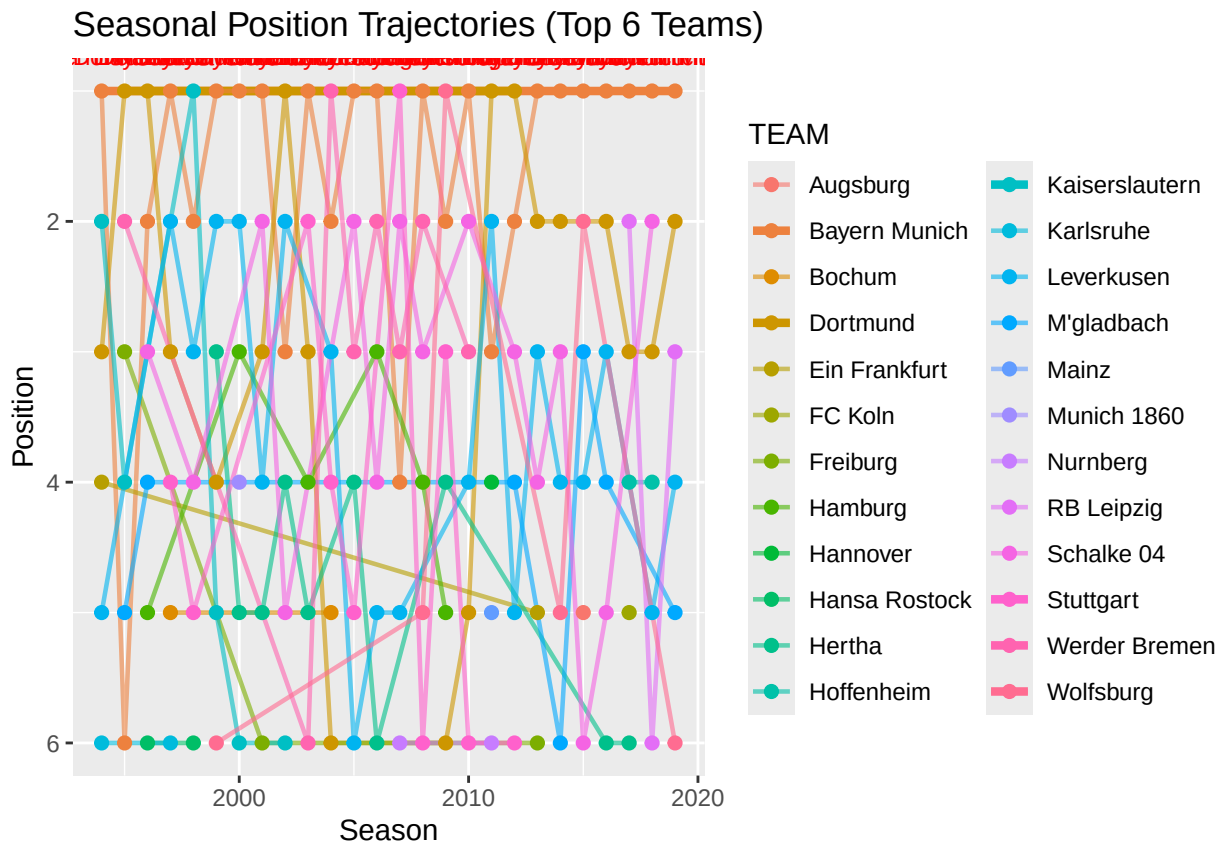
1. Team Trajectories and Volatility • Seasonal Position Trajectories • Line plots showing seasonal ranks for top 6 teams. • Annotate title-winning seasons.

```
bundesliga2 <- read.csv("bundesliga2.csv" )
head(bundesliga2)
```

```
##             TEAM  M  W  D  L GF GA DIFF POINTS POSITION SEASON
## 1  Bayern Munich 34 17 10  7 68 37   31     61        1   1994
## 2 Kaiserslautern 34 18  7  9 64 36   28     61        2   1994
## 3       Dortmund 34 15  9 10 49 45    4     54        3   1994
## 4  Ein Frankfurt 34 15  8 11 57 41   16     53        4   1994
## 5     Leverkusen 34 14 11  9 60 47   13     53        5   1994
## 6      Karlsruhe 34 14 10 10 46 43    3     52        6   1994
```

```
top_6_teams <- bundesliga2 %>% filter(POSITION <= 6)
title_winning_teams <- bundesliga2 %>% filter(POSITION == 1)

ggplot(top_6_teams, aes(x = SEASON, y = POSITION, group = TEAM)) +
  geom_line(
    data = title_winning_teams,
    aes(color = TEAM),
    size = 1.5
```

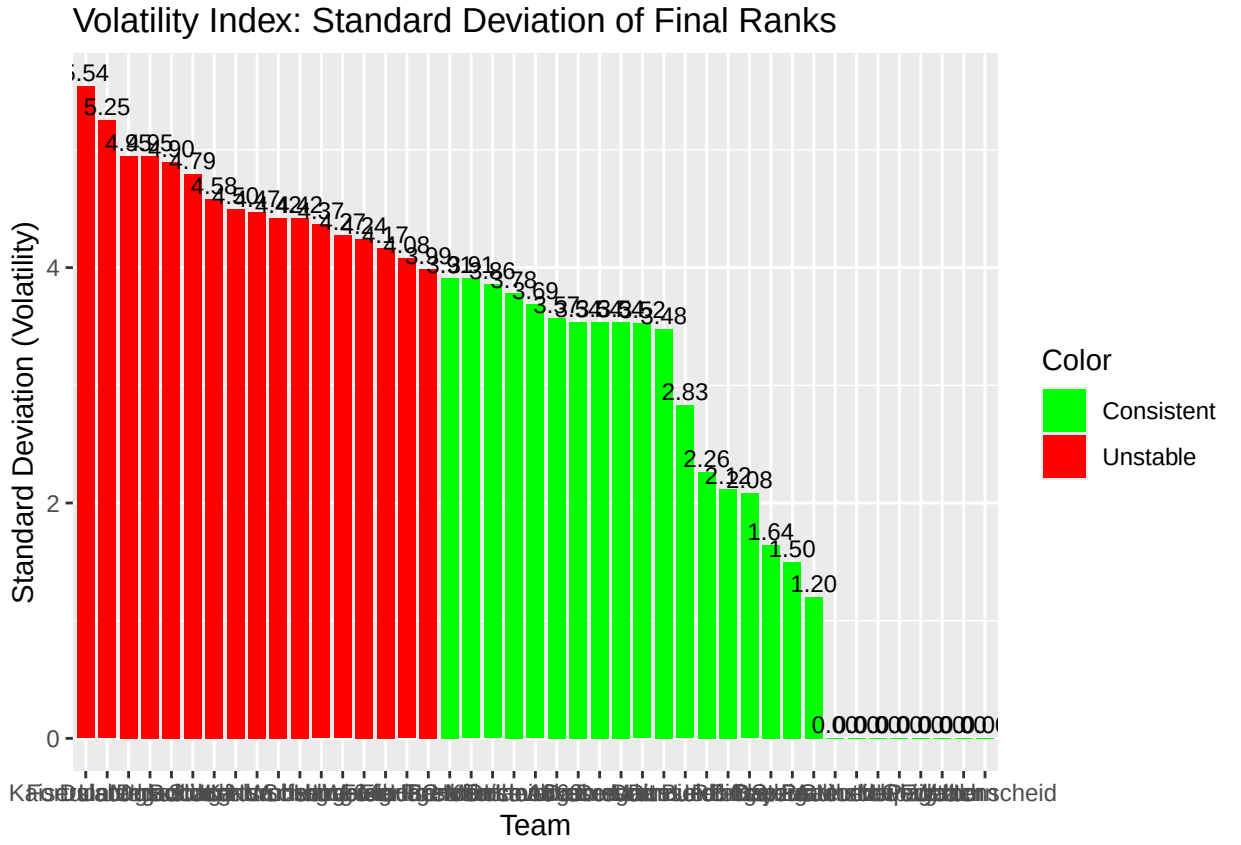
```
  ) +
  geom_line(aes(color = TEAM), size = 0.8, alpha = 0.6) +
  geom_point(aes(color = TEAM), size = 2) +
  geom_text(
    data = title_winning_teams,
    aes(label = TEAM),
    color = "red",
    size = 3,
    vjust = -1.5,
    position = position_jitter(width = 0.2, height = 0)
  ) +
  scale_y_reverse() +
  labs(
    title = "Seasonal Position Trajectories (Top 6 Teams)",
    x = "Season",
    y = "Position"
  )
```



2. Volatility Index • For each team, calculate standard deviation of final rank over all seasons. • Use a bar chart with conditional coloring (e.g., red = unstable, green = consistent). • Add text labels above each bar with exact values.

```
volatility <- bundesliga2 %>%
  group_by(TEAM) %>%
  summarise(Volatility = sd(POSITION, na.rm = TRUE)) %>%
```

```
  mutate(Color = ifelse(Volatility > median(Volatility, na.rm = TRUE), "Unstable", "Consistent"))

volatility$Volatility[is.na(volatility$Volatility)] <- 0
volatility$Color[is.na(volatility$Color)] <- "Consistent"

volatility <- volatility %>% arrange(desc(Volatility))

ggplot(volatility, aes(x = reorder(TEAM, -Volatility), y = Volatility, fill = Color)) +
  geom_bar(stat = "identity", width = 0.8) +
  scale_fill_manual(values = c("Unstable" = "red", "Consistent" = "green")) +
  geom_text(aes(label = sprintf("%.2f", Volatility)), vjust = -0.3, size = 3) +
  labs(
    title = "Volatility Index: Standard Deviation of Final Ranks",
    x = "Team",
    y = "Standard Deviation (Volatility)"
  )
```

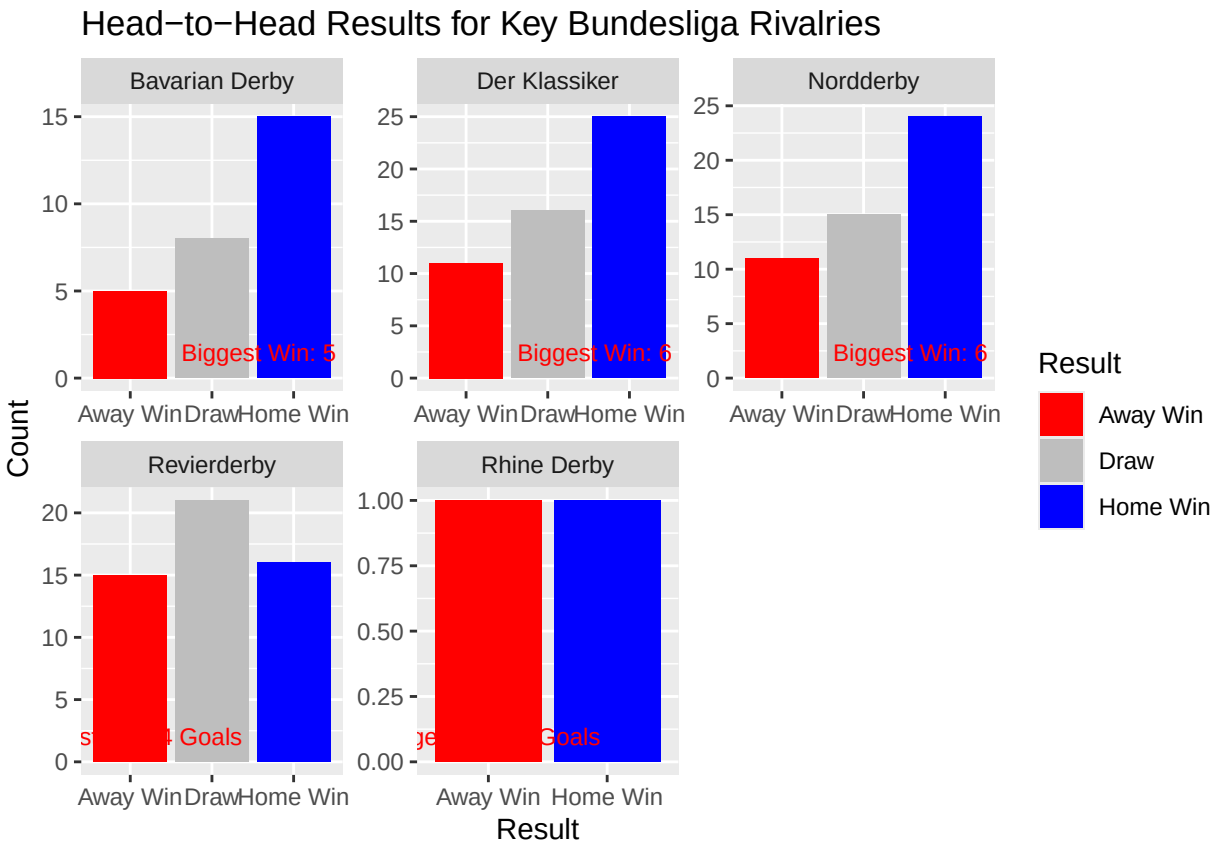


Part 4: Rivalries & Big Match Patterns

1. Head-to-Head Matrix for Selected Rivalries • Select 5 key rivalries more info click here . • Create a facet grid of win/draw/loss bar charts per rivalry. • Annotate biggest win margins.

```
rivalries <- list(
  "Der Klassiker" = c("Bayern Munich", "Dortmund"),
```

```
  "Revierderby" = c("Dortmund", "Schalke 04"),
  "Nordderby" = c("Werder Bremen", "Hamburg"),
  "Rhine Derby" = c("FC Koln", "M'Gladbach"),
  "Bavarian Derby" = c("Bayern Munich", "Nurnberg")
)

df <- bundesliga %>%
  mutate(Result = case_when(
    FTHG > FTAG ~ "Home Win",
    FTHG < FTAG ~ "Away Win",
    TRUE ~ "Draw"
  ))

rivalry_data <- NULL
for (name in names(rivalries)) {
  teams <- rivalries[[name]]
  matches <- df %>%
    filter((HOMETEAM == teams[1] & AWAYTEAM == teams[2]) |
           (HOMETEAM == teams[2] & AWAYTEAM == teams[1])) %>%
    mutate(Rivalry = name)  # Add rivalry name for grouping
  rivalry_data <- bind_rows(rivalry_data, matches)
}

rivalry_data <- rivalry_data %>%
  mutate(Goal_Diff = abs(FTHG - FTAG))

biggest_wins <- rivalry_data %>%
  group_by(Rivalry) %>%
  filter(Goal_Diff == max(Goal_Diff)) %>%
  slice(1)  # Pick one row in case of ties

ggplot(rivalry_data, aes(x = Result, fill = Result)) +
  geom_bar() +
  facet_wrap(~ Rivalry, ncol = 3, scales = "free") +
  geom_text(
    data = biggest_wins,
    aes(x = Result, y = 0, label = paste("Biggest Win:", Goal_Diff, "Goals")),
    color = "red",
    vjust = -1,
    size = 3
  ) +
  labs(
    title = "Head-to-Head Results for Key Bundesliga Rivalries",
    x = "Result",
    y = "Count"
  ) +
  scale_fill_manual(values = c("Home Win" = "blue", "Away Win" = "red", "Draw" = "gray"))
```

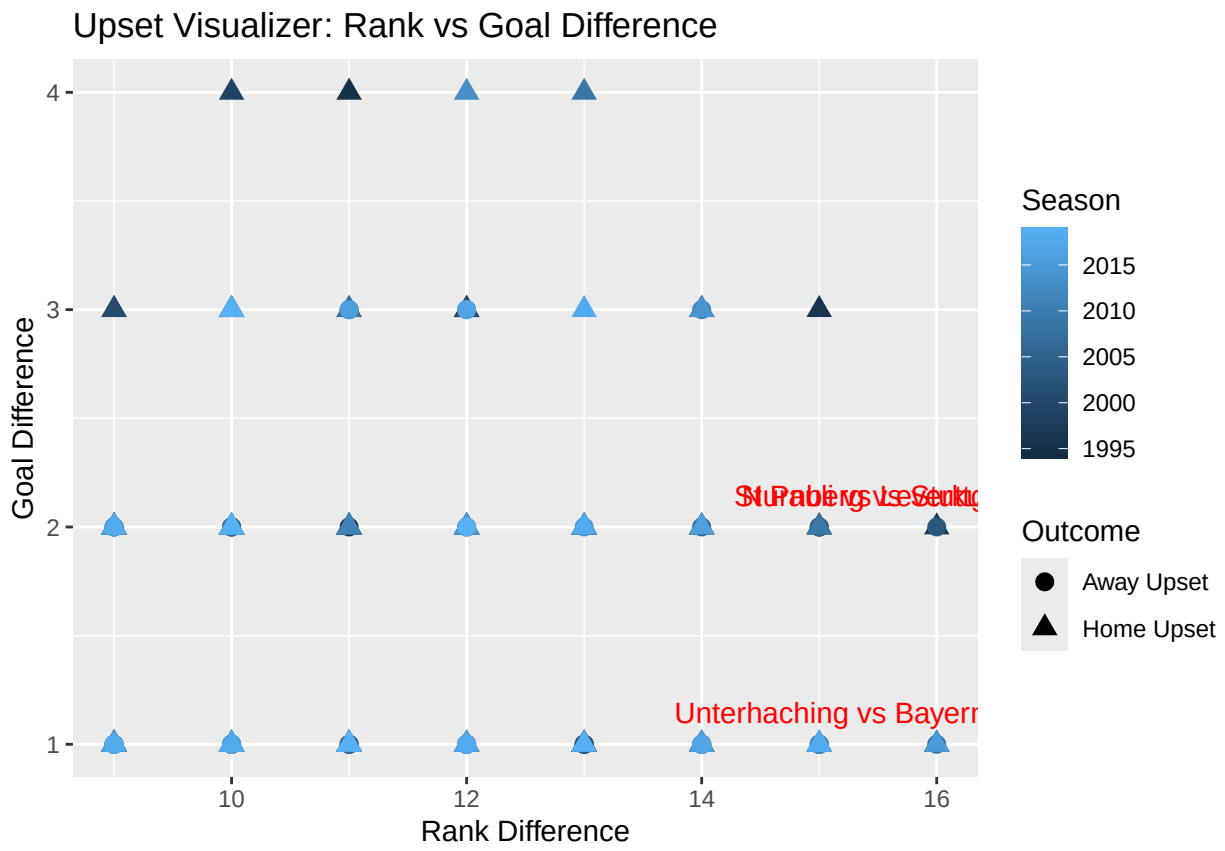


2. Upset Visualizer

- Define "upset" as a team >8 places below beating a top-5 team.
- Scatterplot of upsets: x-axis = rank difference, y-axis = goal difference.
- Encode team colors; highlight and label famous upsets

```
standings <- read.csv("bundesliga2.csv")
matches <- read.csv("bundesliga.csv")

get_top_5_teams <- function(df, season, criterion) {
  season_data <- df %>% filter(SEASON == season)
  if (criterion == "Points") {
    top_5 <- season_data %>% arrange(desc(POINTS)) %>% slice_head(n = 5)
  } else if (criterion == "Most Goals Scored") {
    top_5 <- season_data %>% arrange(desc(GF)) %>% slice_head(n = 5)
  } else if (criterion == "Fewest Goals Conceded") {
    top_5 <- season_data %>% arrange(GA) %>% slice_head(n = 5)
  } else {
    stop("Invalid criterion!")
  }
  return(top_5$TEAM)
}

filter_upsets <- function(matches, standings, criterion = "Points") {
```

```
  upsets <- list()
  for (season in unique(matches$SEASON)) {
    top_5 <- get_top_5_teams(standings, season, criterion)

    season_matches <- matches %>% filter(SEASON == season)
    for (i in 1:nrow(season_matches)) {
      match <- season_matches[i, ]
      home_rank <- standings %>%
        filter(SEASON == season, TEAM == match$HOMETEAM) %>%
        pull(POSITION)
      away_rank <- standings %>%
        filter(SEASON == season, TEAM == match$AWAYTEAM) %>%
        pull(POSITION)

      if (match$FTHG > match$FTAG & match$AWAYTEAM %in% top_5 & home_rank > away_rank + 8) {
        upsets <- append(upsets, list(data.frame(
          Season = season, Winner = match$HOMETEAM, Loser = match$AWAYTEAM,
          Rank_Difference = home_rank - away_rank, Goal_Difference = abs(match$FTHG - match$FTAG),
          Outcome = "Home Upset"
        )))
      }
      if (match$FTAG > match$FTHG & match$HOMETEAM %in% top_5 & away_rank > home_rank + 8) {
        upsets <- append(upsets, list(data.frame(
          Season = season, Winner = match$AWAYTEAM, Loser = match$HOMETEAM,
          Rank_Difference = away_rank - home_rank, Goal_Difference = abs(match$FTAG - match$FTHG),
          Outcome = "Away Upset"
        )))
      }
    }
  }
  return(bind_rows(upsets))
}

upsets <- filter_upsets(matches, standings, criterion = "Points")

famous_upsets <- upsets %>% arrange(desc(Rank_Difference)) %>% slice(1:3)

ggplot(upsets, aes(x = Rank_Difference, y = Goal_Difference, color = Season, shape = Outcome)) +
  geom_point(size = 3) +
  geom_text(data = famous_upsets, aes(label = paste(Winner, "vs", Loser, "(", Season, ")")), vjust = -1
  labs(title = "Upset Visualizer: Rank vs Goal Difference", x = "Rank Difference", y = "Goal Difference
```



Part 5: Overall performance

• Define unique color for each team per season. For each season create horizontal bar plot using total number of points. Highlighting the winner with the unique color that you assigned to it. Save all graphs in pdf.

```
teams <- unique(bundesliga2$TEAM)
team_colors <- setNames(
  scales::hue_pal()(length(teams)),
  teams
)

output_pdf <- "part5-r.pdf"
pdf(output_pdf, width = 12, height = 8)

for (season in unique(bundesliga2$SEASON)) {
  season_data <- bundesliga2 %>% filter(SEASON == season)

  winner <- season_data %>% slice_max(POINTS, n = 1) %>% pull(TEAM)

  season_data <- season_data %>%
    mutate(Color = team_colors[TEAM])

  p <- ggplot(season_data, aes(x = reorder(TEAM, POINTS), y = POINTS, fill = Color)) +
    geom_bar(stat = "identity", width = 0.8) +
```

```
    scale_fill_identity() +
    coord_flip() +
    labs(
      title = paste("Overall Performance - Season", season),
      x = "Team",
      y = "Total Points"
    )

  print(p)
}

dev.off()
```

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

Part 6. Monte Carlo simulation.

Use Monte Carlo simulation to predict how many goals will Bayern Munchen score for next 10 seasons. Repeat the same for Bayer Leverkusen and Borusia Dortmund. Compare results using appropriate visualization technique.

```
teams <- c("Bayern Munich", "Leverkusen", "Dortmund")
historical_data <- bundesliga2 %>%
  filter(TEAM %in% teams) %>%
  select(TEAM, SEASON, GF)

simulate_goals <- function(team_name, historical_data, n_seasons = 10, n_simulations = 1000) {
  team_goals <- historical_data %>%
    filter(TEAM == team_name) %>%
    pull(GF)

  mean_goals <- mean(team_goals)
  sd_goals <- sd(team_goals)

  simulations <- replicate(n_simulations, rnorm(n_seasons, mean = mean_goals, sd = sd_goals))
  colnames(simulations) <- paste("Sim", 1:n_simulations, sep = "_")

  simulation_df <- data.frame(
    Season = rep(1:n_seasons, n_simulations),
    Goals = as.vector(simulations),
    Simulation = rep(1:n_simulations, each = n_seasons)
  )

  simulation_df$TEAM <- team_name
  return(simulation_df)
}

bayern_sim <- simulate_goals("Bayern Munich", historical_data)
leverkusen_sim <- simulate_goals("Leverkusen", historical_data)
dortmund_sim <- simulate_goals("Dortmund", historical_data)
```

```r
all_simulations <- bind_rows(bayern_sim, leverkusen_sim, dortmund_sim)

ggplot(all_simulations, aes(x = TEAM, y = Goals, fill = TEAM)) +
  geom_boxplot(alpha = 0.7, outlier.colour = "red", outlier.shape = 1) +
  labs(
    title = "Monte Carlo Simulation: Predicted Goals for Next 10 Seasons",
    x = "Team",
    y = "Simulated Goals"
  )
```