Albert_Hakobyan2_DV_HW4R

Albert Hakobyan

2025-03-27

```
library(ggplot2)
library(dplyr)
library(ggthemes)
```

```
bliga_df <- read.csv('bundesliga.csv')

head(bliga_df, 10)
```

```
##    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
## 7    1994 Bundesliga 1 1993-08-07    M'Gladbach  Ein Frankfurt  0-4    0    4
## 8    1994 Bundesliga 1 1993-08-07  Wattenscheid     Schalke 04  3-0    3    0
## 9    1994 Bundesliga 1 1993-08-07 Werder Bremen      Stuttgart  5-1    5    1
## 10   1994 Bundesliga 1 1993-08-14       Dresden       Duisburg  0-1    0    1
##    FTTG
## 1     4
## 2     3
## 3     4
## 4     2
## 5     7
## 6     6
## 7     4
## 8     3
## 9     6
## 10    1
```

PART 4: Rivalries & Big Match Patterns (*R*):

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.

```r
rivalries <- list("Bayern Munich vs Dortmund" = list(team1 = "Bayern Munich", team2 = "Dortmund"),
                  "Schalke 04 vs Dortmund" = list(team1 = "Schalke 04", team2 = "Dortmund"),
                  "Bayern Munich vs M'Gladbach" = list(team1 = "Bayern Munich", team2 = "M'Gladbach"),
                  "FC Koln vs M'Gladbach" = list(team1 = "FC Koln", team2 = "M'Gladbach"),
                  "Hamburg vs Werder Bremen" = list(team1 = "Hamburg", team2 = "Werder Bremen"))
```

```r
all_rivalry_matches <- NULL

for (rivalry_name in names(rivalries)) {
  team1 <- rivalries[[rivalry_name]]$team1
  team2 <- rivalries[[rivalry_name]]$team2

  matches <- bliga_df %>%
    filter((HOMETEAM == team1 & AWAYTEAM == team2) | (HOMETEAM == team2 & AWAYTEAM == team1)) %>%
    mutate(Rivalry = rivalry_name,
       Team1IsHome = HOMETEAM == team1,
            Result = case_when(
                        Team1IsHome & FTHG > FTAG ~ "Win",
                       !Team1IsHome & FTAG > FTHG ~ "Win",
                                    FTHG == FTAG ~ "Draw",
                                            TRUE ~ "Loss"),
    GoalDifference = ifelse(Team1IsHome, FTHG - FTAG, FTAG - FTHG))
  all_rivalry_matches <- bind_rows(all_rivalry_matches, matches)}

# Summary of wins/draws/losses
rivalry_summary <- all_rivalry_matches %>%
  group_by(Rivalry, Result) %>%
  summarise(Count = n(), .groups = "drop")

# Defining all_combinations (Ensuring existence)
all_combinations <- expand.grid(Rivalry = unique(all_rivalry_matches$Rivalry),
                                Result = c("Win", "Draw", "Loss"))

rivalry_summary <- left_join(all_combinations, rivalry_summary, by = c("Rivalry", "Result"))
rivalry_summary$Count[is.na(rivalry_summary$Count)] <- 0

# The biggest win margins
biggest_wins <- all_rivalry_matches %>%
  filter(Result == "Win") %>%
  group_by(Rivalry) %>%
  slice_max(order_by = abs(GoalDifference), n = 1) %>%
  ungroup()

max_counts <- rivalry_summary %>% # Max counts per rivalry for annotation positioning
  group_by(Rivalry) %>%
  summarise(MaxCount = max(Count, na.rm = TRUE), .groups = "drop")

biggest_wins <- biggest_wins %>%
  left_join(max_counts, by = "Rivalry")

head(all_rivalry_matches)
```

```
##   SEASON       LEAGUE       DATE      HOMETEAM      AWAYTEAM FTSC FTHG FTAG
## 1   1994 Bundesliga 1 1993-09-25      Dortmund Bayern Munich  1-1    1    1
## 2   1994 Bundesliga 1 1994-03-19 Bayern Munich      Dortmund  0-0    0    0
## 3   1995 Bundesliga 1 1994-10-22      Dortmund Bayern Munich  1-0    1    0
## 4   1995 Bundesliga 1 1995-04-22 Bayern Munich      Dortmund  2-1    2    1
## 5   1996 Bundesliga 1 1995-10-01      Dortmund Bayern Munich  3-1    3    1
## 6   1996 Bundesliga 1 1996-03-30 Bayern Munich      Dortmund  1-0    1    0
##   FTTG                  Rivalry Team1IsHome Result GoalDifference
## 1    2 Bayern Munich vs Dortmund       FALSE   Draw              0
## 2    0 Bayern Munich vs Dortmund        TRUE   Draw              0
## 3    1 Bayern Munich vs Dortmund       FALSE   Loss             -1
## 4    3 Bayern Munich vs Dortmund        TRUE    Win              1
## 5    4 Bayern Munich vs Dortmund       FALSE   Loss             -2
## 6    1 Bayern Munich vs Dortmund        TRUE    Win              1
```

```
head(biggest_wins)
```

```
## # A tibble: 4 x 14
##   SEASON LEAGUE       DATE     HOMETEAM AWAYTEAM FTSC   FTHG  FTAG  FTTG Rivalry
##    <int> <chr>        <chr>    <chr>    <chr>    <chr> <int> <int> <int> <chr>
## 1   2018 Bundesliga 1 2018-03~ Bayern ~ Dortmund 6-0       6     0     6 Bayern~
## 2   1994 Bundesliga 1 1993-10~ Bayern ~ M'Gladb~ 3-1       3     1     4 Bayern~
## 3   2011 Bundesliga 1 2011-02~ Hamburg  Werder ~ 4-0       4     0     4 Hambur~
## 4   2001 Bundesliga 1 2000-09~ Dortmund Schalke~ 0-4       0     4     4 Schalk~
## # i 4 more variables: Team1IsHome <lgl>, Result <chr>, GoalDifference <int>,
## #   MaxCount <dbl>
```

Facet grid of bar charts

```
ggplot(rivalry_summary, aes(x = Result, y = Count, fill = Result)) +
  geom_bar(stat = "identity") +
  facet_wrap(~ Rivalry, ncol = 2) +
  geom_text(data = biggest_wins, aes(x = "Win", y = MaxCount * 1.1, label = paste("Biggest win:", abs(GoalDifference), "goal
s")), vjust = 0.3) +
  labs(title = "Head-to-Head Statistics for Selected Bundesliga Rivalries", x = "Result", y = "Number of Matches") +
  theme_bw() +
  scale_fill_manual(values = c("Win" = "gold", "Draw" = "grey", "Loss" = "red"))
```

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

```r
# First, we need to create a table of season standings to determine team rankings
standings <- bliga_df %>%
  group_by(SEASON, HOMETEAM) %>%
  summarise(
    Points = sum(case_when(
      FTHG > FTAG ~ 3,
      FTHG == FTAG ~ 1,
      TRUE ~ 0)),
    GoalsScored = sum(FTHG),
    GoalsConceded = sum(FTAG),
    .groups = "drop") %>%
  rename(TEAM = HOMETEAM)

# Then we should add "away" games to standings
away_standings <- bliga_df %>%
  group_by(SEASON, AWAYTEAM) %>%
  summarise(Points = sum(case_when(FTAG > FTHG ~ 3,
                                   FTAG == FTHG~ 1,
                                          TRUE ~ 0)),
                    GoalsScored = sum(FTAG),
                    GoalsConceded = sum(FTHG),
                    .groups = "drop") %>%
  rename(TEAM = AWAYTEAM)

head(away_standings)
```

```
## # A tibble: 6 x 5
##   SEASON TEAM          Points GoalsScored GoalsConceded
##    <int> <chr>          <dbl>       <int>         <int>
## 1   1994 Bayern Munich     19          24            27
## 2   1994 Dortmund          13          14            30
## 3   1994 Dresden           16          13            28
## 4   1994 Duisburg          23          20            29
## 5   1994 Ein Frankfurt     24          28            19
## 6   1994 FC Koln           19          24            29
```

```r
# Now we need the combining of home and away standings
total_standings <- bind_rows(standings, away_standings) %>%
  group_by(SEASON, TEAM) %>%
  summarise(Points = sum(Points),
    GoalsScored = sum(GoalsScored),
    GoalsConceded = sum(GoalsConceded),
    GoalDifference = GoalsScored - GoalsConceded,
    .groups = "drop")

head(total_standings)
```

```
## # A tibble: 6 x 6
##   SEASON TEAM          Points GoalsScored GoalsConceded GoalDifference
##    <int> <chr>          <dbl>       <int>         <int>          <int>
## 1   1994 Bayern Munich     61          68            37             31
## 2   1994 Dortmund          54          49            45              4
## 3   1994 Dresden           44          33            44            -11
## 4   1994 Duisburg          50          41            52            -11
## 5   1994 Ein Frankfurt     53          57            41             16
## 6   1994 FC Koln           48          49            51             -2
```

```
# We need to establish a "Rank" system of teams by points (primary) and goal difference (secondary) for each season
season_rankings <- total_standings %>%
                  group_by(SEASON) %>%
arrange(desc(Points), desc(GoalDifference), .by_group = TRUE) %>%
        mutate(Rank = row_number()) %>%
                  ungroup()

top_teams <- season_rankings %>%
  filter(Rank <= 5) %>% # Identifying top 5 teams per season (based on points)
  select(SEASON, TEAM)

head(top_teams, 10)
```

```
## # A tibble: 10 x 2
##    SEASON TEAM
##     <int> <chr>
##  1   1994 Bayern Munich
##  2   1994 Kaiserslautern
##  3   1994 Dortmund
##  4   1994 Ein Frankfurt
##  5   1994 Leverkusen
##  6   1995 Dortmund
##  7   1995 Werder Bremen
##  8   1995 Freiburg
##  9   1995 Kaiserslautern
## 10   1995 M'gladbach
```

```r
upsets <- bliga_df %>%
  # Joining with rankings for home teams
  left_join(season_rankings, by = c("SEASON", "HOMETEAM" = "TEAM")) %>%
  rename(HomeRank = Rank) %>%
  # Joining with rankings for away teams
  left_join(season_rankings, by = c("SEASON", "AWAYTEAM" = "TEAM")) %>%
  rename(AwayRank = Rank) %>%
  # Filter for upsets
  filter(
    # Either home team is > 8 places below and beats a top5 away team
    ((HomeRank - AwayRank > 8) & (FTHG > FTAG) & (AwayRank <= 5)) |
    # Or away team is > 8 places below and beats a top5 home team
    ((AwayRank - HomeRank > 8) & (FTAG > FTHG) & (HomeRank <= 5))) %>%
  # Calculating thee rank difference and goal difference
  mutate(
    RankDifference = case_when(
      (HomeRank - AwayRank > 8) & (FTHG > FTAG) ~ HomeRank - AwayRank,
      (AwayRank - HomeRank > 8) & (FTAG > FTHG) ~ AwayRank - HomeRank,
      TRUE ~ NA_real_),
    GoalDifference = case_when(
      (HomeRank - AwayRank > 8) & (FTHG > FTAG) ~ FTHG - FTAG,
      (AwayRank - HomeRank > 8) & (FTAG > FTHG) ~ FTAG - FTHG,
      TRUE ~ NA_real_),
    UpsetTeam = case_when(
      (HomeRank - AwayRank > 8) & (FTHG > FTAG) ~ HOMETEAM,
      (AwayRank - HomeRank > 8) & (FTAG > FTHG) ~ AWAYTEAM,
      TRUE ~ NA_character_),
    FavoriteTeam = case_when(
      (HomeRank - AwayRank > 8) & (FTHG > FTAG) ~ AWAYTEAM,
      (AwayRank - HomeRank > 8) & (FTAG > FTHG) ~ HOMETEAM,
      TRUE ~ NA_character_),
   MatchLabel = paste0(UpsetTeam, " ",
                       ifelse((HomeRank - AwayRank > 8) & (FTHG > FTAG), FTHG, FTAG),
                       "-",
                       ifelse((HomeRank - AwayRank > 8) & (FTHG > FTAG), FTAG, FTHG),
                       " ", FavoriteTeam, " (", SEASON, ")"))
```

```r
head(upsets, 10)
```

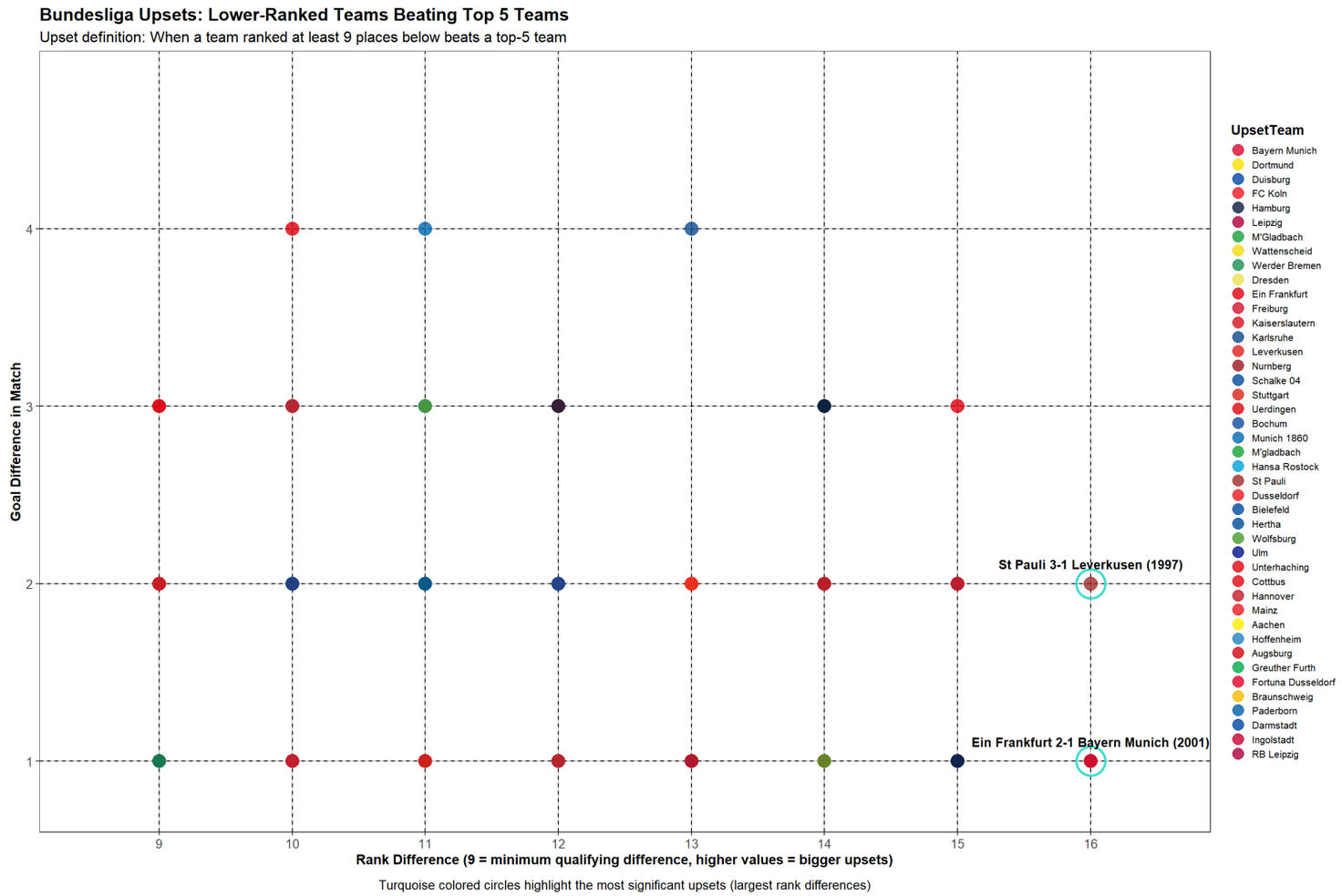
```
##    SEASON       LEAGUE       DATE      HOMETEAM       AWAYTEAM FTSC FTHG FTAG
## 1    1994 Bundesliga 1 1993-08-14    Schalke 04       Dortmund  1-0    1    0
## 2    1994 Bundesliga 1 1993-09-08      Dortmund        Leipzig  0-1    0    1
## 3    1994 Bundesliga 1 1993-10-02       Dresden Kaiserslautern  3-1    3    1
## 4    1994 Bundesliga 1 1993-10-23      Freiburg     Leverkusen  1-0    1    0
## 5    1994 Bundesliga 1 1993-11-06      Nurnberg  Bayern Munich  2-0    2    0
## 6    1994 Bundesliga 1 1993-11-27      Freiburg  Bayern Munich  3-1    3    1
## 7    1994 Bundesliga 1 1993-12-11      Freiburg       Dortmund  4-1    4    1
## 8    1994 Bundesliga 1 1994-02-19       Dresden       Dortmund  3-0    3    0
## 9    1994 Bundesliga 1 1994-02-26 Ein Frankfurt     Schalke 04  1-3    1    3
## 10   1994 Bundesliga 1 1994-03-05    Schalke 04 Kaiserslautern  2-0    2    0
##    FTTG Points.x GoalsScored.x GoalsConceded.x GoalDifference.x HomeRank
## 1     1       39            38              50               -12       14
## 2     1       54            49              45                 4        3
## 3     4       44            33              44               -11       13
## 4     1       38            54              57                -3       15
## 5     2       38            41              55               -14       16
## 6     4       38            54              57                -3       15
## 7     5       38            54              57                -3       15
## 8     3       44            33              44               -11       13
## 9     4       53            57              41                16        4
## 10    2       39            38              50               -12       14
##    Points.y GoalsScored.y GoalsConceded.y GoalDifference.y AwayRank
## 1        54            49              45                 4        3
## 2        20            32              69               -37       18
## 3        61            64              36                28        2
## 4        53            60              47                13        5
## 5        61            68              37                31        1
## 6        61            68              37                31        1
## 7        54            49              45                 4        3
## 8        54            49              45                 4        3
## 9        39            38              50               -12       14
## 10       61            64              36                28        2
##    RankDifference GoalDifference  UpsetTeam   FavoriteTeam
## 1              11              1 Schalke 04       Dortmund
## 2              15              1    Leipzig       Dortmund
## 3              11              2    Dresden Kaiserslautern
## 4              10              1   Freiburg     Leverkusen
## 5              15              2   Nurnberg  Bayern Munich
## 6              14              2   Freiburg  Bayern Munich
## 7              12              3   Freiburg       Dortmund
## 8              10              3    Dresden       Dortmund
## 9              10              2 Schalke 04  Ein Frankfurt
## 10             12              2 Schalke 04 Kaiserslautern
##                            MatchLabel
## 1        Schalke 04 1-0 Dortmund (1994)
## 2           Leipzig 1-0 Dortmund (1994)
## 3     Dresden 3-1 Kaiserslautern (1994)
## 4        Freiburg 1-0 Leverkusen (1994)
## 5     Nurnberg 2-0 Bayern Munich (1994)
## 6     Freiburg 3-1 Bayern Munich (1994)
## 7          Freiburg 4-1 Dortmund (1994)
## 8           Dresden 3-0 Dortmund (1994)
## 9   Schalke 04 3-1 Ein Frankfurt (1994)
## 10 Schalke 04 2-0 Kaiserslautern (1994)
```

```r
# Getting the famous upsets (top 3 by rank difference)
famous_upsets <- upsets %>%
  arrange(desc(RankDifference)) %>% head(3)

team_colors <- data.frame(
  TEAM = c(
    "Bayern Munich", "Dortmund", "Duisburg", "FC Koln", "Hamburg",
    "Leipzig", "M'Gladbach", "Wattenscheid", "Werder Bremen", "Dresden",
    "Ein Frankfurt", "Freiburg", "Kaiserslautern", "Karlsruhe", "Leverkusen",
    "Nurnberg", "Schalke 04", "Stuttgart", "Uerdingen", "Bochum",
    "Munich 1860", "M'gladbach", "Hansa Rostock", "St Pauli", "Dusseldorf",
    "Bielefeld", "Hertha", "Wolfsburg", "Ulm", "Unterhaching",
    "Cottbus", "Hannover", "Mainz", "Aachen", "Hoffenheim",
    "Augsburg", "Greuther Furth", "Fortuna Dusseldorf", "Braunschweig", "Paderborn",
    "Darmstadt", "Ingolstadt", "RB Leipzig"),
  Color = c(
    "#DC052D", "#FDE100", "#0046AD", "#ED1C24", "#0C2240",
    "#B1003C", "#18A33C", "#F7D917", "#1D9053", "#F0E453",
    "#E1000F", "#D31230", "#D3171E", "#0C4C92", "#E32221",
    "#9B1C1F", "#004D9D", "#DA291C", "#E30613", "#144DA3",
    "#006AB3", "#18A33C", "#00A5DC", "#A52A2A", "#EE1D23",
    "#004F9F", "#004B9C", "#4C9E2F", "#001489", "#E30613",
    "#E30613", "#C11D28", "#ED1C24", "#FFED00", "#1E87C5",
    "#D80A14", "#0AAC4A", "#E4002B", "#FBBA00", "#005CA9",
    "#0045A1", "#CC0033", "#B1003C"), stringsAsFactors = FALSE)

# Adding colors to upsets (default color for teams not in the list)
upsets <- upsets %>%
  left_join(team_colors, by = c("UpsetTeam" = "TEAM")) %>%
  mutate(TeamColor = ifelse(is.na(Color), "turquoise", Color))
```

```r
head(upsets, 10)
```

```
##    SEASON       LEAGUE       DATE      HOMETEAM       AWAYTEAM FTSC FTHG FTAG
## 1    1994 Bundesliga 1 1993-08-14    Schalke 04       Dortmund  1-0    1    0
## 2    1994 Bundesliga 1 1993-09-08      Dortmund        Leipzig  0-1    0    1
## 3    1994 Bundesliga 1 1993-10-02       Dresden Kaiserslautern  3-1    3    1
## 4    1994 Bundesliga 1 1993-10-23      Freiburg     Leverkusen  1-0    1    0
## 5    1994 Bundesliga 1 1993-11-06      Nurnberg  Bayern Munich  2-0    2    0
## 6    1994 Bundesliga 1 1993-11-27      Freiburg  Bayern Munich  3-1    3    1
## 7    1994 Bundesliga 1 1993-12-11      Freiburg       Dortmund  4-1    4    1
## 8    1994 Bundesliga 1 1994-02-19       Dresden       Dortmund  3-0    3    0
## 9    1994 Bundesliga 1 1994-02-26 Ein Frankfurt     Schalke 04  1-3    1    3
## 10   1994 Bundesliga 1 1994-03-05    Schalke 04 Kaiserslautern  2-0    2    0
##    FTTG Points.x GoalsScored.x GoalsConceded.x GoalDifference.x HomeRank
## 1     1       39            38              50               -12       14
## 2     1       54            49              45                 4        3
## 3     4       44            33              44               -11       13
## 4     1       38            54              57                -3       15
## 5     2       38            41              55               -14       16
## 6     4       38            54              57                -3       15
## 7     5       38            54              57                -3       15
## 8     3       44            33              44               -11       13
## 9     4       53            57              41                16        4
## 10    2       39            38              50               -12       14
##    Points.y GoalsScored.y GoalsConceded.y GoalDifference.y AwayRank
## 1        54            49              45                 4        3
## 2        20            32              69               -37       18
## 3        61            64              36                28        2
## 4        53            60              47                13        5
## 5        61            68              37                31        1
## 6        61            68              37                31        1
## 7        54            49              45                 4        3
## 8        54            49              45                 4        3
## 9        39            38              50               -12       14
## 10       61            64              36                28        2
##    RankDifference GoalDifference  UpsetTeam   FavoriteTeam
## 1              11              1 Schalke 04       Dortmund
## 2              15              1    Leipzig       Dortmund
## 3              11              2    Dresden Kaiserslautern
## 4              10              1   Freiburg     Leverkusen
## 5              15              2   Nurnberg  Bayern Munich
## 6              14              2   Freiburg  Bayern Munich
## 7              12              3   Freiburg       Dortmund
## 8              10              3    Dresden       Dortmund
## 9              10              2 Schalke 04  Ein Frankfurt
## 10             12              2 Schalke 04 Kaiserslautern
##                              MatchLabel   Color TeamColor
## 1        Schalke 04 1-0 Dortmund (1994) #004D9D   #004D9D
## 2           Leipzig 1-0 Dortmund (1994) #B1003C   #B1003C
## 3     Dresden 3-1 Kaiserslautern (1994) #F0E453   #F0E453
## 4        Freiburg 1-0 Leverkusen (1994) #D31230   #D31230
## 5     Nurnberg 2-0 Bayern Munich (1994) #9B1C1F   #9B1C1F
## 6     Freiburg 3-1 Bayern Munich (1994) #D31230   #D31230
## 7          Freiburg 4-1 Dortmund (1994) #D31230   #D31230
## 8           Dresden 3-0 Dortmund (1994) #F0E453   #F0E453
## 9   Schalke 04 3-1 Ein Frankfurt (1994) #004D9D   #004D9D
## 10 Schalke 04 2-0 Kaiserslautern (1994) #004D9D   #004D9D
```

```
ggplot(upsets, aes(x = RankDifference, y = GoalDifference)) +
  geom_hline(yintercept = c(1, 2, 3, 4), linetype = "dashed") +
  geom_vline(xintercept = seq(9, 16, by = 1), linetype = "dashed") +

  geom_point(aes(color = UpsetTeam), size = 5, alpha = 0.8) +

  geom_point(data = famous_upsets, aes(x = RankDifference, y = GoalDifference),
             size = 10, shape = 1, color = "turquoise", stroke = 1.5) +
  geom_text(data = famous_upsets,
            aes(x = RankDifference, y = GoalDifference, label = MatchLabel),
            vjust = -1.5, hjust = 0.5, size = 3.5, fontface = "bold",
            check_overlap = TRUE) +
  scale_x_continuous(breaks = seq(9, 16, by = 1),
                     minor_breaks = NULL,
                     limits = c(8.5, max(upsets$RankDifference) + 0.5)) +
  scale_y_continuous(breaks = seq(1, 4, by = 1),
                     minor_breaks = NULL,
                     limits = c(min(upsets$GoalDifference) - 0.2, max(upsets$GoalDifference) + 0.8)) +
  labs(title = "Bundesliga Upsets: Lower-Ranked Teams Beating Top 5 Teams",
       subtitle = "Upset definition: When a team ranked at least 9 places below beats a top-5 team",
       x = "Rank Difference (9 = minimum qualifying difference, higher values = bigger upsets)",
       y = "Goal Difference in Match",
       caption = "Turquoise colored circles highlight the most significant upsets (largest rank differences)") +
  theme_bw() +
  theme(plot.title = element_text(face = "bold", size = 14),
        plot.subtitle = element_text(size = 12),
        legend.position = "right",
        legend.title = element_text(face = "bold"),
        legend.text = element_text(size = 8),
        legend.key.size = unit(0.8, "lines"),
        panel.grid.minor = element_blank(),
        panel.grid.major = element_line(color = "gray"),
        axis.title = element_text(face = "bold"),
        axis.text = element_text(size = 10),
        plot.caption = element_text(hjust = 0.5, size = 10, margin = margin(t = 10))) +
scale_color_manual(values = setNames(team_colors$Color, team_colors$TEAM)) +
guides(color = guide_legend(override.aes = list(size = 4), ncol = 1))
```



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. (*R*)

```r
seasons <- unique(bliga_df$SEASON)

pdf("bundesliga_season_standings.pdf", width = 20, height = 8)

for (season in seasons) {
  season_data <- bliga_df %>% filter(SEASON == season)

  home_points <- season_data %>%
    group_by(HOMETEAM) %>%
    summarise(Points = sum(case_when(FTHG > FTAG ~ 3,
                                     FTHG == FTAG~ 1,
                                            TRUE ~ 0))) %>%
    rename(Team = HOMETEAM)

  away_points <- season_data %>%
    group_by(AWAYTEAM) %>%
    summarise(Points = sum(case_when(FTAG > FTHG ~ 3,
                                     FTAG == FTHG~ 1,
                                            TRUE ~ 0))) %>%
    rename(Team = AWAYTEAM)
    # Combining points and sorting
  standings <- bind_rows(home_points, away_points) %>%
    group_by(Team) %>%
    summarise(TotalPoints = sum(Points)) %>%
    arrange(desc(TotalPoints))

  winner <- standings$Team[1]

  season_colors <- standings %>%
    left_join(team_colors, by = c("Team" = "TEAM")) %>%
    mutate(Color = ifelse(is.na(Color), "turquoise", Color)) # Default turquoise for missing teams

  color_values <- setNames(season_colors$Color, season_colors$Team) #colors for scale_fill_manual

  eachPlot <- ggplot(standings, aes(x = reorder(Team, TotalPoints), y = TotalPoints, fill = Team)) +
    geom_bar(stat = "identity") +
    geom_text(aes(label = TotalPoints), hjust = -0.2, size = 3) +
    geom_bar(data = filter(standings, Team == winner),
             aes(x = reorder(Team, TotalPoints), y = TotalPoints),
             stat = "identity", fill = "gold", color = "black") +     # Highlighting the winner
    coord_flip() +
    scale_fill_manual(values = color_values) +
    labs(title = paste("Bundesliga Season", season), subtitle = paste("Winner:", winner), x = "", y = "Total Points") +
    theme_bw() +
    theme(legend.position = "none",
              plot.title = element_text(face = "bold"),
          panel.grid.minor = element_blank())
  print(eachPlot)}

dev.off()
```

```
## png
##   2
```

PART 6: Monte Carlo simulation. *R*

> *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.*

```
# Function to calculate goals per season for a team
calculate_goals_per_season <- function(data, team_name) {
  home_goals <- data %>%
    filter(HOMETEAM == team_name) %>%
    group_by(SEASON) %>%
    summarise(HomeGoals = sum(FTHG))

  away_goals <- data %>%
    filter(AWAYTEAM == team_name) %>%
    group_by(SEASON) %>%
    summarise(AwayGoals = sum(FTAG))

  # Merging and calculating total goals per season
  full_join(home_goals, away_goals, by = "SEASON") %>%
    mutate(HomeGoals = ifelse(is.na(HomeGoals), 0, HomeGoals),
           AwayGoals = ifelse(is.na(AwayGoals), 0, AwayGoals),
          TotalGoals = HomeGoals + AwayGoals) %>%
    arrange(SEASON)}

# Extracting Historical goal data for each team
bayern_goals <- calculate_goals_per_season(bliga_df, "Bayern Munich")
dortmund_goals <- calculate_goals_per_season(bliga_df, "Dortmund")
leverkusen_goals <- calculate_goals_per_season(bliga_df, "Leverkusen")

calculate_team_stats <- function(goals_data) {
  mean_goals <- mean(goals_data$TotalGoals)
    sd_goals <- sd(goals_data$TotalGoals)
   min_goals <- min(goals_data$TotalGoals)
   max_goals <- max(goals_data$TotalGoals)

  return(list(
   mean = mean_goals,
     sd = sd_goals,
    min = min_goals,
    max = max_goals))}

bayern_stats <- calculate_team_stats(bayern_goals)
dortmund_stats <- calculate_team_stats(dortmund_goals)
leverkusen_stats <- calculate_team_stats(leverkusen_goals)
```

```
print("Bayern Munich Statistics:")
```

```
## [1] "Bayern Munich Statistics:"
```

```
print(bayern_stats)
```

```
## $mean
## [1] 74.19231
##
## $sd
## [1] 11.12482
##
## $min
## [1] 55
##
## $max
## [1] 98
```

```
print("Dortmund Statistics:")
```

```
## [1] "Dortmund Statistics:"
```

```
print(dortmund_stats)
```

```
## $mean
## [1] 61
##
## $sd
## [1] 13.33867
##
## $min
## [1] 41
##
## $max
## [1] 82
```

```
print("Leverkusen Statistics:")
```

```
## [1] "Leverkusen Statistics:"
```

```
print(leverkusen_stats)
```

```
## $mean
## [1] 60.88462
##
## $sd
## [1] 8.627059
##
## $min
## [1] 37
##
## $max
## [1] 77
```

Monte Carlo Simulation Function

```
monte_carlo_simulation <- function(team_stats, num_seasons = 10, num_simulations = 1000) {
  # Creating a matrix to store all simulations. Each row is a simulation, each column is a season
  simulation_results <- matrix(0, nrow = num_simulations, ncol = num_seasons)

  for (i in 1:num_simulations) {
    # According to step 1: I should generate random goals for each season based on historical pattern
    # Using a normal distribution with mean and standard deviation from historical data
    # I'll use the log-normal distribution because I should not allow negative goals

    # Turning mean and sd into log-normal parameters
    variance <- team_stats$sd^2
    mu <- log(team_stats$mean^2 / sqrt(variance + team_stats$mean^2))
    sigma <- sqrt(log(1 + (variance / team_stats$mean^2)))

    # Generating random goals for each season
    for (j in 1:num_seasons) {
      # Generating a random number from log-normal distribution
      random_value <- rlnorm(1, meanlog = mu, sdlog = sigma)

      # Rounding to nearest integer for goal count (Almost forgot about this ...)
      simulation_results[i, j] <- round(random_value)}
  }
  return(simulation_results)}
```

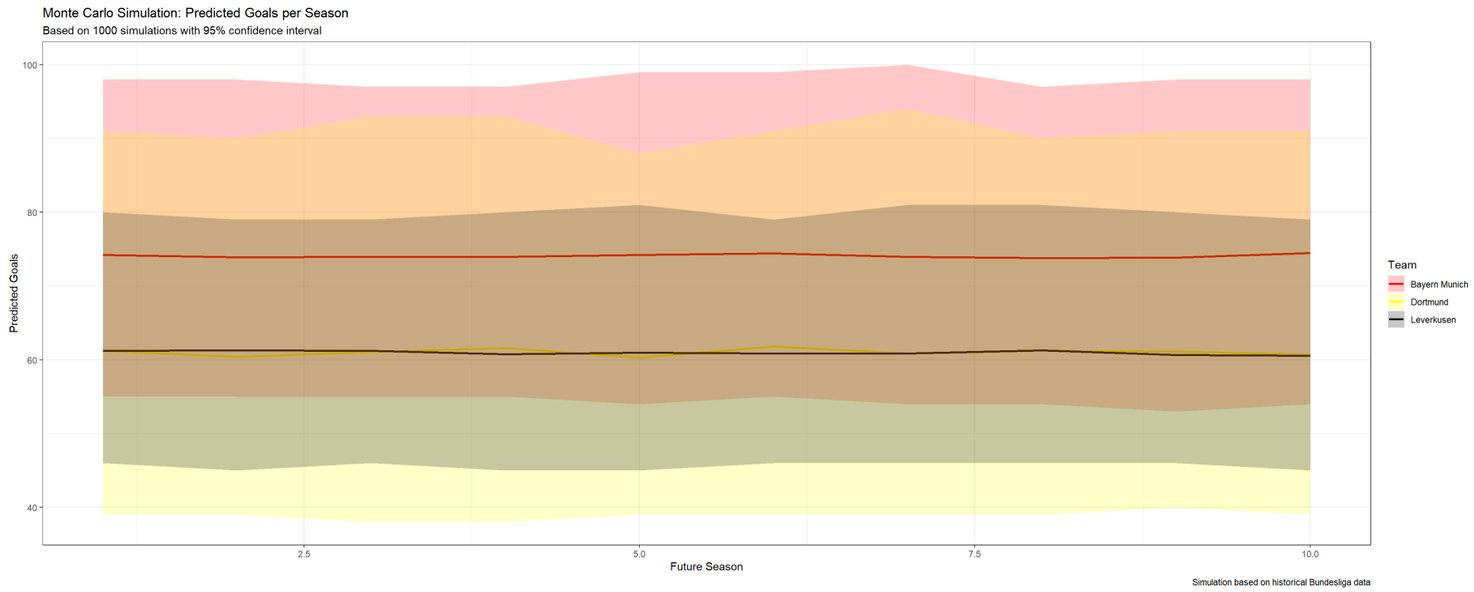
```
set.seed(42)

# Now we need to set the number of simulations and future seasons we should step into.
# P.S.: if this was not Monte Carlo, but a different time series forecasting method (such as AR[q], MA[p] or ARIMA/SARIMA),
# where order of AR or MA is important (both seasonal and trend), I'd use an ACF or PACF and AIC tests to
# reliably determine the number of future seasons we could reliably select in our forecasting.
# But since this is MC, lets go with 10 steps into the future.

num_simulations <- 1000
future_seasons <- 10

# Performing Monte Carlo simulations for each team of interest
bayern_simulations <- monte_carlo_simulation(bayern_stats, future_seasons, num_simulations)
dortmund_simulations <- monte_carlo_simulation(dortmund_stats, future_seasons, num_simulations)
leverkusen_simulations <- monte_carlo_simulation(leverkusen_stats, future_seasons, num_simulations)
```

```
# Setting up a function which calculates the summary statistics from simulations
calculate_simulation_summary <- function(simulations) {
  # Calculating mean, median, and percentiles for each season
  summary_data <- data.frame(
    Season = 1:ncol(simulations),
    Mean = apply(simulations, 2, mean),
    Median = apply(simulations, 2, median),
    Lower_CI = apply(simulations, 2, function(x) quantile(x, 0.025)),
    Upper_CI = apply(simulations, 2, function(x) quantile(x, 0.975)))
  return(summary_data)}

bayern_summary <- calculate_simulation_summary(bayern_simulations)
dortmund_summary <- calculate_simulation_summary(dortmund_simulations)
leverkusen_summary <- calculate_simulation_summary(leverkusen_simulations)

# Converting summaries to a combined format for visualization
bayern_summary$Team <- "Bayern Munich"
dortmund_summary$Team <- "Dortmund"
leverkusen_summary$Team <- "Leverkusen"
combined_summary <- rbind(bayern_summary, dortmund_summary, leverkusen_summary)

head(combined_summary)
```

```
##   Season   Mean Median Lower_CI Upper_CI         Team
## 1      1 74.213     74   55.000   98.000 Bayern Munich
## 2      2 73.884     73   55.000   98.025 Bayern Munich
## 3      3 73.985     73   54.975   97.025 Bayern Munich
## 4      4 73.983     73   55.000   97.000 Bayern Munich
## 5      5 74.213     73   54.000   99.000 Bayern Munich
## 6      6 74.436     73   55.000   99.000 Bayern Munich
```

Creating a visualization of the simulation results

```
ggplot(combined_summary, aes(x = Season, y = Mean, color = Team, fill = Team)) +
  geom_line(size = 1) +
  geom_ribbon(aes(ymin = Lower_CI, ymax = Upper_CI), alpha = 0.2, linetype = 0) +
  labs(title = "Monte Carlo Simulation: Predicted Goals per Season",
    subtitle = "Based on 1000 simulations with 95% confidence interval", x = "Future Season", y = "Predicted Goals", caption
= "Simulation based on historical Bundesliga data") +
  theme_bw() +
  scale_color_manual(values = c("Bayern Munich" = "red", "Dortmund" = "yellow", "Leverkusen" = "black")) +
  scale_fill_manual(values = c("Bayern Munich" =  "red", "Dortmund" = "yellow", "Leverkusen" = "black"))
```

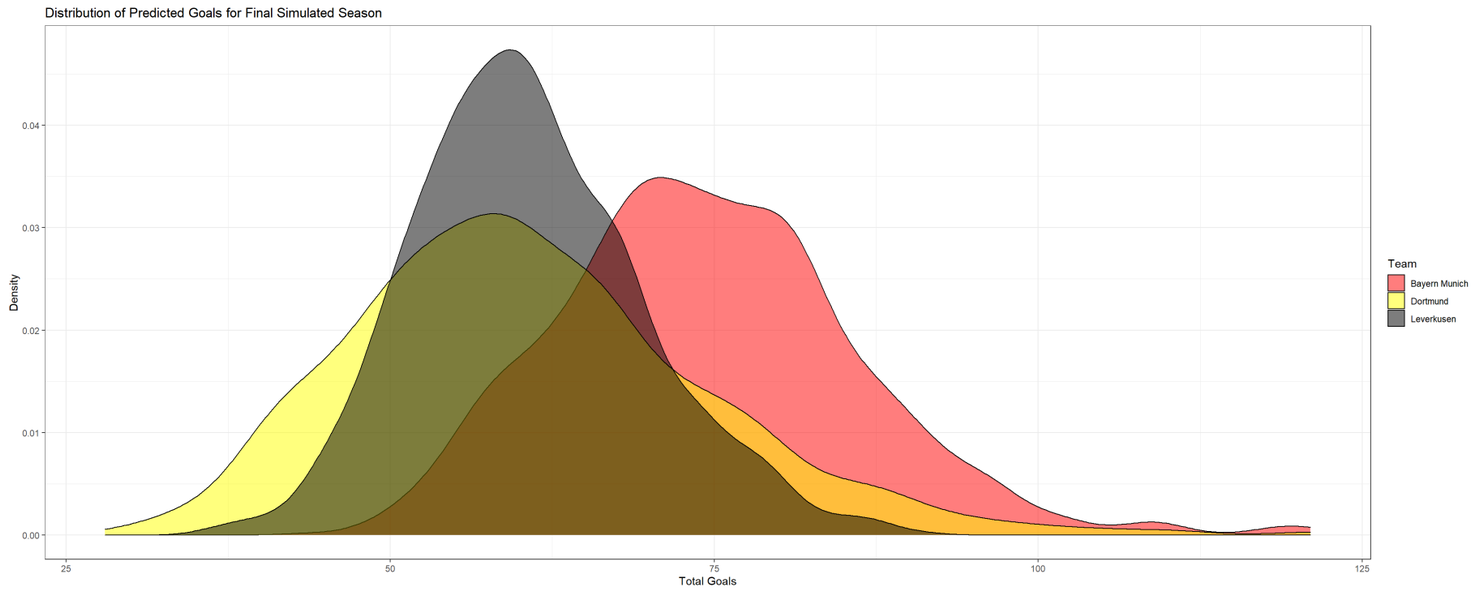


Creating a density plot to show the distribution of goals for the final season

```
final_season <- data.frame(
  Goals = c(bayern_simulations[, future_seasons],
            dortmund_simulations[, future_seasons],
            leverkusen_simulations[, future_seasons]),
  Team = rep(c("Bayern Munich", "Dortmund", "Leverkusen"), each = num_simulations))

head(final_season)
```

```
##   Goals          Team
## 1    73 Bayern Munich
## 2    89 Bayern Munich
## 3    67 Bayern Munich
## 4    74 Bayern Munich
## 5    81 Bayern Munich
## 6    77 Bayern Munich
```

```
ggplot(final_season, aes(x = Goals, fill = Team)) +
  geom_density(alpha = 0.5) +
  labs(title = "Distribution of Predicted Goals for Final Simulated Season", x = "Total Goals", y = "Density") +
  theme_bw() +
  scale_fill_manual(values = c("Bayern Munich" = "red", "Dortmund" = "yellow", "Leverkusen" = "black"))
```



A rolling forecast visualization (I am not satisfied with the results but this is Monte Carlo, what more could I expect from this :))

- Also, I am completely aware that there is a better way of creating a rolling forecast, where it is a good practice to shift the forecast a couple of seasons before the most recent value and having the comparison of each model's forecast line plots from the future with the actual data behavior as well (to calculate the errors and to later build bar plots of those MSE/MAE of each model compared to each other, but I thought of these things too late into the assignment completion and the deadline is too close for all of this). At least my knowledge is exposed through my words.

```
rolling_forecast <- data.frame()

# Last historical season (for connecting historical data to forecast)
last_season <- max(bayern_goals$SEASON)

# Historical data for plotting
historical_data <- rbind(data.frame(Season = bayern_goals$SEASON, Goals = bayern_goals$TotalGoals, Team = "Bayern Munich"),
  data.frame(Season = dortmund_goals$SEASON, Goals = dortmund_goals$TotalGoals, Team = "Dortmund"),
  data.frame(Season = leverkusen_goals$SEASON, Goals = leverkusen_goals$TotalGoals, Team = "Leverkusen"))

head(historical_data)
```

```
##   Season Goals          Team
## 1   1994    68 Bayern Munich
## 2   1995    55 Bayern Munich
## 3   1996    66 Bayern Munich
## 4   1997    68 Bayern Munich
## 5   1998    69 Bayern Munich
## 6   1999    76 Bayern Munich
```

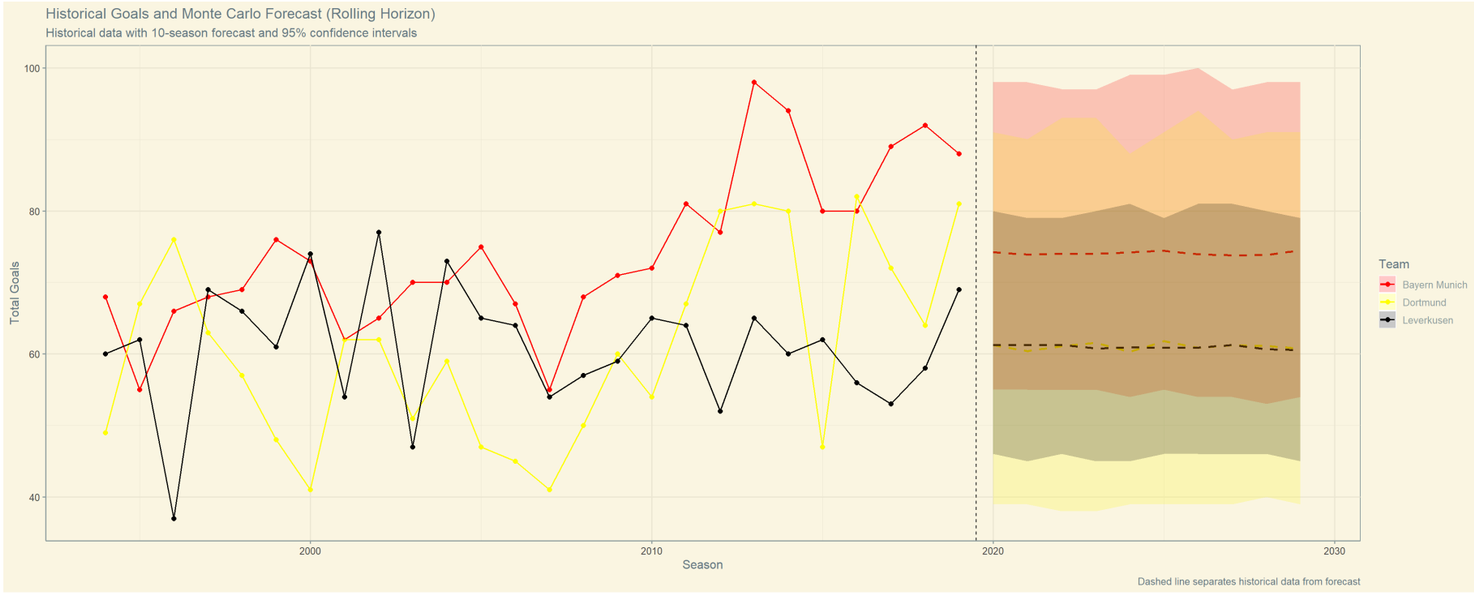
Future data for plotting

```
future_data <- rbind(data.frame(Season = (last_season + 1):(last_season + future_seasons),
              Goals = bayern_summary$Mean,
          Lower_CI = bayern_summary$Lower_CI,
          Upper_CI = bayern_summary$Upper_CI,
              Team = "Bayern Munich"),
  data.frame(Season = (last_season + 1):(last_season + future_seasons),
              Goals = dortmund_summary$Mean,
          Lower_CI = dortmund_summary$Lower_CI,
          Upper_CI = dortmund_summary$Upper_CI,
              Team = "Dortmund"),
  data.frame(Season = (last_season + 1):(last_season + future_seasons),
              Goals = leverkusen_summary$Mean,
          Lower_CI = leverkusen_summary$Lower_CI,
          Upper_CI = leverkusen_summary$Upper_CI,
              Team = "Leverkusen"))
head(future_data)
```

```
##   Season  Goals Lower_CI Upper_CI          Team
## 1   2020 74.213   55.000   98.000 Bayern Munich
## 2   2021 73.884   55.000   98.025 Bayern Munich
## 3   2022 73.985   54.975   97.025 Bayern Munich
## 4   2023 73.983   55.000   97.000 Bayern Munich
## 5   2024 74.213   54.000   99.000 Bayern Munich
## 6   2025 74.436   55.000   99.000 Bayern Munich
```

Plotting historical data with forecast

```
ggplot() +
  # Historical data as points and lines (scatterplotted lineplot)
  geom_point(data = historical_data, aes(x = Season, y = Goals, color = Team), size = 2) +
  geom_line(data = historical_data, aes(x = Season, y = Goals, color = Team), size = 0.7) +

  # Future data as lines with ribbon for confidence interval
  geom_line(data = future_data, aes(x = Season, y = Goals, color = Team), size = 1, linetype = "dashed") +
  geom_ribbon(data = future_data, aes(x = Season, ymin = Lower_CI, ymax = Upper_CI, fill = Team), alpha = 0.2) +

  # Vertical line to separate historical from forecast (imitate a rolling forecast visualization)
  geom_vline(xintercept = last_season + 0.5, linetype = "dashed") +
  labs(title = "Historical Goals and Monte Carlo Forecast (Rolling Horizon)", subtitle = "Historical data with 10-season for
ecast and 95% confidence intervals", x = "Season", y = "Total Goals", caption = "Dashed line separates historical data from
forecast") +
  theme_solarized() +
  scale_color_manual(values = c("Bayern Munich" = "red",  "Dortmund" = "yellow", "Leverkusen" = "black")) +
  scale_fill_manual(values = c("Bayern Munich" =  "red",  "Dortmund" = "yellow", "Leverkusen" = "black"))
```



Prediction summaries

```
print("Bayern Munich Prediction Summary:")
```

```
## [1] "Bayern Munich Prediction Summary:"
```

```
print(bayern_summary)
```

```
##    Season   Mean Median Lower_CI Upper_CI         Team
## 1       1 74.213     74   55.000   98.000 Bayern Munich
## 2       2 73.884     73   55.000   98.025 Bayern Munich
## 3       3 73.985     73   54.975   97.025 Bayern Munich
## 4       4 73.983     73   55.000   97.000 Bayern Munich
## 5       5 74.213     73   54.000   99.000 Bayern Munich
## 6       6 74.436     73   55.000   99.000 Bayern Munich
## 7       7 73.945     73   54.000  100.000 Bayern Munich
## 8       8 73.797     73   54.000   97.000 Bayern Munich
## 9       9 73.835     73   53.000   98.000 Bayern Munich
## 10     10 74.464     74   54.000   98.025 Bayern Munich
```

```
print("Dortmund Prediction Summary:")
```

```
## [1] "Dortmund Prediction Summary:"
```

```
print(dortmund_summary)
```

```
##    Season   Mean Median Lower_CI Upper_CI     Team
## 1       1 61.233     60   39.000   91.000 Dortmund
## 2       2 60.374     59   39.000   90.025 Dortmund
## 3       3 61.089     59   38.000   93.000 Dortmund
## 4       4 61.586     60   38.000   93.025 Dortmund
## 5       5 60.305     59   39.000   88.000 Dortmund
## 6       6 61.798     61   39.000   91.000 Dortmund
## 7       7 60.887     59   39.000   94.025 Dortmund
## 8       8 61.220     60   39.000   90.000 Dortmund
## 9       9 61.152     60   39.975   91.000 Dortmund
## 10     10 60.692     59   39.000   91.000 Dortmund
```

```
print("Leverkusen Prediction Summary:")
```

```
## [1] "Leverkusen Prediction Summary:"
```

```
print(leverkusen_summary)
```

```
##    Season   Mean Median Lower_CI Upper_CI       Team
## 1       1 61.232     61   46.000   80.000 Leverkusen
## 2       2 61.277     61   45.000   79.000 Leverkusen
## 3       3 61.236     61   46.000   79.025 Leverkusen
## 4       4 60.781     60   45.000   80.000 Leverkusen
## 5       5 60.954     61   45.000   81.000 Leverkusen
## 6       6 60.855     60   46.000   79.000 Leverkusen
## 7       7 60.879     60   46.000   81.000 Leverkusen
## 8       8 61.270     61   45.975   81.000 Leverkusen
## 9       9 60.677     60   46.000   80.000 Leverkusen
## 10     10 60.560     60   45.000   79.025 Leverkusen
```

Creating bundesliga3.csv from bundesliga.csv to replicate bundesliga2.csv.

```
bundesliga <- read.csv("bundesliga.csv")
transform_to_standings <- function(match_data) {
  home_stats <- match_data %>%
    group_by(SEASON, HOMETEAM) %>%
    summarise(Home_M = n(),  # Number of home matches
              Home_W = sum(FTHG > FTAG),       # Home wins
              Home_D = sum(FTHG == FTAG),      # Home draws
              Home_L = sum(FTHG < FTAG),       # Home losses
             Home_GF = sum(FTHG),             # Goals scored at home
             Home_GA = sum(FTAG),             # Goals conceded at home
         Home_Points = sum(FTHG > FTAG) * 3 + sum(FTHG == FTAG) * 1) %>%  # Points earned at home
         rename(TEAM = HOMETEAM)

  away_stats <- match_data %>%
    group_by(SEASON, AWAYTEAM) %>%
    summarise(Away_M = n(),                                 # Number of away matches
              Away_W = sum(FTAG > FTHG),
              Away_D = sum(FTAG == FTHG),
              Away_L = sum(FTAG < FTHG),
             Away_GF = sum(FTAG),
             Away_GA = sum(FTHG),
         Away_Points = sum(FTAG > FTHG) * 3 + sum(FTAG == FTHG) * 1) %>%  # Points earned away
         rename(TEAM = AWAYTEAM)

  # Step 3: Combine home and away statistics
  combined_stats <- full_join(home_stats, away_stats, by = c("SEASON", "TEAM")) %>%
    mutate(
      # Calculate total statistics
            M = Home_M + Away_M,              # Total matches
            W = Home_W + Away_W,              # Total wins
            D = Home_D + Away_D,              # Total draws
            L = Home_L + Away_L,              # Total losses
           GF = Home_GF + Away_GF,           # Total goals for
           GA = Home_GA + Away_GA,           # Total goals against
         DIFF = (Home_GF + Away_GF) - (Home_GA + Away_GA),  # Goal difference
       POINTS = Home_Points + Away_Points)  # Total points

  standings <- combined_stats %>%
    group_by(SEASON) %>%
    arrange(desc(POINTS), desc(DIFF), desc(GF), .by_group = TRUE) %>%
    mutate(POSITION = row_number()) %>%
    ungroup() %>%
    # Selecting only the columns needed for the final output
    select(TEAM, M, W, D, L, GF, GA, DIFF, POINTS, POSITION, SEASON)

  return(standings)
}
# Applying the transformation and saving the result
bundesliga_standings <- transform_to_standings(bundesliga)
```

```
## `summarise()` has grouped output by 'SEASON'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'SEASON'. You can override using the
## `.groups` argument.
```

```
# THE FILE IS called:  bundesliga3.csv
write.csv(bundesliga_standings, "bundesliga3.csv", row.names = FALSE)
head(bundesliga_standings)
```

```
## # A tibble: 6 x 11
##   TEAM              M     W     D     L    GF    GA  DIFF POINTS POSITION SEASON
##   <chr>         <int> <int> <int> <int> <int> <int> <int>  <dbl>    <int>  <int>
## 1 Bayern Munich    34    17    10     7    68    37    31     61        1   1994
## 2 Kaiserslaute~    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
```