Project #1

2023-03-07

Project #1 Report: Football Statistics vs. Winning the League

by: Rodrigo Guerra

The text of your report will provide a narrative structure around your code and outputs with R Markdown. Answers without supporting code will not receive credit and outputs without comments will not receive credit either: write full sentences to describe your findings. All code contained in your final project document must work correctly (knit early, knit often)! If a group project, only submit one report.

##Guidelines for the report:

1. Title and Introduction.

```
Write a narrative introduction describing
the datasets you have chosen, how they were acquired (cite your
sources), and why they are interesting to you. Describe what a
unique row represents in the datasets (e.g., states, countries,
dates, sport players, ...), how you can join them (keys), the types
of variables (numeric, categorical) they contain. Expand on
potential relationships, trends, you may expect, if any. Write a
research question (if a group project, each member writes a research
question) that you will answer with your exploratory data analysis.
```

a. Set up

First load the packages you will use throughout the document. *Note: you can hide this step in the resulting output with the option `echo=FALSE`.*

b. Quick description of the dataset(s)

Then call the dataset(s) you will need to join. If stored locally, make sure the datasets are saved in the same folder as your RMarkdown file (the working directory). Otherwise, R would not know where to look for the data!

Remember the following data from the GitHub page about Amazon reviews (originally obtained from https://s3.amazonaws.com/amazon-reviews-pds/readme.html (https://s3.amazonaws.com/amazon-reviews-pds/readme.html)):

```
# Upload data from R which was uploaded
top_scorers <- read_csv("top_scorers.csv")

## Take a look
glimpse(top_scorers)
```

```
## Rows: 660
## Columns: 15
## $ Country                  <chr> "Spain", "Spain", "Spain", "Spain", "Spain",…
## $ League                   <chr> "La Liga", "La Liga", "La Liga", "La Liga", …
## $ Club                     <chr> "(BET)", "(BAR)", "(ATL)", "(CAR)", "(VAL)",…
## $ `Player Names`           <chr> "Juanmi Callejon", "Antoine Griezmann", "Lui…
## $ Matches_Played           <dbl> 19, 36, 34, 32, 21, 29, 23, 30, 25, 31, 27, …
## $ Substitution             <dbl> 16, 0, 1, 3, 10, 0, 6, 0, 7, 7, 5, 2, 2, 0, …
## $ Mins                     <dbl> 1849, 3129, 2940, 2842, 1745, 2634, 1967, 26…
## $ Goals                    <dbl> 11, 16, 28, 13, 13, 25, 11, 13, 19, 11, 16, …
## $ xG                       <dbl> 6.62, 11.86, 23.21, 14.06, 10.65, 24.68, 13.…
## $ `xG Per Avg Match`       <dbl> 0.34, 0.36, 0.75, 0.47, 0.58, 0.89, 0.64, 0.…
## $ Shots                    <dbl> 48, 88, 120, 117, 50, 162, 69, 105, 78, 64, …
## $ OnTarget                 <dbl> 20, 41, 57, 42, 23, 60, 34, 42, 37, 26, 45, …
## $ `Shots Per Avg Match`    <dbl> 2.47, 2.67, 3.88, 3.91, 2.72, 5.84, 3.33, 3.…
## $ `On Target Per Avg Match` <dbl> 1.03, 1.24, 1.84, 1.40, 1.25, 2.16, 1.64, 1.…
## $ Year                     <dbl> 2016, 2016, 2016, 2016, 2016, 2016, 2016, 20…
```

```
# Upload data from R which was uploaded
football_metrics <- read_csv("football_metrics.csv")

## Take a look
glimpse(football_metrics)
```

```
## Rows: 684
## Columns: 24
## $ ...1         <chr> "La_liga", "La_liga", "La_liga", "La_liga", "La_liga", "L…
## $ ...2         <dbl> 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 201…
## $ position     <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17…
## $ team         <chr> "Barcelona", "Real Madrid", "Atletico Madrid", "Valencia"…
## $ matches      <dbl> 38, 38, 38, 38, 38, 38, 38, 38, 38, 38, 38, 38, 38, 38, 3…
## $ wins         <dbl> 30, 30, 23, 22, 23, 16, 15, 13, 14, 15, 13, 11, 11, 10, 9…
## $ draws        <dbl> 4, 2, 9, 11, 7, 12, 10, 12, 8, 4, 10, 13, 8, 7, 10, 14, 1…
## $ loses        <dbl> 4, 6, 6, 5, 8, 10, 13, 13, 16, 19, 15, 14, 19, 21, 19, 17…
## $ scored       <dbl> 110, 118, 67, 70, 71, 48, 42, 47, 42, 46, 47, 44, 35, 33,…
## $ missed       <dbl> 21, 38, 29, 32, 45, 37, 41, 44, 48, 68, 51, 51, 62, 64, 6…
## $ pts          <dbl> 94, 92, 78, 77, 76, 60, 55, 51, 50, 49, 49, 46, 41, 37, 3…
## $ xG           <dbl> 102.98015, 95.76624, 57.04767, 55.06250, 69.52662, 56.768…
## $ xG_diff      <dbl> -7.0198480, -22.2337570, -9.9523300, -14.9375000, -1.4733…
## $ npxG         <dbl> 97.77721, 86.10389, 52.58801, 49.70398, 62.09460, 55.2814…
## $ xGA          <dbl> 28.44429, 42.60720, 29.06911, 39.39257, 47.86274, 40.7018…
## $ xGA_diff     <dbl> 7.4442927, 4.6071980, 0.0691071, 7.3925720, 2.8627420, 3.…
## $ npxGA        <dbl> 24.72791, 38.89081, 26.83927, 33.44648, 41.91653, 38.4719…
## $ npxGD        <dbl> 73.0493053, 47.2130900, 25.7487369, 16.2575010, 20.178070…
## $ ppda_coef    <dbl> 5.683535, 10.209085, 8.982028, 8.709827, 8.276148, 10.072…
## $ oppda_coef   <dbl> 16.367593, 12.929510, 9.237091, 7.870225, 9.477805, 8.679…
## $ deep         <dbl> 489, 351, 197, 203, 305, 242, 183, 287, 184, 147, 173, 14…
## $ deep_allowed <dbl> 114, 153, 123, 172, 168, 171, 171, 207, 184, 219, 205, 16…
## $ xpts         <dbl> 94.0813, 81.7489, 73.1353, 63.7068, 67.3867, 62.7363, 53.…
## $ xpts_diff    <dbl> 0.0813, -10.2511, -4.8647, -13.2932, -8.6133, 2.7363, -1.…
```

The datasets I have chosen have statistics from some of the top leagues in the world of football (soccer). One dataset has statistics for 8 leagues, including the top 5 European leagues, from 2016 to 2020. It's called top_scorers. The other dataset has statistics from 6 leagues, including the top 5 european leagues, from 2014 to 2019. It's called football_metrics.

I acquired both datasets from Kaggle:

https://www.kaggle.com/datasets/mohamedhanyyy/top-football-leagues-scorers (https://www.kaggle.com/datasets/mohamedhanyyy/top-football-leagues-scorers)

https://www.kaggle.com/datasets/slehkyi/extended-football-stats-for-european-leagues-xg (https://www.kaggle.com/datasets/slehkyi/extended-football-stats-for-european-leagues-xg)

In the top_scorers dataset, a unique row represents a football player and their statistics for a certain year.

In the football_metrics dataset, a unique row represents a football club and their statistics for a certain year.

I can join both datasets with the league columns, since they share the top 5 European leagues in common.

The types of variables contained in the top_scorers dataset include: Matches played, minutes, goals, XG (expected goals), shots, and shots on target, to name a few.

The types of variables in the football_metrics dataset include: matches (played), wins, draws, losses, (goals) scored, points, XG, XGA (expected goals and assists), and xpts (expected points), to name a few.

I expect there to be a positive relationship in player performance and that player's team's wins, goals, and points (and expected goals, goals scored and points). If the player performs better, and if their teammates also perform better, then their team will perform better. Similarly, if a player performs poorly, as well as their teammates, then their team will perform poorly.

c. Define a research question

Something we might want to explore is if having 'better' statistics in football has a positive relationship with winning the league. If this is true, then teams and players can focus more on improving their stats in order to win their team titles and championships.

Research Question: Does better performance in terms of football statistics (Goals, xG, etc.) have a positive relationship with winning the league?

2. Tidying.

```
If the datasets are not tidy, you will need to reshape them
so that every observation has its own row and every variable its own
column.\
✓ Use the tidyr functions pivot_longer/gather and/or
pivot_wider/spread. ✓ Document the process (describe in words what
was done). ✓ Depending on your datasets, it might be a good idea to
do this before joining or might be necessary after. Feel free to
rearrange the sections.\
NOTE: If your datasets are already tidy, be sure to use those tidyr
functions somewhere else in your project (e.g., for rearranging
summary statistics).
```

My datasets are already tidy. Every observation has its own row, and every variable has its own column.

I will use tidyr functions somewhere else in my project.

However, I do have to edit some things in my datasets in order to join them, so I will do that here.

```
# Rename the first two variables in the football_metrics dataset
football_metrics <-football_metrics %>%
  rename("League" = 1,
         "Year" = 2)

# Rename the leagues in the football_metrics dataset
football_metrics <- football_metrics %>%
  mutate(League = recode(League, 'La_liga'='La Liga', 'EPL'= 'Premier League', 'Serie_A'
= 'Serie A', 'Ligue_1'='Ligue 1'))

# Rename the French leagues in the top_scorers dataset
top_scorers <- top_scorers %>%
  mutate(League = recode(League, 'France Ligue 1' = 'Ligue 1', 'France Ligue 2' = 'Ligue
1', 'France Ligue 3' = 'Ligue 1', 'France Ligue 4' = 'Ligue 1', 'France Ligue 5' = 'Ligu
e 1', 'France Ligue 6' = 'Ligue 1', 'France Ligue 7' = 'Ligue 1', 'France Ligue 8' = 'Li
gue 1', 'France Ligue 9' = 'Ligue 1', 'France Ligue 10' = 'Ligue 1', 'France Ligue 11' =
'Ligue 1', 'France Ligue 12' = 'Ligue 1', 'France Ligue 13' = 'Ligue 1', 'France Ligue 1
4' = 'Ligue 1', 'France Ligue 15' = 'Ligue 1', 'France Ligue 16' = 'Ligue 1', 'France Li
gue 17' = 'Ligue 1', 'France Ligue 18' = 'Ligue 1', 'France Ligue 19' = 'Ligue 1', 'Fran
ce Ligue 20' = 'Ligue 1'))
```

3. Joining/Merging.

```
Join your 2+ separate data sources into a single
dataset. ✓ Use dplyr join functions on an ID variable (or ID
variables) common to all datasets. ✓ Document the process (describe
in words what was done). ✓ Include the number of:\
o total observations in each dataset before joining, o IDs that
appear in one dataset but not the other, o IDs in common, o IDs that
may have been left out after joining. ✓ Report how many
observations/rows were dropped/added when joining the datasets and
discuss any potential issues.
```

```
# Join the two datasets into a single dataset
football_stats <- left_join(football_metrics, top_scorers, by = c("League"))

football_stats
```

League <chr>	Year.x <dbl>	position <dbl>	team <chr>	matches <dbl>	wins <dbl>	draws <dbl>	loses <dbl>	scored <dbl>	missed <dbl>
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21

League <chr>	Year.x <dbl>	position <dbl>	team <chr>	matches <dbl>	wins <dbl>	draws <dbl>	loses <dbl>	scored <dbl>	missed <dbl>
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21
La Liga	2014	1	Barcelona	38	30	4	4	110	21

1-10 of 10 rows | 1-10 of 38 columns

What was done here is that I joined the datasets using `left_join()` to keep information from the “left” dataset and I added information from the “right” dataset, matching the observations with the key variable.

The “left” dataset is ‘football_metrics’, the “right” dataset is ‘top_scorers’, and the key variable is ‘League’.

```
# Find the number of observations and variables in each dataset
nrow(football_metrics)
```

```
## [1] 684
```

```
ncol(football_metrics)
```

```
## [1] 24
```

```
nrow(top_scorers)
```

```
## [1] 660
```

```
ncol(top_scorers)
```

```
## [1] 15
```

The number of total observations in each dataset before joining was:

football_metrics: 684 observations of 24 variables

top_scorers: 660 observations of 15 variables

IDs that appear in one dataset but not the other

IDs in football_metrics that aren’t in top_scorers:

Position (in table), wins, draws, losses, goals, missed (goals missed), pts (points), XG diff, npxG, xGA, xGA_diff, npxGA, npxGD, ppda_coef, oppda_coef, deep, deep_allowed, xpts, xpts_diff

IDs in top_scorers that aren't in football_metrics:

Country, Player names, mins (minutes played), XG per avg match, shots, on target (shots on target), shots per avg match, on target per avg match

IDs in common

Year, Club (different names), Matches played (different names), Goals scored (different names), XG

IDs that may have been left out after joining

None, but some duplicates made, which can be fixed by tidying.

When joining the datasets, 50352 observations were added to the already existing 1344 observations, retulting in 51696 observations in the joined dataset, 'football_stats'.

1 variable was dropped when joining as well. since they both have Year columns, so the joined dataset includes year columns for both. This can be fixed by tidying.

In order to move on to the next step, we have to tidy our data some more:

```
# Tidy the data some more: We only want data for when the years match
football_stats <- football_stats %>%
  filter(Year.x == Year.y)

football_stats
```

League <chr>	Year.x <dbl>	position <dbl>	team <chr>	matches <dbl>	wins <dbl>	draws <dbl>	loses <dbl>	scored <dbl>	missed <dbl>
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41

1-10 of 10 rows | 1-10 of 38 columns

```
# Tidy the data some more: Remove Year.y, since we're going to be using Year.x for our y
ear column. Rename Year.x to Year.
football_stats <- football_stats %>%
  select(-Year.y) %>%
  rename("Year" = 'Year.x') %>%
  rename("losses" = 'loses') # rename loses to losses

football_stats
```

League <chr>	Year <dbl>	position <dbl>	team <chr>	matches <dbl>	wins <dbl>	draws <dbl>	losses <dbl>	scored <dbl>	missed <dbl>
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41

1-10 of 10 rows | 1-10 of 37 columns

With some further tidying…:

```
# Tidy the data some more: Put League and Year at the front, and arrange by Year
football_stats <- football_stats %>%
  select(League, Year, everything()) %>%
  arrange(Year)

football_stats
```

League <chr>	Year <dbl>	position <dbl>	team <chr>	matches <dbl>	wins <dbl>	draws <dbl>	losses <dbl>	scored <dbl>	missed <dbl>
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41

League <chr>	Year <dbl>	position <dbl>	team <chr>	matches <dbl>	wins <dbl>	draws <dbl>	losses <dbl>	scored <dbl>	missed <dbl>
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41

1-10 of 10 rows | 1-10 of 37 columns

And, we want to look at champions from 2016 to 2019, since those are the years that our two datasets have in common:

```
# Tidy the data some more: Only include champions from the year 2016-2019
football_stats<- football_stats %>%
  filter(position == 1 | position == 2 | position == 3 | position == 4, Year == "2016" |
Year == "2017" | Year == "2018" | Year == "2019")

football_stats
```

League <chr>	Year <dbl>	position <dbl>	team <chr>	matches <dbl>	wins <dbl>	draws <dbl>	losses <dbl>	scored <dbl>	missed <dbl>
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41

1-10 of 10 rows | 1-10 of 37 columns

We also want the teams to match the players' clubs:

```
# Tidy the data some more: Only include players that match their team and club from the
teams that finished in the top 4 from the year 2016-2019
football_stats <- football_stats %>%
  filter((team == 'Barcelona' & Club == '(BAR)') |
        (team == 'Real Madrid' & Club == "(RMA)") |
        (team == 'Chelsea' & Club == "(CHE)") |
        (team == 'Manchester City' & Club == "(MNC)") |
        (team == 'Liverpool' & Club == "(LIV)") |
        (team == 'Bayern Munich' & Club == "(BAY)") |
        (team == 'Juventus' & Club == "(JUV)") |
        (team == 'Paris Saint Germain' & Club == "(PSG)") |
        (team == 'Sevilla' & Club == "(SEV)") |
        (team == 'Valencia' & Club == "(VAL)") |
        (team == 'Atletico Madrid' & Club == "(ATL)") |
        (team == 'Tottenham' & Club == "(TOT)") |
        (team == 'Manchester United' & Club == "(MNU)") |
        (team == 'RasenBallsport Leipzig' & Club == "(RBL)") |
        (team == 'Borussia Dortmund' & Club == "(DOR)") |
        (team == 'Hoffenheim' & Club == "(HOF)") |
        (team == 'Schalke 04' & Club == "(SCH)") |
        (team == 'Bayer Leverkusen' & Club == "(LEV)") |
        (team == 'Borussia M.Gladbach' & Club == "(BMG)") |
        (team == 'Roma' & Club == "(ROM)") |
        (team == 'Napoli' & Club == "(NAP)") |
        (team == 'Atalanta' & Club == "(ATA)") |
        (team == 'Inter' & Club == "(INT)") |
        (team == 'Lazio' & Club == "(LAZ)") |
        (team == 'Lyon' & Club == "(LYO)") |
        (team == 'Lille' & Club == "(LIL)") |
        (team == 'Marseille' & Club == "(MAR)") |
        (team == 'Saint-Etienne' & Club == "(STE)") |
        (team == 'Nice' & Club == "(NIC)") |
        (team == 'Rennes' & Club == "(REN)"))


football_stats
```

League <chr>	Y... <dbl>	position <dbl>	team <chr>	matc... <dbl>	w... <dbl>	dr... <dbl>	losses <dbl>	scor... <dbl>	mis... <dbl>
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	1	Real Madrid	38	29	6	3	106	41
La Liga	2016	2	Barcelona	38	28	6	4	116	37
La Liga	2016	2	Barcelona	38	28	6	4	116	37
La Liga	2016	3	Atletico Madrid	38	23	9	6	70	27
Premier League	2016	1	Chelsea	38	30	3	5	85	33
Premier League	2016	2	Tottenham	38	26	8	4	86	26

League <chr>	Y... <dbl>	position <dbl>	team <chr>	matc... <dbl>	w... <dbl>	dr... <dbl>	losses <dbl>	scor... <dbl>	mis... <dbl>
Premier League	2016	2	Tottenham	38	26	8	4	86	26
Premier League	2016	2	Tottenham	38	26	8	4	86	26
Premier League	2016	3	Manchester City	38	23	9	6	80	39

1-10 of 10 rows | 1-10 of 37 columns

Now our data looks good, and we're ready to move on to the next step, wrangling our data.

4. Wrangling.

Explore your data with summary tables and statistics. ✓ Use all six core dplyr functions (filter, select, arrange, group_by, mutate, summarize) at least once in your report to manipulate your dataset.

✓ Use mutate to create a variable that is a function of at least one other variable 9e.g., change the scale of a numeric variable or create a categorical variable based on a numeric variable).

```
# Use mutate to create the variable avg_goals_per_match, which calculates the average nu
mber of goals scored per match during a season for a team
football_stats <- football_stats %>%
  mutate(avg_goals_per_match = scored / matches) %>%

  # Put avg_goals_per_match before goals scored
  select(1, 2, 3, 4, 5, 6, 7, 8, avg_goals_per_match, everything())

football_stats
```

League <chr>	Y... <dbl>	position <dbl>	team <chr>	matc... <dbl>	w... <dbl>	dr... <dbl>	losses <dbl>	avg_goals_per_m
La Liga	2016	1	Real Madrid	38	29	6	3	2.78
La Liga	2016	1	Real Madrid	38	29	6	3	2.78
La Liga	2016	2	Barcelona	38	28	6	4	3.05
La Liga	2016	2	Barcelona	38	28	6	4	3.05
La Liga	2016	3	Atletico Madrid	38	23	9	6	1.84
Premier League	2016	1	Chelsea	38	30	3	5	2.23
Premier League	2016	2	Tottenham	38	26	8	4	2.26
Premier League	2016	2	Tottenham	38	26	8	4	2.26
Premier League	2016	2	Tottenham	38	26	8	4	2.26
Premier League	2016	3	Manchester City	38	23	9	6	2.10

1-10 of 10 rows | 1-10 of 38 columns

We can see from this that:

The average goals per match in a season is created as a new variable.

✓ Compute summary statistics for at least 2 numeric variables and 1 categorical variable (if a group project, describe at least 3 numeric and 2 categorical variables). If you do not have a categorical variable, create one with mutate!

```
# Compute summary statistics for at least 2 numeric variables and 1 categorical variable
football_stats %>%
  summarize(mean_XG = mean(xG.x, na.rm = T), # ignore NA values
            mean_wins = mean(wins, na.rm = T), # ignore NA values
            n_teams = n_distinct(team)) # number of distinct teams
```

mean_XG <dbl>	mean_wins <dbl>	n_teams <int>
72.08957	23.4902	29

1 row

We can see from these summary statistics that:

The mean expected goals (xG) for the whole dataset is **72.08957** expected goals for a season

The mean number of wins for teams in the top 4 in each league from 2016 to 2019 is **23.4902**

The number of distinct teams that finished in the top 4 in each league from 2016 to 2019 is **29** teams

✓ Discuss all findings/results. Make sure to include units for describing numeric variables.

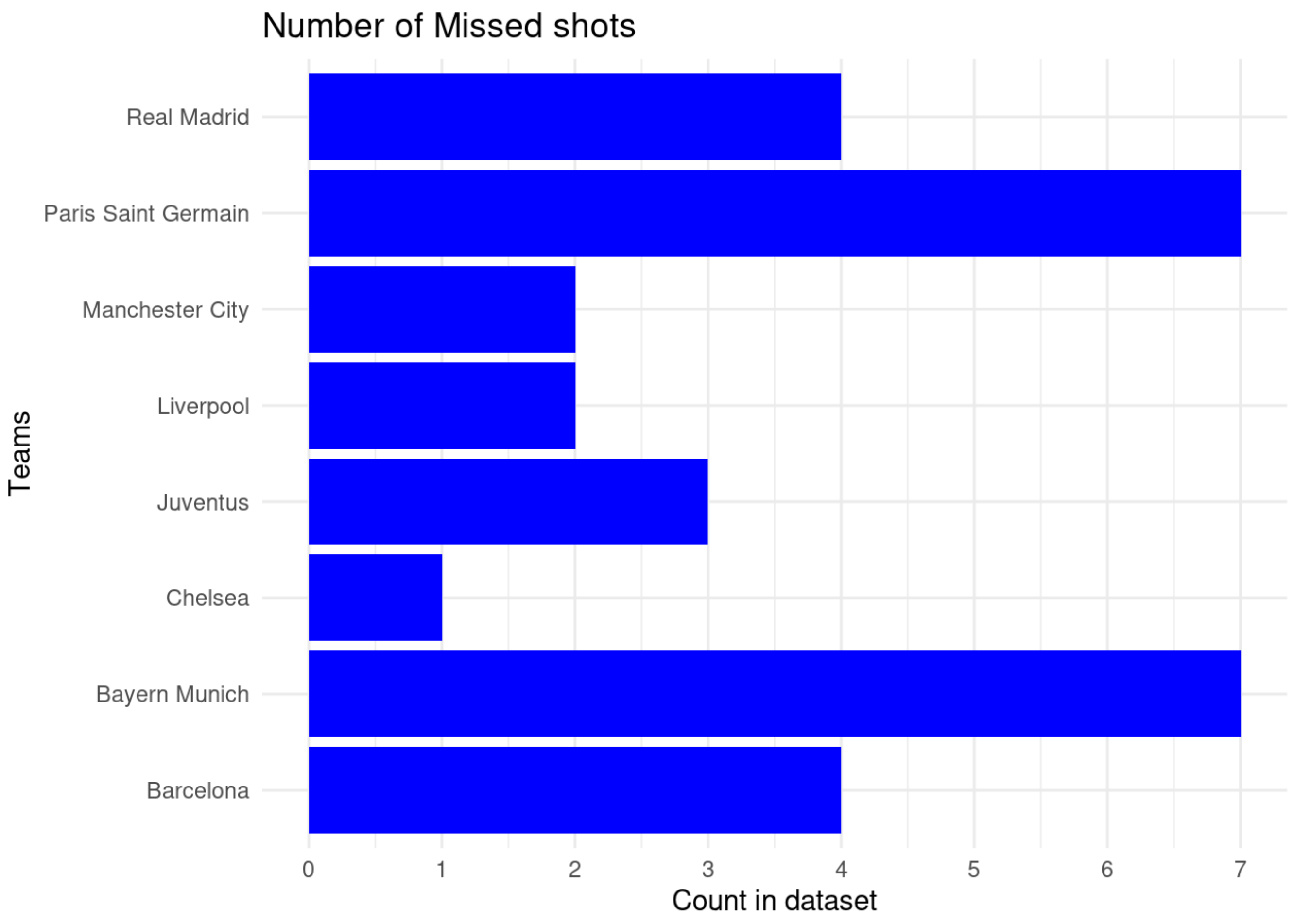
Done!

5. Visualizing.

```
Create visualizations for your data with 3 different
ggplot: include a visualization with 1 variable, another with 2
variables, and one with 3 variables (if a group project, make two
plots of each). ✓ Each plot should have a title and clean labeling
for all mappings. ✓ Modify the default theme and scales on the axes
at least once per plot. ✓ For at least one plot, use a stat
function. ✓ Write a supporting paragraph below each plot describing
what the plot depicts and any relationships/trends that are
apparent.
```

1.) 1 variable:

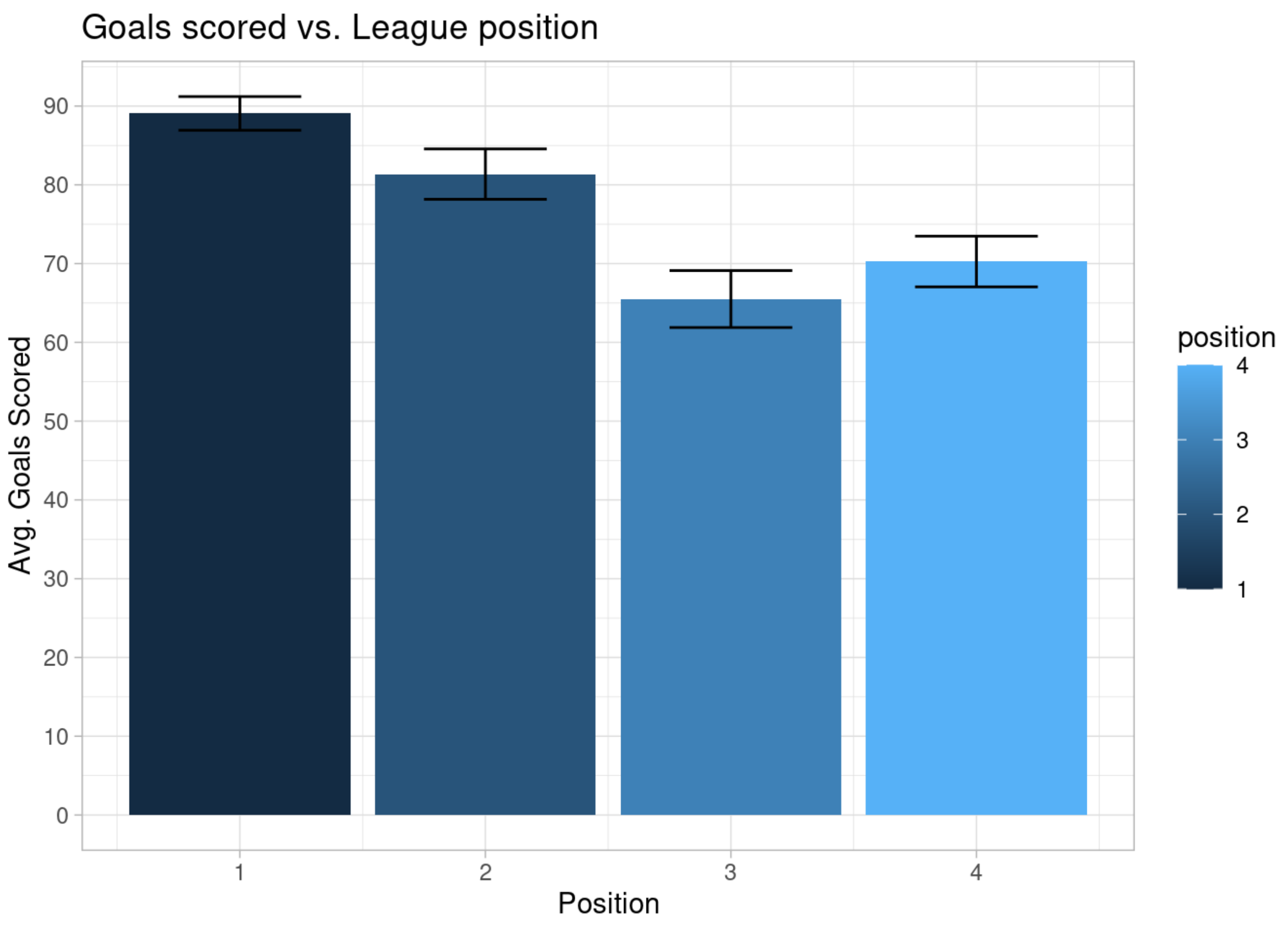
```
# 1 variables: This plot shows the count of the number of players for which we have data
for who played for a team that were champions in each of the top 5 European football lea
gues from 2016 to 2019

football_stats %>%
  filter(position == 1) %>% # filter to only include the league winners
  group_by(team, League, Year, position) %>%

  # Create a bar graph showing the number of teams that finished first in their league
  ggplot(aes(y = team)) +
  geom_bar(fill = 'blue') +
  scale_x_continuous(breaks = seq(0, 10, 1)) + # modify the scales on the axes
  labs(title = "Number of Missed shots",  # adjust labels
       x = "Count in dataset", y = "Teams") +
  theme_minimal() # modify the default theme
```



This plot depicts the number of teams from the top 5 European leagues that finished first in their league from 2016 to 2019. It also shows us which teams were champions, and the number of times they appear in our dataset. This is important to us because it shows us the data we have on the champions, and this could explain further relationships later on.

2.) 2 variables:

```
# 2 variables: This plot shows the relationship between the average number of goals scor
ed and the position at the end of the league's season, for the top 5 European football l
eagues, from 2016 to 2019

football_stats %>%
  group_by(team, League, Year, position) %>%


  # Create a bar graph showing the average number of goals scored and a team's position

  ggplot(aes(y = scored, x = position, fill = position)) +
  geom_bar(stat = "summary", fun = "mean") + # use a stat function


  # Add error bars showing + or - 1 SE
  geom_errorbar(stat = "summary", fun.data = "mean_se", width = 0.5) +

  scale_x_continuous(breaks = seq(0, 5, 1)) + # modify the scales on the axes
  scale_y_continuous(breaks = seq(0, 120, 10)) +
  labs(title = "Goals scored vs. League position",   # adjust labels
       x = "Position", y = "Avg. Goals Scored") +

  theme_light() # modify the default theme
```



This plot depicts the relationship between the position in the league for the top 5 European football leagues and the average number of goals scored in a season (from 2016 to 2019). It also depicts error bars showing + or - 1 standard deviation in average number of goals scored.

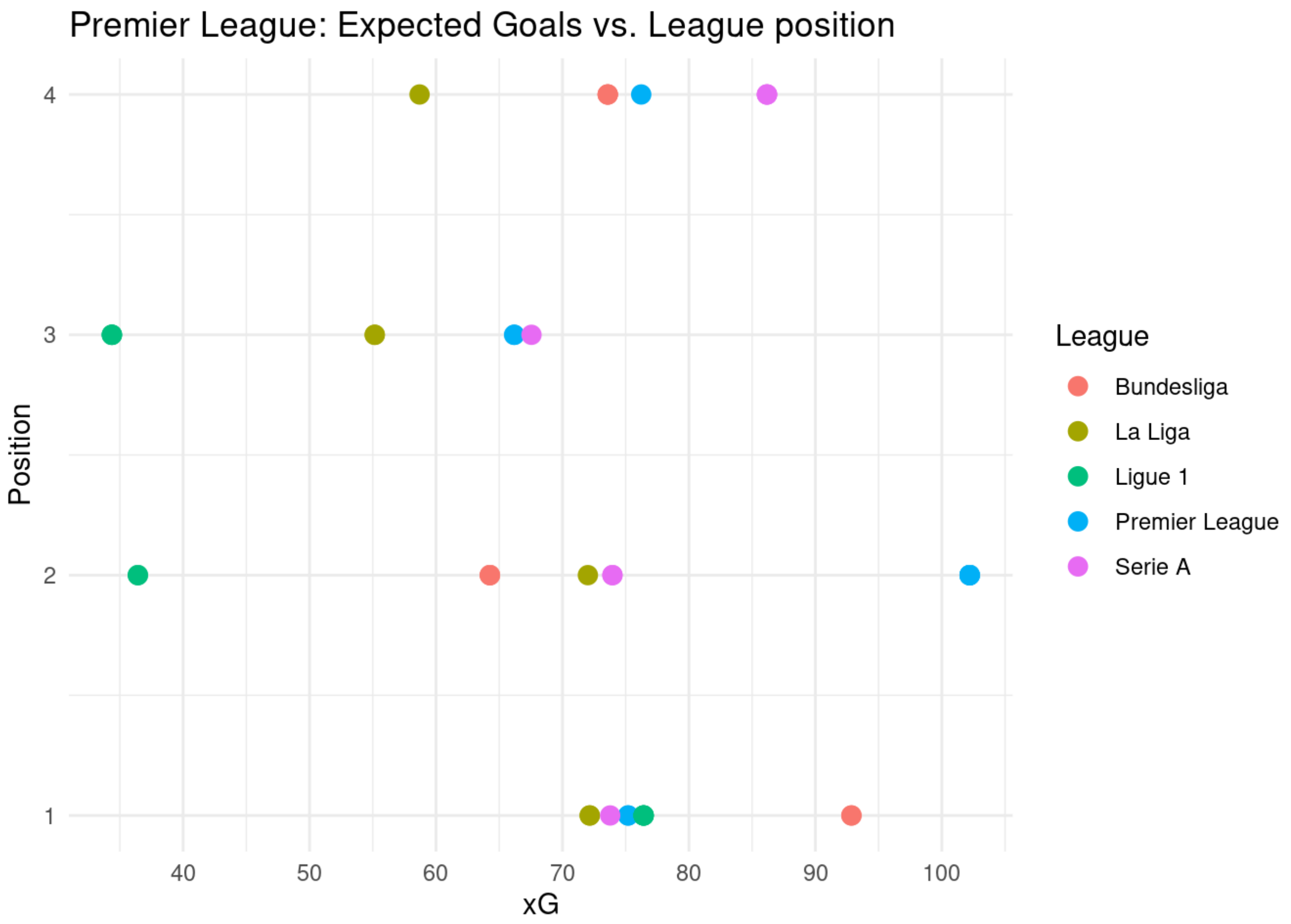
We can see from this plot that, on average, the team that scores the most goals in a season will win the league.

Something I found interesting was that the bar for the fourth position in the league was higher than that of the third position. This can be explained by a number of factors, such as the team in third winning by a small number of goals in lots of games compared to the team in fourth winning by a large number of goals in a few games.

Also, all teams in the top 4 qualify for the Champions League, which means that the third team doesn't have to try as hard during the end of the season, but the teams fighting for fourth do, so they may try harder and score more goals towards the end of the season

3.) 3 variables:

```
# 3 variables: This plot shows the relationship between expected goals (xG) and position
in the league, for the top 5 European football leagues, in the year 2019 (the most recen
t year for which there's complete data)

football_stats %>%
  filter(Year == 2019) %>% # only include data for the year 2019
  group_by(League, Year, position) %>%

  # Create a scatterplot showing the relationship between expected goals and a team's po
sition in the league
  ggplot(aes(x = xG.x, y = position, color = League)) +
  geom_point(size = 3) + # Use geom_bar()

  scale_x_continuous(breaks = seq(30, 110, 10)) + # modify the scales on the axes
  scale_y_continuous(breaks = seq(0, 5, 1)) +

  labs(title = "Premier League: Expected Goals vs. League position", # adjust labels
       x = "xG", y = "Position") +
    theme_minimal() # modify the default theme
```



This plot depicts the relationship between the expected number of goals in the league (xG) for the top 5 European football leagues and the position in the league at the end of the season (in the year 2019)

We can see from this plot that: teams with a higher number of expected goals tend to finish higher than teams with a lower amount of expected goals.

Something I found interesting was that the teams that finished in the fourth position in the league had a very high amount of expected goals compared to the second and third position, sometimes even a greater amount!

This agrees with what we saw in the last plot, where teams that finished fourth had a higher number of goals scored on average in the dataset than teams that finished third.

This could be explained by the fact that getting the fourth position means getting a spot in the Champions League, the top European club competition, which leads teams to fight very hard for that fourth spot and create more chances towards the end of the season.

Ultimately, this could also mean that: a higher amount of expected goals could be correlated to a higher amount of goals scored, looking at this plot and the last one.

✓ Modify the default theme and scales on the axes

at least once per plot.

Done!

✓ For at least one plot, use a stat

function.

Done!

✓ Write a supporting paragraph below each plot describing

what the plot depicts and any relationships/trends that are

apparent.

Done!

6. Discussion.

```
Putting it all together, what did you learn from your
data? ✓ Answer your research question(s) citing important statistics
and visualizations (it is a good idea to number your visualizations
to refer to them in this section for example). ✓ Reflect on the
process of conducting this project. What was challenging, what have
you learned from the process itself?\
✓ Include acknowledgements for any help received (if a group
project, that is where you report the contribution of each member:
who did what).
```

Research Question: Does better performance in terms of football statistics (Goals, xG, etc.) have a positive relationship with winning the league?

Looking at the analysis done in this project, the wrangling and the visualizations, I learned a lot from my data. I learned that the greater average number of goals scored in a season seems to lead to finishing higher in the league, and the greater number of expected goals in a season seems to lead to finishing higher in the league.

Answering my research question: I can say that better performance in terms of football statistics does in fact have a positive relationship with winning the league. We can see this from visualization #2, which depicts the average number of goals scored vs. league position, and visualization #3, which depicts expected goals (xG) vs. league position.

✓ Reflect on the

process of conducting this project. What was challenging, what have

you learned from the process itself?

The process of conducting this project was pretty challenging.

Tidying my data took the longest, and trying to figure out how to make my data make sense in terms of what I was looking for was difficult and I had to refer to lots of concepts and techniques to do so.

Doing the visualizations was also challenging, because I wasn't sure what I wanted to visualize at first or if it made sense, so I kept trying different things. Also, trying to figure out why my visualizations didn't look the way I wanted them to was a challenge.

I have learned that the concepts and work in data science and R/R Studio can be straightforward at times, but they need careful attention and diligence to truly make the most out of analyzing one's data.

✓ Include acknowledgements for any help received (if a group

project, that is where you report the contribution of each member:

who did what).

I received help from our TAs and Dr. Guyot on some things. Other than that, nothing else really.

Used the worksheets and homeworks to help me with things I had trouble with.

7. Formatting.

```
Create the report using R Markdown, with headers for
each section; include comments to the R code; include references
(datasets, context). The final report should be no more than 20
pages (the number of pages can vary greatly depending on the
cleaning process). It is extremely important that you select pages
when submitting on Gradescope.
```