title: “The Guardian Knowledge July 2019” author: “Robert Hickman” date: ‘2019-07-07’ output: html_document: df_print: paged header: caption: ” image: ” slug: guardian_knowledge_july tags: - rstats - football - the_knowledge categories: [] —

```
library(tidyverse)
library(broom)
library(engsoccerdata)
library(rvest)

set.seed(3459)
```

Question 1

The first question this week is

I wonder if any of any sporting leagues have ever ended in alphabetical order? pic.twitter.com/you6u8Uzwz

— P A Hunt (@TeachFMaths) June 15, 2019

```
league_data <- engsoccerdata::england %>%
  select(season = Season, division, home, visitor, hgoal, vgoal) %>%
  gather("location", "team", -season, -division, -hgoal, -vgoal) %>%
  mutate(
    g_for = case_when(
      location == "home" ~ hgoal,
      location == "visitor" ~ vgoal
    ),
    g_ag = case_when(
      location == "home" ~ vgoal,
      location == "visitor" ~ hgoal
    )) %>%
  mutate(
    points = case_when(
      g_for > g_ag & season < 1981 ~ 2,
      g_for > g_ag & season > 1980 ~ 3,
      g_for == g_ag ~ 1,
      g_for < g_ag ~ 0
    ),
    gd = g_for - g_ag
  ) %>%
  group_by(season, division, team) %>%
  summarise(points = sum(points),
            gd = sum(gd),
            g_for = sum(g_for)) %>%
  arrange(-points, -gd, -g_for) %>%
  mutate(league_pos = rank(-points, ties.method = "first"),
         alph_order = rank(team, ties.method = "first")) %>%
  select(season, division, team, league_pos, alph_order) %>%
  split(., f = list(.$season, .$division)) %>%
  keep(function(x) nrow(x) > 0)
```

```
correlations <- league_data %>%
  map_df(., function(data) {
    cor.test(
      data$league_pos,
      data$alph_order,
      method = "spearman"
    ) %>%
      tidy() %>%
      mutate(season = unique(data$season),
             division = unique(data$division))
  }) %>%
  filter(p.value < 0.05)
```

Let's imagine a very small league (say 8 teams).

```
first_letter_names <- league_data %>%
  bind_rows() %>%
  ungroup() %>%
  mutate(first_letter = gsub("(^.)(.*)", "\\1", team)) %>%
  filter(season > 1992 &
           division == 1 &
           first_letter %in% toupper(letters[1:6])
         ) %>%
  filter(!duplicated(first_letter)) %>%
  select(team) %>%
  arrange(team) %>%
  print()
```

```
## # A tibble: 6 x 1
##   team
##   <chr>
## 1 Arsenal
## 2 Blackburn Rovers
## 3 Coventry City
## 4 Derby County
## 5 Everton
## 6 Fulham
```

So for the league to finish in alphabetical order, we first need the team that is first alphabetically (Arsenal) to finish in first position. Assuming all teams have an equal chance of winning the league, the chance of this is obviously

$$p(Arsenal = 1) = \frac{1}{n}$$

Then we need the second team (Blackburn Rovers), to finish in second. This is predicated on Arsenal already finishing in first position, so the chance becomes

$$p(Blackburn = 2|Arsenal = 1) = \frac{1}{n-1}$$

and so on until the last team (Fulham) just have to slot into the only position left (n, 6th in our example)

Thus the total chance becomes

$$\frac{1}{n} \cdot \frac{1}{n-1} ... \cdot \frac{1}{1}$$

which can also be written

$$p(ordered) = \prod_{n=1}^{N} \frac{1}{n}$$

which multiplies out to

$$p(ordered) = \frac{1}{n!}$$

so for our very small league the chance of n (assumed equally strong teams)

```
factorial(nrow(first_letter_names))
```

```
## [1] 720
```

so we have a 1/720 chance that this league ends perfectly in alphabetical order. For bigger leagues (for reference most large European leagues contain 18-24 teams) this number *super-exponentially* and is tiny.

For the English Premier League (20 teams) for instance the chance becomes

```
league_data %>%
  bind_rows() %>%
  ungroup() %>%
  filter(season == max(season) & division == 1) %>%
  nrow() %>%
  factorial()
```

```
## [1] 2.432902e+18
```

or 1 in 2.4 quintillion. In short, if it's assumed that there's no relation between order of names and team strength, we might expect the universe to end before all 20 teams finish in perfect order.

We can test if our predictions bear out by looking at tiny leagues with small numbers of teams, e.g. the group stages of the Champions/Europa Leagues.

First we need to scrape the final tables for the last 8 years of data from both competitions:

```
#website to scrape group stage data from
fb_data <- "https://footballdatabase.com"
ucl_links <- sprintf(
  "/league-scores-tables/uefa-champions-league-20%s-%s",
  10:18, 11:19
)
europa_links <- sprintf(
  "/league-scores-tables/uefa-europa-league-20%s-%s",
  10:18, 11:19
)

#function to scrape the data from these links
get_competition_data <- function(competition, links) {
  data <- links %>%
    paste0(fb_data, .) %>%
    map_df(., function(year) {
      page_read <- read_html(year)

      groups <- letters[1:8] %>%
        map_df(., function(group) {
          page_read %>%
            html_nodes(sprintf("#total-group-%s > div > table", group)) %>%
            html_table(fill = TRUE) %>%
            as.data.frame() %>%
            mutate(group)
        }) %>%
        mutate(year = gsub("(.*-)([0-9]{4}-[0-9]{2})", "\\2", year))
    }) %>%
    mutate(competition)
}

#scrape and bind the data
uefa_data <- bind_rows(
  get_competition_data("champions", ucl_links),
  get_competition_data("europa", europa_links)
)

#print a cutdown version of the scraped data
head(uefa_data %>% select(club = Club, points = P, year, competition))
```

```
##                 club points    year competition
## 1 Tottenham Hotspur     11 2010-11   champions
## 2       Inter Milan     10 2010-11   champions
## 3         FC Twente      6 2010-11   champions
## 4     Werder Bremen      5 2010-11   champions
## 5        Schalke 04     13 2010-11   champions
## 6              Lyon     10 2010-11   champions
```

So now we have 128 (8 groups x 8 years x 2 competitions) 'mini-leagues' each of 4 teams.

We can then munge this data to find all the groups where the teams finish in alphabetical order. We'd expect 128/4! leagues to finish in alphabetical order (or 5.33 to be exact).

```
ordered_groups <- uefa_data %>%
  #select relevant informatiob
  select(team = Club, league_pos = X., group, year, competition) %>%
  #by group find where teams finish in alphabetical order
  group_by(year, group, competition) %>%
  mutate(alph_order = rank(team, ties.method = "first")) %>%
  filter(league_pos == alph_order) %>%
  #keep only group where all (4) teams finish in order
  summarise(n = n()) %>%
  filter(n == 4) %>%
  #join and filter back data
  left_join(uefa_data, ., by = c("group", "year", "competition")) %>%
  filter(!is.na(n)) %>%
  #select useful information
  select(team = Club, points = P, gd = X..., league_pos = X.,
         group, year, competition) %>%
  #split groups up
  split(., list(.$year, .$group, .$competition)) %>%
  keep(function(x) nrow(x) > 0)
```

which leaves us with 5 leagues that have finished in order! almost exactly what we'd predict by chance if the first letter of a teams name had no effect on the outcome.

```r
ordered_groups
```

```
## $`2011-12.c.champions`
##                 team points gd league_pos group    year competition
## 5            Benfica     12  4          1     c 2011-12   champions
## 6           FC Basel     11  1          2     c 2011-12   champions
## 7 Manchester United       9  3          3     c 2011-12   champions
## 8      Otelul Galati      0 -8          4     c 2011-12   champions
##
## $`2015-16.c.champions`
##                 team points gd league_pos group    year competition
## 9    Atlético Madrid     13  8          1     c 2015-16   champions
## 10           Benfica     10  2          2     c 2015-16   champions
## 11       Galatasaray      5 -4          3     c 2015-16   champions
## 12 Lokomotiv Astana      4 -6          4     c 2015-16   champions
##
## $`2010-11.f.champions`
##              team points  gd league_pos group    year competition
## 1      Chelsea FC     15  10          1     f 2010-11   champions
## 2       Marseille     12   9          2     f 2010-11   champions
## 3 Spartak Moskva       9  -3          3     f 2010-11   champions
## 4          Žilina      0 -16          4     f 2010-11   champions
##
## $`2015-16.g.champions`
##                   team points  gd league_pos group    year competition
## 13          Chelsea FC     13  10          1     g 2015-16   champions
## 14         Dynamo Kyiv     11   4          2     g 2015-16   champions
## 15            FC Porto     10   1          3     g 2015-16   champions
## 16 Maccabi Tel Aviv FC      0 -15          4     g 2015-16   champions
##
## $`2018-19.h.champions`
##                  team points gd league_pos group    year competition
## 17          Juventus     12  5          1     h 2018-19   champions
## 18 Manchester United     10  3          2     h 2018-19   champions
## 19          Valencia      8  0          3     h 2018-19   champions
## 20        Young Boys      4 -8          4     h 2018-19   champions
```

We can also do a larger test by randomly selecting teams out of the English league data we looked at earlier. To do this I need two quick functions= one to sample randomly from the data, and another to carry out the correlation test.

The first takes a number of samples (how many tests to run) and then selects a number of teams from each league sample. For instance, if I chose 3 teams, it might select Livepool, Manchester United, and Watford, from the latest season of the Premier League. These teams finished 2nd, 6th, and 11th respecitvely, so this 'sampled league' would fulfill the criteria of finishing in alphabetical order.

```r
#take a random sample of leagues and teams withing those leagues
sample_cutdown_leagues <- function(nteams, nsamples, data) {
  samples <- sample(length(data), nsamples, replace = TRUE)

  sampled_league_data <- data[samples]

  league_team_serials <- sampled_league_data %>%
    lapply(., nrow) %>%
    lapply(., sample, size = nteams)
  #carry out the correlation test
  league_cor_test <- map2_df(
    .x = sampled_league_data,
    .y = league_team_serials,
    .f = cor_test_data
  )
}

#function for correlation test
cor_test_data <- function(full_league_data, sampled_teams) {
  sampled_league <- full_league_data[sampled_teams,] %>%
    arrange(league_pos)
  cor_test <- cor.test(
    sampled_league$league_pos,
    sampled_league$alph_order,
    method = "spearman"
  ) %>%
    tidy() %>%
    #mutate on information about that season and teams chosen
    mutate(teams = paste(sampled_league$team, collapse = ", "),
           season = unique(sampled_league$season),
           division = unique(sampled_league$division))
}
```

So for instance if I just run it once, randomly selecting 4 teams:

```
test <- sample_cutdown_leagues(4, 1, league_data)

#print the teams selected
test$teams
```

```
## [1] "Walsall, Gillingham, Notts County, Lincoln City"
```

```
test
```

```
## # A tibble: 1 x 8
##   estimate statistic p.value method    alternative teams    season division
##      <dbl>     <dbl>   <dbl> <chr>     <chr>       <chr>     <dbl> <chr>
## 1     -0.4        14    0.75 Spearman~ two.sided   Walsall~   1998 3
```

It gives me 4 teams from the 1924 division 2 championship which indeed also

We can then carry this out with 10000 samples for nteam numbers of 2:6 to see if we get roughly the expected numbers of exactly correlated league finish positions (this will take 1-2mins) by finidng out how many tests give an estimate of 1 (finished exactly correlated with alphabetical order) or -1 (finished exactly anti-correlated with alphabetical order).

Both these numbers should be roughly equal to the number of samples (10000) divided by the factorial of the number of teams selected.

```
test_n_numbers <- function(nteams) {
  sampling <- sample_cutdown_leagues(nteams, 10000, league_data)

  correlated <- length(which(sampling$estimate == max(sampling$estimate)))
  anti_correlated <- length(which(sampling$estimate == min(sampling$estimate)))

  expected <- nrow(sampling) / factorial(nteams)
  df <- data.frame(n = nteams,
                   sample_cor = correlated,
                   sample_anticor = anti_correlated,
                   sample_expected = expected)
}

#run the function
testing <- map_df(2:6, test_n_numbers)

#print results
print(testing)
```

```
##   n sample_cor sample_anticor sample_expected
## 1 2       4985           5015      5000.00000
## 2 3       1609           1650      1666.66667
## 3 4        408            412       416.66667
## 4 5         66             50        83.33333
## 5 6         12             11        13.88889
```

Finally, we can do a Pearson's product moment correlation test to see if there is any relationship between alphabetical team name order and final league finish for all out our English league data

```
all_data <- league_data %>%
  bind_rows()

pearsons_test <- cor.test(all_data$alph_order,
                          all_data$league_pos) %>%
  tidy() %>%
  print()
```

```
## # A tibble: 1 x 8
##   estimate statistic p.value parameter conf.low conf.high method
##      <dbl>     <dbl>   <dbl>     <int>    <dbl>     <dbl> <chr>
## 1   0.0134      1.29   0.199      9186 -0.00704    0.0339 Pears~
## # ... with 1 more variable: alternative <chr>
```

And can see that, as expected, there's no real relationship between the two and so it makes sense that we only see it happen on a chance level.

Question 2

Which team has had to travel the shortest combined distance in a cup run? (excluding regional competitions, just to make it interesting)

— Chris van Thomas (@chrisvanthomas) July 10, 2019