One of the most consistent [fonts of posts on this blog](https://www.robert-hickman.eu/project/guardian_knowledge/) is The Guardian’s football trivia page [The Knowledge](https://www.theguardian.com/football/series/theknowledge). A particular reason for this is that the small contained questions lend themselves to small blogposts that I can turn around in an hour or two, as opposed to being endlessly redrafted until I lose interest.

However, I still sometimes don’t quite get round to finishing some of these posts, or have trouble justifying a blog post on a very small and ‘trivial’ answer to a question. Therefore, as a sort of end-of-year round up, and a Christmas present to myself, I wanted to push out answers to questions I found particularly interesting over the last year and hadn’t quite got round to 1. I’ll probably add them all to this post as I finish them up.

**2nd December – Everything in its right place**

I wonder if any of any sporting leagues have ever ended in alphabetical order? [pic.twitter.com/you6u8Uzwz](https://t.co/you6u8Uzwz)

— P A Hunt (@TeachFMaths) [June 15, 2019](https://twitter.com/TeachFMaths/status/1139832761295024128?ref_src=twsrc%5Etfw)

**Answer – yes, kind of. But also no.**

This question has actually [been answered](https://www.theguardian.com/football/2011/mar/09/has-league-ever-finished-alphabetical-order) (as many of these will have been). For a league of 20 teams (like the English Premier League), we might imagine if would have happened over the last ~150 years, but it’s almost certain from some basic maths that it won’t have, and moreover, will never happen.

Let’s load some data and see why.

#as per usual, going to heavily rely on tidyverse

#and engsoccerdata throughout these posts

library(tidyverse)

library(engsoccerdata)

#load English league data

league\_data <- engsoccerdata::england %>%

#select and gather match results

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

)) %>%

#get correct point for a win/loss

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 and league and get final tables

group\_by(season, division, team) %>%

summarise(points = sum(points),

gd = sum(gd),

g\_for = sum(g\_for)) %>%

arrange(-points, -gd, -g\_for) %>%

#rank league order and alphabetical order

mutate(league\_pos = rank(-points, ties.method = "first"),

alph\_order = rank(team, ties.method = "first")) %>%

select(season, division, team, league\_pos, alph\_order) %>%

#split by league and season

split(., f = list(.$season, .$division)) %>%

keep(function(x) nrow(x) > 0)

#print the top of the first league table

head(league\_data[[1]])

## # A tibble: 6 x 5

## # Groups: season, division [1]

## season division team league\_pos alph\_order

##

## 1 1888 1 Preston North End 1 9

## 2 1888 1 Aston Villa 2 2

## 3 1888 1 Wolverhampton Wanderers 3 12

## 4 1888 1 Blackburn Rovers 4 3

## 5 1888 1 Bolton Wanderers 5 4

## 6 1888 1 West Bromwich Albion 6 11

We can then run a load of Spearman’s rank correlation tests on the data to see which ones are perfectly correlated or anti-correlated in both league and alphabetical ranking. We’ll use the very handy [broom](https://cran.r-project.org/web/packages/broom/vignettes/broom.html) package to tidy the results of our many tests into one data.frame (remove the filter at the end of the pipe chain to see what gets output).

#use broom to tidily do stats

library(broom)

#correlate league and alphabetical order by year

exact\_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))

}) %>%

#take only significantly

filter(abs(statistic) == 1)

print(exact\_correlations)

## # A tibble: 0 x 7

## # ... with 7 variables: estimate , statistic , p.value ,

## # method , alternative , season , division

And so we find no exact correlations. There are no instances in 363 separate seasons of English league football where teams line up in either alphabetical, or anti-alphabetical order.

Let’s see why this is. To make things simpler, I’m going to imagine a cutdown league of only 6 teams using teams starting with each of the first 6 letter of the alphabet:

first\_letter\_names <- league\_data %>%

bind\_rows() %>%

ungroup() %>%

#get first letter of a team name

mutate(first\_letter = gsub("(^.)(.\*)", "\\1", team)) %>%

filter(season > 1992 &

division == 1 &

first\_letter %in% toupper(letters[1:6])

) %>%

#get one team beginning with A, B, C...

filter(!duplicated(first\_letter)) %>%

select(team) %>%

arrange(team) %>%

print()

## # A tibble: 6 x 1

## team

##

## 1 Arsenal

## 2 Blackburn Rovers

## 3 Coventry City

## 4 Derby County

## 5 Everton

## 6 Fulham

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 denominator grows *super-exponentially* and becomes 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](https://en.wikipedia.org/wiki/Order_of_magnitude). 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](https://en.wikipedia.org/wiki/2018%E2%80%9319_UEFA_Champions_League_group_stage).

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

library(rvest)

#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](http://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.

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 Liverpool, Manchester United, and Watford, from the [last season of the Premier League](https://en.wikipedia.org/wiki/2018%E2%80%9319_Premier_League). These teams finished 2nd, 6th, and 11th respectively, so this ‘sampled league’ would fulfill the criteria of finishing in alphabetical order.

set.seed(3459)

#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] "Brentford, Bristol Rovers, Brighton & Hove Albion, Chester"

test

## # A tibble: 1 x 8

## estimate statistic p.value method alternative teams season division

##

## 1 0.8 2.00 0.333 Spearma~ two.sided Brentfor~ 1994 3

It gives me 4 teams from the 1994 division 3 who didn’t finish in alphabetical order (though, amusingly, all have a very similar starting letter).

We can then carry this out with 10000 samples for n\_team 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 finding 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) {

#run sampling function n times

#10k should do

sampling <- sample\_cutdown\_leagues(nteams, 10000, league\_data)

#find exactly correlated and anti-correlated examples

#where teams are in exact alphabetical order ascending or descending

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 5010 4990 5000.00000

## 2 3 1676 1665 1666.66667

## 3 4 367 398 416.66667

## 4 5 101 81 83.33333

## 5 6 14 15 13.88889

And the numbers line up, as we would expect if there is no effect of the first letter of a team’s name upon final league position.

Finally, we can do a Kendall’s correlation test to really see if there is any relationship between alphabetical team name order and final league finish for all out our English league data. We use Kendall instead of a Spearman test here because we grouping all the data together we’re going to have a lot of ties (one team has to finish 1st in every league each year).

all\_data <- league\_data %>%

bind\_rows()

#do a big correlation test

kendall\_test <- cor.test(all\_data$alph\_order,

all\_data$league\_pos,

alternative = "two.sided",

method = "kendall") %>%

tidy() %>%

print()

## # A tibble: 1 x 5

## estimate statistic p.value method alternative

##

## 1 0.0135 1.74 0.0826 Kendall's rank correlation tau two.sided

And we can see that, even though our p-value is [‘approaching significance’](https://mchankins.wordpress.com/2013/04/21/still-not-significant-2/), it’s not significant at our fairly liberal threshold of 0.05. Even then, the effect size (0.013) is tiny, so there’s no need for Watford to start worrying [just yet](https://www.bbc.co.uk/sport/football/50619972).

1. SMALL DIGRESSION: I love blogging on this site and it also has been a great help to me in numerous ways (practice coding/writing, feeling like a “programmer”, for job interviews), but quite a lot of the time feel posts are not quite where I want them (I’m sure this feeling isn’t restricted to me) and so won’t put them up and so that time (sometimes quite a few hours!) I put into them in my spare time feels wasted and makes me feel worse about myself. I’m hoping that pushing out fairly rushed/half formed ideas like this will help with this.