Post Abril

Dominic Royé

18 de abril de 2019

Cuando pretendemos estimar la correlación entre múltiples variables, la tarea se complica para obtener un resultado simple y limpio. Una forma sencilla es usar la función `tidy()` del paquete *{broom}*. Como ejemplo, en este post vamos a estimar la correlación entre la precipitación anual de varias ciudades españolas y varios índices de teleconexiones climáticas: descarga. Los datos de las teleconexiones están preprocesados, pero pueden ser descargados directamente desde crudata.uea.ac.uk. La preciptiación proviene de ECA&D.

Paquetes

En este post usaremos los siguientes paquetes:

Paquete	Descripción
tidyverse	Conjunto de paquetes (visualización y manipulación de datos): ggplot2, dplyr, purrr,etc.
broom	Convierte resultados de funciones estadísticas (lm, t.test, cor.test, etc.) en bonitas tablas
fs	Proporciona una interfaz uniforme y multiplataforma para las operaciones del sistema de archivos
lubridate	Fácil manipulación de fechas y tiempos

```
#instalamos los paquetes si hace falta
if(!require("tidyverse")) install.packages("tidyverse")
if(!require("broom")) install.packages("broom")
if(!require("fs")) install.packages("fs")
if(!require("lubridate")) install.packages("lubridate")

#librerías
library(tidyverse)
library(broom)
library(fs)
library(lubridate)
```

Importar datos

Primero debemos importar la precipitación diaria de las estaciones meteorológicas seleccionadas.

1. Creamos un vector con todos los archivos de precipitación con la función `dir_ls()` del paquete *{fs}*.
2. Importamos los datos con ayuda de la función `map_df()` del paquete *{purrr}* que aplica otra función a un vector o lista, y los une en una única tabla.
3. a) Seleccionamos únicamente las columnas que nos interesan, b) Convertimos la fecha en objeto *date* con la función `ymd()` del paquete *{lubridate}*, c) Creamos una nueva columna *yr* con el año, d) Dividimos la precipitación entre 10 y reclasificamos valores ausentes -9999 por NA, e) Por último, reclasificamos la ID de cada estación meteorológica, creando un factor con nuevas etiquetas.

Más detalles sobre el uso de las funciones `dir_ls()` y `map_df()` en este último post.

```
#archivos de la precipitación
files <- dir_ls(regexp = "txt")
files
```

```
## RR_STAID001393.txt RR_STAID001394.txt RR_STAID002969.txt
```

```
## RR_STAID003946.txt RR_STAID003969.txt
```

```r
#importamos todos, uniéndolos en una única tabla
pr <- files %>% map_df(read_csv,skip=20)
```

```
## Parsed with column specification:
## cols(
##   STAID = col_double(),
##   SOUID = col_double(),
##   DATE = col_double(),
##   RR = col_double(),
##   Q_RR = col_double()
## )
## Parsed with column specification:
## cols(
##   STAID = col_double(),
##   SOUID = col_double(),
##   DATE = col_double(),
##   RR = col_double(),
##   Q_RR = col_double()
## )
## Parsed with column specification:
## cols(
##   STAID = col_double(),
##   SOUID = col_double(),
##   DATE = col_double(),
##   RR = col_double(),
##   Q_RR = col_double()
## )
## Parsed with column specification:
## cols(
##   STAID = col_double(),
##   SOUID = col_double(),
##   DATE = col_double(),
##   RR = col_double(),
##   Q_RR = col_double()
## )
## Parsed with column specification:
## cols(
##   STAID = col_double(),
##   SOUID = col_double(),
##   DATE = col_double(),
##   RR = col_double(),
##   Q_RR = col_double()
## )
```

```r
pr
```

```
## # A tibble: 133,343 x 5
##    STAID SOUID     DATE    RR  Q_RR
##    <dbl> <dbl>    <dbl> <dbl> <dbl>
##  1  1393 20611 19470301     0     0
##  2  1393 20611 19470302     5     0
##  3  1393 20611 19470303     0     0
##  4  1393 20611 19470304    33     0
##  5  1393 20611 19470305    15     0
```

```
##  6  1393 20611 19470306     0     0
##  7  1393 20611 19470307    85     0
##  8  1393 20611 19470308     3     0
##  9  1393 20611 19470309     0     0
## 10  1393 20611 19470310     0     0
## # ... with 133,333 more rows
```

```
#creamos los niveles del factor
id <- unique(pr$STAID)

#las etiquetas correspondientes
lab <- c("Bilbao","Santiago","Barcelona","Madrid","Valencia")

#primeros cambios
pr <- select(pr,STAID,DATE,RR)%>%
        mutate(DATE=ymd(DATE),
               RR=ifelse(RR==-9999,NA,RR/10),
               STAID=factor(STAID,id,lab),
               yr=year(DATE))
pr
```

```
## # A tibble: 133,343 x 4
##    STAID  DATE          RR    yr
##    <fct>  <date>     <dbl> <dbl>
##  1 Bilbao 1947-03-01   0    1947
##  2 Bilbao 1947-03-02   0.5  1947
##  3 Bilbao 1947-03-03   0    1947
##  4 Bilbao 1947-03-04   3.3  1947
##  5 Bilbao 1947-03-05   1.5  1947
##  6 Bilbao 1947-03-06   0    1947
##  7 Bilbao 1947-03-07   8.5  1947
##  8 Bilbao 1947-03-08   0.3  1947
##  9 Bilbao 1947-03-09   0    1947
## 10 Bilbao 1947-03-10   0    1947
## # ... with 133,333 more rows
```

Lo que todavía nos hace falta es filtrar y calcular la suma anual de precipitación. En principio, no es lo más correcto sumar la precipitación sin tener en cuenta que haya valores ausentes, pero nos sirve igualmente para este ensayo. Después, cambiamos el formato de la tabla con la función `spread()`, pasando de una tabla larga a una ancha, es decir, queremos obtener una columna por estación meteorológica.

```
pr_yr <- filter(pr,DATE>="1950-01-01",DATE<"2018-01-01")%>%
           group_by(STAID,yr)%>%
             summarise(pr=sum(RR,na.rm = TRUE))
pr_yr
```

```
## # A tibble: 324 x 3
## # Groups:   STAID [5]
##    STAID     yr    pr
##    <fct>  <dbl> <dbl>
##  1 Bilbao  1950 1342
##  2 Bilbao  1951 1306.
##  3 Bilbao  1952 1355.
##  4 Bilbao  1953 1372.
##  5 Bilbao  1954 1428.
##  6 Bilbao  1955 1062.
```

```
##  7 Bilbao  1956 1254.
##  8 Bilbao  1957  968.
##  9 Bilbao  1958 1272.
## 10 Bilbao  1959 1450.
## # ... with 314 more rows
```

```
pr_yr <- spread(pr_yr,STAID,pr)
pr_yr
```

```
## # A tibble: 68 x 6
##       yr Bilbao Santiago Barcelona Madrid Valencia
##    <dbl>  <dbl>    <dbl>     <dbl>  <dbl>    <dbl>
##  1  1950  1342     1800.      345     NA        NA
##  2  1951  1306.    2344.     1072.   798.       NA
##  3  1952  1355.    1973.      415.   524.       NA
##  4  1953  1372.     973.      683.   365.       NA
##  5  1954  1428.    1348.      581.   246.       NA
##  6  1955  1062.    1769.      530.   473.       NA
##  7  1956  1254.    1533.      695.   480.       NA
##  8  1957   968.    1599.      635.   424.       NA
##  9  1958  1272.    2658.      479.   482.       NA
## 10  1959  1450.    2847.     1006    665.       NA
## # ... with 58 more rows
```

El siguiente paso es importar los índices de las teleconexiones.

```
#teleconexiones
telecon <- read_csv("teleconnections_indices.csv")
```

```
## Parsed with column specification:
## cols(
##   yr = col_double(),
##   NAO = col_double(),
##   WeMO = col_double(),
##   EA = col_double(),
##   `POL-EUAS` = col_double(),
##   `EATL/WRUS` = col_double(),
##   MO = col_double(),
##   SCAND = col_double(),
##   AO = col_double()
## )
```

```
telecon
```

```
## # A tibble: 68 x 9
##       yr   NAO   WeMO     EA `POL-EUAS` `EATL/WRUS`    MO    SCAND       AO
##    <dbl> <dbl>  <dbl>  <dbl>      <dbl>       <dbl> <dbl>    <dbl>    <dbl>
##  1  1950  0.49  0.555 -0.332     0.0217     -0.0567 0.335  0.301   -1.99e-1
##  2  1951 -0.07  0.379 -0.372     0.402      -0.419  0.149 -0.00667 -3.65e-1
##  3  1952 -0.37  0.693 -0.688    -0.0117     -0.711  0.282  0.0642  -6.75e-1
##  4  1953  0.4  -0.213 -0.727    -0.0567     -0.0508 0.216  0.0233  -1.64e-2
##  5  1954  0.51  1.20  -0.912     0.142      -0.318  0.386  0.458   -5.83e-4
##  6  1955 -0.64  0.138 -0.824    -0.0267      0.154  0.134  0.0392  -3.62e-1
##  7  1956  0.17  0.617 -1.29     -0.197       0.0617 0.256  0.302   -1.63e-1
##  8  1957 -0.02  0.321 -0.952    -0.638      -0.167  0.322 -0.134   -3.42e-1
##  9  1958  0.12  0.941 -0.243     0.138       0.661  0.296  0.279   -8.68e-1
## 10  1959  0.49 -0.055 -0.23     -0.0142      0.631  0.316  0.725   -7.62e-2
```

```
## # ... with 58 more rows
```

Por último nos falta unir ambas tablas por año.

```
data_all <- left_join(pr_yr,telecon,by="yr")
data_all
```

```
## # A tibble: 68 x 14
##       yr Bilbao Santiago Barcelona Madrid Valencia   NAO   WeMO     EA
##    <dbl>  <dbl>    <dbl>     <dbl>  <dbl>    <dbl> <dbl>  <dbl>  <dbl>
##  1  1950  1342     1800.      345     NA        NA  0.49  0.555 -0.332
##  2  1951  1306.    2344.     1072.   798.       NA -0.07  0.379 -0.372
##  3  1952  1355.    1973.      415.   524.       NA -0.37  0.693 -0.688
##  4  1953  1372.     973.      683.   365.       NA  0.4  -0.213 -0.727
##  5  1954  1428.    1348.      581.   246.       NA  0.51  1.20  -0.912
##  6  1955  1062.    1769.      530.   473.       NA -0.64  0.138 -0.824
##  7  1956  1254.    1533.      695.   480.       NA  0.17  0.617 -1.29
##  8  1957   968.    1599.      635.   424.       NA -0.02  0.321 -0.952
##  9  1958  1272.    2658.      479.   482.       NA  0.12  0.941 -0.243
## 10  1959  1450.    2847.     1006    665.       NA  0.49 -0.055 -0.23
## # ... with 58 more rows, and 5 more variables: `POL-EUAS` <dbl>,
## #   `EATL/WRUS` <dbl>, MO <dbl>, SCAND <dbl>, AO <dbl>
```

Test de correlación

Un test de correlación lo podemos hacer con la función `cor.test()` de *R Base*. En este caso entre la precipitación anual de Bilbao y el índice de NAO.

```
cor_nao_bil <- cor.test(data_all$Bilbao,data_all$NAO,
                        method="spearman")
```

```
## Warning in cor.test.default(data_all$Bilbao, data_all$NAO, method =
## "spearman"): Cannot compute exact p-value with ties
```

```
cor_nao_bil
```

```
##
##  Spearman's rank correlation rho
##
## data:  data_all$Bilbao and data_all$NAO
## S = 44372, p-value = 0.2126
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.1531149
```

Vemos que el resultado está en un formato poco manejable. Nos resume la correlación con todos los parametros estadísticos necesarios para sacar una conclusión sobre la relación. La función `tidy()` del paquete *{broom}* nos permite convertir el resultado en formato de tabla.

```
tidy(cor_nao_bil)
```

```
## # A tibble: 1 x 5
##   estimate statistic p.value method                          alternative
##      <dbl>     <dbl>   <dbl> <chr>                           <chr>
## 1    0.153    44372.   0.213 Spearman's rank correlation rho two.sided
```

Aplicar el test de correlación a múltiples variables

El objetivo es aplicar el test de correlación a todas las estaciones meteorológicas e índices de teleconexiones.

Primero, debemos pasar la tabla al formato largo, o sea, crear una columna de la ciudad y el valor de la precipitación correspondiente. Después lo repetimos para las teleconexiones.

```
data <- gather(data_all,city,pr,Bilbao:Valencia)%>%
                    gather(telecon,index,NAO:AO)
data
```

```
## # A tibble: 2,720 x 5
##       yr city      pr telecon index
##    <dbl> <chr>  <dbl> <chr>   <dbl>
##  1  1950 Bilbao 1342  NAO      0.49
##  2  1951 Bilbao 1306. NAO     -0.07
##  3  1952 Bilbao 1355. NAO     -0.37
##  4  1953 Bilbao 1372. NAO      0.4
##  5  1954 Bilbao 1428. NAO      0.51
##  6  1955 Bilbao 1062. NAO     -0.64
##  7  1956 Bilbao 1254. NAO      0.17
##  8  1957 Bilbao  968. NAO     -0.02
##  9  1958 Bilbao 1272. NAO      0.12
## 10  1959 Bilbao 1450. NAO      0.49
## # ... with 2,710 more rows
```

Para poder aplicar el test a todas las ciudades, debemos tener las correspondientes agrupaciones. Por ello, usamos la función `group_by()` indicando los dos grupos (*city* y *telecon*), y además, aplicamos la función `nest()` del paquete *{tidyr}*, colección *{tidyverse}*, con el objetivo de crear listas de tablas encajadas por fila. En otras palabras, en cada fila de cada ciudad y teleconexión tendremos una nueva tabla que contiene correspondientemente el año, la precipitación y el valor del índice.

```
data_nest <- group_by(data,city,telecon)%>%nest()
data_nest
```

```
## # A tibble: 40 x 3
##    city      telecon data
##    <chr>     <chr>   <list>
##  1 Bilbao    NAO     <tibble [68 x 3]>
##  2 Santiago  NAO     <tibble [68 x 3]>
##  3 Barcelona NAO     <tibble [68 x 3]>
##  4 Madrid    NAO     <tibble [68 x 3]>
##  5 Valencia  NAO     <tibble [68 x 3]>
##  6 Bilbao    WeMO    <tibble [68 x 3]>
##  7 Santiago  WeMO    <tibble [68 x 3]>
##  8 Barcelona WeMO    <tibble [68 x 3]>
##  9 Madrid    WeMO    <tibble [68 x 3]>
## 10 Valencia  WeMO    <tibble [68 x 3]>
## # ... with 30 more rows
```

```
str(slice(data_nest,1))
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    1 obs. of  3 variables:
##  $ city   : chr "Bilbao"
##  $ telecon: chr "NAO"
##  $ data   :List of 1
##   ..$ :Classes 'tbl_df', 'tbl' and 'data.frame': 68 obs. of  3 variables:
##   .. ..$ yr   : num  1950 1951 1952 1953 1954 ...
```

```
##   .. ..$ pr   : num  1342 1306 1355 1372 1428 ...
##   .. ..$ index: num  0.49 -0.07 -0.37 0.4 0.51 -0.64 0.17 -0.02 0.12 0.49 ...
```

El siguiente paso es crear una función, en la que definimos el test de correlación y lo pasamos al formato limpio, que aplicamos a cada agrupación.

```
cor_fun <- function(df) cor.test(df$pr,df$index,method="spearman")%>%tidy()
```

Ahora sólo nos queda por aplicar nuestra función a la columna que contiene las tablas por cada combinación entre ciudad y teleconexión. Para ello, usamos la función `map()` que aplica otra función sobre un vector o lista. Lo que hacemos es crear una nueva columna que contiene el resultado, una tabla del resumen estadístico, por cada fila de cada combinación.

```
data_nest <- mutate(data_nest, model = map(data,cor_fun))
data_nest
```

```
## # A tibble: 40 x 4
##    city      telecon data              model
##    <chr>     <chr>   <list>            <list>
##  1 Bilbao    NAO     <tibble [68 x 3]> <tibble [1 x 5]>
##  2 Santiago  NAO     <tibble [68 x 3]> <tibble [1 x 5]>
##  3 Barcelona NAO     <tibble [68 x 3]> <tibble [1 x 5]>
##  4 Madrid    NAO     <tibble [68 x 3]> <tibble [1 x 5]>
##  5 Valencia  NAO     <tibble [68 x 3]> <tibble [1 x 5]>
##  6 Bilbao    WeMO    <tibble [68 x 3]> <tibble [1 x 5]>
##  7 Santiago  WeMO    <tibble [68 x 3]> <tibble [1 x 5]>
##  8 Barcelona WeMO    <tibble [68 x 3]> <tibble [1 x 5]>
##  9 Madrid    WeMO    <tibble [68 x 3]> <tibble [1 x 5]>
## 10 Valencia  WeMO    <tibble [68 x 3]> <tibble [1 x 5]>
## # ... with 30 more rows
```

```
str(slice(data_nest,1))
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    1 obs. of  4 variables:
##  $ city   : chr "Bilbao"
##  $ telecon: chr "NAO"
##  $ data   :List of 1
##   ..$ :Classes 'tbl_df', 'tbl' and 'data.frame': 68 obs. of  3 variables:
##   .. ..$ yr   : num  1950 1951 1952 1953 1954 ...
##   .. ..$ pr   : num  1342 1306 1355 1372 1428 ...
##   .. ..$ index: num  0.49 -0.07 -0.37 0.4 0.51 -0.64 0.17 -0.02 0.12 0.49 ...
##  $ model  :List of 1
##   ..$ :Classes 'tbl_df', 'tbl' and 'data.frame': 1 obs. of  5 variables:
##   .. ..$ estimate   : num 0.153
##   .. ..$ statistic  : num 44372
##   .. ..$ p.value    : num 0.213
##   .. ..$ method     : chr "Spearman's rank correlation rho"
##   .. ..$ alternative: chr "two.sided"
```

¿Cómo podemos deshacer la lista de tablas en cada fila de nuestra tabla?

Pues bien, primero eliminamos la columna con los datos y después aplicamos simplemente la función `unnest()`.

```
corr_pr <- select(data_nest,-data)%>%unnest()
corr_pr
```

```
## # A tibble: 40 x 7
##    city    telecon estimate statistic  p.value method          alternative
```

```
##    <chr>   <chr>      <dbl>     <dbl>    <dbl> <chr>            <chr>
##  1 Bilbao  NAO        0.153     44372. 0.213    Spearman's rank~ two.sided
##  2 Santia~ NAO       -0.181     61902. 0.139    Spearman's rank~ two.sided
##  3 Barcel~ NAO       -0.0203    53460. 0.869    Spearman's rank~ two.sided
##  4 Madrid  NAO       -0.291     64692. 0.0169   Spearman's rank~ two.sided
##  5 Valenc~ NAO       -0.113     27600. 0.422    Spearman's rank~ two.sided
##  6 Bilbao  WeMO       0.404     31242. 0.000706 Spearman's rank~ two.sided
##  7 Santia~ WeMO       0.332     35014. 0.00594  Spearman's rank~ two.sided
##  8 Barcel~ WeMO       0.0292    50862  0.813    Spearman's rank~ two.sided
##  9 Madrid  WeMO       0.109     44660  0.380    Spearman's rank~ two.sided
## 10 Valenc~ WeMO      -0.252     31056. 0.0688   Spearman's rank~ two.sided
## # ... with 30 more rows
```

El resultado es una tabla en la que podemos ver las correlaciones y su significación estadística para cada ciudad y teleconexiones.

Heatmap de los resultados

Finalmente, hacemos un *heatmap* del resultado obtenido. Antes creamos una columna que indica si la correlación es significativa con p-valor menor de 0,05.

```
corr_pr <- mutate(corr_pr,sig=ifelse(p.value<0.05,"Sig.","Non Sig."))
```

```
ggplot()+
  geom_tile(data=corr_pr,
            aes(city,telecon,fill=estimate),
            size=1,
            colour="white")+
  geom_tile(data=filter(corr_pr,sig=="Sig."),
            aes(city,telecon),
            size=1,
            colour="black",
            fill="transparent")+
  geom_text(data=corr_pr,
            aes(city,telecon,label=round(estimate,2),
            fontface=ifelse(sig=="Sig.","bold","plain")))+
  scale_fill_gradient2(breaks=seq(-1,1,0.2))+
  labs(x="",y="",fill="",p.value="")+
  theme_minimal()+
  theme(panel.grid.major = element_blank(),
        panel.border = element_blank(),
        panel.background = element_blank(),
        axis.ticks = element_blank())
```


WeMO
SCAND
POL−EUAS
NAO
MO
EATL/WRUS
EA
AO
0.03
0.4
0.11
0.33
−0.25
0.16
0.36
0.31
0.44
−0.08
0.18
0.15
0.08
0.05
0.1
−0.02
0.15
−0.29
−0.18
−0.11
−0.25
−0.05
−0.5
−0.19
−0.05
0.16
0.02
0.06
0.18
0.05
−0.29
−0.26
−0.3
−0.01
−0.07
−0.01
−0.18
−0.31
−0.44
0.21
Barcelona
Bilbao
Madrid
Santiago
Valencia
0.4
0.2
0.0
−0.2
−0.4